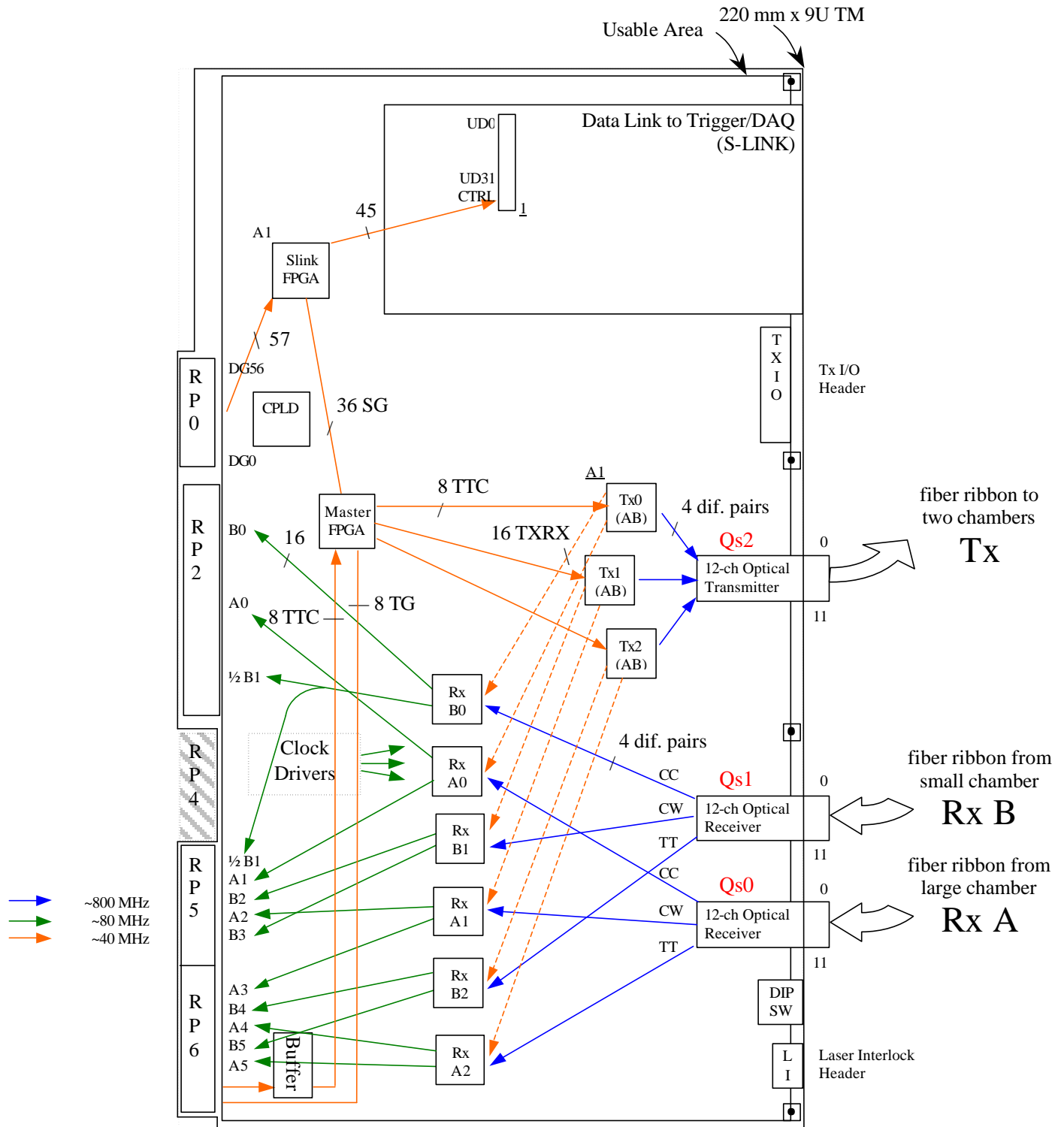


CSC Transition Module (CTM) Layout



Tx = XC2VP2 FPGA utilizing four MGT serializers

Rx = XC2VP2 FPGA utilizing four MGT deserializers

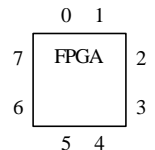
TXRX: Tx's and Rx's are related, because Rx needs SCA info and Tx needs lock info.

CC, CW, TT: From counterclockwise edge, clockwise edge, and top edge (transverse, two fibers unused) ASMIIT's. ** 16-bit data busses to ROD are organized one bus per layer (or all transverse layers).

Qsk = Opto.TempSensor.Qsensek temperature sense transistor.

** = see ChamberNomenclatureXX.xls

FPGA I/O Bank Locations



CSC Transition Module Features

CTM-Specific Features

Chambers supported:	2	chambers
Data fibers:	20+4	fibers *
Control fibers:	10+2	fibers *
Data stream from ASMII's	32 40	bits/ASMII MHz, including padding
Data stream to ROD:	16 80	bits/SPU MHz
Control stream from ROD:	16 40	bits MHz
Clock phase verniers:	1	per control fiber
Clock phase vernier step:	1.25	ns

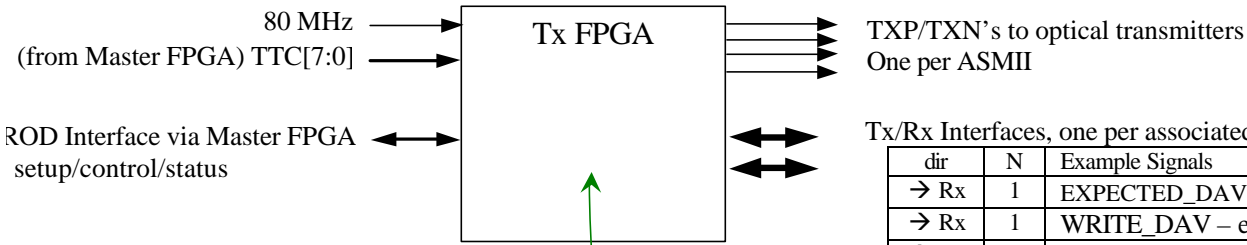
* Spare data fibers can be used for power supply current monitoring.

Spare control fibers can be used for calibration signals, e.g., DAC setting and CAL pulse.

System Features

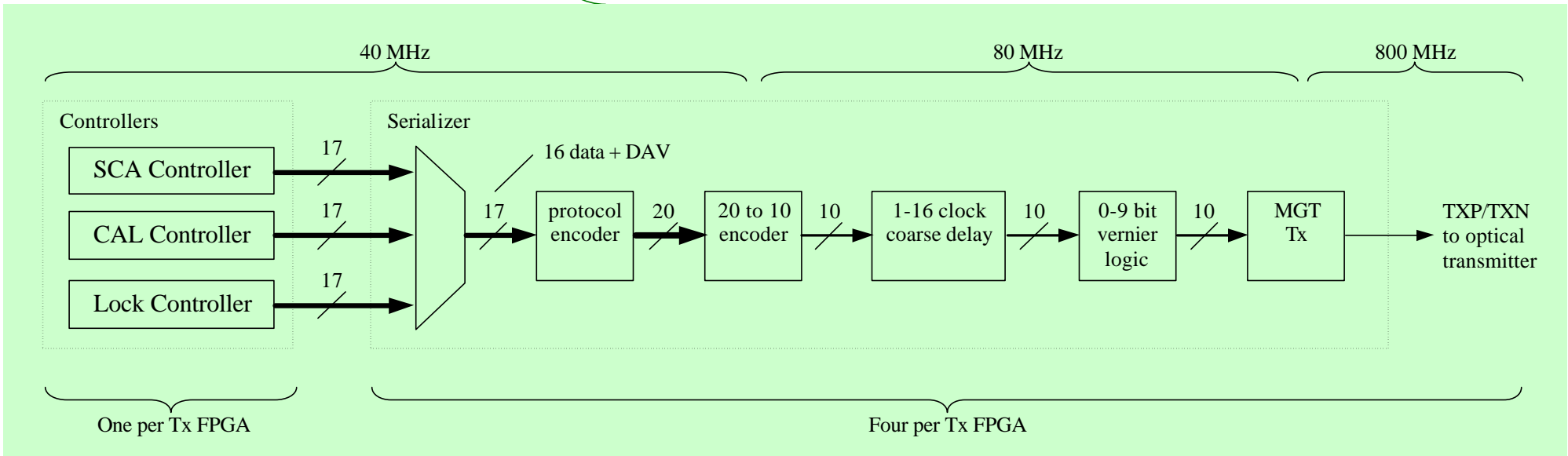
Data Synchronization:	The ASMII must connect one bit of its control stream to DAV* on both of its serializers.	
Lock Establishment:	The TM helps establish lock and recover from loss of lock by transmitting fill frames when data links lose lock.	on a per-ASMII basis
Optical Compatibility:	Optical Fibers, Transmitters and Receivers must be compatible.	
Optical Power Budget:	The CTM must operate within budgeted optical power limits.	
Laser safety:	The CTM disables all transmitters if there is excessive loss of lock. The CTM disables all transmitters when the rack door interlock signal is not present.	on a per-CTM basis

Tx FPGA

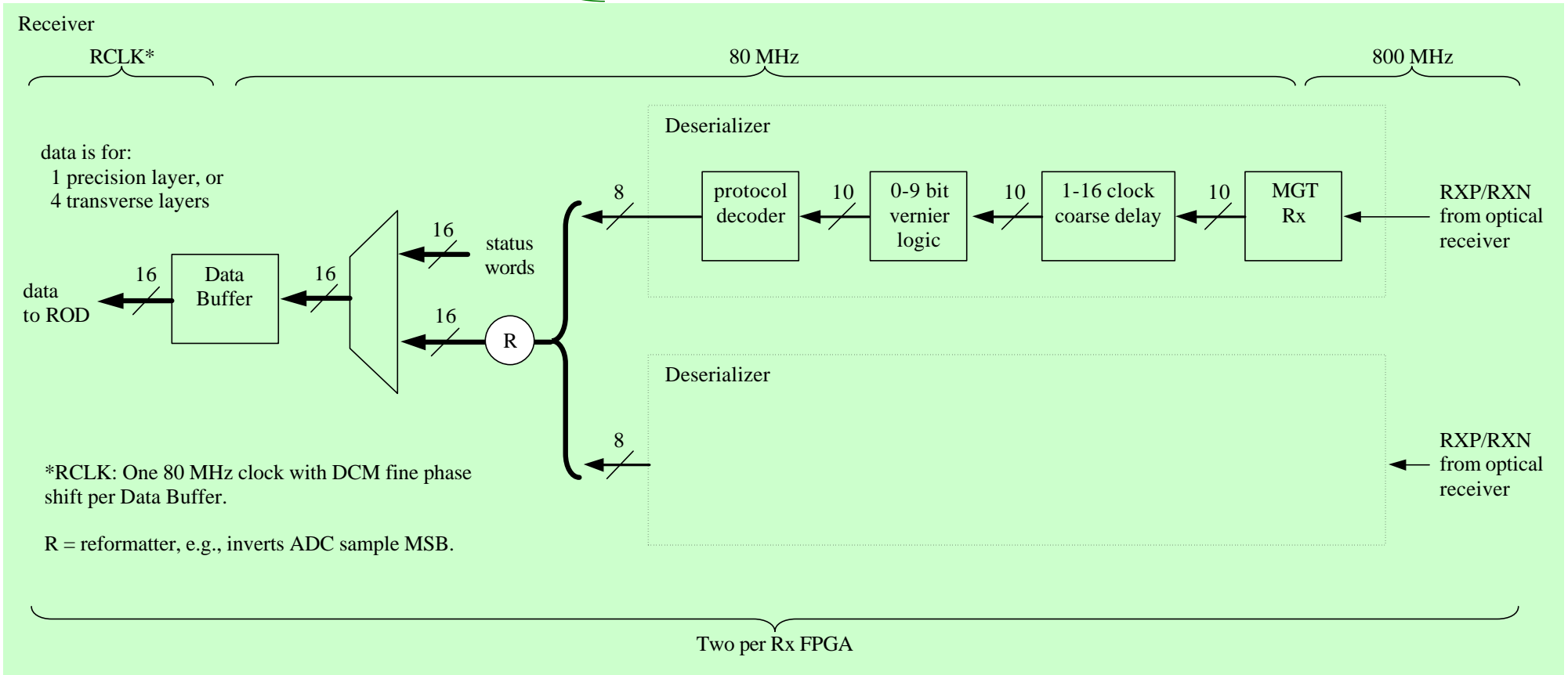
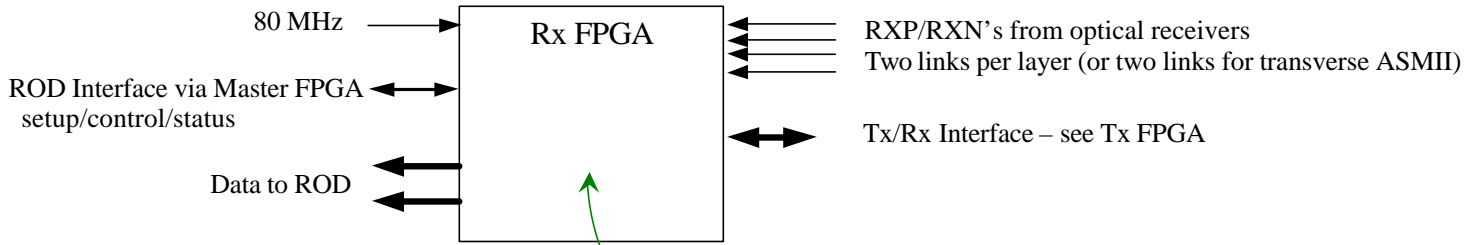


dir	N	Example Signals
→ Rx	1	EXPECTED_DAV – must match DAV's from Rx deserializers *
→ Rx	1	WRITE_DAV – enables write of link data to Rx data buffer *
→ Rx	1	SERIAL – serial time slice descriptor:
		START_BIT – indicates start of new slice
		FIRST – indicates first time slice of trigger
		PHASE – indicates even/odd trigger phase
		TYPE – trigger type, hi or lo priority
		SCA_ADDR – SCA read address
		SCAC_FAULT_CODE – fault code
→ Rx	1	SYNC_RESET – resets parts of Rx FPGA
→ Rx	1	TRIGGER – optional
← Rx	4	LOCK_STATUS – Rx deserializer lock status, details TBD
	8	spare

