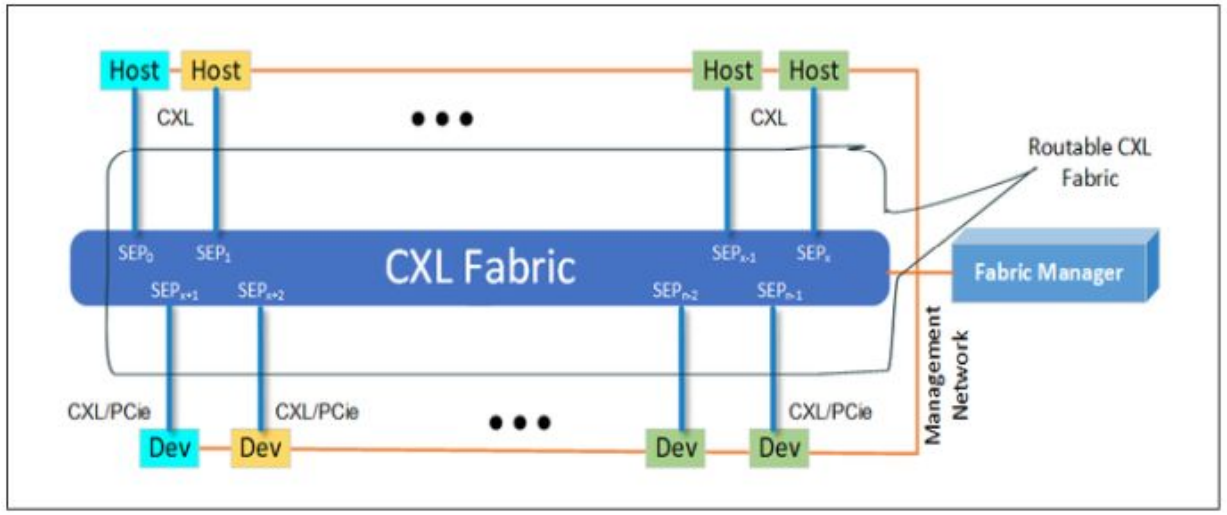
CXL FABRIC MANAGER (FM) ARCHITECTURE

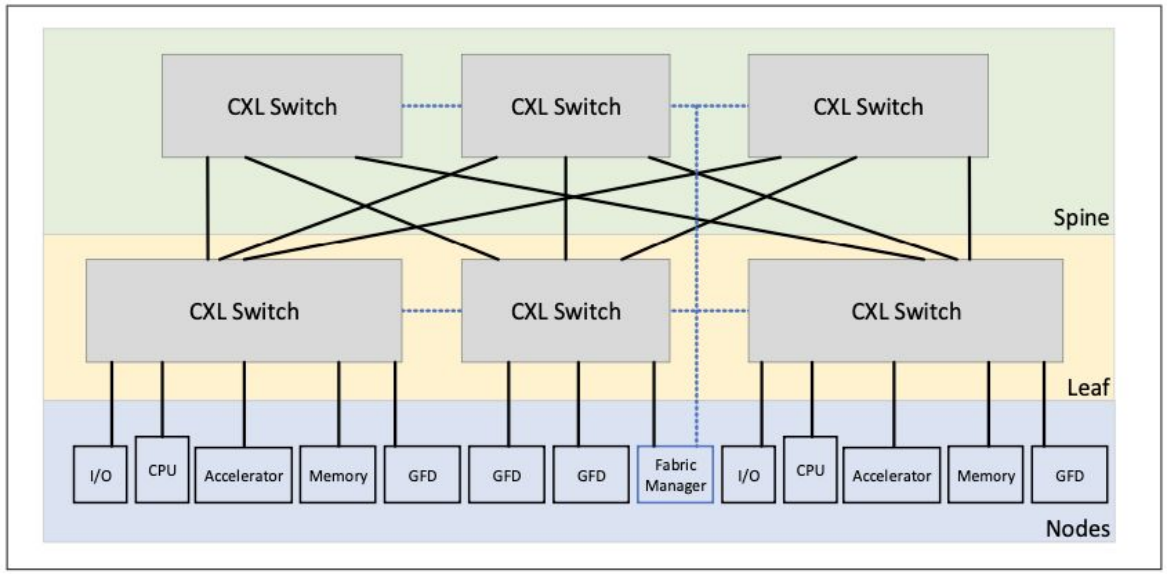
Viacheslav Dubeyko <slava@dubeyko.com>

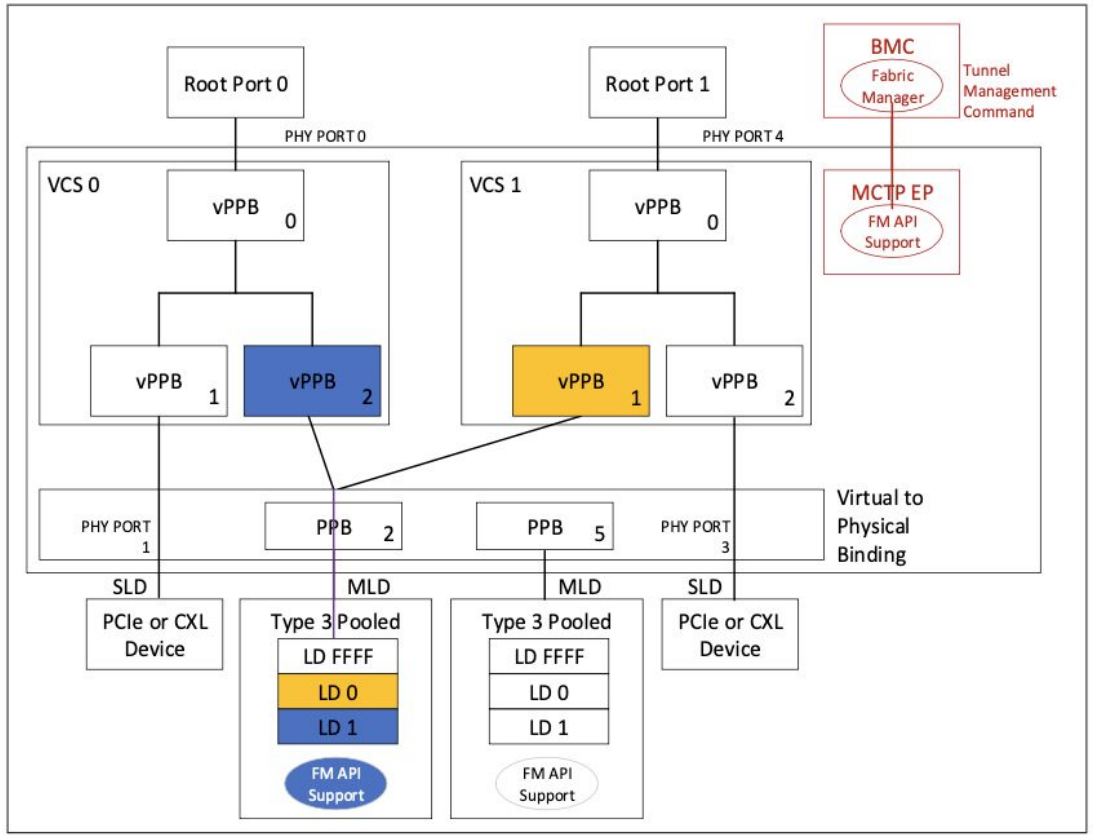
Content

1. Architecture cases.
2. FM Infrastructure.
3. CXL FM configuration tool.
4. CXL FM daemon.
5. Orchestrator
6. Current status.
7. Open questions.

What is Fabric Manager (FM)?





