

Accelerating Verification of CXL based Designs

Prashant Dixit
Siemens EDA

Agenda

Functional Verification Solution

Verification IP

VIP Use Models

Host/Device BFM

Verification Plan and Stimuli

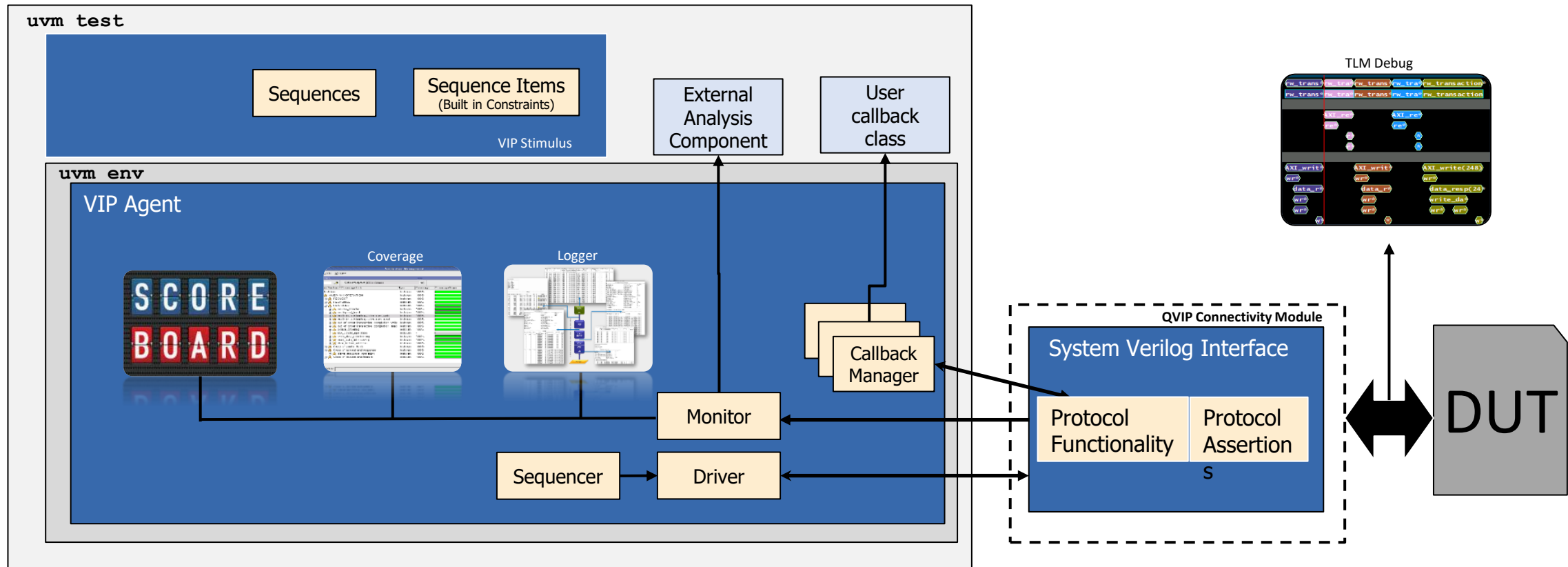
Protocol Director

Debugging Methods

Protocol Checks and Coverage

Summary

Functional Verification Solution



Verification IP

- ☐ Host, Device and Monitor BFM
- ☐ Verification Plan with Stimulus
 - ☐ Compliance Test Suite
 - ☐ SIG compliance
- ☐ Protocol Director
- ☐ Debug : Text and GUI Based Loggers
- ☐ Protocol Checks
- ☐ Scoreboard
- ☐ Coverage

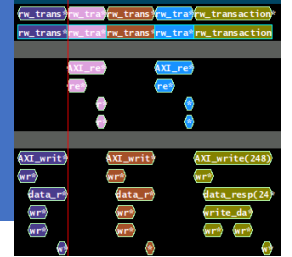
Verification IP

Architected for ease of use

- Standards based SV UVM support
- Configuration & TB generation GUI

Intuitive Debug

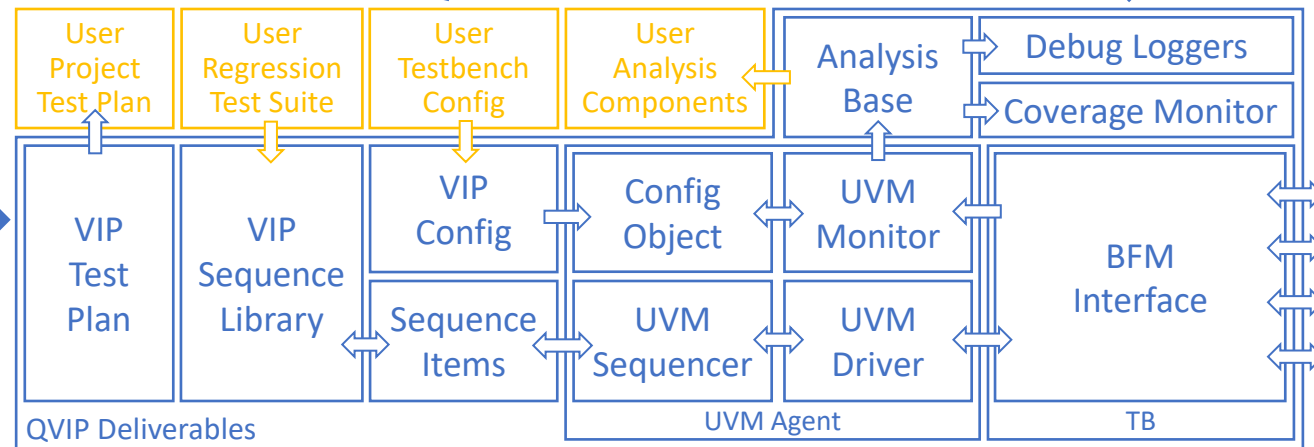
- Transaction viewing
- Scoreboard comparison
- Tracker files



