Shift Left CXL Product Readiness and Validation via OpenCIS

Presenters:

Karthik Balan, Associate Director, Samsung Electronics

Arun Bosco J, Associate Director, Samsung Electronics

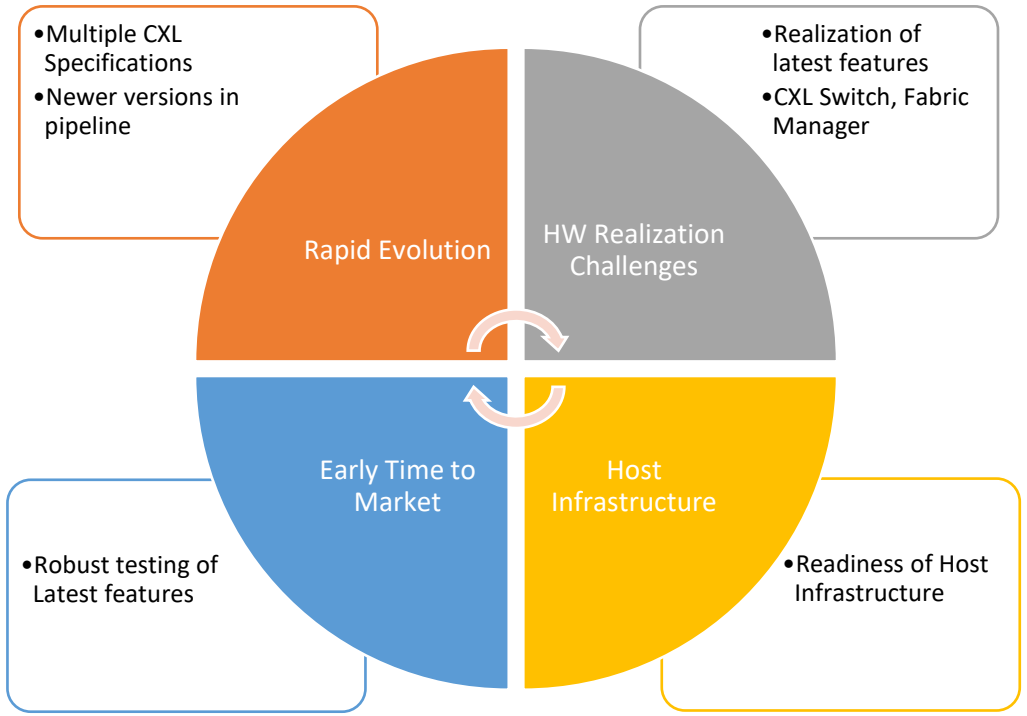
Uthank Nandin NN, Associate Staff Engineer, Samsung Electronics

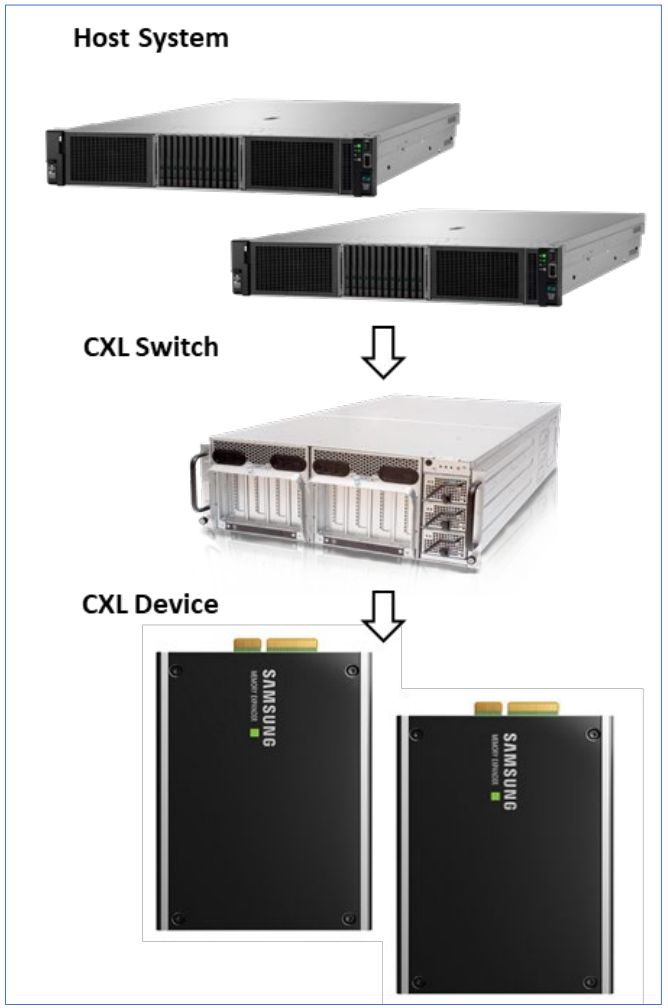
Outline

- Background
- Objective
- View of OpenCIS Ecosystem
- Functional Block of OpenCIS
- Architecture of Emulated CXL Switch
- OpenCIS based MH-MD Ecosystem & Output View
- Takeaways

Background

- '***Continuously Evolving CXL Spec and Features***' -> Challenges for Host Infra readiness
- Need of Shift Left Readiness for realization & verification of Feature ahead of ASIC

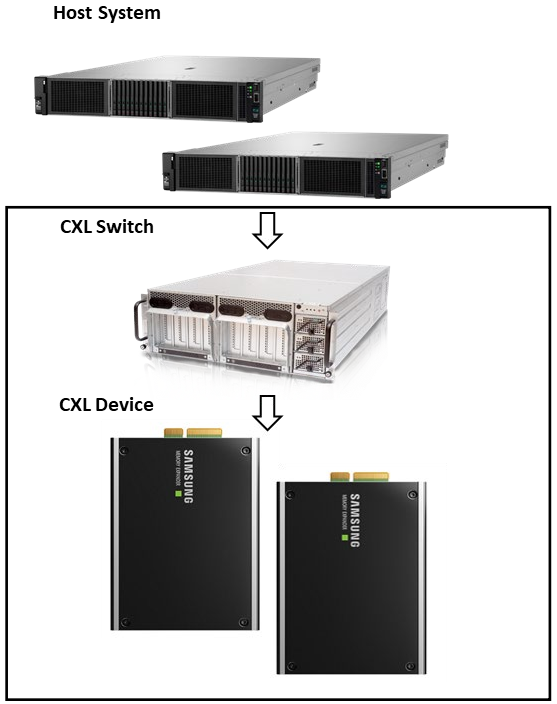




Hardware Ecosystem of CXL

Objective

- ***Awareness*** of ecosystem for Shift left development and validation of CXL Products.
- Host Test Infra readiness in ***parallel*** with CXL device availability
- Flexible open source environment for prototype end-to-end SW for future features