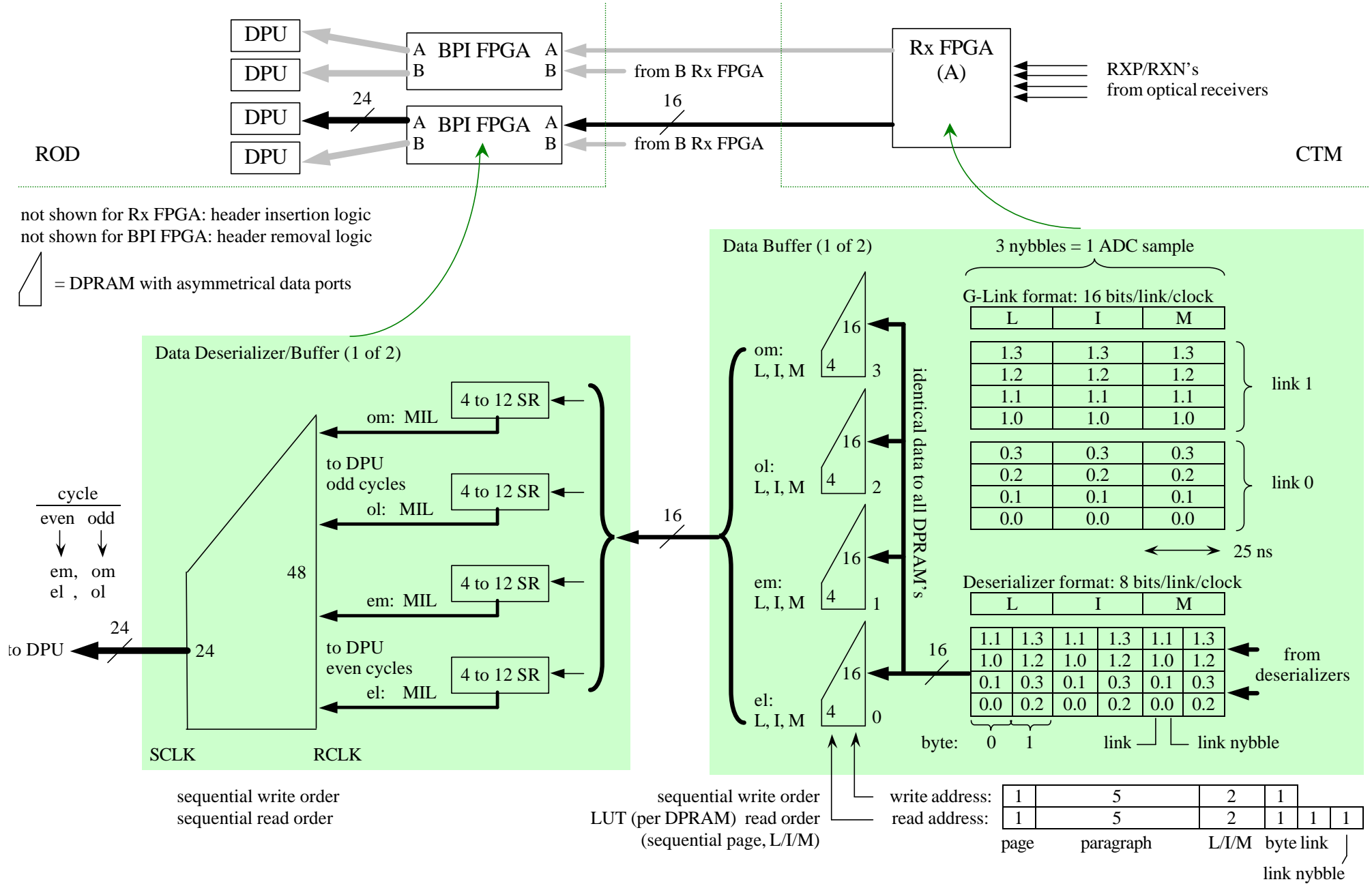
* DAV is periodically pulsed even when ASMII is not being read out.



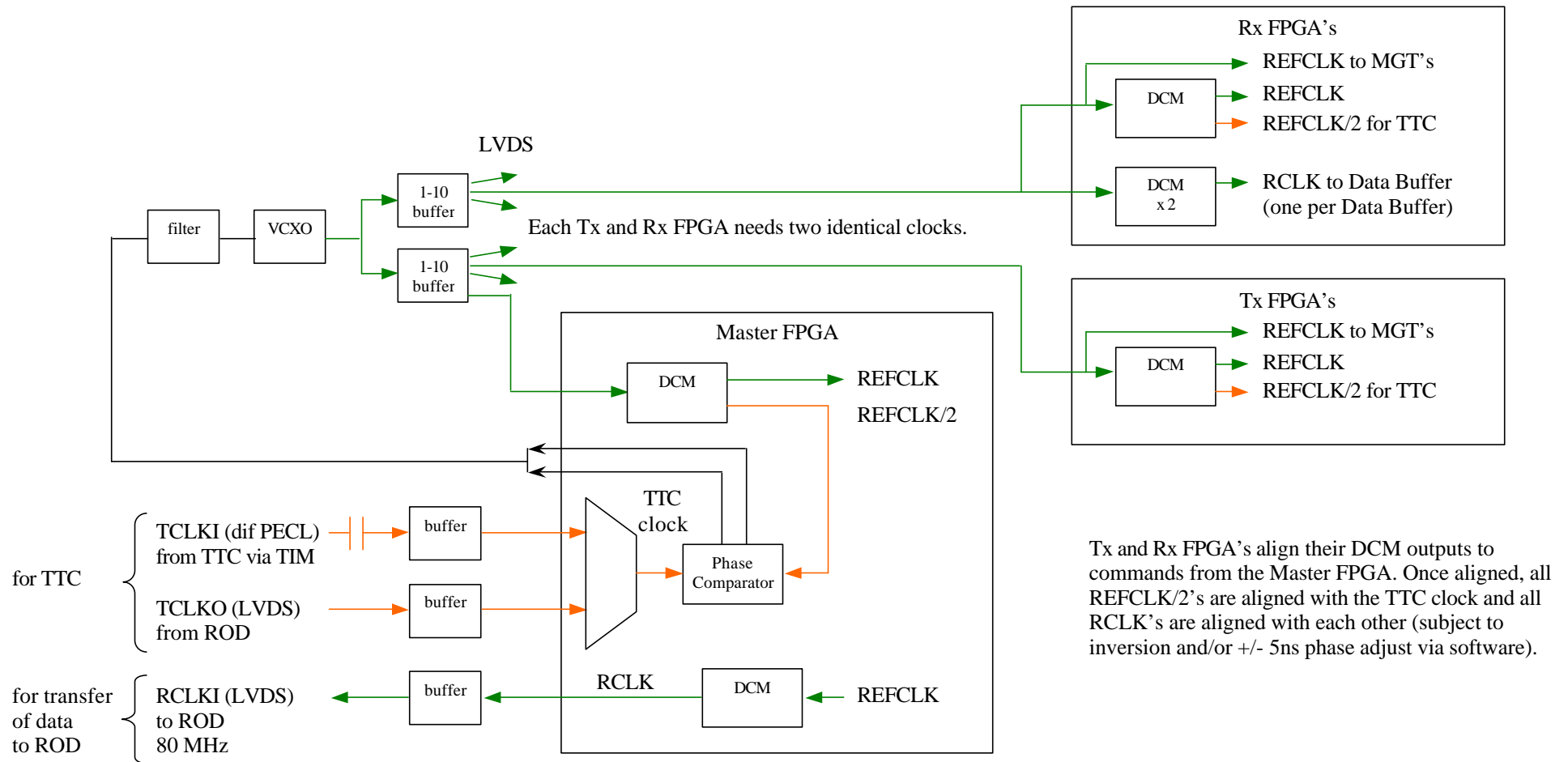
Rx FPGA



Data Buffering Detail



e/o = even/odd SCLK cycle (one 24-bit word to DPU each cycle)
m/l = most/least significant half of 24-bit word to DPU
M/I/L = 3 nybbles of 12-bit ADC sample, M = MSN, L = LSN



Tx and Rx FPGA's align their DCM outputs to commands from the Master FPGA. Once aligned, all REFCLK/2's are aligned with the TTC clock and all RCLK's are aligned with each other (subject to inversion and/or +/- 5ns phase adjust via software).

Sercom commands from Mx to Tx and Rx FPGA's:
Write, Read, Reset

TTC = ATLAS Timing Trigger and Control functionality at 40 MHz, e.g., L1A, BCR, etc.

Master FPGA:

receives TTC clk

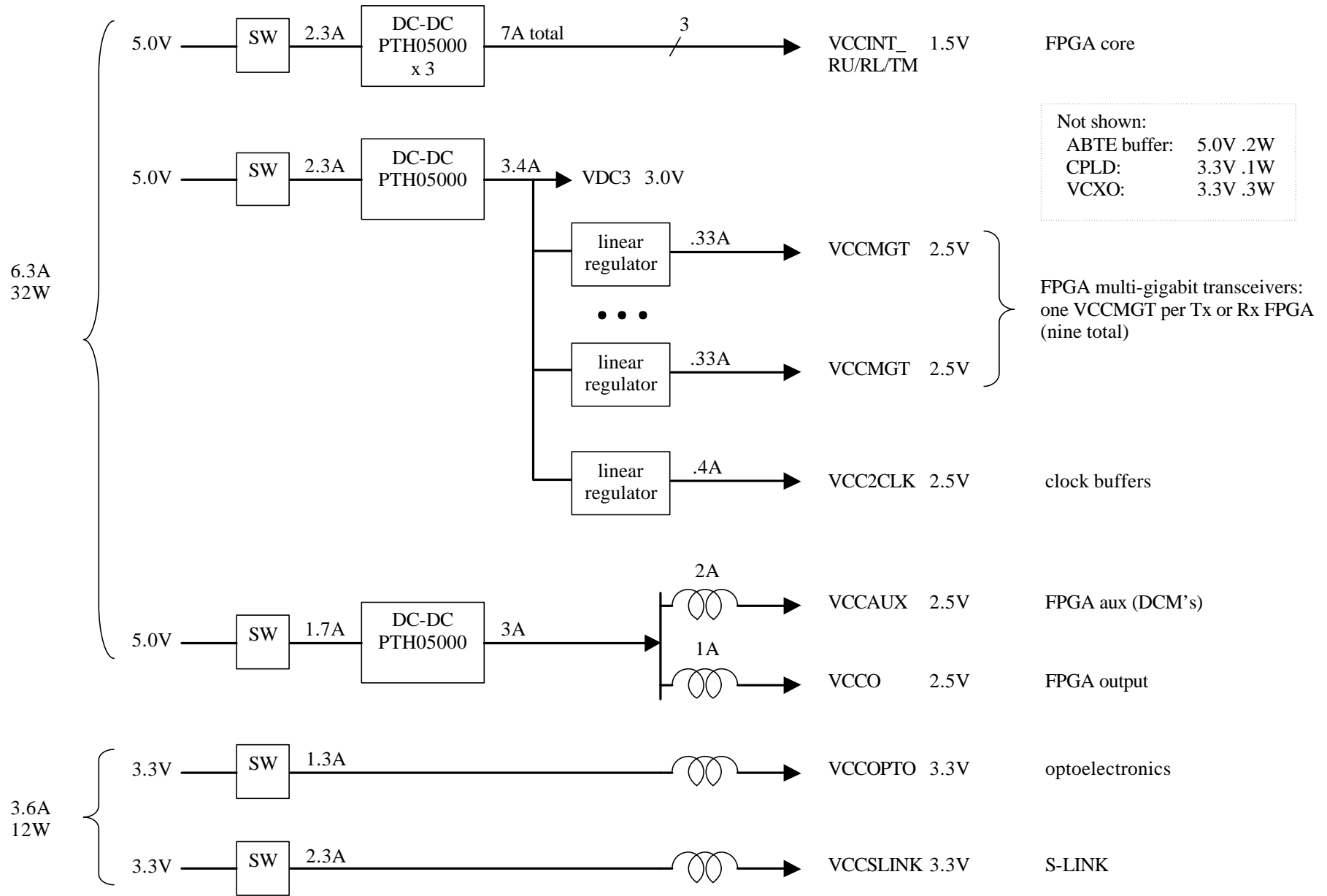
phase locks VCXO to TTC clk

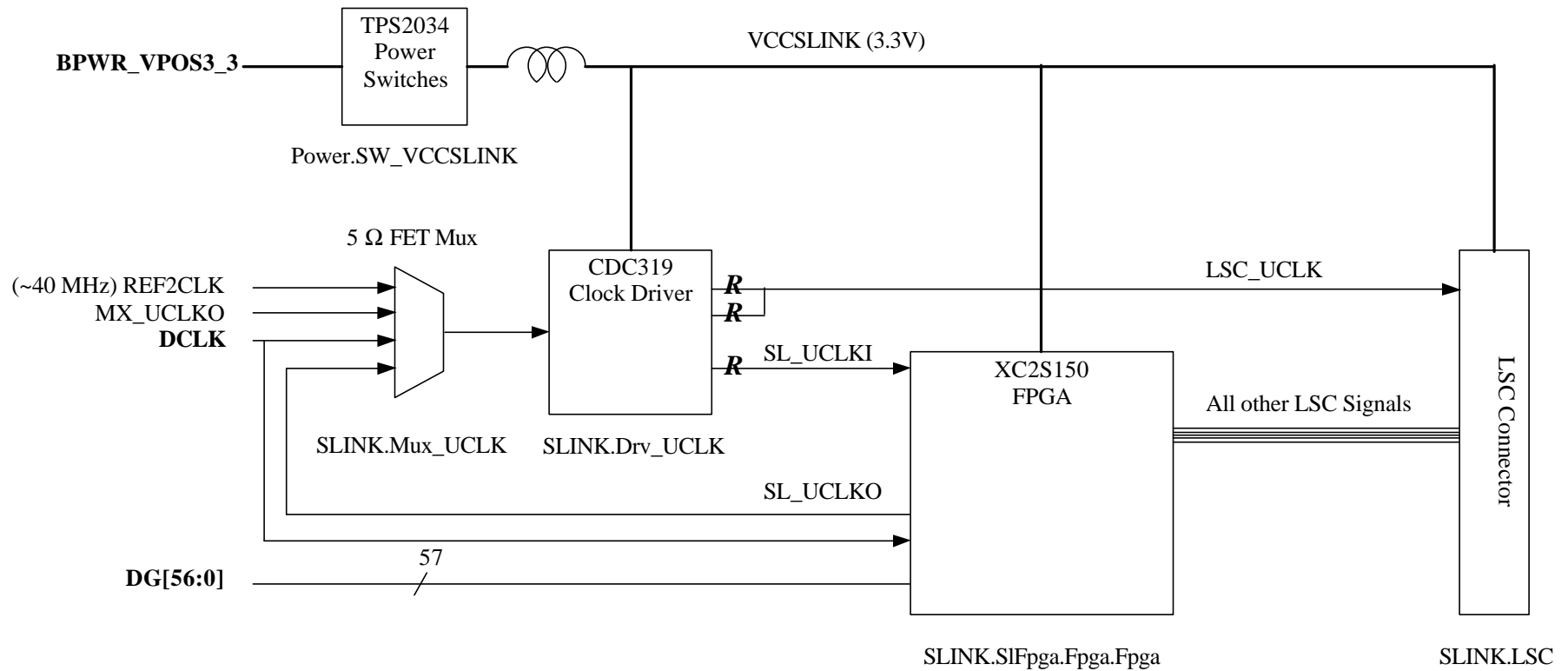
distributes TTC signals to Tx FPGA's

provides ROD with random access to registers in all Tx and Rx FPGA's

outputs SYNC pulse (via sercom) to slave Tx's and Rx's to align their 40 MHz clocks to TTC clk

outputs 80 MHz RCLK to ROD to be used to clock data arriving from CTM





Notes:

Nets with bold names are connected to the backplane. (An LVDS receiver on the CTM translates BK_DCLKOP/N to TTL).

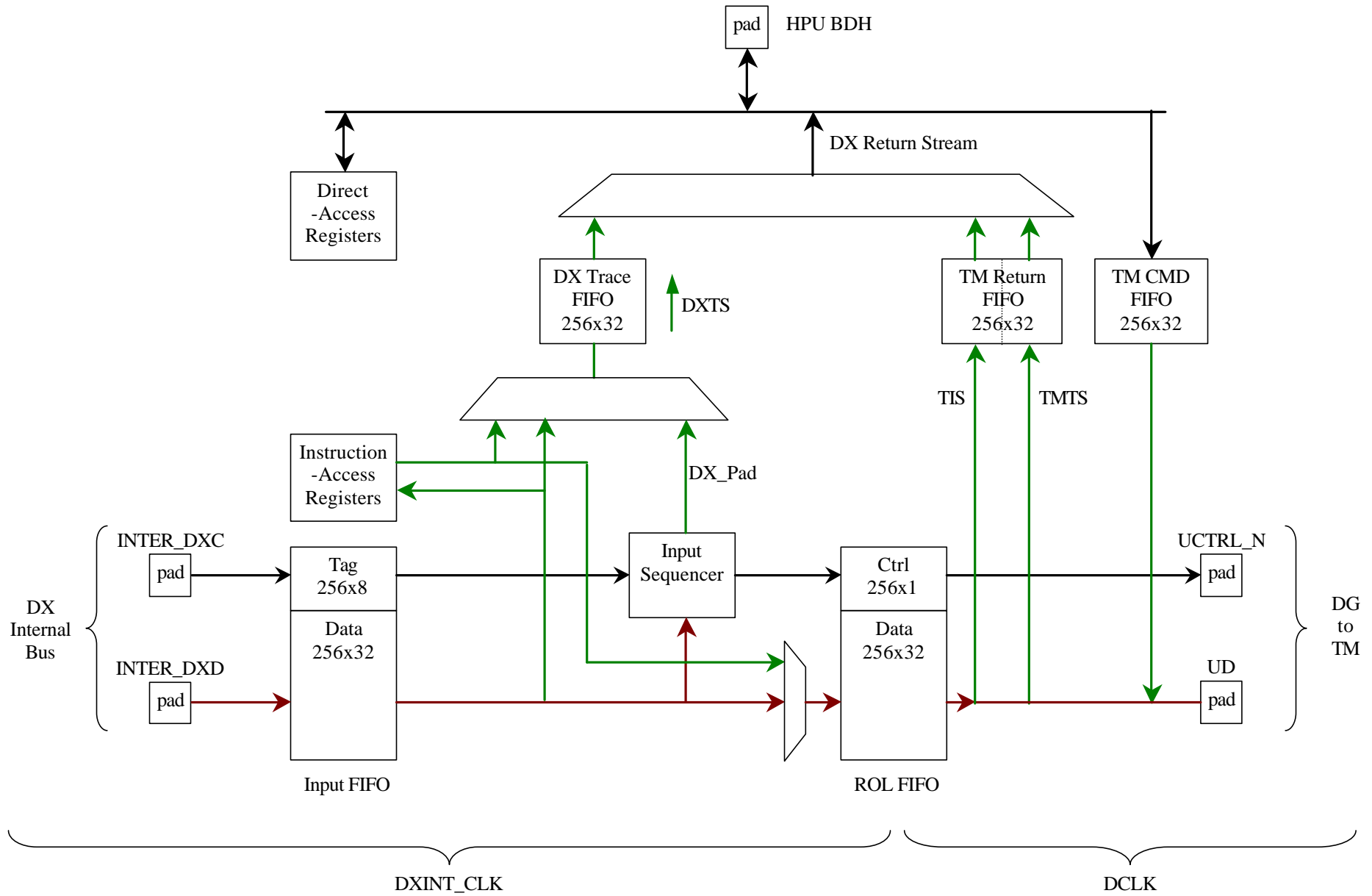
REF2CLK is equivalent to the ~40 MHz LHC BC clock. Use of other clocks is optional. Unused clocks can be disabled.

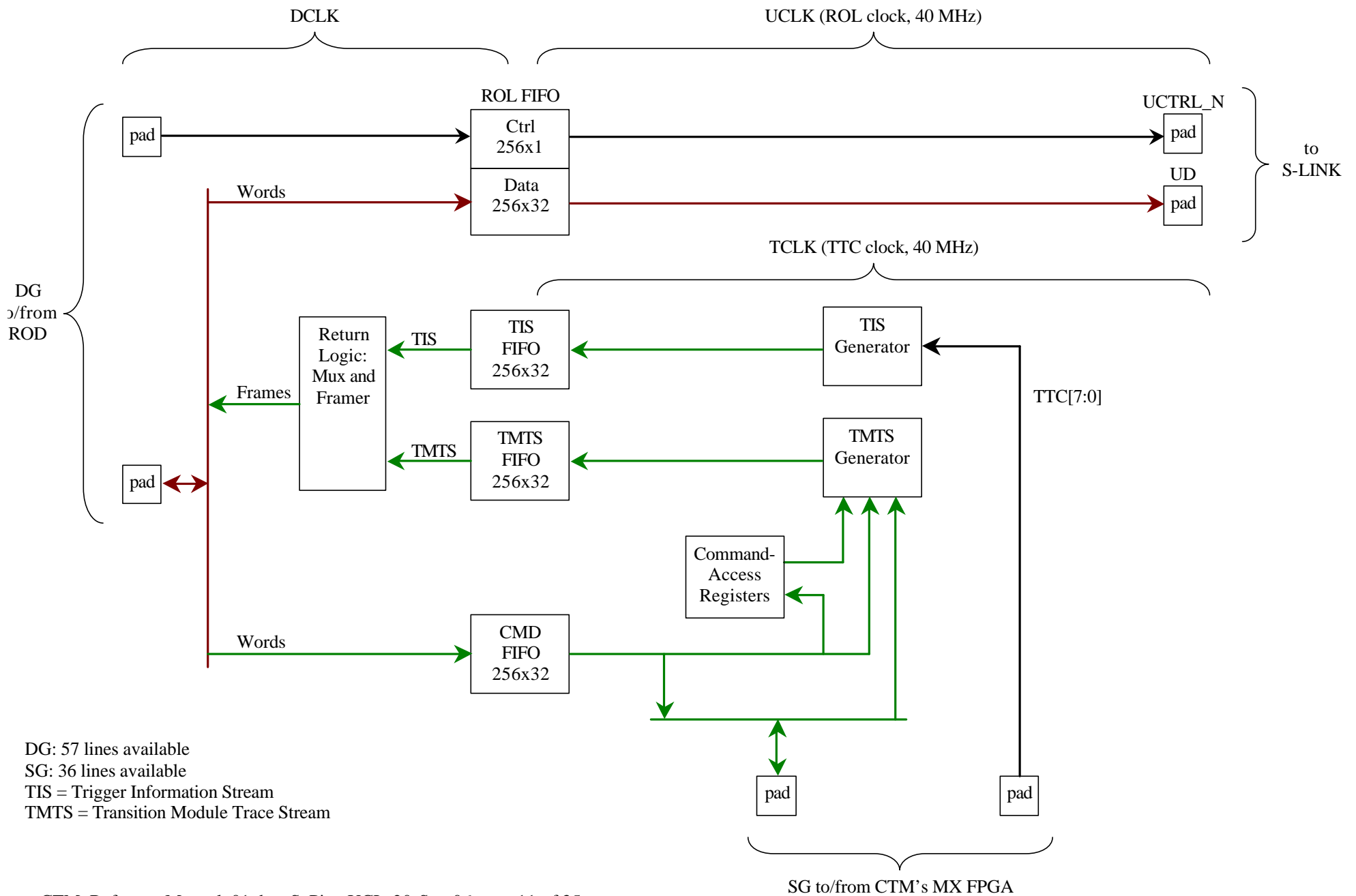
R = zero ohm resistor—for optional source termination.

LSC_UDW0 is connected to an FPGA I/O pin as well as an FPGA GCLK input. The GCLK connection is for possible future non-ATLAS, non-SLINK applications.