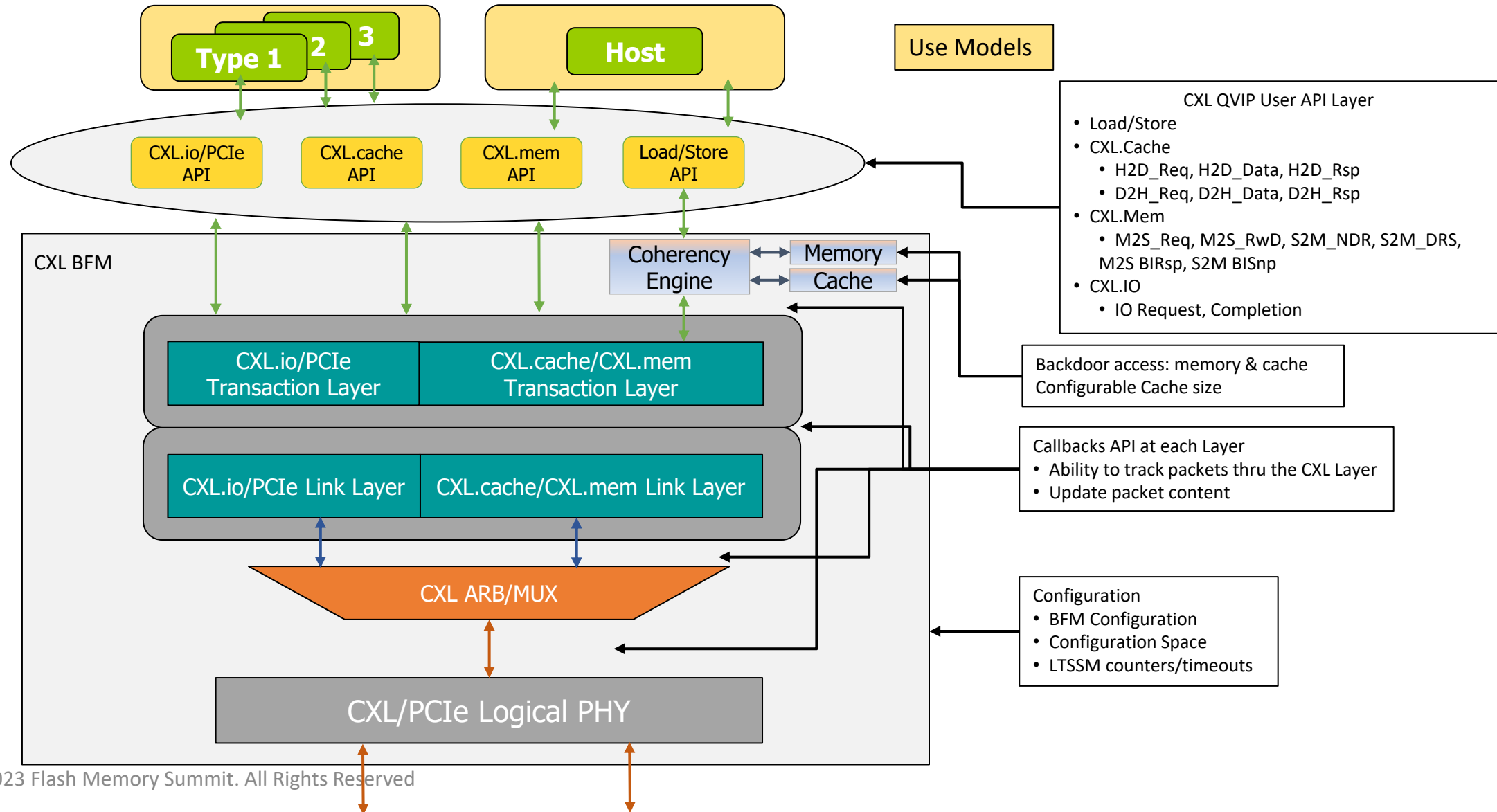
Comprehensive validation

- Test suites
- Protocol checks
- Functional coverage

#	Section	Description
1	AMBA AXI OPERATION	
1.1	REQUEST	
1.1.1	Operation	This covers the basic AMBA AXI operation types read and write operations
1.1.2	burst_length	This covers no. of data transfer that occurs in each burst. This includes length from 1 to 16. For wrapping bursts, length can be 2, 4, 8, 16
1.1.3	transfer_size_write	This covers maximum no. of data bytes to transfer data transfer. This includes 1, 2, 4, 8, 16, 32, 64
1.1.4	transfer_size_read	This covers maximum no. of data bytes to transfer data transfer. This includes 1, 2, 4, 8, 16, 32, 64
1.1.5	burst_type	This covers different type of burst to transfer data. This includes 1, 2, 4, 8, 16, 32, 64
1.3.9	write_data_interleaving	This covers write data sent from master to slave transaction (having different IC) are interleaved
1.3.10	read_data_interleaving	This covers read data sent from slave to master transaction (having different IC) are interleaved
1.3.11	data_before_address	This covers whether a new write data phase has before the address phase. This includes yes and no



QVIP Block Diagram



QVIP Use Models

- ❑ Intelligent modeling allows the QVIP to mimic either host or device (Type 1/2/3) behavior based on DUT type at the other end of the CXL bus
- ❑ Monitor BFM can be plugged into existing env as passive component
 - ❑ monitors bus traffic and provides various verification capabilities like protocol checking, coverage and logging

Host/Device BFM

☐ Load/Store APIs

- ☐ Abstraction for providing high-level API
- ☐ breaks down into lower-level traffic of D2H requests depending upon the cache line states
- ☐ Follows coherency protocol and biasing rules

☐ Automatic responder

- ☐ DCOH engine/CA responds to the CXL.cache/mem requests automatically
- ☐ Provides appropriate responses based on protocol

☐ Built-in Cache Model and Snoop Filter

- ☐ Highly configurable cache model with options to set line replacement algorithms, sets, ways, line-width etc.
- ☐ Automated checks at the cache level to verify protocol compliance

Abstractions to run stimuli

- ☐ UVM-compliant
- ☐ Abstractions
 - ☐ Load/Store APIs
 - ☐ Sequences
 - ☐ Exhaustive set of ready to use sequences
 - ☐ Sequence Items
 - ☐ For Cache/mem req, data, response
 - ☐ For IO request/completion
 - ☐ Complete control of every field, can be randomized also.

Verification Plan

CXL CTS - Advanced	TL	B-2 Table 313	Type 3 Memory Request	Host illegally sends Metafield as Metastate f	mem_rd_data_illegal_metafield	icvip_cxl_mem_rd_data_illegal_metafield_sequence		
CXL CTS - Advanced	TL	B-4 Table 315	Type 3 Memory RwD	The Host wants to update memory. Host se	mem_wr_with_metavalue	icvip_cxl_mem_wr_with_metavalue_sequence		
CXL CTS - Advanced	TL	B-4 Table 315	Type 3 Memory RwD	The Host wants to update memory. Host se	mem_wr_ptl_with_metavalue	icvip_cxl_mem_wr_ptl_with_metavalue_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host wants to update memory and kee	mem_wr_with_exclusive_cachel	icvip_cxl_mem_wr_with_exclusive_cacheline_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host wants to update memory and kee	mem_wr_with_shared_cacheline	icvip_cxl_mem_wr_with_shared_cacheline_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host wants to update memory and will	mem_wr_with_invalid_cacheline	icvip_cxl_mem_wr_with_invalid_cacheline_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host wants to write the line back to me	mem_wr_with_line_write_back	icvip_cxl_mem_wr_with_line_write_back_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host wants to update memory and kee	mem_wr_ptl_with_exclusive_cac	icvip_cxl_mem_wr_ptl_with_exclusive_cacheline_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host wants to update memory and kee	mem_wr_ptl_with_shared_cach	icvip_cxl_mem_wr_ptl_with_shared_cacheline_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The host wants to update memory and will	mem_wr_ptl_with_invalid_cache	icvip_cxl_mem_wr_ptl_with_invalid_cacheline_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host wants to write the line back to me	mem_wr_ptl_with_line_write_ba	icvip_cxl_mem_wr_ptl_with_line_write_back_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host Send illegal Snoop Type for S-state	mem_rwd_illegal_snp_type_with	icvip_cxl_mem_rwd_illegal_snp_type_with_meta_shared_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host Send illegal Snoop Type as Snoop t	mem_rwd_illegal_snp_data_with	icvip_cxl_mem_rwd_illegal_snp_data_with_meta_invalid_sequence		
CXL CTS - Advanced	TL	B-3 Table 314	Type 2 Memory RwD	The Host Send illegal Snoop Type as Snoop t	mem_rwd_illegal_snp_current_v	icvip_cxl_mem_rwd_illegal_snp_current_with_meta_invalid_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants an exclusive copy of the line	mem_rd_with_exclusive_cacheli	icvip_cxl_mem_rd_with_exclusive_cacheline_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host requesting a shared copy of the lin	mem_rd_with_shared_cacheline	icvip_cxl_mem_rd_with_shared_cacheline_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host requesting a non-cacheable but cu	mem_rd_with_current_cacheline	icvip_cxl_mem_rd_with_current_cacheline_and_flush_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host requesting a non-cacheable but cu	mem_rd_with_current_cacheline	icvip_cxl_mem_rd_with_current_cacheline_no_flush_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants to read line without changir	mem_rd_with_invalid_cacheline	icvip_cxl_mem_rd_with_invalid_cacheline_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants a current value of the line v	mem_rd_with_current_cacheline	icvip_cxl_mem_rd_with_current_cacheline_no_state_change_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants a cacheable copy in either e	mem_rd_data_with_shared_or_e	icvip_cxl_mem_rd_data_with_shared_or_exclusive_cacheline_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants ownership of the line witho	mem_inv_ownership_with_no_d	icvip_cxl_mem_inv_ownership_with_no_data_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants the device to degrade to S i	mem_inv_shared_cacheline_wit	icvip_cxl_mem_inv_shared_cacheline_with_no_data_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants the device to invalidate the	mem_inv_invalid_cacheline_no	icvip_cxl_mem_inv_invalid_cacheline_no_metastate_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants the device to invalidate the	mem_inv_invalid_cacheline_wit	icvip_cxl_mem_inv_invalid_cacheline_with_metastate_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants ownership of the line witho	mem_inv_nt_ownership_with_n	icvip_cxl_mem_inv_nt_ownership_with_no_data_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants the device to degrade to S i	mem_inv_nt_shared_cacheline	icvip_cxl_mem_inv_nt_shared_cacheline_with_no_data_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants the device to invalidate the	mem_inv_nt_invalid_cacheline_r	icvip_cxl_mem_inv_nt_invalid_cacheline_no_metastate_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	The Host wants the device to invalidate the	mem_inv_nt_invalid_cacheline_v	icvip_cxl_mem_inv_nt_invalid_cacheline_with_metastate_sequence		
CXL CTS - Advanced	TL	B-1 Table 311	Type 2 Memory Request	Host illegally sends Snoop_type(SnpInv, SnpC	mem_rd_data_illegal_snp_type	icvip_cxl_mem_rd_data_illegal_snp_type_sequence		
CXL CTS - Advanced	TL	B-4 Table 315 and	Type 3 And Type2 Memory R	The Host Send illegal Metafield	mem_rwd_illegal_metafield	icvip_cxl_mem_rwd_illegal_metafield_sequence		
CXL CTS - Advanced	TL	B-4 Table 315 and	Type 3 And Type2 Memory R	The Host Send illegal Snoop Type for A-state	mem_rwd_illegal_snp_type	icvip_cxl_mem_rwd_illegal_snp_type_sequence		
CXL CTS - Advanced	TL	8.1.3.8.4	DVSEC CXL Range registers	The Host initiates Memory read command c	mem_rd_non_hdm	icvip_cxl_mem_rd_non_hdm_sequence		