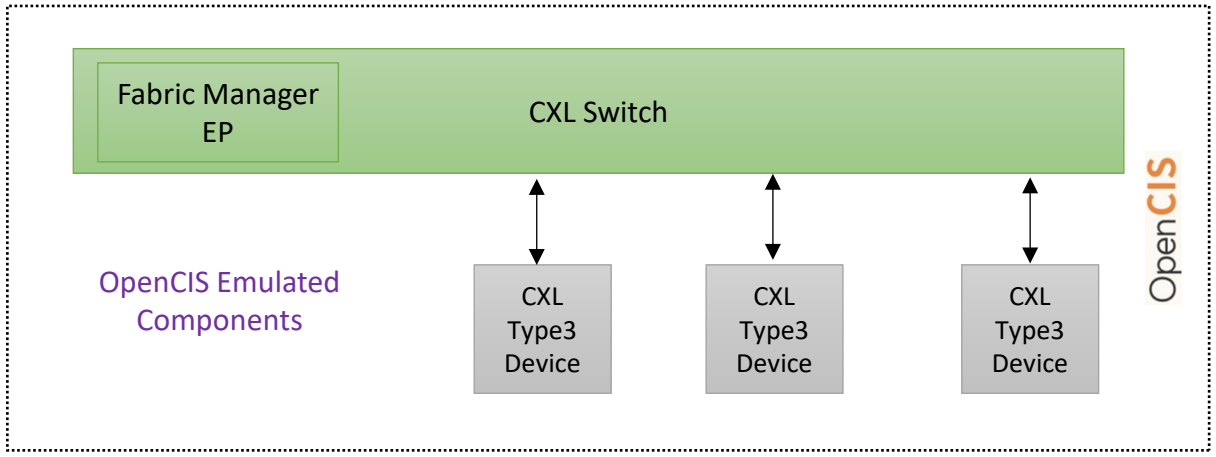
CXL Switch, CXL Device

OpenCIS

[1] https://www.opencis.io/

View of OpenCIS Ecosystem

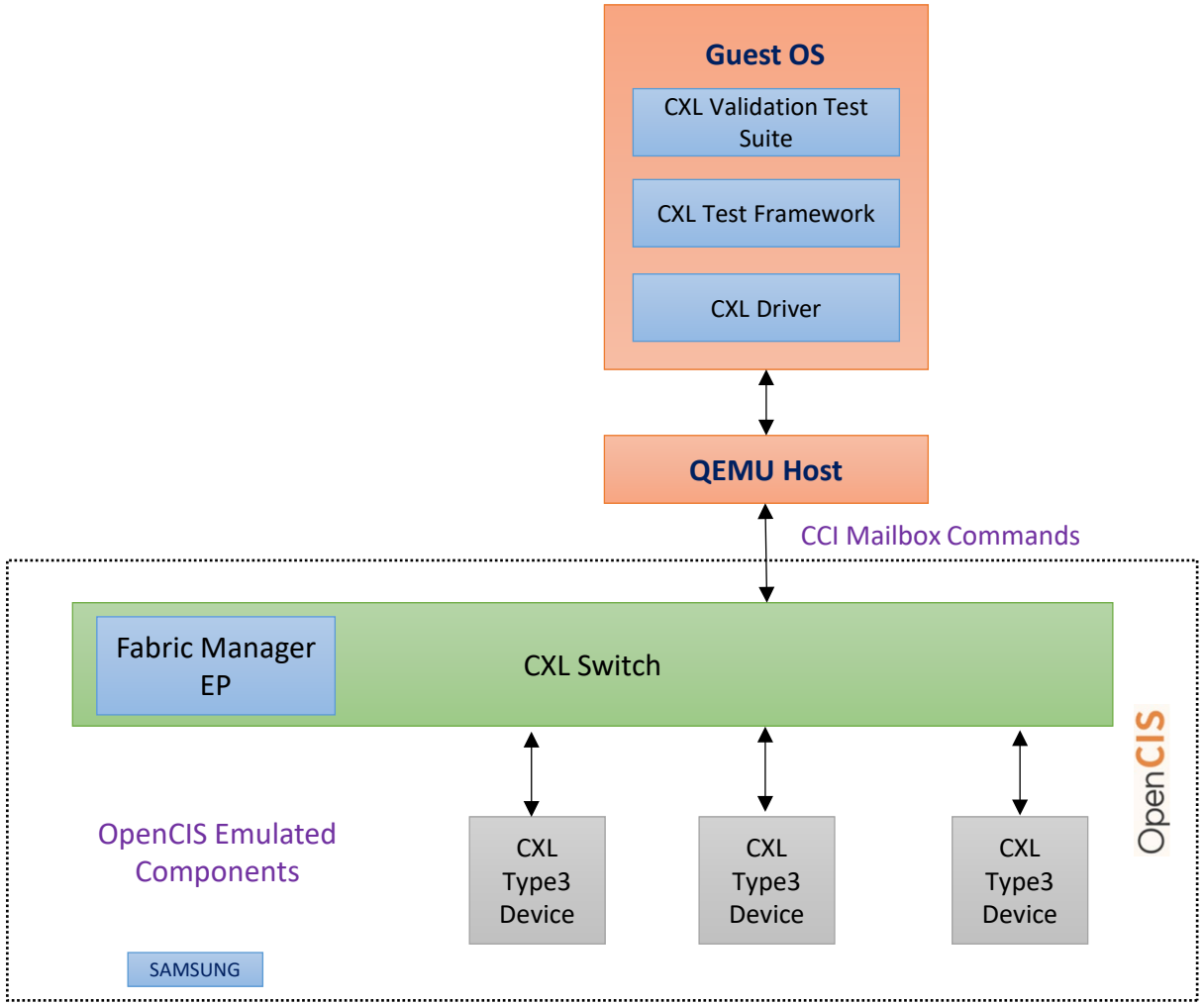
- Supported Emulation Modules by OpenCIS for CXL Device and Ecosystem



- ❖ OpenCIS Emulated CXL Components
  - ✓ CXL Switch
  - ✓ CXL Fabric Manager API End Point
  - ✓ CXL Type 3 device