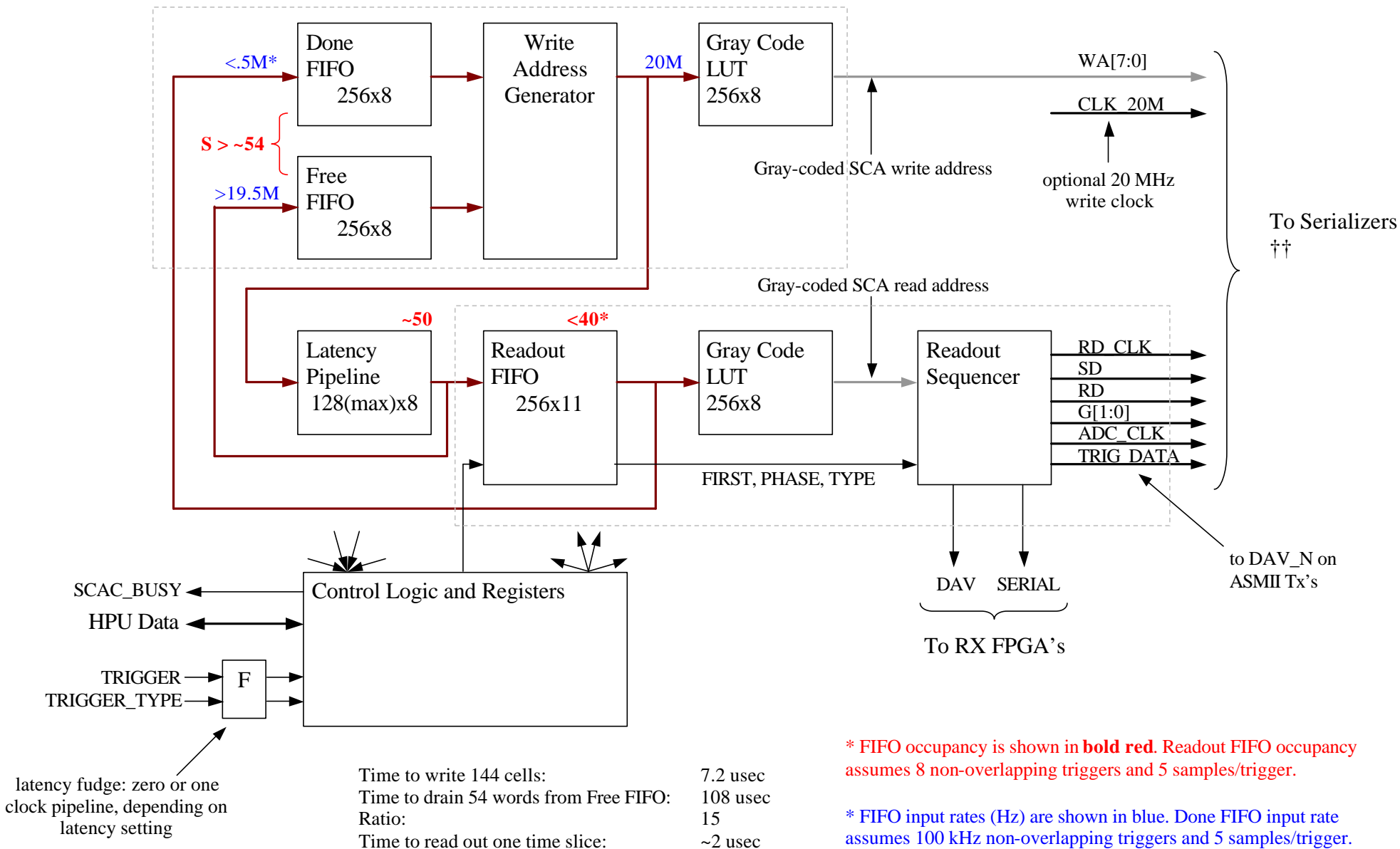
LSC VCC decoupling includes six .22 uF ceramic capacitors near LSC VCC pins and two 33 uF low-ESR tantalum capacitors.

LSC trace lengths should be less than 12 cm. FPGA output impedance can be varied to approximately match trace impedance.





DG: 57 lines available
 SG: 36 lines available
 TIS = Trigger Information Stream
 TMS = Transition Module Trace Stream



CTM Registers

All CTM Registers are accessible to the HPU via the TM Command/Response stream. The HPU sends commands to the DXB FPGA's TM Command FIFO. The commands are forwarded to the SL FPGA's Command FIFO. If a response is enabled, the TM returns the response to the HPU via the TMTS FIFO and TM Return FIFO. See DX_NotesXX.doc. The command and response have similar format:

```

                                     B   A   <-- RRR side
1111 CCCC ErWF NNNN  rrSM rTTT rRRR rRRR  <-- command word
AAAA AAAA AAAA AAAA  DDDD DDDD DDDD DDDD  <-- address/data word

```

E = enable response for this command

W = warning: the fault or reset bit for one of the targetted sercom's was set—
its data was replaced with iodoFault (0xfafa) or iodoReset (0xfefe)

F = hard fault -- due to Verilog bug, MX FPGA not operating (e.g., not yet configured), or electrical fault on the SG lines

N = number of expected response data words

N = number of words to follow (response stream)

N is 1 for writes and SL/MX reads; count of set T, R bits for TX/RX reads.

S = target enable for SL FPGA

M = target enable for MX FPGA (ignored if S is set)

T, R = target enable for TX and RX FPGA's (ignored if S or M is set)

CCCC = command opcode:

```

tmcmdPad      (0)    // pad (ok to ignore remainder of frame, legal in response stream only)
tmcmdWrite    (1)    // write to targets (S, M, or TTT RRR RRR)
tmcmdRead     (2)    // read from targets (S, M, or TTT RRR RRR)

```

Each command consists of two words. Response length is variable:

command opcode	length of response in words	notes
tmcmdPad	1	entire response word is zero
tmcmdWrite	2	entire response is identical to command
tmcmdRead	1+N	all fields except D are identical to corresponding fields in command

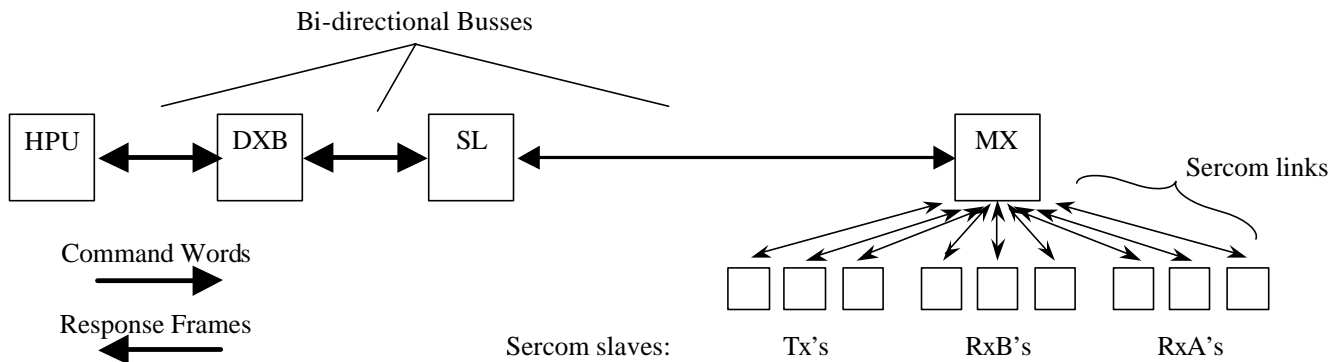
Words between a pad word and the end of the frame are guaranteed to be pad words.

No response will cross a frame boundary.

All r fields must be zero.

Illegal commands are ignored, but a fault flag is set.

Register Access Overview



TM Status Register in the ROD's DXB FPGA

In addition to registers on the CTM itself, the DXB FPGA on the ROD contains a summary status register. The SL FPGA continuously sends a 32-bit status word to the DXB FPGA via a dedicated serial line. The HPU can read the word from either of these DXB registers:

DXBR_TM_Status	direct-access register	-- HPU can read directly
DXBI_TM_Status	instruction-access register	-- HPU can read via the DX instruction stream

The registers contain:

<i>In DXB FPGA:</i>		
DXBR_TM_Status DXBI_TM_Status (read-only)	I rMMMMMM TTTTTTTT SSSSSSSS SSSSSSSS I = 1 → all 32 bits are invalid M = (MS to LS) ~AAL_LOCKED AAL_PHASE, ~ROFIFO_EF, ~TISFIFO_EF, ~TMTSFIFO_EF, ~CMFIFO_EF, T = T_CTM_SL::TMTS_Status[7:0] S = T_CTM_SL::Status	SL serial status. For access information, see DX_NotesXX.doc. ~ indicates logical not → most fields are zero in the absence of any unusual condition. E.g., all FIFO's should be empty at the end of a run. The DXB FPGA sets the I bit if the 31 LSB's were not recently updated by the TM, e.g., if the TM powered itself down.

The bits of T_CTM_SL::STATUS are detailed in the Keep-Alive section.

CTM Register Summary

The register summary tables below are intended to be used in conjunction with documentation in these CTM driver source files:

hpu_ctm.h	declarations of FPGA classes and most of their ports
hpu_ctm_scac.h	declarations of SCAC ports, converted from previous SCAC driver code with minimal modifications

CTM registers are accessed through hardware encapsulation classes designed specifically for the CTM. See hpu_ctm_bases.h. The classes exploit the order within the CTM's FPGA/module/register hierarchy.

```
class T_TransitionModule ... {
    ...
    T_CTM_SL      SL;          // 1 SL FPGA
    T_CTM_MX      MX;          // 1 MX FPGA
    T_CTM_TX      TX;          // 3 TX FPGA's
    T_CTM_RX      RX;          // 6 RX FPGA's      (0-2 = A, 3-5 = B)
    T_CTM_SCAC    SCA;         // 3 SCA controllers (in 3 TX FPGA's)
    T_CTM_SER     SER;         // 12 serializers      (in 3 TX FPGA's)
    T_CTM_RCV     RCV;         // 12 receivers        (in 6 RX FPGA's)
    ...
};
```

The constructor of each T_CTM_... member declared above associates a group with each of its member ports. The group determines the number of physical registers associated with each port. See hpu_ctm.h. Groups are defined in hpu_ctm.cpp.

A uniform syntax is provided for port and field writing:

Example port access:

```
TM.RX.Test() = 1;           // write 1 to the test register in each RX FPGA
TM.RX.Test( 4 ) = 9;        // write 9 to the test register in RX FPGA #4
TM.SCA.LatencyCells() = 0;  // write zero to the LatencyCells ports of all 3 SCA controllers
```

Example field access:

```
TM.SCA.LatencyCells.LATENCY() = 100; // set latency of 100 for all three SCA controllers
TM.SCA.LatencyCells.LATENCY( 2 ) = 99; // set latency of 99 for SCA controller #2
```

There are two classes of ports: TCTM_Ind_Port and TCTM_All_Port. The syntax for their use is identical, but when writing to a TCTM_All_Port, the index in parenthesis is ignored and all registers in the port's group are written. TCTM_All_Port is more efficient than TCTM_Ind_Port, both in speed and memory use. To allow writing of individual fields, TCTM_Ind_Port must maintain an image variable for each of its group's members while TCTM_All_Port maintains a single image variable for all of its group's members.

Each TCTM_Field has an associated port, which determines the field's group and access behavior. TCTM_Field's are typically declared as members of a class derived from TCTM_Ind_Port. These classes are named PI_... as a reminder that they are derived from a port whose group members may be accessed individually.

In general, high level code does not access CTM registers directly. Instead, it calls CTM driver functions, e.g.,

```
TM.Reset();
TM.SetNormalMode();
```

SL FPGA:

class T_CTM_SL		SL global registers	
TCTM_Ind_Port	Status	SSSSSSSS SSSSSSSS See details in the Keep-Alive section.	SL status register.
TCTM_Ind_Port	AAL_Status	SSSS00SS HHHHHHHH S (MS to LS): AAL_MANUAL, AAL_MANUAL_PHASE, AAL_LOCK, DG_OUT_PHASE, AAL_LOCKED, AAL_PHASE H = AAL_HIST	Auto-alignment status register.
TCTM_Ind_Port	TMTS_Status	rrrrrrrr rFFFFFFF F (MS to LS): FAULT_CMDX_STATE, FAULT_SG_IDLE_ACK, FAULT_SG_FAIL, FAULT_SG_WARN, FAULT_ILLEGAL_CMD, FAULT_TMTSFIFO_OVF, FAULT_CMFIFO_OVF	Status from TMTS generator.
TCTM_Ind_Port	TMTS_Release	Manual frame release register. Writing any value to this register enables the current response frame to be padded and released for readout as soon as the TMTS return path is empty. The “current response frame” will include the response of the command that wrote this register if the comand’s response enable (E) bit was set. If manual frame release is not used, partial response frames are padded and released 50 microseconds after the TMTS return path goes empty. (See <code>ctm_tmts_generator.v</code> .)	