Stimuli

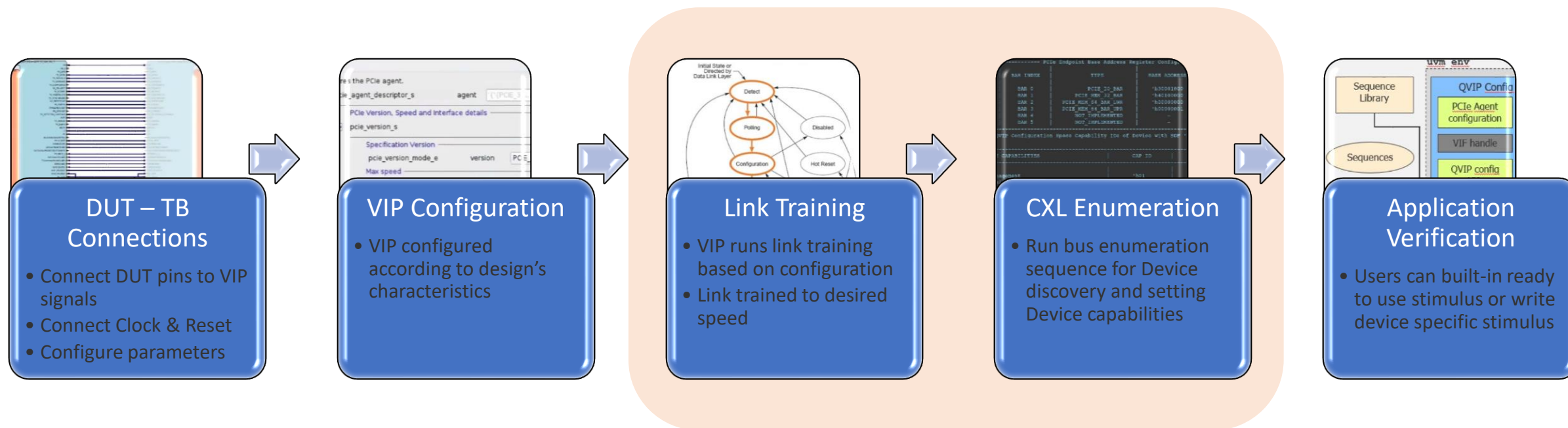
☐ Compliance Test Suite (~1000)

- ☐ Exhaustive and ready to use test suite
- ☐ For CXL.io, CXL.cache and CXL.mem
- ☐ Covers all layers : TL, LL, Arbiter and Flex bus
- ☐ Power Management
- ☐ CXL.cache/mem request and response combinations for HDM and main memory
- ☐ Error injection scenarios for various errors with poison and viral
- ☐ Cacheability restrictions/Buried cache scenarios
- ☐ Resets – CXL, FLR, Hot, Cold
- ☐ Security – IDE, TDISP, SPDM/CMA, DOE, KM
- ☐ GUI based tool (Protocol Director) to enable feature/layer wise selection and run stimuli

Stimuli

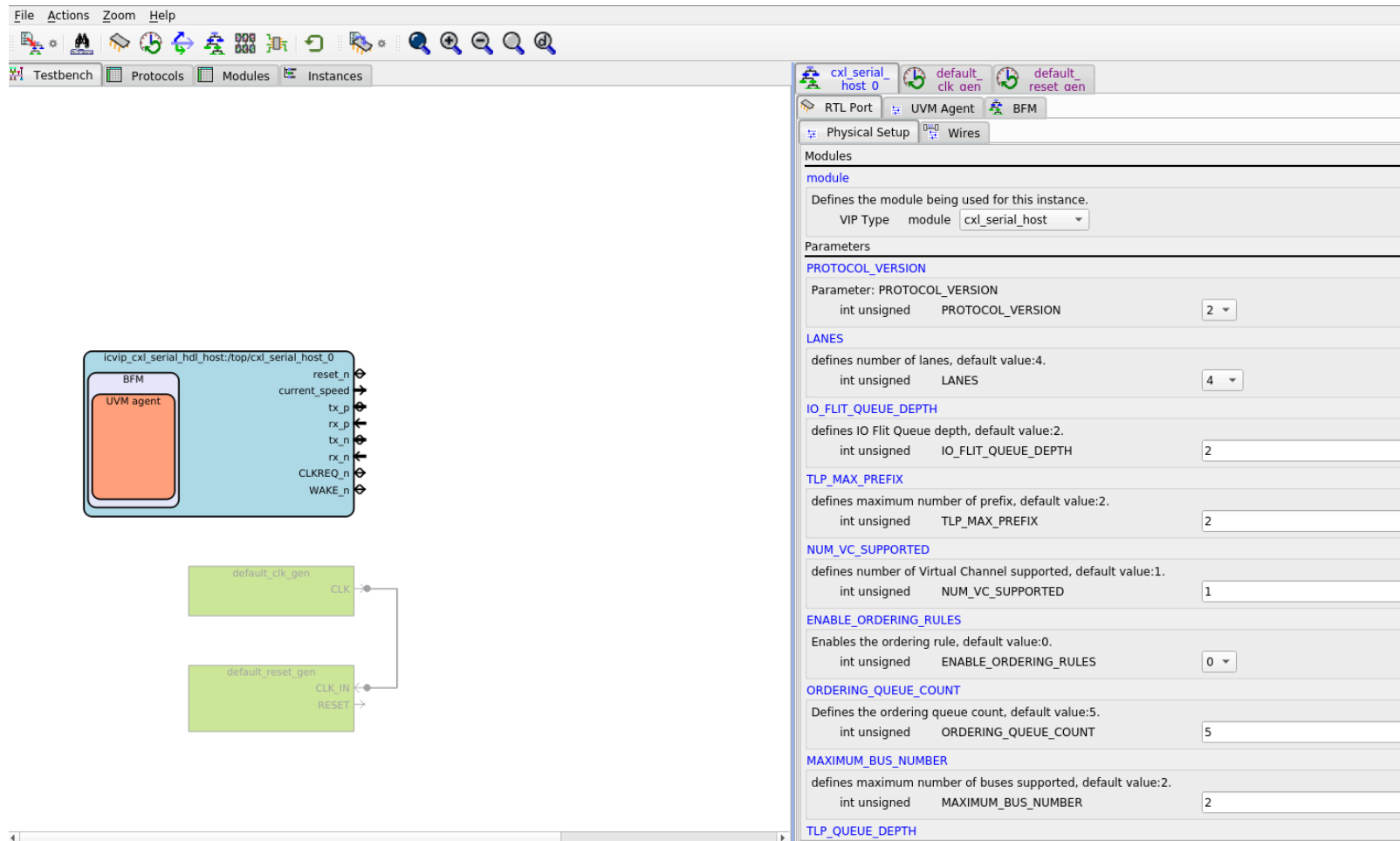
- ❑ CXL SIG compliance suite (~200)
 - ❑ As per CXL specs
 - ❑ Various layers, arbiter, Power management
 - ❑ Configuration Registers, RAS
 - ❑ Security
 - ❑ Memory Mapped registers
 - ❑ Reset and Initialization