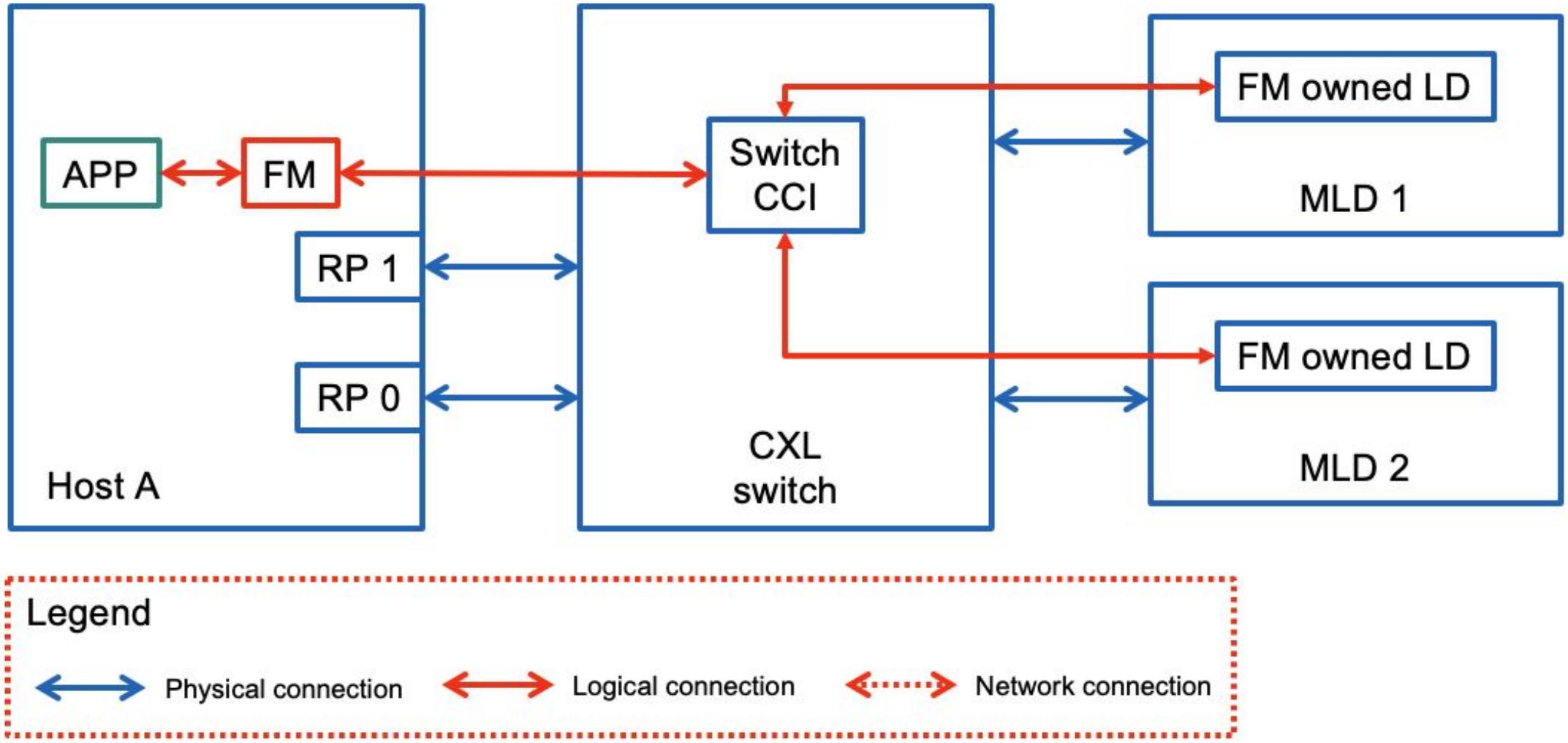


What is Fabric Manager (FM)?

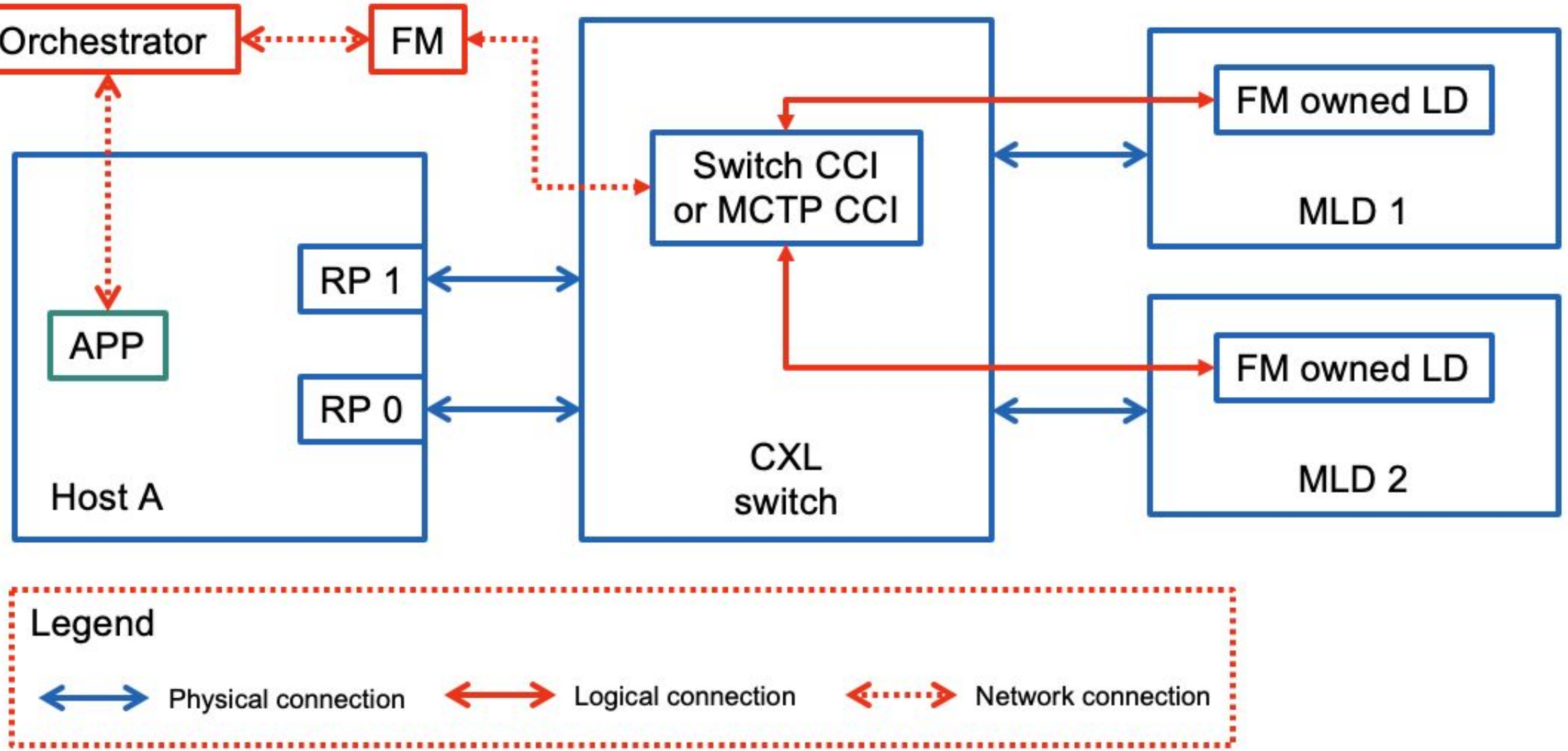
CXL Specification 3.0 defines Fabric Manager as:

“CXL devices can be configured **statically** or **dynamically** via a **Fabric Manager** (FM), an external logical process that queries and configures the system’s operational state using the FM commands defined in this specification. The FM is defined as the **logical process** that decides when reconfiguration is necessary and initiates the commands to perform configurations. It can take any form, including, but not limited to, **software** running on a **host machine**, embedded software running on a **BMC**, embedded **firmware** running on another CXL device or CXL switch, or a **state machine** running within the CXL device itself.”