SL FPGA (cont.):

PI_SL_Control	Control	SL main control register
PI_SL_FPGA_Program	FPGA_Program	FPGA PROGRAM_N control register
PI_LED_Control	LED_Control	<p>MMMM MMMM EEEE EEEE LED control register</p> <p>M = minimum flash duration in ms</p> <p>E = source enables (MS to LS):</p> <p>CMFIFO_WEN, // active when ROD writes word to CMD FIFO</p> <p>ROFIFO_WEN, // active when ROD writes word to ROL FIFO</p> <p>TIS_FIFO_REN, // active only when frame is being read out to ROD</p> <p>TMTS_FIFO_REN, // active only when frame is being read out to ROD</p> <p>SERINFO_LED, // active if any MX serial stream fault</p> <p>TIS_LED, // e.g., active if any TIS fault</p> <p>TMTS_LED, // e.g., active if any TMTS fault</p> <p>1'b1 // constant ON source --> HPU can control LED</p>
TCTM_Ind_Port	Test	see standard test register note
SL Trigger Information Stream registers		
TCTM_Ind_Port	TIS_Status	<p>rrrrrILB ssssssss Trigger Information Stream status register.</p> <p>I = input frame state machine fault, should never happen</p> <p>L = L1ID rollover before ECR</p> <p>B = unexpected BCR or BCID value</p> <p>s = specific faults—same as in TIS record</p>
TCTM_Ind_Port	TIS_Watch	watch timeout period register, units is TCLK cycles (e.g. 25 ns)
TCTM_Ind_Port	BCID_Reload	BCID reload--MS nybble is ignored
TCTM_Ind_Port TCTM_Ind_Port	EL1ID_MS EL1ID_LS	Extended L1ID counter access registers. Reading these registers returns the EL1ID counter value captured via the SLR_TIS_SET register. Writing these registers saves the value to be loaded into the counter via the SLR_TIS_SET register.
TCTM_Ind_Port TCTM_Ind_Port	Orbit_MS Orbit_LS	Orbit counter access registers. Reading these registers returns the orbit counter value captured via the SLR_TIS_SET register. Writing these registers saves the value to be loaded into the counter via the SLR_TIS_SET register.
PI_SL_TIS_Control	TIS_Control	misc. control bits, including RUN and RUN_ORBIT *1
PI_SL_TIS_Timeout	TIS_Timeout	timeout values: TIS_FIFO_TMO_T, SER_IN_TMO_T *2
PI_SL_BCID_Max	BCID_Max	BCID_MAX_D, BCID_MAX *3
PI_SL_TIS_Set	TIS_Set	misc. load/capture, etc. bits, including bit to generate watch record

*1: While RUN is zero:

The user may set the EL1ID counter value.

The TM's TIS FIFO is reset (after any frame readout already in progress).

TIS queues are reset → all events, including watch events, are ignored.

When RUN is released:

The EL1ID is incremented after each L1A.

When RUN_ORBIT is zero:

The user may set the orbit counter value.

When RUN_ORBIT is released:

The first L1A enables incrementing of the orbit counter (at the next BCR).

The BCID counter is not affected by RUN or RUN_ORBIT.

*2: TIS_FIFO_TMO_T is the maximum time an incomplete TIS frame will remain in the TIS FIFO before being padded and released for readout. This timeout occurs only if the remainder of the TIS return path (TIS FIFO's on TM and ROD) is empty and no event is ready to be recorded in the incomplete TIS frame. SER_IN_TMO_T is the time after which an L1A's TTC serial ID or trigger type is considered lost. This timeout results in a fault.

*3: The value in the BCID_MAX field is used to detect BCR faults, i.e., instances of the BCR arriving with unexpected BCID counter value. The minimum non-fault counter value upon BCR is BCID_MAX-BCID_MAX_D; the maximum is BCID_MAX+BCID_MAX_D. When internal BCR generation is used, internal BCR's are issued such that the maximum BCID counter value is BCID_MAX.

MX FPGA:

class T_CTM_MX		
TCTM_Ind_Port	SlaveEnables *1	one bit per sercom slave FPGA, 0 --> reset slave, 1 --> unreset slave
TCTM_Ind_Port	Status	<pre> rrrrrrrrr SSSSSSSS MX status register. S (MS to LS): OTF_DETECTED, // * 1 --> LM83 supervisor detected overtemp or diode fault ~OPIO_OK, // * 1 --> failure of SL/MX one-pin communication ~INTERLOCKED_1, // * 1 --> rack rear door interlock failed, one bit per interlock signal ~INTERLOCKED_0, EXLOL_RB, // * 1 --> RXB FPGA's report excessive loss of lock EXLOL_RA, // * 1 --> RXA FPGA's report excessive loss of lock SLAVE_BAD, // * 1 --> bad sercom communication with a TX or RX FPGA MX_DIS_LASERS // * 1 --> lasers will be disabled, e.g., due to excessive loss of lock </pre> <p>* = contributes to MX keep-alive logic -- see the Keep-Alive section.</p>
TCTM_Ind_Port	TG_Debug	current value of TG -- typically useful only for DC debugging of TG/TTC lines
TCTM_Ind_Port	SlaveFaults *1	sercom fault for each slave
		flags from slaves, of dubious utility
TCTM_Ind_Port	SlaveFlags_RA	4 bits per RX FPGA, MSN is zero -- set bit indicates excessive loss of lock for each the FPGA's four links: deserializer/link = 1/1 1/0 0/1 0/0
TCTM_Ind_Port	SlaveFlags_RB	
TCTM_Ind_Port	SlaveFlags_TX	currently zero
PI_MX_Control	Control	control register, including fields USE_SOFT_TTC, ENABLE_BUSY, others
PI_LED_Control	LED_Control	<pre> MMMM MMMM EEEE EEEE LED control register M = minimum flash duration in ms E = source enables (MS to LS): !BUSY_IN_N, // active when ROD or CTM is driving backplane BUSY_N line BUSY_OUT, // active when CTM is driving backplane BUSY_N line 1'b0, // not currently used !LASER_EN, // active when MX believes lasers should be disabled SLAVE_BAD, // any sercom fault or reset SRUN, // active when sercom sequence is running LED_LOC_CYCLE, // active when SL FPGA accesses MX command-access resources 1'b1 // constant ON source --> HPU can control LED </pre>
TCTM_Ind_Port	SoftTrigger	Write-only: if USE_SOFT_TTC is true, the halfword written to this register is shifted out as ~TTC[0] (L1A). Shifting is MSB first, e.g. writing 1000010000000001 results in three triggers: at t = x, t = x+5, t = x+15.
TCTM_Ind_Port	Test	see standard test register note

*1: These registers have standard target format:

```

      B      A      <-- RX side
rrrr rTTT rRRR rRRR   T = TX, R = RX

```

TX FPGA:

class T_CTM_TX		
TCTM_Ind_Port	Status	<p>PMFFFrFFF I I I I I I I I I I TX status register. P=PHASE_FAULT, // incorrect CLK40 phase seen M=~PHASE_MATCH, // correct CLK40 phase was never seen F (MS to LS): FAULT_RXS_DAV, // 1 --> SCAC DAV was active at same time as periodic DAV FAULT_RXS_START, // 1 --> next slice starts before all serial info sent to RX FPGA (reserved) FAULT_SCAC_FREE_FIFO, // these three faults are from SCAC FAULT_SCAC_WRITE_ADDR, FAULT_READOUT_SEQ, I = INVALID_BA, // deserializer output invalid *</p> <p>M may take several milliseconds to clear after CTM initialization. Otherwise, set F or M bits indicate a hardware failure. PHASE_FAULT is only set after PHASE_MATCH is set. * of dubious utility, see Verilog for details</p>
PI_TX_Control	Control	control register, including field SCAC_RUN
PI_TX_CAL_MainControl	CAL_MainControl	Calibration Board (CALB) main control register
PI_TX_CAL_LinkControl	CAL_LinkControl	Calibration Board (CALB) link control register -- many bits are connected directly to corresponding G-Link data lines
PI_LED_Control	LED_Control	<p>MMMM MMMM EEEE EEEE LED control register M = minimum flash duration in ms E = source enables (MS to LS): LS_NO_NEW_BA, // active when any RX FPGA does not send its serial status SLICE_START, // active at beginning of each time slice CAL_LED, // active when cal controller is enabled and outputs cal pulse (-LS_LOCK_SUM_BA), // active when any RX link reports loss of lock 1'b0, // not currently used 1'b0, // not currently used IO_STB_N, // active when ROD accesses any register (even if illegal page) 1'b1 // constant ON source --> HPU can control LED</p>
TCTM_Ind_Port	Test	see standard test register note
class T_CTM_SCAC		
PI_SCAC_Setup	SCAC_Setup	general setup
PI_ROSEQ_Setup	ROSEQ_Setup	readout sequencer setup
PI_LatencyCells	LatencyCells	latency and cell count
		the LUT ports represent the base of each LUT space
TCTM_All_Port	ROS_LUT	readout sequencer lookup table
TCTM_All_Port	GRY_LUT	graycode lookup table
class T_CTM_SER		
PI_SER_Control	Control	general setup
PI_SER_Delay	Delay	misc. delay values
TCTM_Ind_Port	TestData	data output on G-Link when SER_SRCE is set to srceTest
PI_SER_Loopback	Loopback	loopback control register—used by driver to measure and minimize MGT TX latency
TCTM_Ind_Port	CV_Time	TX→RX on-chip loopback delay measurement—used by driver to measure MGT TX latency: zero if no valid measurement format is 0CCV, with units = CC: 10 bit periods, V: 1 bit period

RX FPGA:

class T_CTM_RX		
TCTM_Ind_Port	Status	<p>PMrrrEEEE LLLLLLLL RX status register.</p> <p>P=PHASE_FAULT, // incorrect CLK40 phase seen</p> <p>M=~PHASE_MATCH, // correct CLK40 phase was never seen</p> <p>E=EXCESSIVE_LOL, // excessive loss-of-lock—one bit per deserializer</p> <p>L=LS_STATE // lock status—two bits per deserializer:</p> <p> lsReset // lock system resetting</p> <p> lsVernAlign // lock system attempting vernier (bit) alignment</p> <p> lsWordAlign // lock system attempting word alignment</p> <p> lsLocked // lock system locked</p> <p>M may take several milliseconds to clear after CTM initialization. Otherwise, set F or M bits indicate a hardware failure. PHASE_FAULT is only set after PHASE_MATCH is set.</p>
PI_RX_Control	Control	RX control register, contains TTC_LATENCY field.
PI_LED_Control	LED_Control	<p>MMMM MMMM EEEE EEEE LED control register</p> <p>M = minimum flash duration in ms</p> <p>E = source enables (MS to LS):</p> <p> EXPECTED_DAV, // active when TX expects RX to see DAV on data links</p> <p> SLICE_START, // active at beginning of each time slice</p> <p> 1'b0, // not currently used</p> <p> 1'b0, // not currently used</p> <p> ANY_LOL, // active when any link is experiencing any loss of lock</p> <p> EXCESSIVE_LOL, // active when any link is experiencing excessive loss of lock</p> <p> IO_STB_N, // active when ROD accesses any register (even if illegal page)</p> <p> 1'b1 // constant ON source --> HPU can control LED</p>
PI_LOL_Override	LOL_Override	<p>loss-of-lock override register</p> <p>*** See notes in ctm_rx.v before using this register. ***</p> <p>This register allows the HPU to override some loss-of-lock logic that might otherwise shut down parts of the CTM.</p> <p>A limited number of links (two per RX FPGA) can have their loss-of-lock logic permanently overridden, allowing the CTM to operate with some unused RX links. Two RX links per fiber ribbon are unused in ATLAS, and PI_LOL_Override::Initialize() automatically overrides their loss-of-lock logic.</p> <p>PI_LOL_Override::OverrideTemporary() allows loss-of-lock logic for all four links of any set of RX FPGA's to be overridden for up to twelve seconds. This feature is intended for initial debugging only and must be disabled in production HPU software, because it can defeat laser safety mechanisms.</p> <p>*** See file hpu.h: Env_AllowTemp_LOL_Override must not be defined in production HPU software. ***</p>
TCTM_Ind_Port	ClearRun	Writing any value clears RUN bit. The RUN bit is set by the first slice of a run, enabling incrementing of accounting registers and slice and event counters.
TCTM_Ind_Port	Test	see standard test register note