Workflow



Getting Started
possible in hours

Automatic TB generation (Configurator)



The screenshot displays the Automatic TB generation (Configurator) interface. The left pane shows a testbench diagram with a module named `icvip_cxl_serial_hdl_host/top/cxl_serial_host_0` containing a `BFM` and a `UVM agent`. The right pane shows the configuration for the `cxl_serial_host_0` module, including parameters like `PROTOCOL_VERSION`, `LANES`, `IO_FLIT_QUEUE_DEPTH`, `TLP_MAX_PREFIX`, `NUM_VC_SUPPORTED`, `ENABLE_ORDERING_RULES`, `ORDERING_QUEUE_COUNT`, `MAXIMUM_BUS_NUMBER`, and `TLP_QUEUE_DEPTH`.

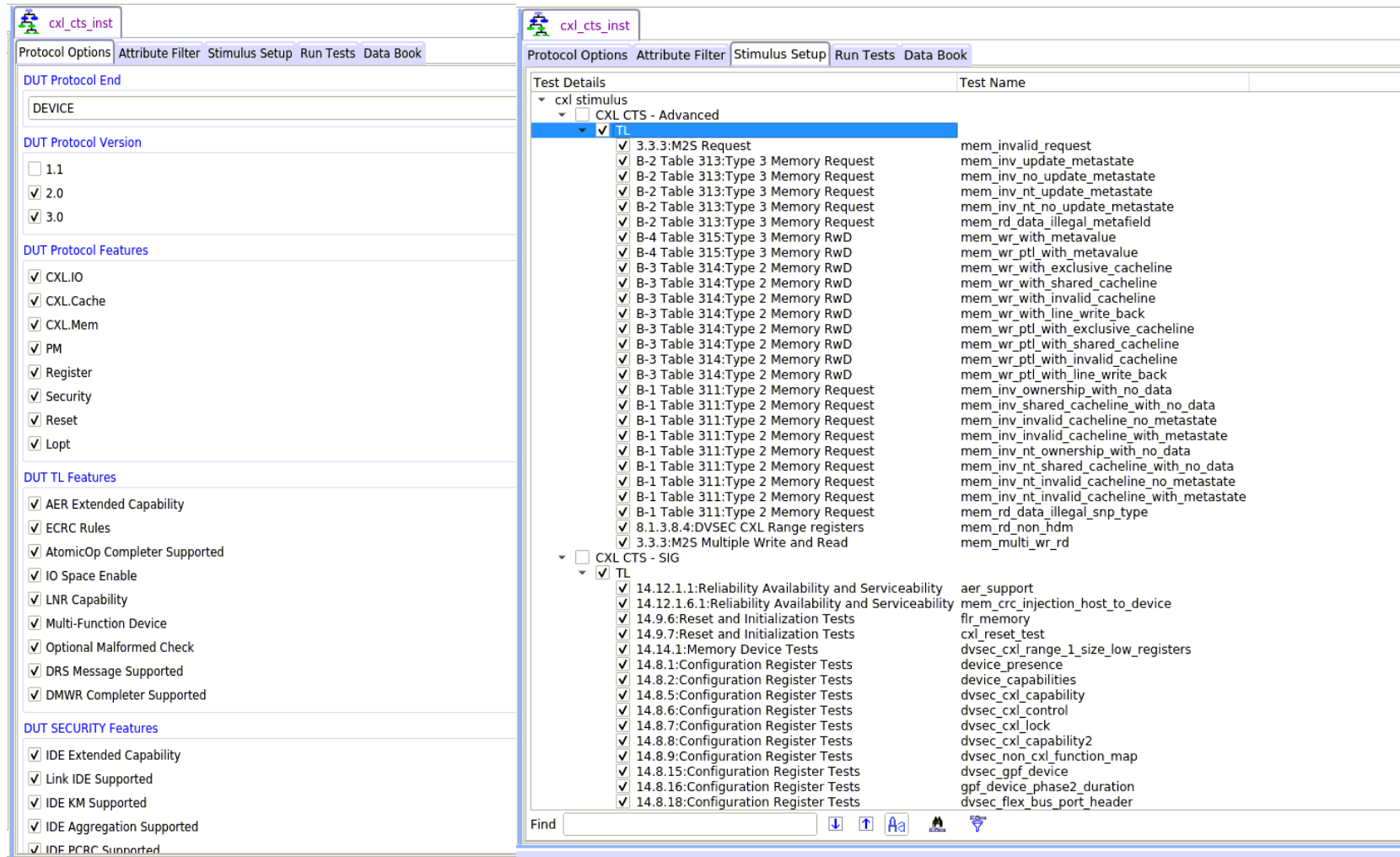
Testbench Diagram:

- Module: `icvip_cxl_serial_hdl_host/top/cxl_serial_host_0`
- Sub-modules: `BFM`, `UVM agent`
- Inputs: `reset_n`, `current_speed`, `tx_p`, `rx_p`, `tx_n`, `rx_n`, `CLKREQ_n`, `WAKE_n`
- Outputs: `CLK` (from `default_clk_gen`), `CLK_IN` (to `default_reset_gen`), `RESET` (from `default_reset_gen`)

Configuration Panel:

- RTL Port**: `cxl_serial_host_0`
- UVM Agent**: `default_clk_aen`, `default_reset_aen`
- Physical Setup**: `Wires`
- Modules**: `module`
- Parameters**:
 - PROTOCOL_VERSION**: `Parameter: PROTOCOL_VERSION`, `int unsigned`, `PROTOCOL_VERSION`, `2`
 - LANES**: `defines number of lanes, default value:4.`, `int unsigned`, `LANES`, `4`
 - IO_FLIT_QUEUE_DEPTH**: `defines IO Flit Queue depth, default value:2.`, `int unsigned`, `IO_FLIT_QUEUE_DEPTH`, `2`
 - TLP_MAX_PREFIX**: `defines maximum number of prefix, default value:2.`, `int unsigned`, `TLP_MAX_PREFIX`, `2`
 - NUM_VC_SUPPORTED**: `defines number of Virtual Channel supported, default value:1.`, `int unsigned`, `NUM_VC_SUPPORTED`, `1`
 - ENABLE_ORDERING_RULES**: `Enables the ordering rule, default value:0.`, `int unsigned`, `ENABLE_ORDERING_RULES`, `0`
 - ORDERING_QUEUE_COUNT**: `Defines the ordering queue count, default value:5.`, `int unsigned`, `ORDERING_QUEUE_COUNT`, `5`
 - MAXIMUM_BUS_NUMBER**: `defines maximum number of buses supported, default value:2.`, `int unsigned`, `MAXIMUM_BUS_NUMBER`, `2`
 - TLP_QUEUE_DEPTH**: `defines maximum number of buses supported, default value:2.`, `int unsigned`, `MAXIMUM_BUS_NUMBER`, `2`

Protocol Director



The screenshot displays the Protocol Director software interface, specifically the 'cxl_cts_inst' configuration window. The interface is divided into two main panes. The left pane contains configuration options for the DUT (Device Under Test) protocol and features. The right pane displays a list of test details, organized into two sections: 'CXL CTS - Advanced' and 'CXL CTS - SIG'.