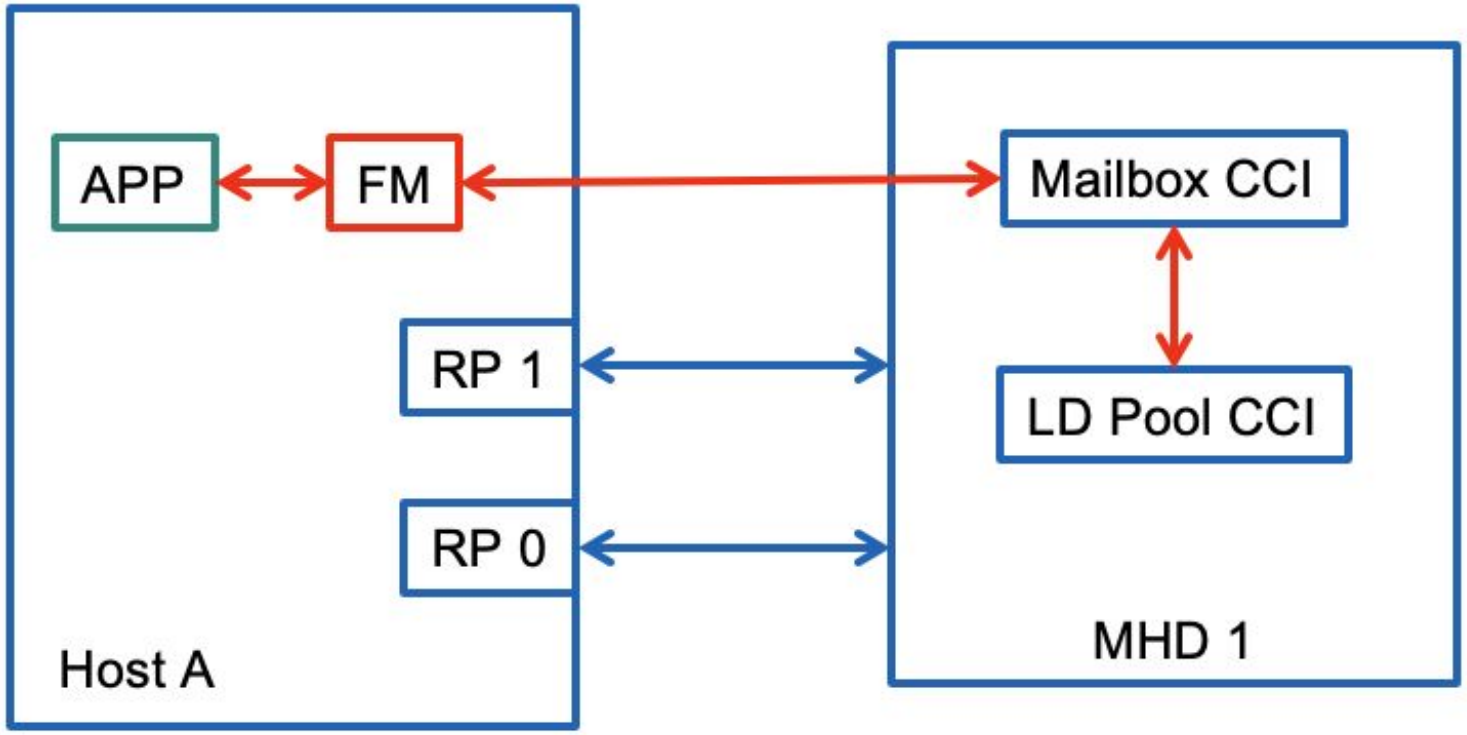
Single host with FM

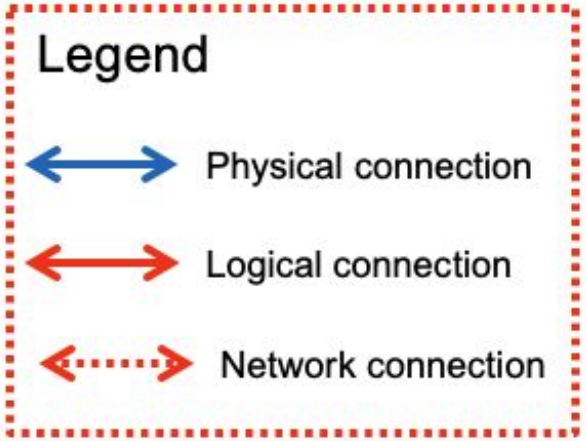


Single host with Orchestrator

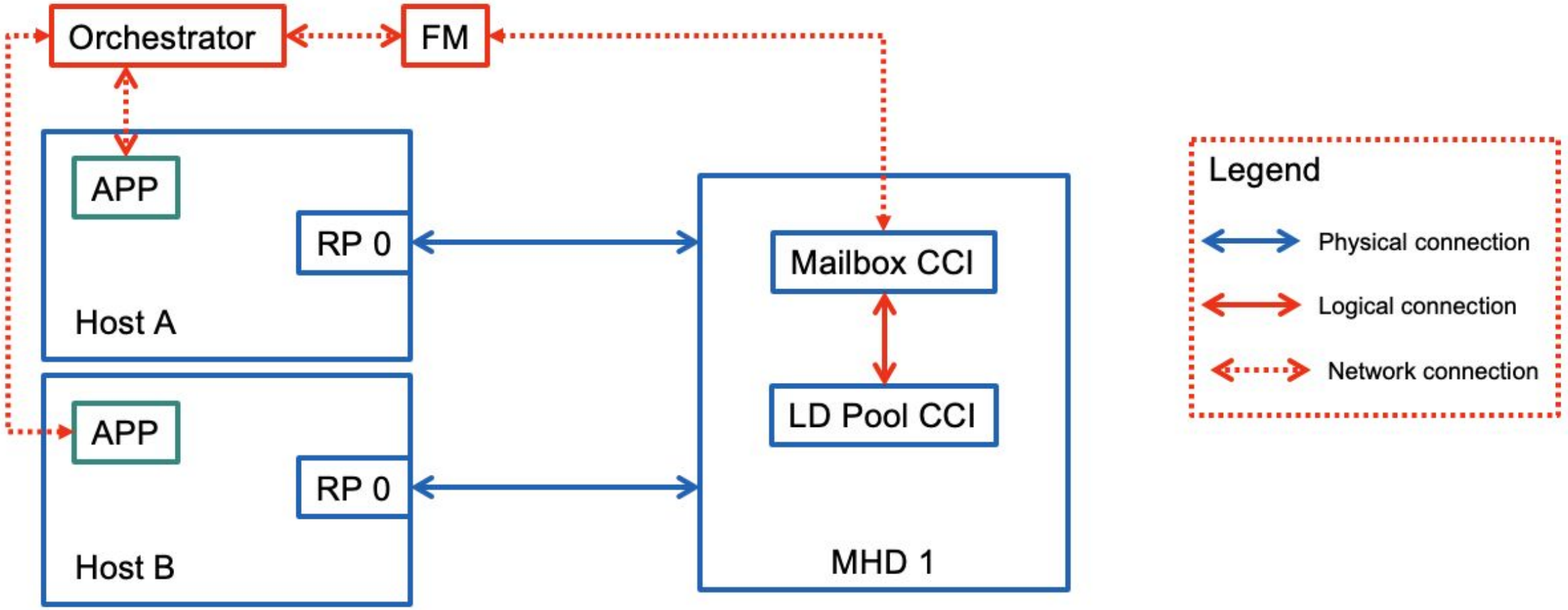


Multi-headed device