RX FPGA (cont.):

class T_CTM_RCV		receiver registers (two receivers in each RX FPGA)
PI_RCVR_Control	RCVR_Control	general setup
PI_RSEQ_Control	RSEQ_Control	readout sequencer setup
PI_ACC_Control	ACC_Control *	Accounting control register. By default, accounting for all conditions is enabled. The ACC_Incidents register counts incidents of conditions. The ACC_Time register counts the duration of incidents of conditions. Bits are provided to disable counting incidents of the following conditions: RX link protocol error (one bit per link) RX link flag error (one bit per link) RX link DAV error (one bit per link) RX link loss of lock (one bit per link) TX FPGA asserting INVALIDATE, as extended in the RX FPGA Bits are provided for counting the duration of the following conditions: RX link loss of lock (one bit per link) TX FPGA asserting INVALIDATE, as extended in the RX FPGA RX link protocol error, as extended in the RX FPGA RX link loss of lock, as extended in the RX FPGA In addition, the ACC_ONLY_ON_RUN bit enables counting only when the RX FPGA's RUN bit is set. The RUN bit is normally set by the arrival of the first slice of a run. The default value of ACC_ONLY_ON_RUN (zero) allows counting before the run starts, e.g., to monitor link performance.
TCTM_Ind_Port	ACC_Incidents *	incident accounting saturating counter -- counts incidents selected in ACC_Control
TCTM_Ind_Port	ACC_Time *	time accounting saturating counter -- counts duration in ms of incidents selected in ACC_Control
PI_ReformatMask	ReformatMask	reformat mask and receiver ID
TCTM_Ind_Port	RCLK_Phase	RCLK phase control -- writing 0/1 steps phase +/- (hardware saturates phase in both directions)
TCTM_Ind_Port	Delay	CCCC VVVV cccc vvvv Deserializer delay register, read-only CV: for deserializer 1 cv: for deserializer 2 C/c = coarse (10-bit byte) delay V/v = vernier (bit) delay Delays are determined by the deserializer's lock acquisition mechanism. A deserializer's delay halfword is zero when the link is not locked.
TCTM_Ind_Port TCTM_Ind_Port	ReadoutLUT_X ReadoutLUT_Y	data buffer readout lookup tables, one group for X, one for Y

* ACC_ suggests accounting or accumulation. A better choice might have been QOS_ for quality of service. These registers are intended to provide a summary of link performance, e.g., over the course of a several-hour run. By disabling some conditions, they can also be used for debugging hardware.

Standard Test Register Note:

Any value may be written. The value is post-incremented after each read.

Atlas Trigger Rules

The following rules are not currently verified by the CTM:

T triggers in 80 microseconds, max → “Currently, the baseline is to have less than 8 L1A within 80 μs.”
 K clocks between triggers, min → “After each L1A a 4 BC dead-time is introduced by the CTP.”

Input to Gigabit Serializers

The TX_MODE field of the TM.SCA.SCAC_Setup register determines the output mode of the SCA controller.

The SER_SRCE field of the TM.SER.Control register determines the input to the gigabit serializer logic.

When SER_SRCE is srceSCA_C, TX_MODE chooses one of the following:

TX_MODE	input to gigabit serializer
txmSCAC	normal SCA control stream
txmTestPattern	test pattern: 0x0001, 0x0002, 0x0004, 0x0008, ..., 0x4000, 0x8000, 0x0001, 0x0002 ...
txmSimulateASMI	Simulated ASMI output. This mode is intended for use with a loopback fiber ribbon between CTM Tx and one CTM Rx. In this case the DPU(s) will receive simulated ASMI data such that each data halfword arriving at the DPU has the format described below.
txmReserved	undefined

Simulated ASMI Output Format

0000TTTTCCCC0AMM:

bits	range	description
TTTT	0-15	Zero for the first time slice, incremented for each subsequent time slice.
CCCC	0-11	The SCA channel. Each SCA has 12 channels.
A	0-1	The ASMI ADC that converted the data. Each ADC services one SCA.
MM	0-3	The ASMI mux chip that carried the data—this corresponds with the GLink nybble that carried the data (0→Glink bits 0-3, 1→ Glink bits 4-7, etc.) Each mux services two ADC’s.

Data are transmitted on the GLink in triplets:

MSN			LSN	description	increment
0A11	0A10	0A01	0A00	0AMM, A is the same for all nybbles	A toggles every triplet
CCCC	CCCC	CCCC	CCCC	SCA channel: same for all nybbles	CCCC increments every other triplet
TTTT	TTTT	TTTT	TTTT	time slice: same for all nybbles	TTTT increments every time slice *

* The actual value transmitted is T*T T T (the MSB of TTTT is inverted). The receivers in RX FPGA’s undo this inversion such that the halfwords arrive at the DPU in the format shown in the first table above.

Timeslice Status Words

After it has buffered all data words for a slice, a receiver writes the following to its buffer:

```

STAT_TRAILER    STAT_HEADER    // 0xfae          0xfed          constants
STAT_SCAC_H     STAT_SCAC_L     // CCCCCAAAAAAA  IIIIXrrrrBTPF SCAC slice info
STAT_EVENT      STAT_SLICE      // event          slice          counters
STAT_ERR_SUM_H  STAT_ERR_SUM_L  // see below      see below      error summary
  
```

SCAC slice info sent serially from the TX FPGA to its associated RX FPGA's includes:

C = SCAC fault code
 A = binary write address, i.e., address before gray encoding
 I = receiver ID (added by RX FPGA)
 X = slice info invalid (added by RX FPGA)
 B = slice is bad due to SCAC initialization—should only be set for the first few slices of a run *
 T = trigger type: 0 if F is 0, else 1 – this is an anachronism from the System Integration Test Verilog
 P = trigger phase: 0 if F is 0, else trigger phase with respect to 20-MHz SCA write cycles
 F = first slice of trigger
 * The bits associated with INVALIDATE (I1/0, XI) will also be set during this time.

The error summary is detailed below. The slice's data should be considered invalid when the error summary is nonzero.

Bit:	11	10	9	8	7	6	5	4	3	2	1	0
STAT_ERR_SUM_H	rr	XP	XL	XI	I1	rr	rr	rr	L1	D1	F1	P1
STAT_ERR_SUM_L	rr	rr	SX	SC	I0	rr	rr	rr	L0	D0	F0	P0

Each two-character field in the table above is one bit:

XP	=	protocol error on either link, extended by ~5 μ s
XL	=	lock error (loss of lock) on either link, extended by ~20 μ s
XI	=	INVALIDATE error for either link, extended by ~80 μ s—see I1/0
I1/0	=	for each link: TX FPGA's INVALIDATE, e.g., when SCAC is not running or TX is sending fill frames
L1/0	=	for each link: lock error
D1/0	=	for each link: data error, i.e., DAV from ASMI does not match EXPECTED_DAV
F1/0	=	for each link: flag error, i.e., G-Link flag did not alternate on successive data words
P1/0	=	for each link: protocol error, i.e., violation of G-Link protocol
SX	=	serial slice error (due to Verilog bug or faulty wiring on the CTM), equal to the X bit in SCAC slice info
SC	=	SCAC fault, equal to the OR of the CCCC bits in SCAC slice info

XP, XL, and XI are summaries of other errors, extended in time in order to invalidate possibly corrupt data. E.g., data read from the SCA's will be corrupt for up to 80 microseconds after the TX FPGA stops sending fill frames.

All bits except SX and SC reflect errors that occurred at any time during writing of the slice's data to the buffer. SX and SC are evaluated when serial transmission of slice info should be complete, i.e., at the time the status words are written to the buffer.

The buffer readout lookup table determines readout order. For compatibility with existing DPU software, the status words are:

Status Words	Processing by DPU software
STAT_TRAILER STAT_HEADER	constant value checked by DPU software
STAT_TRAILER STAT_HEADER	ignored
STAT_TRAILER STAT_HEADER	ignored
STAT_TRAILER STAT_HEADER	ignored
STAT_TRAILER STAT_HEADER	ignored
STAT_SCAC_H STAT_SCAC_L	recorded in event output (?)
STAT_EVENT STAT_SLICE	checked for appropriate incrementing
STAT_ERR_SUM_H STAT_ERR_SUM_L	must be zero, else entire slice is invalid

SCA Controller Register Reference from CSC_TTC_and_BPI_6.doc

The table below documents registers of the SCA controller that was used in the System Integration Test. The CTM version of the SCAC is nearly identical to the SIT SCAC. LUT addressing is now direct, eliminating the need for bit L and bit G.

<i>for SCA controller:</i>				
TTCR_SCAC_Setup	R/W		<p>TTLWWRPS ssssssss</p> <p>Tx mode: 00 = drive Tx with SCA control 01 = drive Tx with test pattern 10 = drive Tx with simulated ASMI data 11 = reserved</p> <p>L = reset lookup table address register W = write rate: 00 = 20 MHz, CLK_20M toggled, 0 phase 01 = 20 MHz, CLK_20M toggled, 180 phase 10 = 20 MHz, CLK_20M held low 11 = 40 MHz, CLK_20M held low</p> <p>R = read clock rate: 0 = 5 MHz, 1 = 6.67 MHz P = read clock phase w.r.t. write addresses S = enable simultaneous read/write s = number of time slices to read out per trigger</p>	<p>SCA Controller setup: To reset the lookup table address register, TTCR_GrayLUT must be read while L is set—see >>> notes in sit_scac.v.</p> <p>See note below for Tx test modes.</p> <p>CLK_20M is a GLINK data line that can be selected via jumper on the ASMI to drive the SCA WCK. Phase of WCLK is with respect to write address changes.</p>
TTCR_ROSEQ_Setup	R/W		<p>rAAArDDD GrTTTTTT</p> <p>A = ADC clock phase w.r.t. RDCLK D = SD phase w.r.t. RDCLK G = LUT select – see TTCR_LUT T = additional delay (min = 2) for TRIG_DATA</p>	<p>Readout sequencer setup. SD is the SCA read address serial data. TRIG_DATA becomes DAV_N for ASMI transmitters.</p> <p>All phases are in RCLK periods (25 ns).</p>
TTCR_LatencyCells	R/W		<p>LLLLLLLL CCCCCC</p> <p>L = depth of latency pipeline in clock cycles C = number of SCA cells to use</p>	<p>depth of the latency pipeline, number of SCA cells to use</p>
TTCR_LUT	R/W		<p>rrrrrrrr vvvvvvvv</p>	<p>Data to/from lookup tables: gray code LUT (G == 1) readout sequencer LUT (G == 0)</p> <p>G is in TTCR_ROSEQ_Setup. The address is incremented after each access and reset by L bit in TTCR_SCAC_Setup.</p>

Temperature Sensors

The CTM has 5 LM83 temperature sensors, all on the same SMBUS. Each LM83 provides three remote and one local temperature measurements. The LM83 updates each measurement every 460/600 typ/max ms.

The MX FPGA contains control logic for writing and reading the sensors, though access to the SMBUS is via pins on the SL FPGA. Two values hard-coded in `ctm_sensor_readout.v` help protect the CTM in case of hardware or software bugs:

Name	Value	Used...
LM83_MAX_HIGH	105	to limit user-specified HIGH and CRIT register values to ≤ 105
LM83_OTF_THRESH	105	as a threshold for asserting OTF_DETECTED

The LM83 supervisor logic in the MX FPGA maps LM83 registers into a dual-port RAM that is accessible to the HPU. The DPRAM is organized as shown in the tables below. The supervisor overrides any user-specified setpoint that does not meet the LM83_MAX_HIGH test: User-specified HIGH and CRIT values will be saturated to LM83_MAX_HIGH. The comparison is unsigned, so negative values will be converted to LM83_MAX_HIGH as long as $\text{LM83_MAX_HIGH} < 128$.

The choice of 105 for LM83_MAX_HIGH allows for a very conservative 20 degree LM83 error. The maximum Virtex junction temperature is 125 C. A PN junction on each Virtex die is used to sense junction temperature.

The supervisor logic sets OTF_DETECTED while any Temperature register value equals zero or exceeds LM83_OTF_THRESH (signed comparison). From the LM83 datasheet: "If a D+ input is shorted to VCC or floating then the temperature reading will be +127 °C, and its OPEN bit in the Status Register will be set. If a D+ is shorted to GND or D-, its temperature reading will be 0 °C and its OPEN bit in the Status Register will not be set." Activation of either OTF_DETECTED or the SMBUS INT_N line for more than ~5s causes the SL FPGA to power down the CTM. See the Keep-Alive section.

Because the DPRAM initially contains 0x60 in every byte, the initial HIGH threshold is 96 C.