Left Pane Configuration:

- DUT Protocol End:** DEVICE
- DUT Protocol Version:** 1.1, 2.0, 3.0 (3.0 is selected)
- DUT Protocol Features:** CXL.IO, CXL.Cache, CXL.Mem, PM, Register, Security, Reset, Lopt (all are checked)
- DUT TL Features:** AER Extended Capability, ECRC Rules, AtomicOp Completer Supported, IO Space Enable, LNR Capability, Multi-Function Device, Optional Malformed Check, DRS Message Supported, DMWR Completer Supported (all are checked)
- DUT SECURITY Features:** IDE Extended Capability, Link IDE Supported, IDE KM Supported, IDE Aggregation Supported, IDE PCRC Supported (all are checked)

Right Pane Test Details:

Test Details	Test Name
<input checked="" type="checkbox"/> CXL CTS - Advanced	
<input checked="" type="checkbox"/> TL	
<input checked="" type="checkbox"/> 3.3.3:M2S Request	mem_invalid_request
<input checked="" type="checkbox"/> B-2 Table 313:Type 3 Memory Request	mem_inv_update_metastate
<input checked="" type="checkbox"/> B-2 Table 313:Type 3 Memory Request	mem_inv_no_update_metastate
<input checked="" type="checkbox"/> B-2 Table 313:Type 3 Memory Request	mem_inv_nt_update_metastate
<input checked="" type="checkbox"/> B-2 Table 313:Type 3 Memory Request	mem_inv_nt_no_update_metastate
<input checked="" type="checkbox"/> B-2 Table 313:Type 3 Memory Request	mem_rd_data_illegal_metafield
<input checked="" type="checkbox"/> B-4 Table 315:Type 3 Memory RdD	mem_wr_with_metavalue
<input checked="" type="checkbox"/> B-4 Table 315:Type 3 Memory RdD	mem_wr_ptl_with_metavalue
<input checked="" type="checkbox"/> B-3 Table 314:Type 2 Memory RdD	mem_wr_with_exclusive_cacheline
<input checked="" type="checkbox"/> B-3 Table 314:Type 2 Memory RdD	mem_wr_with_shared_cacheline
<input checked="" type="checkbox"/> B-3 Table 314:Type 2 Memory RdD	mem_wr_with_invalid_cacheline
<input checked="" type="checkbox"/> B-3 Table 314:Type 2 Memory RdD	mem_wr_with_line_write_back
<input checked="" type="checkbox"/> B-3 Table 314:Type 2 Memory RdD	mem_wr_ptl_with_exclusive_cacheline
<input checked="" type="checkbox"/> B-3 Table 314:Type 2 Memory RdD	mem_wr_ptl_with_shared_cacheline
<input checked="" type="checkbox"/> B-3 Table 314:Type 2 Memory RdD	mem_wr_ptl_with_invalid_cacheline
<input checked="" type="checkbox"/> B-3 Table 314:Type 2 Memory RdD	mem_wr_ptl_with_line_write_back
<input checked="" type="checkbox"/> B-1 Table 311:Type 2 Memory Request	mem_inv_ownership_with_no_data
<input checked="" type="checkbox"/> B-1 Table 311:Type 2 Memory Request	mem_inv_shared_cacheline_with_no_data
<input checked="" type="checkbox"/> B-1 Table 311:Type 2 Memory Request	mem_inv_invalid_cacheline_no_metastate
<input checked="" type="checkbox"/> B-1 Table 311:Type 2 Memory Request	mem_inv_invalid_cacheline_with_metastate
<input checked="" type="checkbox"/> B-1 Table 311:Type 2 Memory Request	mem_inv_nt_ownership_with_no_data
<input checked="" type="checkbox"/> B-1 Table 311:Type 2 Memory Request	mem_inv_nt_shared_cacheline_with_no_data
<input checked="" type="checkbox"/> B-1 Table 311:Type 2 Memory Request	mem_inv_nt_invalid_cacheline_no_metastate
<input checked="" type="checkbox"/> B-1 Table 311:Type 2 Memory Request	mem_inv_nt_invalid_cacheline_with_metastate
<input checked="" type="checkbox"/> 8.1.3.8.4:DVSEC CXL Range registers	mem_rd_data_illegal_snp_type
<input checked="" type="checkbox"/> 3.3.3:M2S Multiple Write and Read	mem_rd_non_hdm
<input checked="" type="checkbox"/> CXL CTS - SIG	
<input checked="" type="checkbox"/> TL	
<input checked="" type="checkbox"/> 14.12.1.1:Reliability Availability and Serviceability	aer_support
<input checked="" type="checkbox"/> 14.12.1.6.1:Reliability Availability and Serviceability	mem_crc_injection_host_to_device
<input checked="" type="checkbox"/> 14.9.6:Reset and Initialization Tests	flr_memory
<input checked="" type="checkbox"/> 14.9.7:Reset and Initialization Tests	cxl_reset_test
<input checked="" type="checkbox"/> 14.14.1:Memory Device Tests	dvsec_cxl_range_1_size_low_registers
<input checked="" type="checkbox"/> 14.8.1:Configuration Register Tests	device_presence
<input checked="" type="checkbox"/> 14.8.2:Configuration Register Tests	device_capabilities
<input checked="" type="checkbox"/> 14.8.5:Configuration Register Tests	dvsec_cxl_capability
<input checked="" type="checkbox"/> 14.8.6:Configuration Register Tests	dvsec_cxl_control
<input checked="" type="checkbox"/> 14.8.7:Configuration Register Tests	dvsec_cxl_lock
<input checked="" type="checkbox"/> 14.8.8:Configuration Register Tests	dvsec_cxl_capability2
<input checked="" type="checkbox"/> 14.8.9:Configuration Register Tests	dvsec_non_cxl_function_map
<input checked="" type="checkbox"/> 14.8.15:Configuration Register Tests	dvsec_gpf_device
<input checked="" type="checkbox"/> 14.8.16:Configuration Register Tests	gpf_device_phase2_duration
<input checked="" type="checkbox"/> 14.8.18:Configuration Register Tests	dvsec_flex_bus_port_header



Protocol Checks

- ❑ Exhaustive Set of checks (~1200)
 - ❑ CXL.io, CXL.cache and CXL.mem
 - ❑ TL, LL, Arbiter, PL and PM
 - ❑ Config Space, Memory Mapped registers
 - ❑ Coherency checks

ERROR @ 24329536 ps: **ICVIP_CXL_INVALID_MAX_M2S_BIRSP_COUNT_128B_GROUP** -
ICVIP_CXL[DEVICE]:

ICVIP_CXL[DEVICE]: Maximum message count for BIResponse message exceeded in rolling 128B group

ICVIP_CXL[DEVICE]: - Rolling group: ICVIP_CXL_ROLLING_GROUP_DA

ICVIP_CXL[DEVICE]: - Message count within 128B Group: 11

ICVIP_CXL[DEVICE]: The maximum message rules are applicable on a rolling 128B group in which the groups are A=Slot 0 to 3, B=Slot 4 to 7, C=Slot 8 to 11, D=Slot 12 to 14. Refer Table 4-17 : For M2S BIRsp
- Maximum message count per 128B group is 3. CXL Version 3.0(4.3.4)