View of OpenCIS Ecosystem

- View of QEMU based Host Test Platform

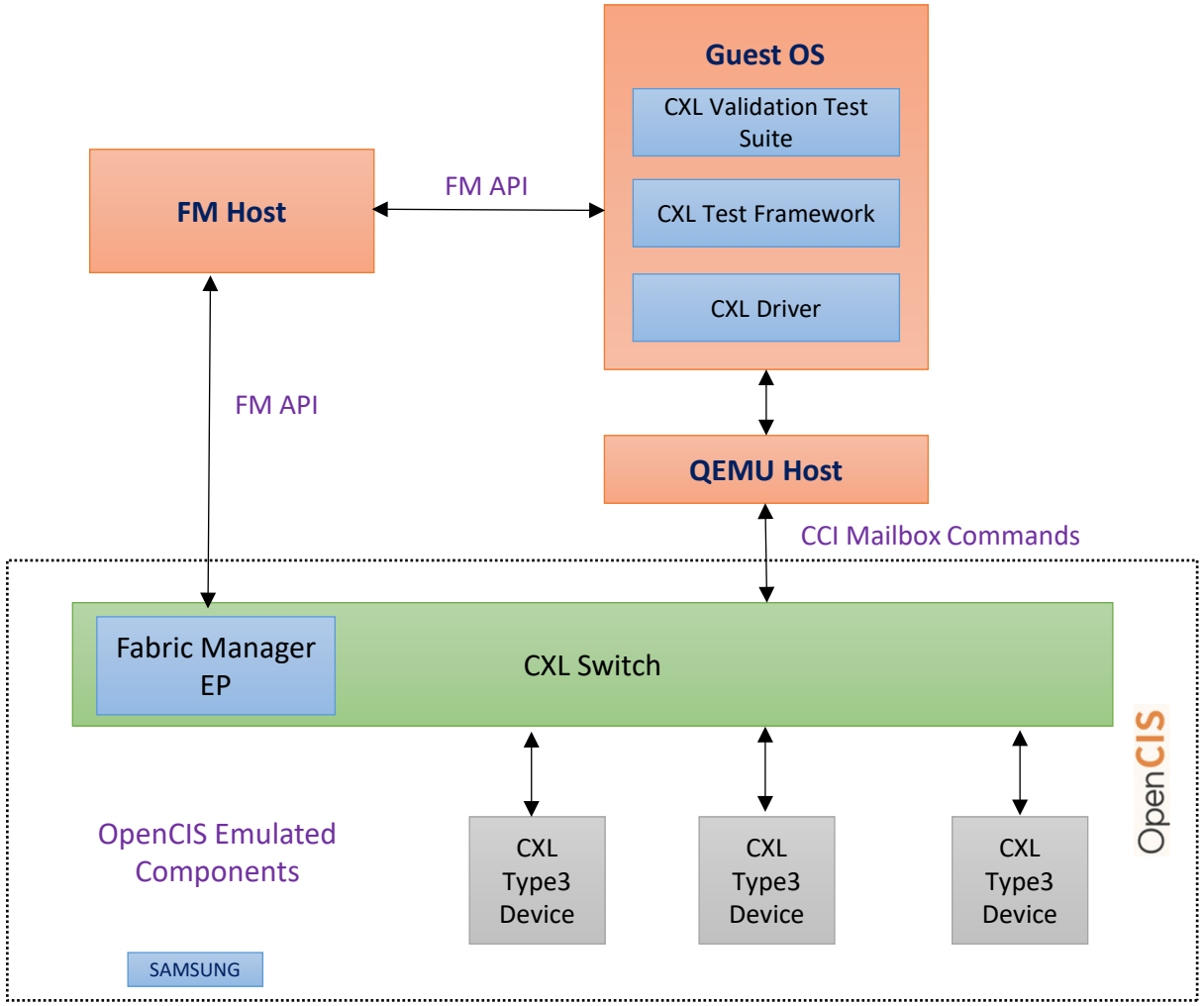


- ❖ OpenCIS Emulated CXL Components
  - ✓ CXL Switch
  - ✓ CXL Fabric Manager
  - ✓ CXL Type 3 device

- ❖ CXL Host Test Platform
  - ✓ Validation Suite
  - ✓ QEMU Host
  - ✓ Connected to Emulated CXL

View of OpenCIS Ecosystem

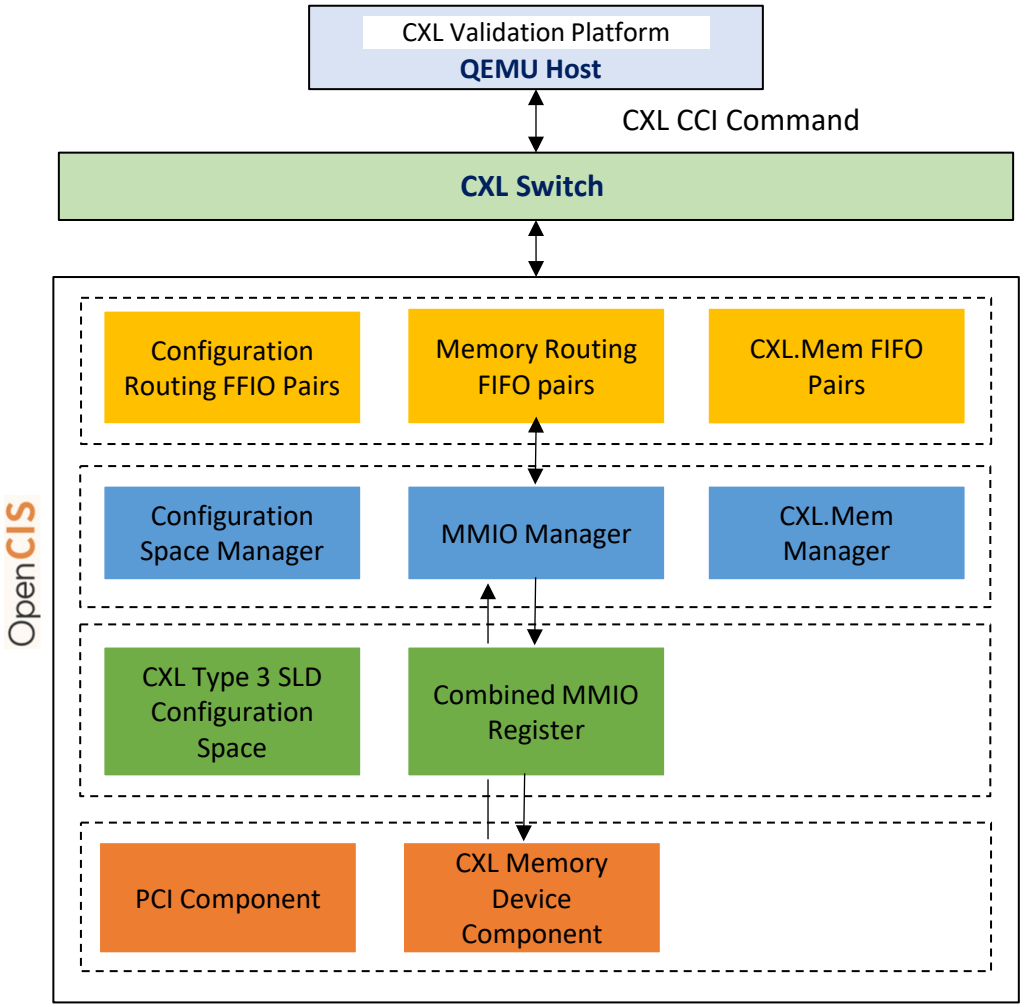
- View of Fabric Manager establishment and flow