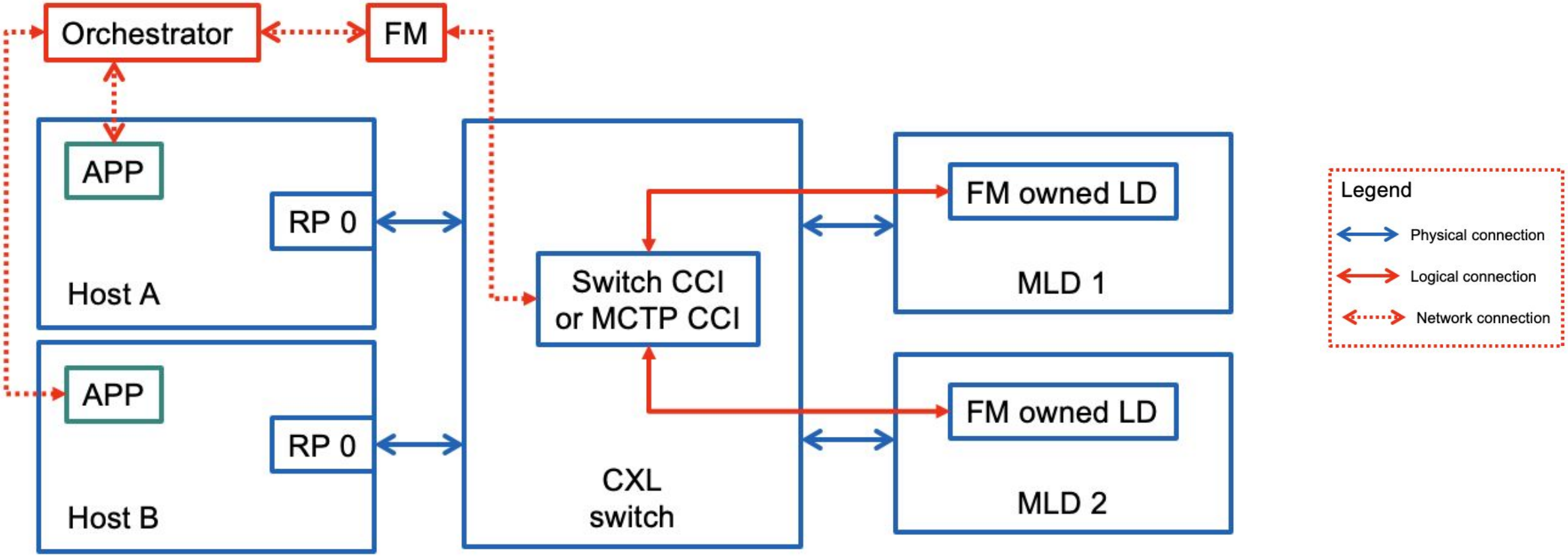




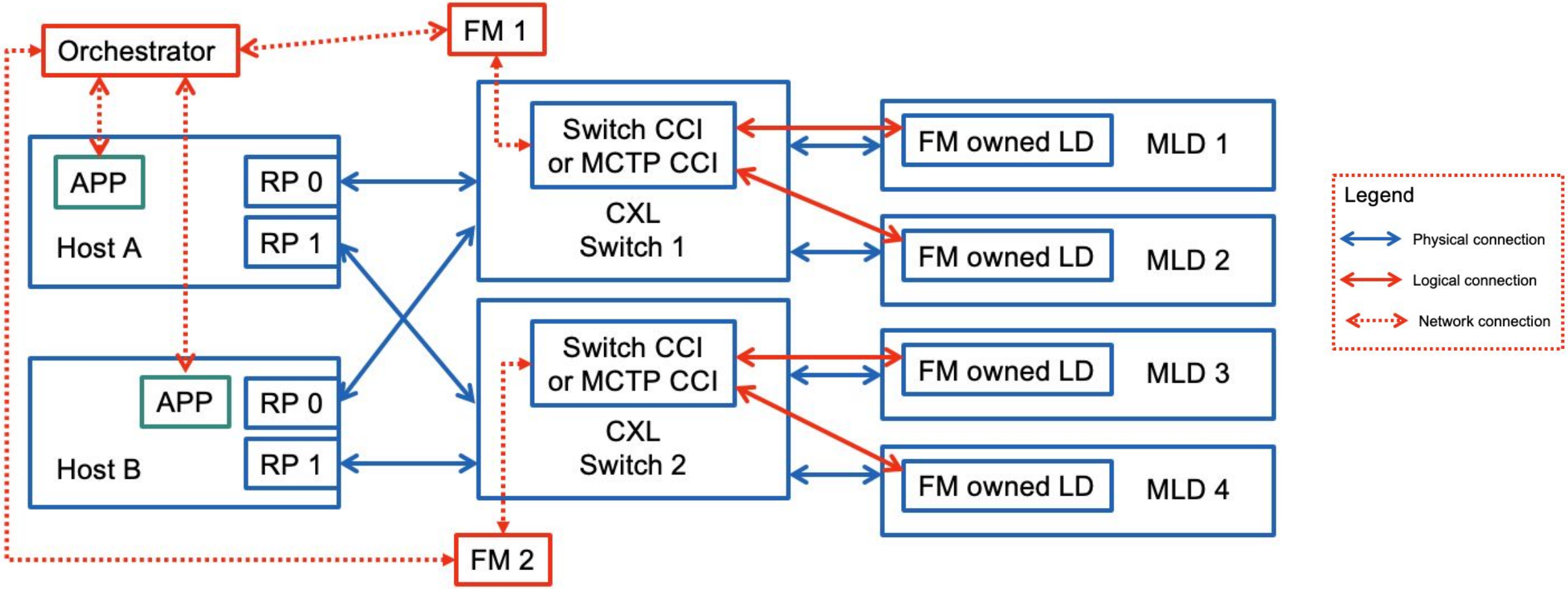
Multi-headed device with multiple hosts



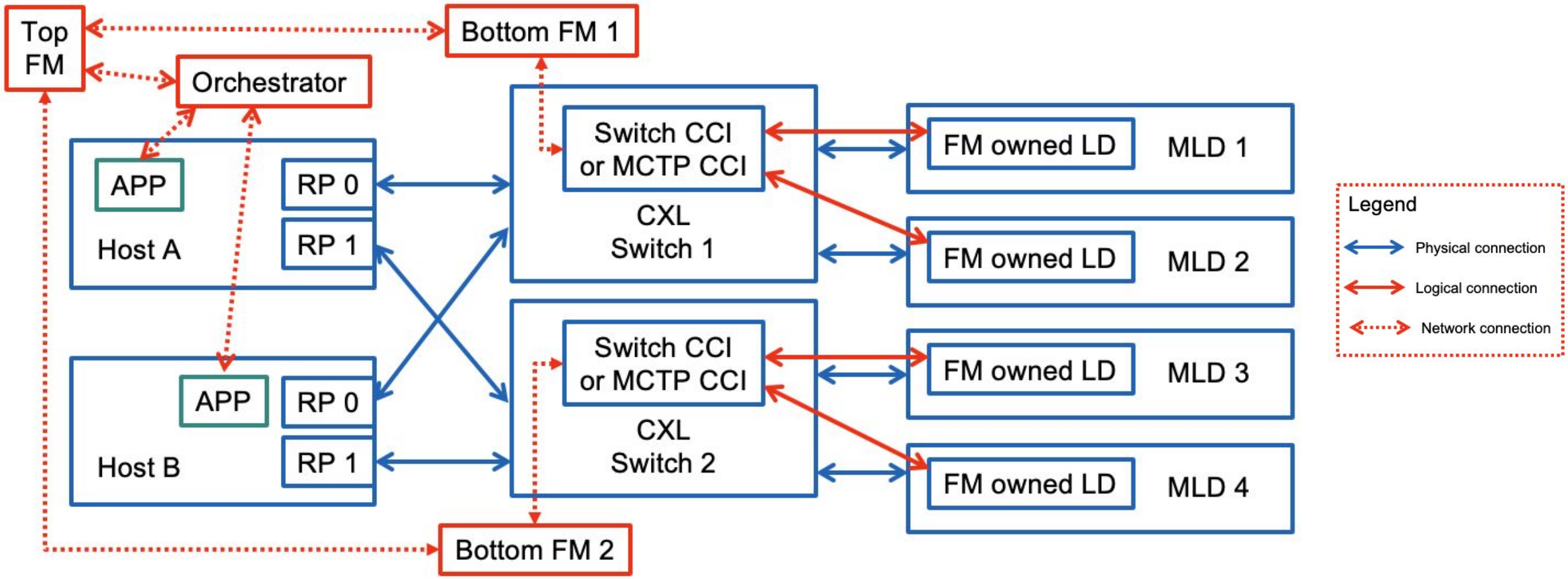
Multiple hosts with FM