ERROR @ 16848875 ps: **ICVIP_CXL_H2D_DATA_POISONED** - ICVIP_CXL[DEVICE]:

ICVIP_CXL[DEVICE]: H2D data is detected with poison bit = 1'b1

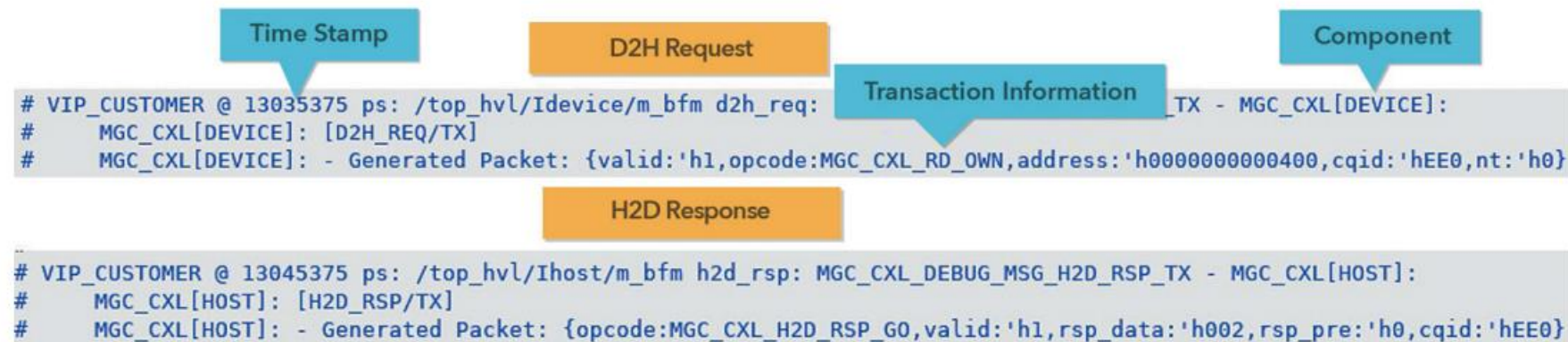
```
ICVIP_CXL_IDE_MAC_NOT_RX_WHEN_EXPECTED, \
ICVIP_CXL_IDE_TMAC_RX_WHEN_NOT_EXPECTED, \
ICVIP_CXL_IDE_KEY_CHANGE_PROTOCOL_FLIT_EARLY_RX, \
ICVIP_CXL_IDE_TX_KEY_REFRESH_TIME_LESS_THAN_RX_MIN, \
ICVIP_CXL_IDE_PROTOCOL_FLIT_RX_EARLY_AFTER_TMAC, \
ICVIP_CXL_IDE_MAC_RX_LINK_INSECURE_STATE, \
ICVIP_CXL_IDE_TMAC_RX_LINK_INSECURE_STATE, \
ICVIP_CXL_IDE_MAC_RX_WHEN_NOT_EXPECTED, \
ICVIP_CXL_IDE_CAPABLE_NOT_SET, \
ICVIP_CXL_IDE_MAC_CHECK_FAILURE, \
ICVIP_CXL_IDE_SKID_MODE_SET_WHEN_NOT_SUPP, \
ICVIP_CXL_CACHEMEM_IDE_KM_QUERY_INVALID_LENGTH, \
ICVIP_CXL_CACHEMEM_IDE_KM_KEY_PROG_INVALID_LENGTH, \
ICVIP_CXL_INVALID_CAPABILITY_ID, \
ICVIP_CXL_PM_TARGET_AGENT_ID_RSVD_PM2IP_MSG, \
ICVIP_CXL_PM_TARGET_AGENT_ID_RSVD_IP2PM_MSG, \
ICVIP_CXL_PM_CAPABILITY_VECTOR_IP2PM_MSG_MISMATCH, \
ICVIP_CXL_PM_ZERO_CREDIT_IP2PM_MSG, \
ICVIP_CXL_PM_MSG_VDM_CODE_INVALID, \
ICVIP_CXL_PM_MSG_LOGICAL_PM_OPCODE_INVALID, \
ICVIP_CXL_PM_EXCESS_CREDIT_RECEIVED, \
ICVIP_CXL_ILLEGAL_G1_SLOT_RECEIVED, \
ICVIP_CXL_ILLEGAL_G4_SLOT_RECEIVED, \
ICVIP_CXL_ILLEGAL_H12_SLOT_RECEIVED, \
ICVIP_CXL_ILLEGAL_H15_SLOT_RECEIVED, \
ICVIP_CXL_TYPE1_G14_RECEIVED_WITH_INVALID_MEM_RWD, \
ICVIP_CXL_TYPE1_G15_RECEIVED_WITH_INVALID_MEM_DRS, \
ICVIP_CXL_H2D_NOT_TIGHTLY_PACKED_WITHIN_G1_SLOT, \
ICVIP_CXL_H2D_NOT_TIGHTLY_PACKED_WITHIN_G12_SLOT, \
ICVIP_CXL_S2M_NOT_TIGHTLY_PACKED_WITHIN_G15_SLOT, \
ICVIP_CXL_H12_RECEIVED_WITH_FIRST_INVALID_DATA_HDR, \
ICVIP_CXL_H14_RECEIVED_WITH_INVALID_RWD, \
ICVIP_CXL_H15_RECEIVED_WITH_FIRST_INVALID_DRS, \
ICVIP_CXL_G12_RECEIVED_WITH_FIRST_INVALID_DATA_HDR, \
ICVIP_CXL_G14_RECEIVED_WITH_INVALID_RWD, \
ICVIP_CXL_G15_RECEIVED_WITH_FIRST_INVALID_DRS, \
ICVIP_CXL_BI_RSP_NOT_TIGHTLY_PACKED_WITHIN_H5_SLOT, \
ICVIP_CXL_S2M_NDR_NOT_TIGHTLY_PACKED_IN_GROUP, \
ICVIP_CXL_S2M_BISNP_NOT_TIGHTLY_PACKED_IN_GROUP, \
ICVIP_CXL_D2H_DATA_HDR_NOT_TIGHTLY_PACKED_IN_GROUP, \
ICVIP_CXL_D2H_REQ_NOT_TIGHTLY_PACKED_IN_GROUP, \
ICVIP_CXL_D2H_RSP_NOT_TIGHTLY_PACKED_IN_GROUP, \
ICVIP_CXL_H13_RECEIVED_AND_DATA_SLOT_EXCEEDS_16, \
ICVIP_CXL_H15_RECEIVED_AND_DATA_SLOT_EXCEEDS_16, \
ICVIP_CXL_INVALID_MAX_S2M_BISNP_REQ_COUNT_128B_GROUP, \
```


Debugging methods

- ☐ Debug Messages
- ☐ Loggers and transcript
- ☐ Protocol Aware Debug GUI

Debug Messages

- ❑ print the traffic and transactions information in a transcript or on a shell as first step of debugging without looking into the loggers
- ❑ Text loggers – layer wise
- ❑ Debug GUI



```
# VIP_CUSTOMER @ 13035375 ps: /top_hvl/Idevice/m_bfm d2h_req: TX - MGC_CXL[DEVICE]:  
# MGC_CXL[DEVICE]: [D2H_REQ/TX]  
# MGC_CXL[DEVICE]: - Generated Packet: {valid:'h1,opcode:MGC_CXL_RD_OWN,address:'h0000000000400,cqid:'hEE0,nt:'h0}  
  
# VIP_CUSTOMER @ 13045375 ps: /top_hvl/Ihost/m_bfm h2d_rsp: MGC_CXL_DEBUG_MSG_H2D_RSP_TX - MGC_CXL[HOST]:  
# MGC_CXL[HOST]: [H2D_RSP/TX]  
# MGC_CXL[HOST]: - Generated Packet: {opcode:MGC_CXL_H2D_RSP_G0,valid:'h1,rsp_data:'h002,rsp_pre:'h0,cqid:'hEE0}
```