- ❖ OpenCIS Emulated CXL Components
  - ✓ CXL Switch
  - ✓ CXL Fabric Manager
  - ✓ CXL Type 3 device
- ❖ CXL Host Test Platform
  - ✓ Validation Suite
  - ✓ QEMU Host
  - ✓ Connected to Emulated CXL
- ❖ Fabric Manager Interface
  - ✓ FM Host System to send commands to Fabric manager
  - ✓ Host *can* have separate FM Host or use direct FM API via CLI

Functional Block of OpenCIS

- 4 key architecture blocks in OpenCIS tool
- Flow of CCI Command is sequential for CXL Type 3 device



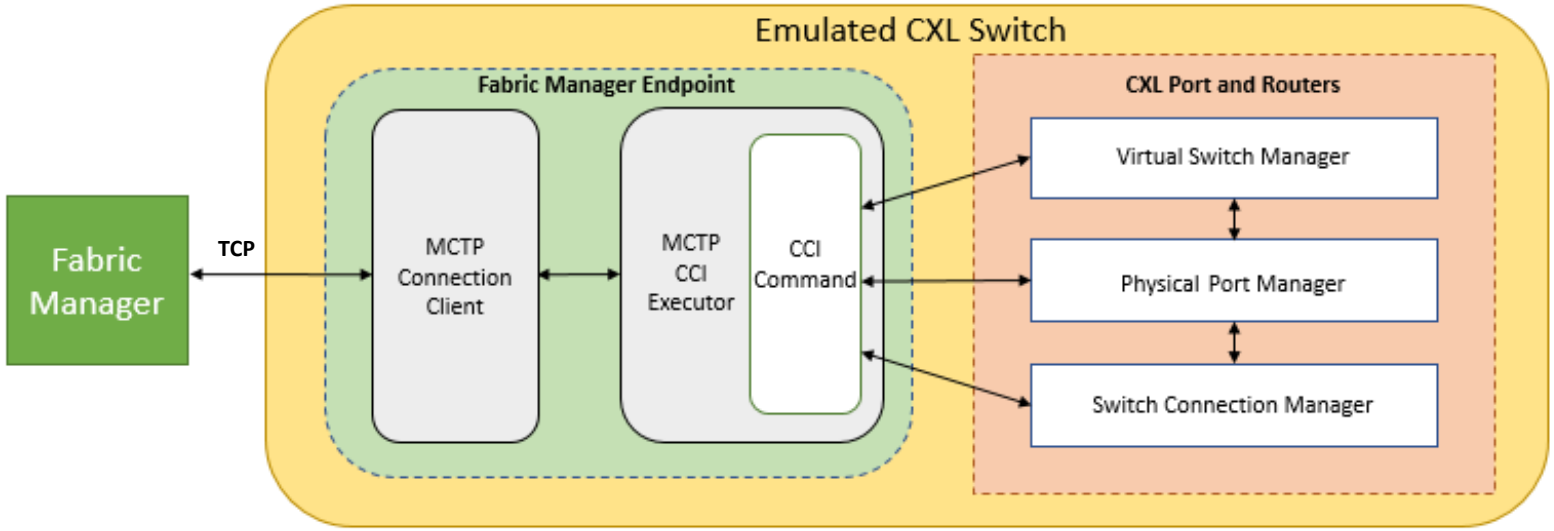
CXL Device

CXL Connection Client	Setting up TCP socket connections and active routing of packets to next layers
Packet Processing Layer	Processes the packet based on packet type in the header
Register Interface Layer	These classes process packets from upper layer and access emulated hardware logic when needed.
H/W Emulation Layer	These simulate necessary behaviors for CXL Type 3 device operations.

Architecture of Emulated CXL Switch

- Emulated Switch Contains *Two* Key Components
- 1) FM Endpoint 2) CXL Port and Routers

CXL Switch



❖ Fabric Manager Endpoint Module

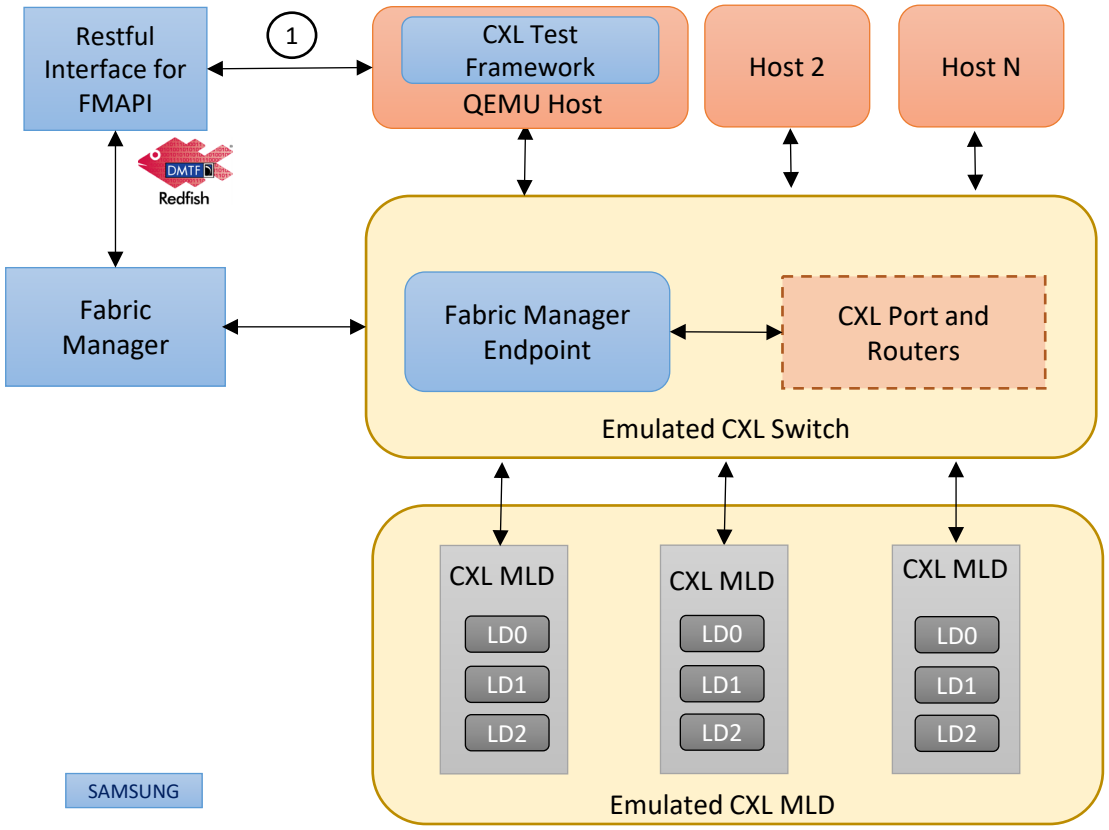
- ➢ MCTP Connection Client
  - Establish MCTP Connection with FM and Executer
- ➢ MCTP CCI Executor
  - Processing of Switch related FM API
  - Interacts with Port Manager for Binding Ops