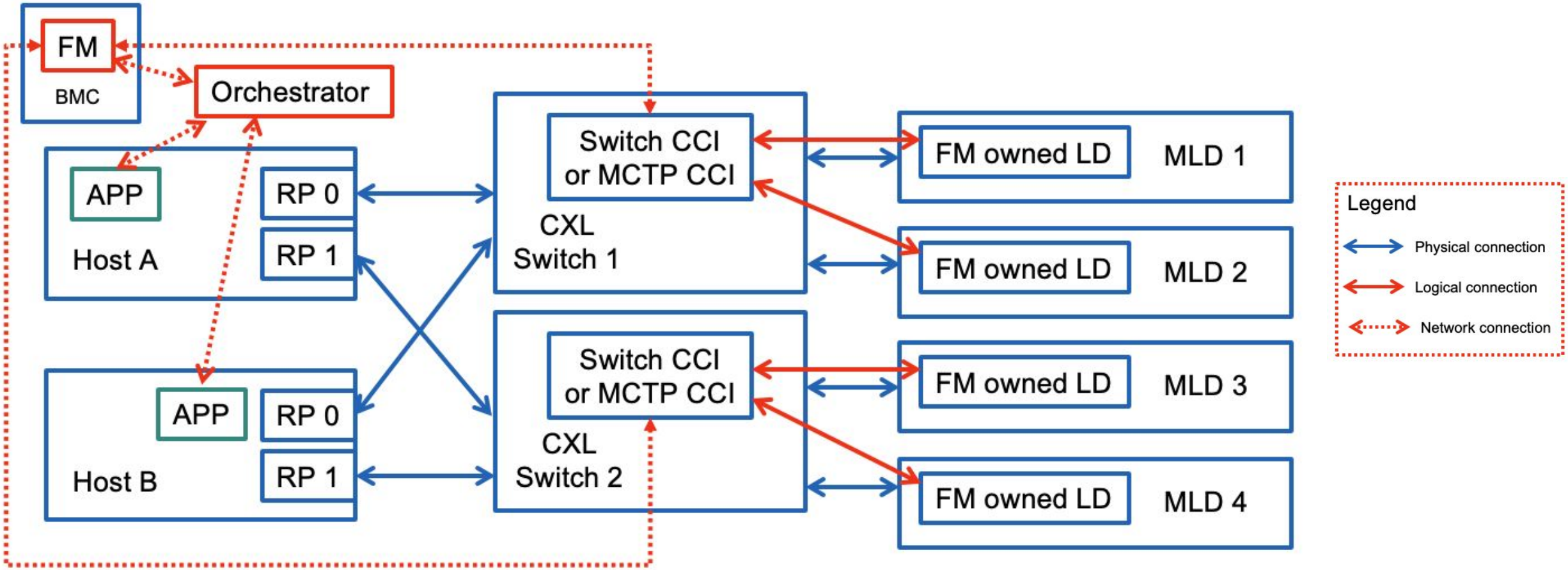
Distributed FM



Layered FM and separate Orchestrator



BMC based FM


FM
BMC
Orchestrator
APP
RP 0
RP 1
Host A
APP
RP 0
RP 1
Host B
Switch CCI
or MCTP CCI
CXL
Switch 1
Switch CCI
or MCTP CCI
CXL
Switch 2
FM owned LD
MLD 1
FM owned LD
MLD 2
FM owned LD
MLD 3
FM owned LD
MLD 4
Legend
Physical connection
Logical connection
Network connection