Temperature sensor summary:

Opto.TempSensor.Sensor (OPU1):

- 0 (D1): OPQ1 above OAU1 (lower opto RX module)
- 1 (D2): OPQ2 above OBU1 (upper opto RX module)
- 2 (D3): OPQ3 above OTU1 (opto TX module)
- 3 (local): OPU1 between opto TX module and upper opto RX module

Master.TempSensor.Sensor(MXU13):

- 0 (D1): MXU1 FPGA junction
- 1 (D2): MXQ4 near PWU54 (VDC3 switching regulator)
- 2 (D3): MXQ5 near PWU64 (VCCAUX/VCCO switching regulator)
- 3 (local): MXU13 near upper left corner of MX FPGA

Tx.TempSensor.Sensor (TXU1):

- 0 (D1): TX0U1 FPGA junction
- 1 (D2): TX1U1 FPGA junction
- 2 (D3): TX2U1 FPGA junction
- 3 (local): TXU1 near lower left corner of MX FPGA

RxA.TempSensor.Sensor (RAU1):

- 0 (D1): RA0U1 FPGA junction
- 1 (D2): RA1U1 FPGA junction
- 2 (D3): RA2U1 FPGA junction
- 3 (local): RAU1 near upper right corner of RA1U1

RxB.TempSensor.Sensor (RBU1):

- 0 (D1): RB0U1 FPGA junction
- 1 (D2): RB1U1 FPGA junction
- 2 (D3): RB2U1 FPGA junction
- 3 (local): RBU1 near upper right corner of RB0U1 (near MXU13)

Temperature Sensor DPRAM:

	HPU side	LM83 side
Overall size	128x16	256x 8
Initial contents	6060h	60h
LM83 register writes	LS half	LS half
LM83 register reads	MS half	MS half
Locations allocated per LM83:		
for writes to LM83	8 halfwords	16 bytes – 6 locations used
for reads from LM83	8 halfwords	16 bytes – 6+8 locations used
LM83 base address within half:		
Opto (OPU1)	00h	00h
Master (MXU13)	08h	10h
Tx (TXU1)	10h	20h
RxA(RAU1)	18h	30h
RxB(RBU1)	20h	40h
unused	28h	50h
unused	30h	60h
unused	38h	70h

A complete set of register accesses is performed every ~120 ms.

Register list from the LM83 data sheet (sorted, reserved locations omitted):

Address in LM83	Name	Function	DPRAM byte index	Notes
50h	WD1RHS	Write D1 Remote HIGH Setpoint	0	Verilog saturates to LM83_MAX_HIGH.
0Dh	WD2RHS	Write D2 Remote HIGH Setpoint	1	“
52h	WD3RHS	Write D3 Remote HIGH Setpoint	2	“
0Bh	WD2LHS	Write Local HIGH Setpoint	3	“
5Ah	WTCS	Write T_CRIT Setpoint	4	“ – typically not used
09h	WC	Write Configuration	5	Verilog ANDs value with 2 → no masks allowed. *
38h	RD1RHS	Read D1 Remote HIGH Setpoint	0	
07h	RD2RHS	Read D2 Remote HIGH Setpoint	1	
3Ah	RD3RHS	Read D3 Remote HIGH Setpoint	2	
05h	RLHS	Read Local HIGH Setpoint	3	
42h	RTCS	Read T_CRIT Setpoint	4	
03h	RC	Read Configuration	5	
30h	RD1RT	Read D1 Remote Temperature	6	
01h	RD2RT	Read D2 Remote Temperature	7	
31h	RD3RT	Read D3 Remote Temperature	8	
00h	RLT	Read Local Temperature	9	
02h	RSR1	Read Status Register 1	10	
35h	RSR2	Read Status Register 2	11	
FEh	RMID	Read Manufacturers ID	12	
FFh	RSR	Read Stepping or Die Revision	13	

* Writing 2 to the WC register inverts INT_N. This can be used to force an INT_N, e.g., for testing the SMBUS, CTM logic, etc.

If the Verilog detects an error during an LM83 register read, it substitutes 0xc0 (-64) for the register's contents. No LM83 register, with the possible exception of the die revision/stepping register (RSR) should ever contain this value.

Loss of Lock

Loss of Lock (LOL):

According to radiation studies, the duration of SEU-induced loss of lock is ~300/600 microseconds, typical/maximum. Only the HDMP-1022 transmitter was considered in these studies. It is also possible for SEU's to induce loss of lock in the ASMII's HDMP-1024 receiver.

The CTM does not use G-Link components. Instead, it interprets the G-Link protocol in FPGA logic. In addition to requiring correct G-Link protocol ("protocol lock"), the RX FPGA requires correctly aligned data words ("word lock"). On each 40 MHz clock cycle, signals from TX FPGA to RX FPGA indicate whether links should receive a data word or a fill word. The RX FPGA can change the length of its word pipelines (0-7 words) in order to achieve correct word alignment for each link.

The total RX FPGA lock time is less than 30 microseconds.

Establishing/Reestablishing Lock:

The deserializers in the RX FPGA's have a local reference clock that is identical to the G-Link bit rate, so they do not need special frames to establish frequency lock. The G-Link receiver on the ASMII has no local reference clock. When it loses lock it is dependent on fill frames to first establish frequency lock before it can achieve final phase lock.

RX links establish/reestablish protocol lock automatically whenever a valid RX stream is present. TX links are more complicated. An eight-bit map in each serializer's PI_SER_Control register associates RX links with the serializer's TX link. If all (both) RX links listed in the map experience loss of protocol lock for 1 ms (see FDELAY_BEG in hpu_ctm.h), the serializer assumes that the multiple RX LOL is due to the ASMII's G-Link receiver losing lock with the serializer's output stream. To regain lock, the serializer sends fill frames rather than the normal SCA control stream. The serializer continues to send fill frames for 5 ms (see FDELAY_END in hpu_ctm.h) after one or more of the associated RX links regain protocol lock. Once transmission of the normal SCA control stream resumes, the RX links can regain word lock.

Excessive Loss of Lock (EXLOL):

The RX FPGA's determine excessive loss of lock on a per-link basis. These constants defined in ctm_rx.v are applicable to each link:

EXLOL_DURATION_MS	=	10	LOL is excessive after this many milliseconds of no lock, or
EXLOL_MAX_INCIDENTS	=	8	LOL is excessive after this many incidents of no lock, but
EXLOL_FORGIVE_MS	=	25	forgive one LOL incident after this many milliseconds

EXLOL typically indicates a hardware malfunction. Persistent EXLOL on several links can cause ATLAS DCS to power down a chamber. Persistent EXLOL can cause the CTM to power down its laser supply, VCCOPTO. See the Keep Alive Details section.

An RX ribbon is considered to be experiencing EXLOL when three or more of its links have EXLOL. This total excludes those links for which HPU software has permanently disabled EXLOL, e.g., the two unused links per RX ribbon. See next section.

Overriding EXLOL:

CTM hardware provides limited means of overriding the EXLOL mechanisms. (When a link's EXLOL detection is overridden, the link will not contribute to the ANY_LOL signal, which is used only to flash the RX FPGA's LED upon loss of lock. Overriding EXLOL detection has no effect on a link's LS_STATE.)

For normal operation, HPU software may initialize the CTM such that EXLOL detection is permanently disabled for at most two links per RX FPGA. By default, the CTM driver permanently disables EXLOL for links associated with the two unused fibers per RX ribbon.

For lab/debug Operation, HPU software may temporarily disable EXLOL for all links in any combination of RX FPGA's. For example, this could be used to prevent the CTM from powering itself down during TX/RX loopback tests in which one RX optoelectronics module has no input. The maximum override interval is 12.6 seconds per call to the OverrideTemporary() driver function. As a safety precaution, the body of this function is to be removed from ATLAS production software.

Laser Safety Mechanisms

All CSC lasers are either Class 1 or Class 1M, and are safe under intended operating conditions, but might not be safe if a laser module, fiber, or fiber ribbon is viewed at close range or with optical instruments. The HFBR-772/782 data sheet states:

Eye Safety: These 850 nm VCSEL-based modules provide eye safety by design. The HFBR-772B has been registered with CDRH and certified by TUV as a Class 1M device under Amendment 2 of IEC 60825-1. See the Regulatory Compliance Table for further detail. If Class 1M exposure is possible, a safety-warning label should be placed on the product stating the following:

LASER RADIATION
DO NOT VIEW DIRECTLY WITH
OPTICAL INSTRUMENTS
CLASS 1M LASER PRODUCT

CSC Laser Safety Features:

Feature	Description
ROD rack door interlock	The CTM disables optoelectronic power if the two rack rear door switches do not return correct signals. This will lead to chamber power-down via ATLAS DCS.
Excessive loss-of-lock logic (EXLOL)	The CTM monitors RX links for excessive loss of lock. Depending on the severity of the condition, CTM optoelectronic power may be disabled and/or one or both chambers may be powered down via ATLAS DCS. See table below.
Warning lights	At the CTM's rear panel, red LED's surround the parallel laser module. The LED's are lit whenever CTM optoelectronic power is up.
Keep-alives	The CTM's CPLD will power down one or more of the CTM supply voltages if the keep-alive signals from the SL and MX FPGA's are not present. See Keep-Alive section below.
Laser power control	Logic on the CTM can power down the CTM's parallel laser module. ASMII lasers can be powered down only by powering down their chamber.
Labeling	See data sheet excerpt above.

EXLOL Summary (see Loss of Lock section above):

Condition	Possible Causes	Reaction
Only one RX ribbon shows EXLOL.	1. The RX ribbon is disconnected or severed. 2. The chamber is not powered up.	The chamber is powered down via ATLAS DCS. CTM lasers remain enabled.
Both RX ribbons show EXLOL.	1. Both RX ribbons are disconnected or severed. 2. Both chambers are not powered up. 3. The TX ribbon is disconnected or severed. 4. The CTM's optoelectronic power is disabled.	CTM optoelectronic power is disabled (except for brief retries—see Keep-Alive section below). If the condition persists, the CTM powers itself down and both chambers are powered down via ATLAS DCS.

Chamber Power-Down Summary

Condition	Possible Causes	Reaction (via ATLAS DCS) *
The ROD is not responding.	1. The ROD has not completed the startup sequence. 2. The ROD has malfunctioned. 3. Etc.	Both of the ROD's chambers are powered down.
The CTM is not powered.	1. The CTM has not completed the startup sequence. 2. The CTM powered itself down due to some fault.	Both of the CTM's chambers are powered down.
The ROD rack door interlock failed.	1. The ROD rack rear door was opened. 2. A rear door switch malfunctioned. 3. Etc.	All of the rack's chambers are powered down.
Only one RX ribbon shows EXLOL	See EXLOL Summary above.	The chamber associated with the RX ribbon is powered down.

* In all cases DCS starts the power-down as quickly as it can, i.e., any appropriate delays are implemented in non-DCS software or hardware. Also in all cases, DCS powers up chambers as soon as the condition has cleared.

Continued on next page.

Laser Safety Mechanisms (continued)

Because ASMII's never disable their lasers and because only DCS can power down chambers, ASMII laser safety is dependent upon several software components, including:

- ROD software
- RCC software
- DCS software

During system initialization, chambers must be powered up while the CTM attempts to acquire initial lock.

The ROD/CTM/Chamber power-up sequence should be something like:

- chambers are initially off
- ROD j may be powered up at any time, but its CTM is initially not powered
- ROD j performs its phase I initialization immediately following power up (or HPU code download)
- DCS powers up the two chambers associated with ROD j
- ROD j is instructed to perform phase II initialization
- DCS powers down the two chambers associated with ROD j if initialization failed

Keep-Alive Details

The CTM's CPLD receives four keep-alive signals:

CPLD action when keep-alive is absent	Keep-Alives from SL FPGA		Keep-Alives from MX FPGA	
	ALIVE_SL	ALIVE_SO	ALIVE_MX	ALIVE_MO
Power down CTM:	except within one second of power-up	no action	(only via SL FPGA)	no action
Power down VCCOPTO:	immediately	immediately	immediately	immediately

If ALIVE_SL is absent, the CPLD powers down the CTM, except during a one-second interval following CTM power-up. This interval allows time for the ROD to configure the SL FPGA. When powered down, the CTM will begin a power-up sequence only upon command from the ROD. CTM power nets VCC5 and VCC3_3, which power the CPLD and related logic, are enabled even during power-down.

The SL FPGA disables ALIVE_SL and ALIVE_SO only after certain fault conditions have persisted. Details are in the table below. The SL FPGA never automatically re-enables a disabled keep-alive—see *D note on next page.