❖ CXL Port and Routers

- ➢ Virtual Switch Manager
  - Manages the Virtual Ports connection
  - Maintains Internal port mapping with Physical Port
- ➢ Physical Port Manager
  - Manages the Physical Ports connection
  - External interaction with Host System/FM
- ➢ Switch Connection Manager
  - Establish connection b/w Switch & Device
  - Link Training & Negotiation Ops

OpenCIS based MH-MD Ecosystem View

- Reduces dependency on Real Switch or Real MLD device for Shift Left Readiness of Host Infra

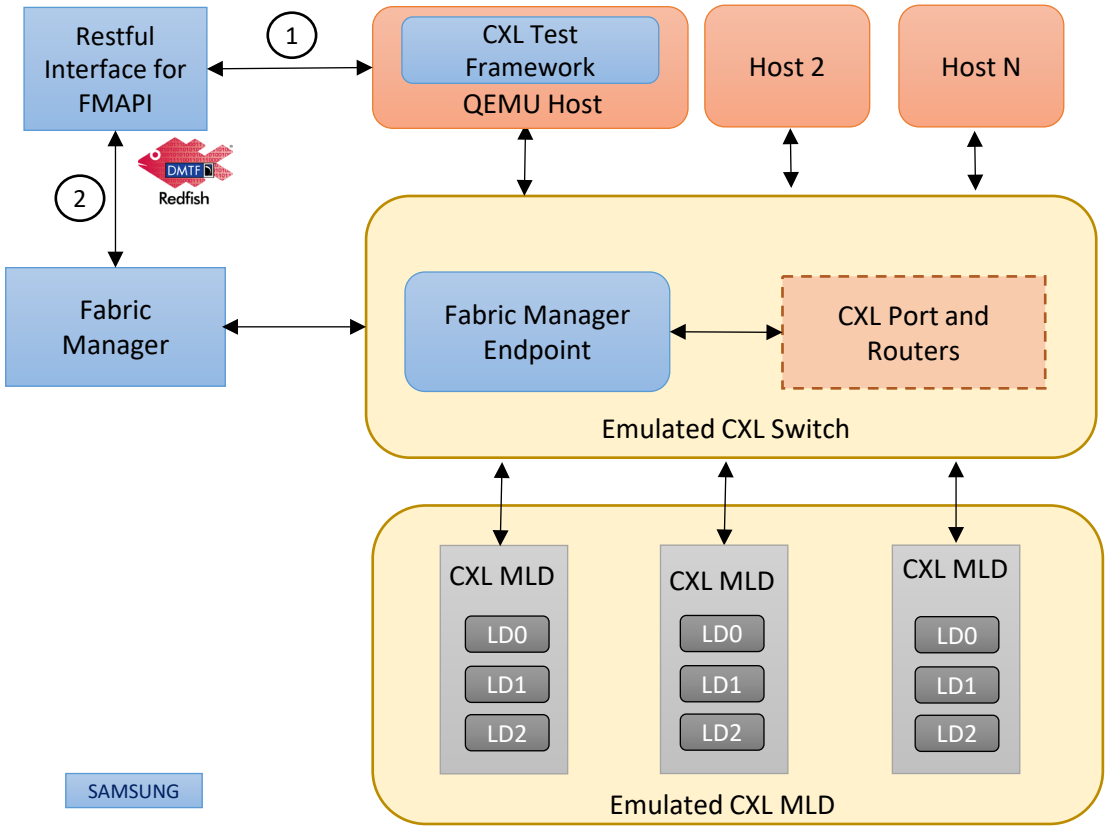


LD – Logical Device, MLD – Multi Logical Devices

1. Host issues *http* request of FM API Command
   - FM API defined by Specification
   - Supports multiple interaction methods b/w Host and FM, one is http approach

OpenCIS based MH-MD Ecosystem View

- Multi-Host (MH) – Multi-Device (MD) Ecosystem is achieved by Switch Emulation
- Establish connection via Restful Interface for http based communication

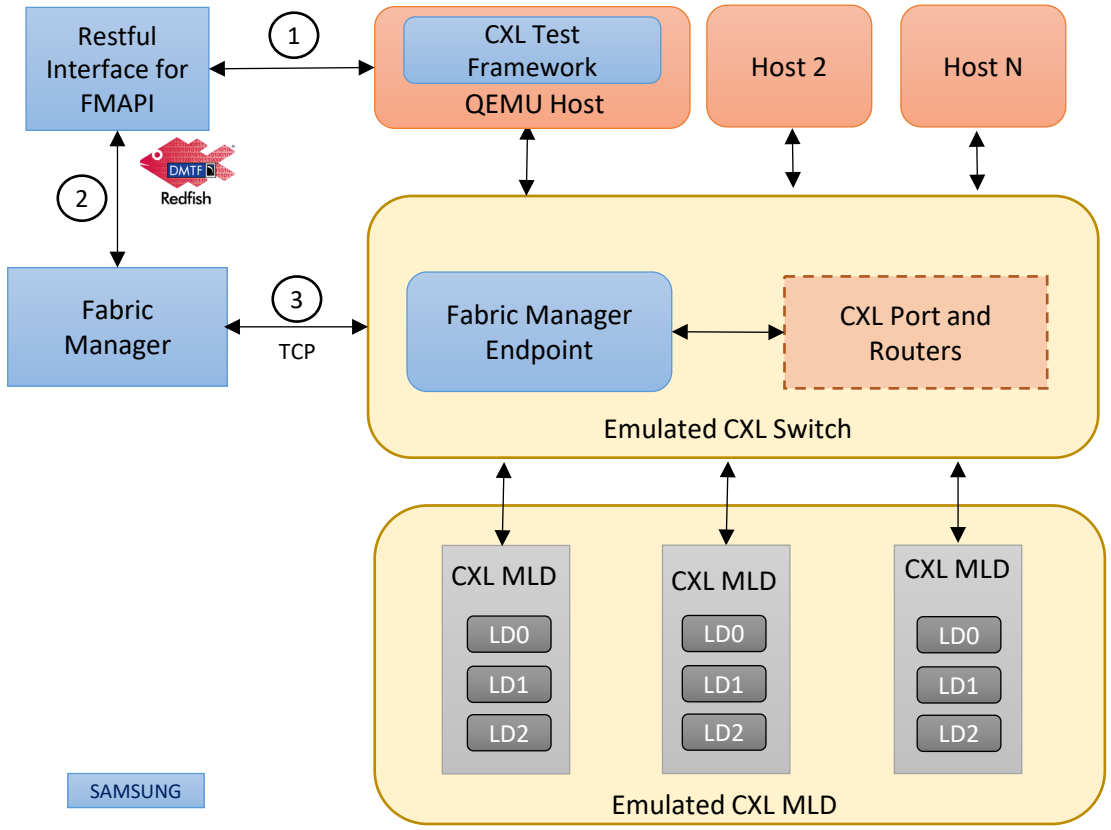


LD – Logical Device, MLD – Multi Logical Devices

1. Host issues *http* request of FM API Command
2. Restful Interface calls FM command based on host request
   - Restful Interface does mapping of Host Request to FM Command Set
   - Interface could be In-house development based on custom needs