Loggers

- ❑ layer wise loggers
- ❑ Cache and mem loggers

ICVIP_CXL_LOGGER_DIR
ICVIP_CXL_PL_LOGGER
ICVIP_CXL_RAW_DATA_LOGGER
ICVIP_CXL_TL_DLL_LOGGER
ICVIP_CXL_DEBUG_LOGGER
ICVIP_CXL_MEM_LOGGER
ICVIP_CXL_FLIT_LOGGER
ICVIP_CXL_CACHE_LOGGER

Mem Logger Snippet

Request/Response type	Packet fields: Value
31841875 M2S_RWD	rx_completed:0 rwd: {ld_id:0, tc:0, poison:0, address:00000076EA30, tag:0000, meta_value:MEM_META_INVALID, meta_field:MEM_META_STATE, snp_type:M2S_SNP_NOP, mem_opcode:RWD_WR, valid:1} data: {6E, 72, B8, 77, ...}
31901125 S2M_RSP_NDR	dev_load:1 ld_id:0 tag:0000 meta_value:MEM_META_INVALID meta_field:MEM_META_NOP mem_opcode:S2M_NDR_CMP valid:1
31901375 M2S_REQ	ld_id:0 tc:0 addr:00000076EA30 tag:0001 meta_value:MEM_META_INVALID meta_field:MEM_META_NOP snp_type:M2S_SNP_NOP mem_opcode:REQ_MEM_RD valid:1
31957125 S2M_RSP_DRS	rx_completed:0 drs: {dev_load:1, ld_id:0, poison:0, tag:0001, meta_value:MEM_META_INVALID, meta_field:MEM_META_NOP, mem_opcode:S2M_DRS_DATA, valid:1} size:64B data: {00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, ...}
31957375 S2M_RSP_DRS	rx_completed:0 drs: {dev_load:1, ld_id:0, poison:0, tag:0001, meta_value:MEM_META_INVALID, meta_field:MEM_META_NOP, mem_opcode:S2M_DRS_DATA, valid:1} size:64B data: {6E, 72, B8, 77, C0, 4B, F3, 0F, 31, 12, 97, FE, ...}
31957625 S2M_RSP_DRS	rx_completed:0 drs: {dev_load:1, ld_id:0, poison:0, tag:0001, meta_value:MEM_META_INVALID, meta_field:MEM_META_NOP, mem_opcode:S2M_DRS_DATA, valid:1} size:64B data: {6E, 72, B8, 77, C0, 4B, F3, 0F, 31, 12, 97, FE, ...}
31958125 S2M_RSP_DRS	rx_completed:0 drs: {dev_load:1, ld_id:0, poison:0, tag:0001, meta_value:MEM_META_INVALID, meta_field:MEM_META_NOP, mem_opcode:S2M_DRS_DATA, valid:1} size:64B data: {6E, 72, B8, 77, C0, 4B, F3, 0F, 31, 12, 97, FE, ...}
31958375 S2M_RSP_DRS	rx_completed:1 drs: {dev_load:1, ld_id:0, poison:0, tag:0001, meta_value:MEM_META_INVALID, meta_field:MEM_META_NOP, mem_opcode:S2M_DRS_DATA, valid:1} size:64B data: {6E, 72, B8, 77, C0, 4B, F3, 0F, 31, 12, 97, FE, ...}

Mem traffic from Master and subordinate

Protocol Aware Debug GUI

☐ Physical Layer

☐ IO traffic

Physical layer Flit Viewer IO Traffic Memory Traffic Cache Traffic											
IO Overview											
Item	Time	Origin	Traffic Type	Address	Tag	BDF	CPL_Status	Vc_id	Shared	Crc	Start_time
DEVICE_CMPL	31658675000	Device	CPLD		187		CPL_SC				3165867
HOST_TLP	31659175000	Host	CFGWD0	00000000000005B8	188	0100					3165867
DEVICE_DLLP	31674425000	Device	ACK					0	0	5D53	
HOST_DLLP	31675925000	Host	ACK					0	0	F58F	
DEVICE_CMPL	31675925000	Device	CPLD		188		CPL_SC				3167592
HOST_TLP	31676425000	Host	CFGWD0	00000000000005B8	189	0100					3167592
DEVICE_DLLP	31691675000	Device	ACK					0	0	BE7F	
HOST_DLLP	31693175000	Host	ACK					0	0	5494	
DEVICE_CMPL	31693175000	Device	CPLD		189		CPL_SC				3169317
HOST_TLP	31693675000	Host	CFGWD0	00000000000005B0	18A	0100					3169317
DEVICE_DLLP	31708925000	Device	ACK					0	0	1F64	
HOST_DLLP	31710675000	Host	ACK					0	0	B7B8	
DEVICE_CMPL	31710675000	Device	CPLD		18A		CPL_SC				3171067
HOST_TLP	31711175000	Host	CFGWD0	00000000000005B0	18B	0100					3171067
DEVICE_DLLP	31726425000	Device	ACK					0	0	70FB	
HOST_DLLP	31727925000	Host	ACK					0	0	16A3	
DEVICE_CMPL	31727925000	Device	CPLD		18B		CPL_SC				3172792
HOST_TLP	31728425000	Host	CFGWD0	00000000000005B4	18C	0100					3172792

IO Data View

Data

{03,00,00,80}

{}

{00,03,A5,00}

{00,03,BC,00}

{03,00,00,00}

{03,00,00,00}

{00,03,A6,00}

{00,03,BD,00}

{}

{00,03,A7,00}

{00,03,BE,00}

{02,00,00,00}

{}

{00,03,A8,00}

{00,03,BF,00}

{02,00,00,00}

{98,1E,00,00}

Physical layer

Flit Viewer

IO Traffic

Memory Traffic

Cache Traffic

Link Status

Link Speed: GEN5

LTSSM Sub-State: L0_SUB

PL States View

Item	Time	Origin	OS Count	S 0	S 1	S 2	S 3	S 4	S 5	S 6	S 7
▼ PL STATE CHANGES											
STATE_CHANGE	0	DETECT_QUIET	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	400000	DETECT_QUIET	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	3000000	DETECT_QUIET	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	1022800000	DETECT_ACTIVE	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	3046800000	POLLING_ACTIVE	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	5214800000	POLLING_CONFIGURATION	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	6510800000	CONFIG_LINKWIDTH_START	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	7230800000	CONFIG_LINKWIDTH_ACCEPT	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	7598800000	CONFIG_LANENUM_WAIT	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	8254800000	CONFIG_LANENUM_ACCEPT	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	8386800000	CONFIG_COMPLETE	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	10094800000	CONFIG_IDLE	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	10346800000	L0_SUB	DL_INACTIVE	PM_LDn	GEN1						
STATE_CHANGE	10350800000	L0_SUB	DL_INACTIVE	PM_L0	GEN1						
STATE_CHANGE	10354800000	RECOVERY_RCVRCLOCK	DL_INACTIVE	PM_L0	GEN1						
STATE_CHANGE	10962800000	RECOVERY_RCVRCFG	DL_INACTIVE	PM_L0	GEN1						
STATE_CHANGE	13226800000	RECOVERY_SPEED	DL_INACTIVE	PM_L0	GEN1						
STATE_CHANGE	13750800000	RECOVERY_SPEED	DL_INACTIVE	PM_L0	GEN5						
STATE_CHANGE	15753175000	RECOVERY_RCVRCLOCK	DL_INACTIVE	PM_L0	GEN5						
STATE_CHANGE	15796925000	RECOVERY_RCVRCFG	DL_INACTIVE	PM_L0	GEN5						
STATE_CHANGE	15875175000	RECOVERY_IDLE	DL_INACTIVE	PM_L0	GEN5						
STATE_CHANGE	15907925000	L0_SUB	DL_INACTIVE	PM_L0	GEN5						
STATE_CHANGE	15908175000	L0_SUB	DL_FEATURE	PM_L0	GEN5						
STATE_CHANGE	15949675000	L0_SUB	DL_INIT1	PM_L0	GEN5						
STATE_CHANGE	15977675000	L0_SUB	DL_INIT2	PM_L0	GEN5						