SL status register bit	ALERT	SO	DCONF	SL	PWRDWN	Example Causes
	?	?	?	?	?	
15 ~ALIVE_MX	Y	5	5	9	9	MX FPGA not configured or it detected a fatal fault.
14 FAULT_OPIO	Y	5	5	9	9	MX FPGA not configured, short or open on PCB.
13 FAULT_OVERTEMP_SL	Y	5	5	9	9	Pulse seen on temp sensor bus INT_N, i.e., overtemperature.
12 MX_CMDX2	Y	5	5	9	9	See MX_CMDX table.
11 MX_CMDX1	Y	5	5	9	9	See MX_CMDX table.
10 MX_CMDX0	Y	5	5	9	9	See MX_CMDX table.
9 MX_INTERLOCKED_N	Y	5	N	N	N	Rack rear door opened.
8 MX_EXLOL_RB	Y	5B	N	N	N	Ribbon unplugged, chamber powered down.
7 MX_EXLOL_RA	Y	5B	N	N	N	Ribbon unplugged, chamber powered down.
6 ~ALIVE_MO	Y	5	N	N	N	MX FPGA not configured or it detected a fault.
5 FAULT_LASER	Y	5	N	N	N	Laser module failed.
4 ~VCCOPTO_OK	N	N	N	N	N	Some condition disabled VCCOPTO.
3 SLINK_DOWN	N	N	N	N	N	SLINK reports link down.
2 FAULT_SLINK_LOSS	N	N	N	N	N	Driver or Verilog bug, SLINK in wrong mode.
1 FAULT_SLINK_FAIL	N	N	N	N	N	Driver or Verilog bug, SLINK in wrong mode.
0 FAULT_ROFIFO_OVF	N	N	N	N	N	Verilog bug, ROD or CTM PCB defect.

ALERT?	=	Will the estatALERT bit in the HPU EMIF FPGA be set as soon as the condition occurs?
SO/SL?	=	Will ALIVE_SO/SL be disabled after 5-6/9-10 seconds?
DCONF?	=	Will the TX/RA/RB FPGA's be deconfigured after 5-6 seconds?
PWRDWN?	=	Will the CTM be powered down after 9-10 seconds?
Y/N	=	Yes/no.
5B	=	Both EXLOL_RB and EXLOL_RA must be active before action is taken.
5/5B	=	The condition must persist 5-6s before KEEP_ALIVE_SO or DECONFIG is affected.
9	=	The condition must persist 9-10s before the SL FPGA disables ALIVE_SL.

MX_CMDX	Meaning
0	MX FPGA reports no fatal conditions. For the MX FPGA, "fatal fault" = nonzero MX_CMDX.
1	MX FPGA detected CMDX_WR fault.
2	MX FPGA detected CMDX_ADDR fault.
3	MX FPGA detected CMDX_AUTO fault (timeout of serial info request from SL FPGA).
4	MX FPGA detected OPIO fault.
5	MX FPGA detected overtemperature fault.
6	MX FPGA detected slave (RX or TX FPGA) fault.
7	All MX_... fields in the SL status register are invalid.

All SL status register bits whose names begin with MX_ are from the MX FPGA via the SER_INFO serial link.

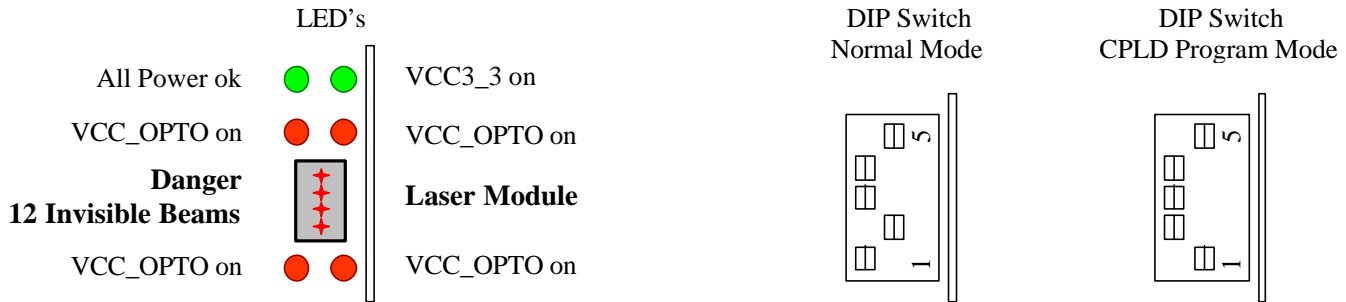
The MX FPGA disables ALIVE_MX and ALIVE_MO as soon as certain fault conditions are detected. Details are in the table below. The MX FPGA re-enables disabled keep-alives as soon as the fault condition clears. Some fault conditions, e.g., FAULT_CMDX_WR and FAULT_CMDX_ADDR, are cleared only by an MX FPGA reset.

MX internal signal	MO ?	MX ?	Example Causes
MX_CMDX (FATAL_FAULT): 1 = FAULT_CMDX_WR 2 = FAULT_CMDX_ADDR 3 = SERINFO_AUTO 4 = ~OPIO_OK 5 = OTF_DETECTED 6 = SLAVE_FAULTS	Y	Y	Defective PCB, MX FPGA detected overtemperature, misconfigured RX or TX FPGA.
MX_DIS_LASERS *D	Y	N	Excessive loss of lock on both ribbons.
~INTERLOCKED	Y	N	Either rack rear door switch open.

MO/MX? = Will ALIVE_MO/MX be disabled immediately?
 Y/N = Yes/no.
 | = OR of all bits in signal

*D = This condition is subject to a ~150 ms retry interval starting as soon as EXLOL is detected and recurring after ~4 seconds. During the retry interval, the MX FPGA will output ALIVE_MO (unless another condition in the MX FPGA disables ALIVE_MO). Because the SL FPGA will deactivate ALIVE_SO after ~5 seconds of EXLOL, there are at most two retry intervals per EXLOL. Once ALIVE_SO is deactivated, the SL FPGA will not automatically reactivate it. HPU software may reactivate ALIVE_SO by clearing then setting the RETRY_VCCOPTO field of the SL Control register. For reasons of laser safety, the RETRY_VCCOPTO feature should only be used if absolutely necessary, preferably upon command from DCS.

CTM Rear Panel LED's and DIP Switch [see standard ESD precautions section]



All Power ok: This LED flashes as each supply is powered on. It is steadily lit once all supplies except VCC_OPTO are at valid levels.

VCC3_3: This LED is connected to VCC3_3, which is only enabled when both backplane voltages (5V and 3.3V) are present and the ejector switch is closed (or a jumper is installed at PWH1 or PWH2).

VCC_OPTO on: These LED's are all connected to VCC_OPTO. When they are lit the laser module is powered (though it may be disabled). See the Laser Safety section.

RearPanel.DIP_Switch (RPSW1):

Switch	net	Meaning when ON (0) (near PCB)	Meaning when OFF (1) (away from PCB)
5	GND_INTLK	ground net GND_INTLK	do not ground net GND_INTLK
4	RP_DIPSW1	TBD	TBD
3	RP_DIPSW0	TBD	TBD
2	JTAG_MUX_SEL1	see table	see table
1	JTAG_MUX_SEL0	see table	see table

GND_INTLK is connected to several pins of the interlock header (see below). Closing this switch on all CTM's connected to the same interlock cable is recommended, but will create a ground loop. This loop is likely harmless. Closing the GND_INTLK switch on only on CTM per interlock cable eliminates the ground loop. Grounding GND_INTLK provides a small extra margin of safety against the interlock mechanism being inadvertently defeated, but the mechanism should still function with GND_INTLK floating.

RP_DIPSW0:1 are connected to the the CTM's power CPLD, Power.PowerPLD.Cpld (PWU1). The Verilog for the CPLD does not currently use the signals except to forward them to the SL FPGA, which does not currently use them.

JTAG_MUX_SEL1:0 set JTAG and power options:

JTAG_MUX_SEL1:0	Mode	Comments
0	Power Disabled	All power is disabled except VCC5 and VCC3_3 if the CPLD is properly programmed. This mode is not generally used.
1	Normal	Normal operating mode, FPGA's can be configured only in this mode
2	CPLD Program	Use this mode only when programming the CTM's CPLD.
3	Power Force	VCCINT, VDC3, VCCAUX/VCCO, VCCSLINK are forced on to allow FPGA JTAG. Use this mode only if/when performing boundary scan testing of the CTM.

Rear Panel Connectors and Headers [see standard ESD precautions section]

The standard layout for IDC connectors, when viewed from the component side of the CTM, is:

1	2
3	4
5	6
7	8
9	10
...	...

pin 1 is marked with an arrow on the CTM's connector

additional rows, if any

RearPanel.InterlockHeader (RPH1):

This header is the connection for the rear door interlock cable. Up to 32 CTM's may be connected in parallel to a single cable. The interlock mechanism is designed to shut off CTM lasers when the correct signals are not seen on the interlock cable, e.g., when one or both of the rack rear door switches is open. Additional ESD protection circuitry is incorporated into the interlock circuitry.

The header pinout is designed to minimize inadvertent defeating of the interlock mechanism, e.g., by a shorted cable.

header pin	net	note
IN[1] (RPH1.7)	INTERLOCK_IN1__1	One rack door switch connects IN[1] to OUT[1] when door is closed.
IN[0] (RPH1.3)	INTERLOCK_IN0__1	One rack door switch connects IN[0] to OUT[0] when door is closed.
OUT[1] (RPH1.4)	INTERLOCK_OUT1	
OUT[0] (RPH1.9)	INTERLOCK_OUT0	
NC (RPH1.1 5)	/NC	no connection on CTM
GND_JMP (RPH1.2 6 8 10)	GND_INTLK	optional ground connection—see GND_INTLK note, above

RearPanel.TXIO_Header (RPH2):

This header provides general-purpose I/O to/from the TX FPGA's. The TX FPGA Verilog currently uses these I/O's for test/debug only. Two input bits are bussed to all three TX FPGA's. Two output bits are dedicated to each TX FPGA.

The buffer chip is a SN74ABTE16245 powered by VCC5. Output levels are TTL. Input levels are improved TTL. Key parameters are:

Voltage range applied to any output in the high state or power-off state:	-0.5 to 5.5 V	absolute max
Input clamp current (VI < 0):	-18 mA	absolute max
High level input voltage	1.6 V	min
Low level input voltage	1.4 V	max
High level output current	-60 mA	max
Low level output current	90 mA	max

RPH2 connections from the CTM symbolic netlist are:

header pin	net	buffer pin	110 ohm pullup/terminator to VCC3_3
IN[1] (RPH2.3)	RP_TX_IN1	RearPanel.Buf_TXIO.B[3] (RPU4.6)	RearPanel.PullupTXIN1.B (RPR20.2)
IN[0] (RPH2.1)	RP_TX_IN0	RearPanel.Buf_TXIO.B[1] (RPU4.3)	RearPanel.PullupTXIN0.B (RPR19.2)
OUT0_[1] (RPH2.7)	RP_TX_OUT0_1	RearPanel.Buf_TXIO.A[7] (RPU4.37)	
OUT0_[0] (RPH2.5)	RP_TX_OUT0_0	RearPanel.Buf_TXIO.A[5] (RPU4.40)	
OUT1_[1] (RPH2.11)	RP_TX_OUT1_1	RearPanel.Buf_TXIO.A[11] (RPU4.32)	
OUT1_[0] (RPH2.9)	RP_TX_OUT1_0	RearPanel.Buf_TXIO.A[9] (RPU4.35)	
OUT2_[1] (RPH2.15)	RP_TX_OUT2_1	RearPanel.Buf_TXIO.A[15] (RPU4.26)	
OUT2_[0] (RPH2.13)	RP_TX_OUT2_0	RearPanel.Buf_TXIO.A[13] (RPU4.29)	
GND (RPH2.2 4 6 8 10 12 14 16)	GND		

Power.LatchSwitch (PWH1) and Power.LatchOverride (PWH2):

PWH1 and PWH2 are connected in parallel. Their function is similar to that of the ROD's PWH2 and PWH3 headers. They control the power switches that connect backplane supply voltages to the remainder of the CTM. Both of the following are required for the switches to be enabled:

Both backplane supply voltages (5.0V and 3.3V) must be above minimum required thresholds.

PWH1 and/or PWH2 must have a short between their two pins.

A switch in one of the CTM's ejectors should be connected to either PWH1 or PWH2. When the CTM is fully installed in the crate, the switch should be closed.

Standard ESD Precautions

Electrostatic discharge can damage the CTM.

Failure of CTM's may occur long after damaging ESD incidents.

An antistatic wrist strap should be worn in all of these cases:

1. whenever the CTM is handled,
2. whenever the CTM's rear panel is accessed, e.g.,
 - a. when installing or removing a cable or fiber ribbon
 - b. when changing a DIP switch setting
3. while a CTM is being packed,
4. if practical, while a CTM is being transported short distances by foot,
5. whenever the CTM is probed, e.g., with an oscilloscope or voltmeter

When not installed in a crate, CTM's should be stored in an antistatic bag.

Antistatic packing material should be used when shipping CTM's.

When being transported, either by vehicle or by foot, a CTM should be in an antistatic bag AND in a cardboard box. Only the cardboard box should be handled during transportation.

The following are poor, but sometimes unavoidable, substitutes for an antistatic strap:

Substitute	Example
Touching a surface at the same potential as the CTM before handling the CTM.	Touching a workbench before handling a CTM on the workbench.
Touching a surface at the new potential for the CTM before unpacking the CTM.	Touching a workbench before unpacking a CTM on the workbench.