OpenCIS based MH-MD Ecosystem View

- Socket communication establishment and FM Endpoint write requests

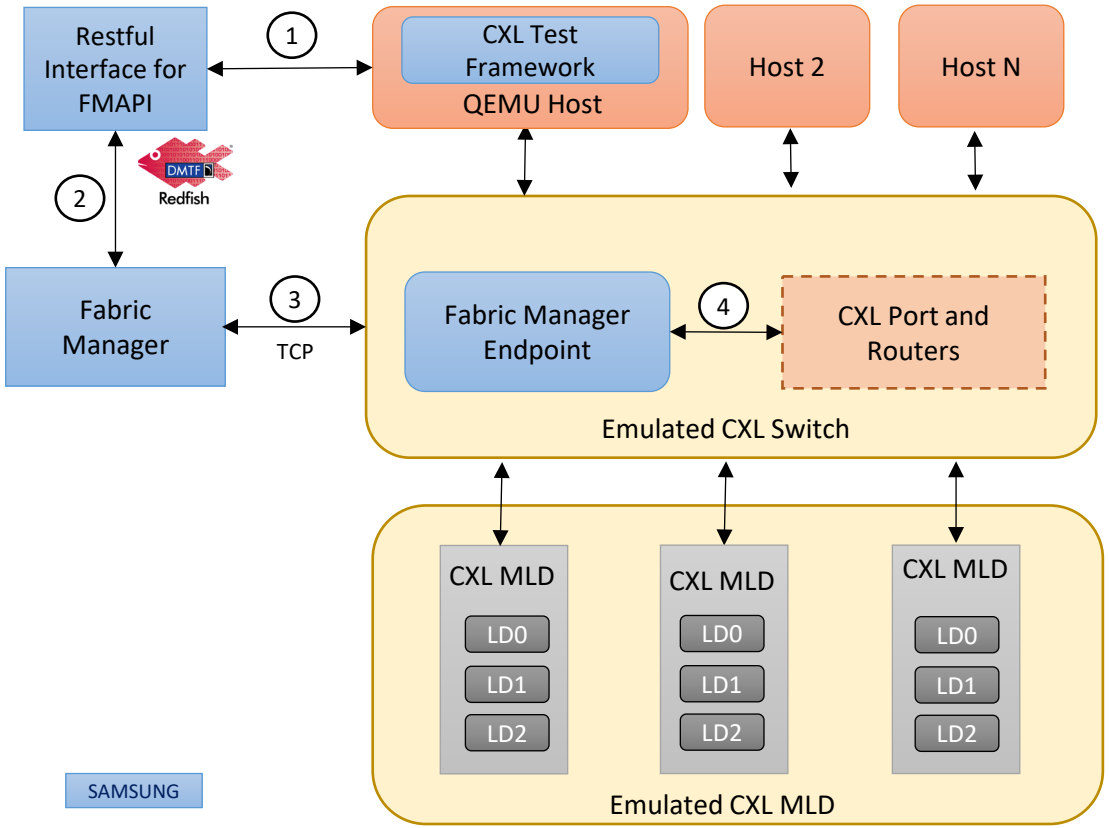


LD – Logical Device, MLD – Multi Logical Devices

1. Host issues *http* request of FM API Command
2. Restful Interface calls FM command based on host request
3. Fabric Manager request to socket ops to FM endpoint
   - Socket Writes are requested by FM to FM Endpoint
   - FM Endpoint is part of OpenCIS Emulated Switch