Find

Protocol Aware Debug GUI

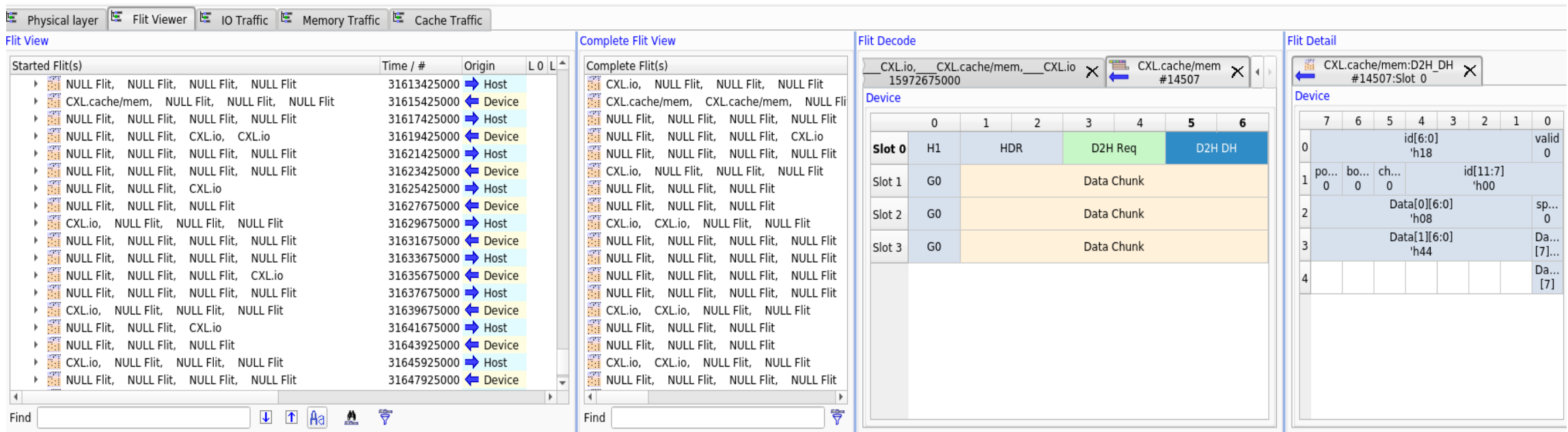
Cache/Mem traffic

Physical layer Flit Viewer IO Traffic Memory Traffic Cache Traffic										
Cache Overview										
Item	Time	Origin	BFM_id	Start_time	Valid	Opcode	Address	Cache_id	Nt	Req
D2H_REQ	25341175000	Device	23	25341175000	1	RD_OWN	000000000000	0	0	000
D2H_DATA	25343425000	Device	11	25343425000	1					000
H2D_RSP	25343675000	Host	23	25343425000	1	H2D_RSP_GO		0		000
H2D_DATA	25343925000	Host	12	25343425000	1			0		000
D2H_REQ	25363675000	Device	24	25363675000	1	DIRTY_EVICT	000000000000	0	0	000
H2D_RSP	25363925000	Host	24	25363675000	1	H2D_RSP_GO_WR_PULL		0		000
D2H_DATA	25383175000	Device	12	25383175000	1					000
D2H_REQ	26343175000	Device	25	26343175000	1	RD_OWN	000000000000	0	0	000
H2D_RSP	26343425000	Host	25	26343175000	1	H2D_RSP_GO		0		000
H2D_DATA	26343675000	Host	13	26343175000	1			0		000
D2H_REQ	26364925000	Device	26	26364925000	1	DIRTY_EVICT	000000000000	0	0	000
H2D_RSP	26365175000	Host	26	26364925000	1	H2D_RSP_GO_WR_PULL		0		000
D2H_REQ	26383175000	Device	27	26383175000	1	RD_OWN	000000000000	0	0	000
D2H_DATA	26385425000	Device	13	26385425000	1					000
H2D_RSP	26385675000	Host	27	26385425000	1	H2D_RSP_GO		0		000

Cache Data View	
Data	B_enable
{AA,AA,AA,AA,AA,AA,AA,AA,AA,AA,A...	FFFFFFFFFFFFFFFF
{AA,AA,AA,AA,AA,AA,AA,AA,AA,AA,A...	
{AA,AA,AA,AA,AA,AA,AA,AA,AA,AA,A...	FFFFFFFFFFFFFFFF
{AA,AA,AA,AA,AA,AA,AA,AA,AA,AA,A...	
{AA,AA,AA,AA,AA,AA,AA,AA,AA,AA,A...	FFFFFFFFFFFFFFFF

Protocol Aware Debug GUI

- ☐ Flit Viewer
- ☐ IO/Cache/Mem traffic



The screenshot displays the Protocol Aware Debug GUI with four main panels:

- Flit View:** A list of started flits with columns for Time / #, Origin, and L0. It shows a sequence of flits between Host and Device, including NULL flits and CXL.io traffic.
- Complete Flit View:** A detailed view of a specific flit, showing its components (CXL.io, NULL Flit, CXL.cache/mem) and their origins.
- Flit Decode:** A table showing the decoded structure of a flit across slots. Slot 0 contains H1, HDR, D2H Req, and D2H DH. Slots 1, 2, and 3 contain Data Chunks.
- Flit Detail:** A detailed view of the flit's data fields, including id[6:0], id[11:7], Data[0][6:0], and Data[1][6:0].

Scoreboard

- ☐ Passive component that implements the reference memory to perform read data checking
- ☐ Updates the reference memory with the write data reported by monitor when write occurs
- ☐ Compares the data from read transaction with the data from reference memory when read occurs
- ☐ Logs UVM INFO or ERROR in logs

Coverage

- ❑ Executable verification plan that hierarchically correlates to the CXL specification sections
- ❑ Each cover point has a corresponding test in compliance test suite

Section	Title	Description	Link	Type	Weight	Goal
1	Introduction	This section provides the overview of CXL and its related nomenclature.			0	0
2	Compute Express Link System Architecture	This section describes the performance advantages and key features of CXL.			0	0
3	Compute Express Link Transaction Layer	This section provides transaction layer constructs.				
3.1	CXL.io	This sections specifies the CXL.io transaction layer packet formatting, credit-based flow control etc.				
3.2	CXL.cache	This section specifies the CXL.cache transaction layer constructs.				
3.2.1	Overview	This section defines the interactions between the device and Host as a number of requests that each have at least one associated response message and sometimes a data transfer.				
3.2.2	CXL.cache Channel Description	This section defines the cache channels and provide their credit-based flow information			0	0
3.2.3	CXL.cache Wire Description	This Section defines fields for each CXL.cache channel			1	100
3.2.3.1	D2H Request opcode	This covers the D2H request opcode field	cache_cov::d2h_req_opcode	CoverPoint	1	100
3.2.3.1.1	D2H Request cqid	This covers the D2H request CQID field	cache_cov::d2h_req_cqid	CoverPoint	1	100
3.2.3.1.2	D2H Request NT	This covers the D2H request Non Temoral Field	cache_cov::d2h_req_nt	CoverPoint	1	100

Spec tagging for every section

Detailed coverage of every field

Full flexibility for handling cover points

Summary

- ❑ Automatic UVM-TB generation
- ❑ Compliance test suite (ready to use exhaustive stimulus)
- ❑ Debug capabilities
- ❑ Discussed features enabled our customers to verify their designs faster and achieve lesser time to market

Thank You !!

Please visit booth # for more details and demo