OpenCIS based MH-MD Ecosystem View

- FM Endpoint processing the FM Command, integral part of Emulated Switch

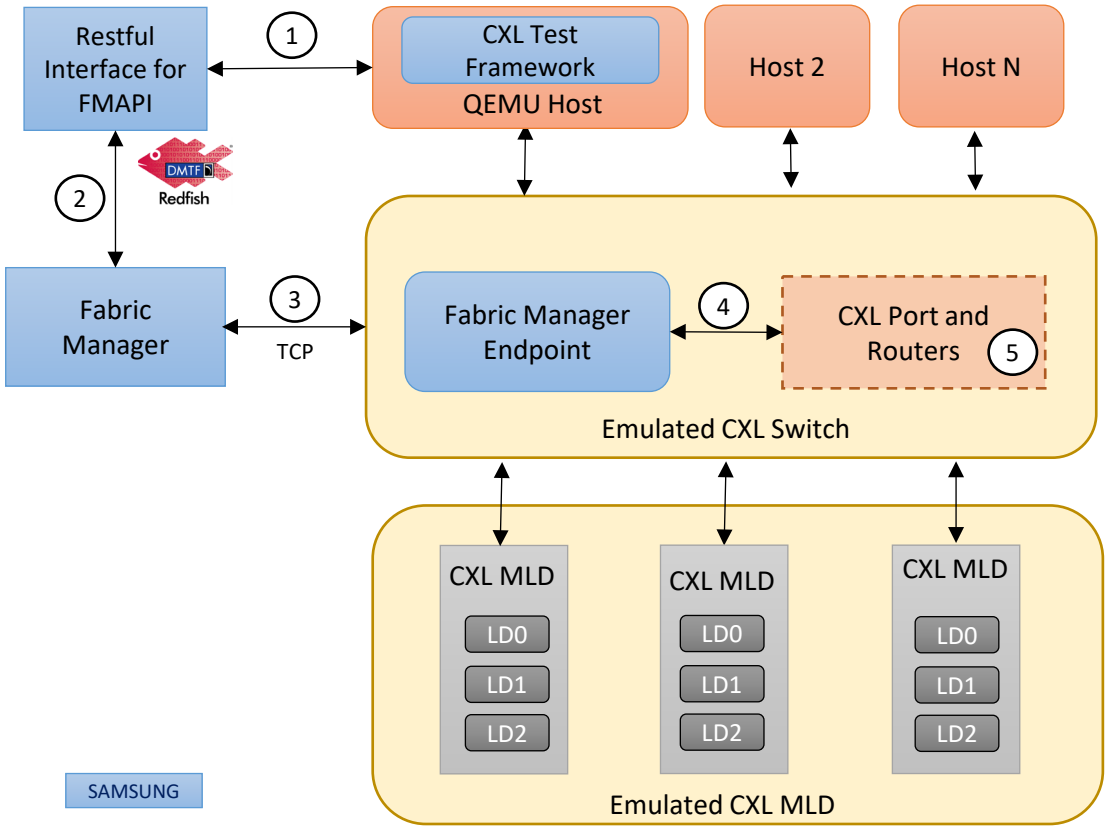


LD – Logical Device, MLD – Multi Logical Devices

1. Host issues *http* request of FM API Command
2. Restful Interface calls FM command based on host request
3. Fabric Manager request to socket ops to FM endpoint
4. Fabric Manager Endpoint processes the FM command
   - Processing of Switch related FM API
   - Interacts with Port Manager for Binding Ops

OpenCIS based MH-MD Ecosystem View

- Managing of bindings inside the CXL Port and Routers Module

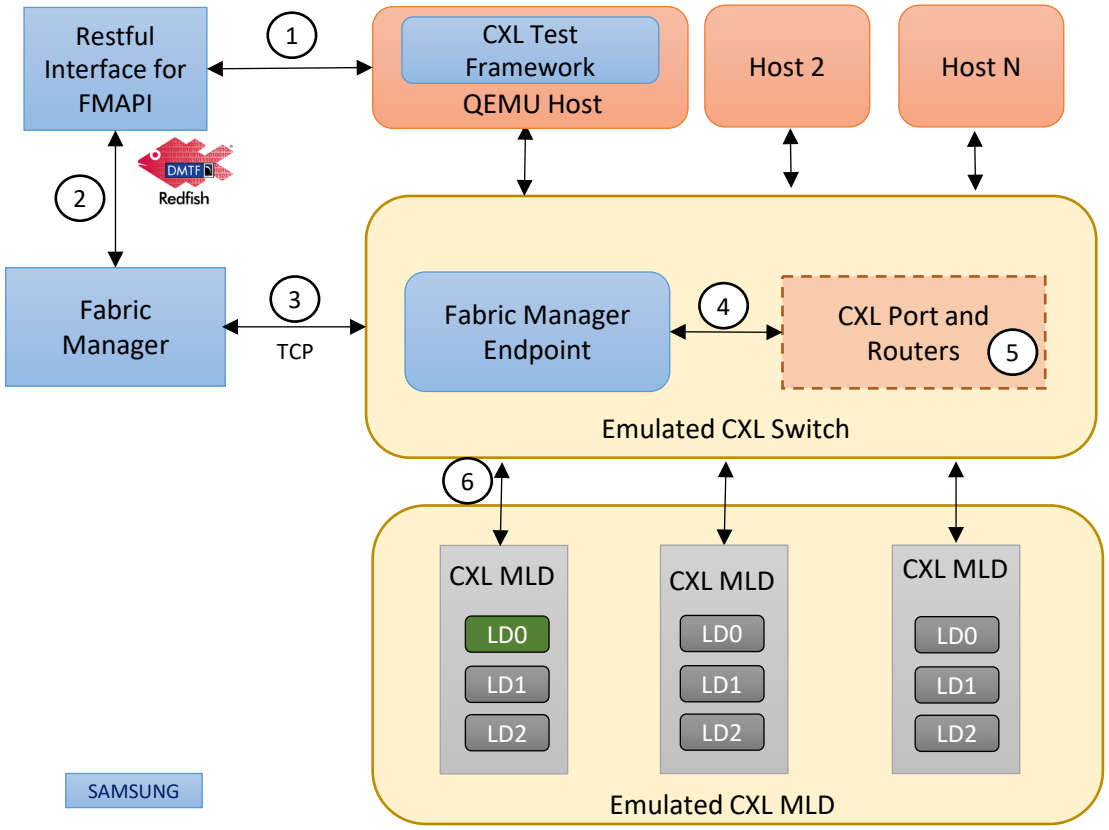


LD – Logical Device, MLD – Multi Logical Devices

1. Host issues *http* request of FM API Command
2. Restful Interface calls FM command based on host request
3. Fabric Manager request to socket ops to FM endpoint
4. Fabric Manager Endpoint processes the FM command
5. Switch managers performs CXL Port Binding
   - Receive bind request from port
   - Establish connection with vPPB and Physical port
   - Routes command through established connection

OpenCIS based MH-MD Ecosystem View

- Establishment of Host Connection with Logic Device from the MLD pool

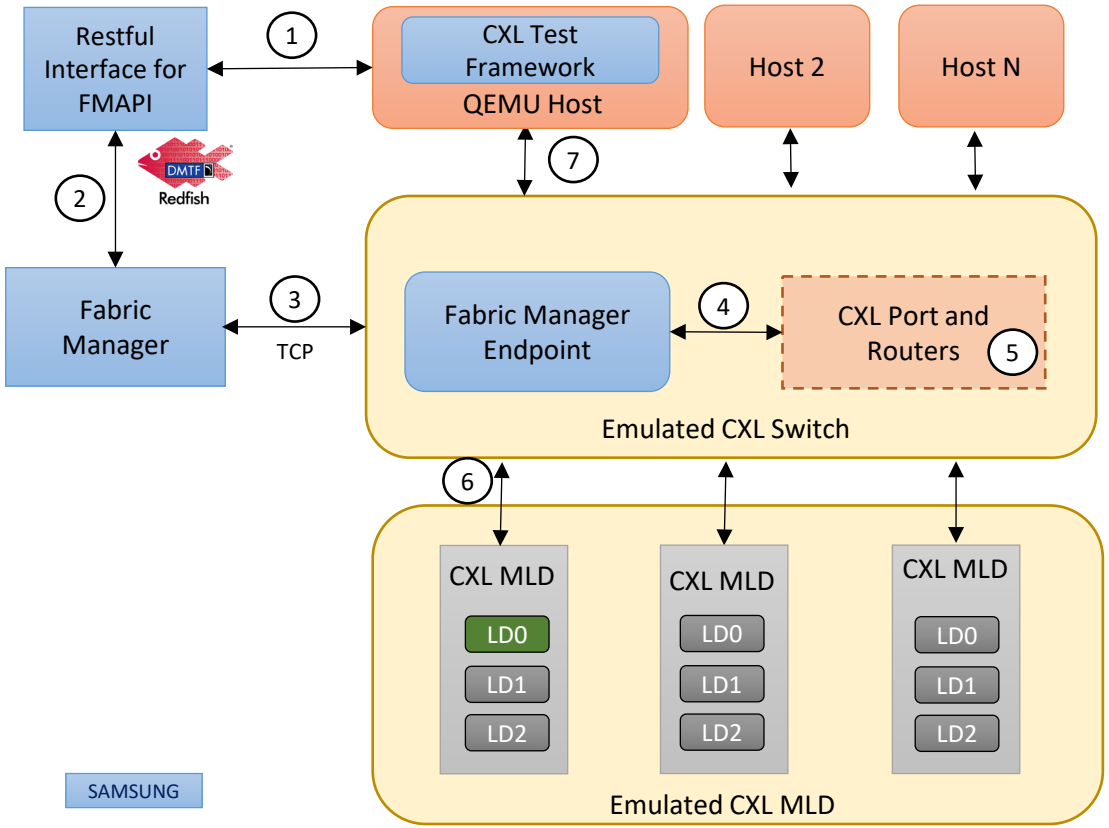


LD – Logical Device, MLD – Multi Logical Devices

1. Host issues *http* request of FM API Command
2. Restful Interface calls FM command based on host request
3. Fabric Manager request to socket ops to FM endpoint
4. Fabric Manager Endpoint processes the FM command
5. Switch managers performs CXL Port Binding
6. Switch connection manager receives request to establish connection with Logical Device (LD0)

OpenCIS based MH-MD Ecosystem View

- CXL CCI/.mem operations by the Host via Switch (normal flow)



LD – Logical Device, MLD – Multi Logical Devices

1. Host issues *http* request of FM API Command
2. Restful Interface calls FM command based on host request
3. Fabric Manager request to socket ops to FM endpoint
4. Fabric Manager Endpoint processes the FM command
5. Switch managers performs CXL Port Binding
6. Switch connection manager receives request to establish connection with Logical Device (LD0)
7. Further Host can perform CXL operation via CCI / .mem path via Switch

Output of CXL Switch Validation

- Supports Virtual Switch Command Set & Physical Switch Command Set
- Support of Virtual to Physical Binding for Multi Host – Multi Device Ecosystem

HTTP Request for LD Bind (Virtual Switch CMD)

```
curl -X POST
"http://10.0.2.2:5000/redfish/v1/Fabrics/0/CXL/Switches/0/VCS/1/Bind
" -d "vppbId=2" -d "physicalPortId=0" -d "ldId=3"
```

Response

```
{"error":"","result":"SUCCESS"}
```

```
[BindVppbCommand] Only DSP port can bind to vPPBHost bind
notification root_port 2, device_vppb 2, val 545
[SYS-SW] Received Binding Root Port: 2, vPPB: 2, val 545 from FM,
vPPB: 2[FabricManagerSocketIoServer] Response: {'error': '', 'result':
'SUCCESS}
```

Example for Bind request from host to bind the Vppb ID :2 and Physical port ID :0 with LD ID:3

HTTP Request for Identify Switch (Physical Switch CMD)

```
curl -X POST
"http://10.0.2.2:5000/redfish/v1/Fabrics/0/CXL/Switches/0”
```

Response

```
{'activePortBitmask': 7,
'activeVcsBitmask': 3,
'ingressPortId': 0,
'numBoundVppbs': 1,
'numHdmDecoders': 32,
'numPhysicalPorts': 2,
'numVcss': 2,
'totalNumVppbs': 4}
```

Example for Request for Identify Switch Command

Output of CXL CCI Validation

- Identify command response view of Emulated Device
- Memory read and Memory Write operation using ***cxl-util*** tool

```
[00:00:01:099] [LOG_MSG] CXLPython : Get Query Response Code
LOG_INFO ----------
LOG_MSG ----------
LOG_MSG Script Version Information (Dates in DD-MM-YYYY)
LOG_MSG ----------
LOG_MSG Test Script Last Modified : 29-12-2023
LOG_MSG Library Last Modified : 09-06-2023
LOG_MSG Library verion : 1.08
LOG_MSG ----------
LOG_INFO ----------
LOG_INFO CALL RUN
LOG_INFO STEP 1: Execute Identify Memory Device Command, validate return code Success
LOG_MSG Port ID[self.cxl_base_port] command Completed with Status Code: 0x0  as expected
LOG_INFO Identify Memory device command PASSED
LOG_INFO STEP 2: log Output payload fields and values
LOG_INFO Identify memory device output payload fields and values
LOG_INFO FW Revision             : EEUM EMU 1.0
LOG_INFO Total Capacity          : 1
LOG_INFO Volatile Only Capacity  : 1
LOG_INFO Persistent Only Capacity : 0
LOG_INFO Partition Alignment     : 0
LOG_INFO Informational Event Log Size : 1
LOG_INFO Warning Event Log Size  : 1
LOG_INFO Failure Event Log Size  : 1
LOG_INFO Fatal Event Log Size    : 1
LOG_INFO LSA Size                : 0
LOG_INFO Inject Poison Limit     : 0
LOG_INFO Poison Handling Capabilities : 0
LOG_INFO QoS Telemetry Capabilities   : 0
LOG_INFO RUN PASSED
LOG_INFO ----------
LOG_INFO CALL TEARDOWN
LOG_INFO TEARDOWN PASSED
LOG_INFO ++++++++++++++++++++++++++++++++++++
```

Memory Write Command

```
./cxl-util mem write 0 0x40
0x4f70656e43584c20204f70656e43584c4f70656e43584c20204f70656e43584
```

Response

```
Starting Timestamp: 2024-09-27 02:24:07.848844
CXL-Host[Port0]: Start CXL.mem Write: addr=0x40
data=0x4f70656e43584c20204f70656e43584c4f70656e43584c20204f70656e43584
CXL-Host[Port0]: CXL.mem Write success
```

Memory Read Command

```
./cxl-util mem read 0 0x40
```

Response

```
Starting Timestamp: 2024-09-27 02:24:39.158851
CXL-Host[Port0]: Start CXL.mem Read: addr=0x40
CXL-Host[Port0]: CXL.mem Read success
Data:
00000000:  4f 70 65 6e 43 58 4c 20 20 4f 70 65 6e 43 58 4c  |OpenCIS..OpenCIS|
00000010:  4f 70 65 6e 43 58 4c 20 20 4f 70 65 6e 43 58
```

Takeaways

- OpenCIS provides stable and evolving ecosystem for Shift Left Readiness of CXL Device and Host Test Infra.
- Reduces dependency on Real Device / ASIC for PoC and Feature Validation
- Platform that combines Simulation strength of QEMU + Flexibility of Python.
- Feature rich ecosystem for CXL - FM API, Mailbox, Memory R/W, Register R/W, and Table Access DOE and CXL.mem.
- Simplifies Development of CXL components for *Rapid* prototype of new features
- More @ https://www.opencis.io/

Q & A

Thank You!