FM Infrastructure

FM infrastructure can include:

1. FM CLI tool
2. FM daemon
3. Orchestrator daemon
4. CXL device/switch or QEMU emulation

FM CLI tool responsibilities

- Configure Fabric Manager (FM)
- Manage CXL switch(es) and CXL device(s) by means of Fabric Manager (FM)
- Issue requests to Fabric Manager by means of FM API
- Extract event records
- Extract errors details

FM CLI commands:

- Discover available agents
- Manage Fabric Manager
- Manage CXL switch
- Manage logical devices
- Manage PCI-to-PCI bridge
- Manage physical ports
- Manage Multi-Logical Devices
- Manage Dynamic Capacity Devices

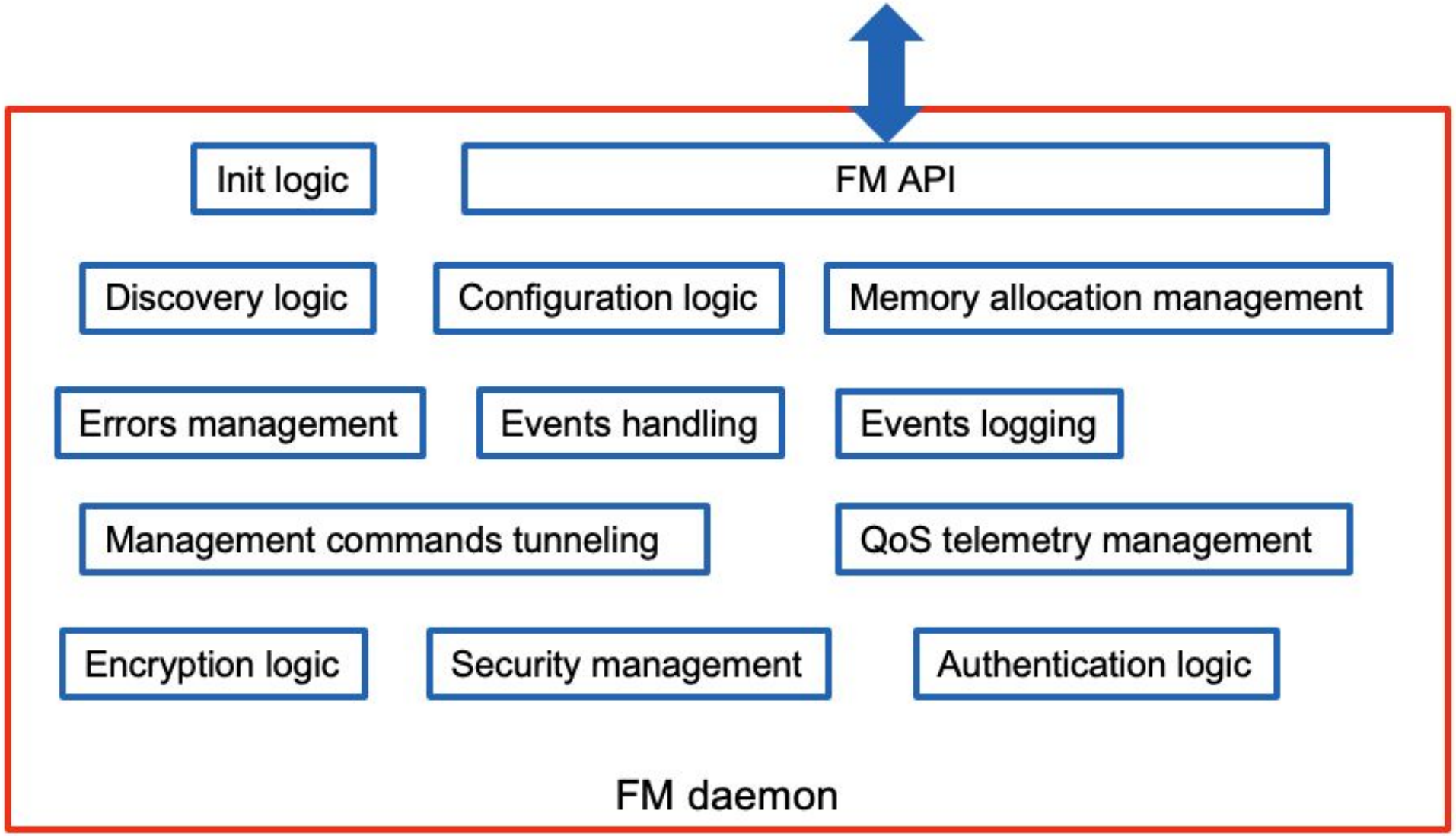
FM daemon responsibilities

- Discover CXL switches and CXL devices
- Query CXL switch information and configuration details
- Bind or unbind ports
- Discover Multi-Logical Devices
- Logical devices binding/unbinding
- Register to receive and handle event notifications from the CXL switch
- Memory allocation management
- QoS telemetry management
- Security management (Logical Device erasure after unbinding, for example)
- Error handling

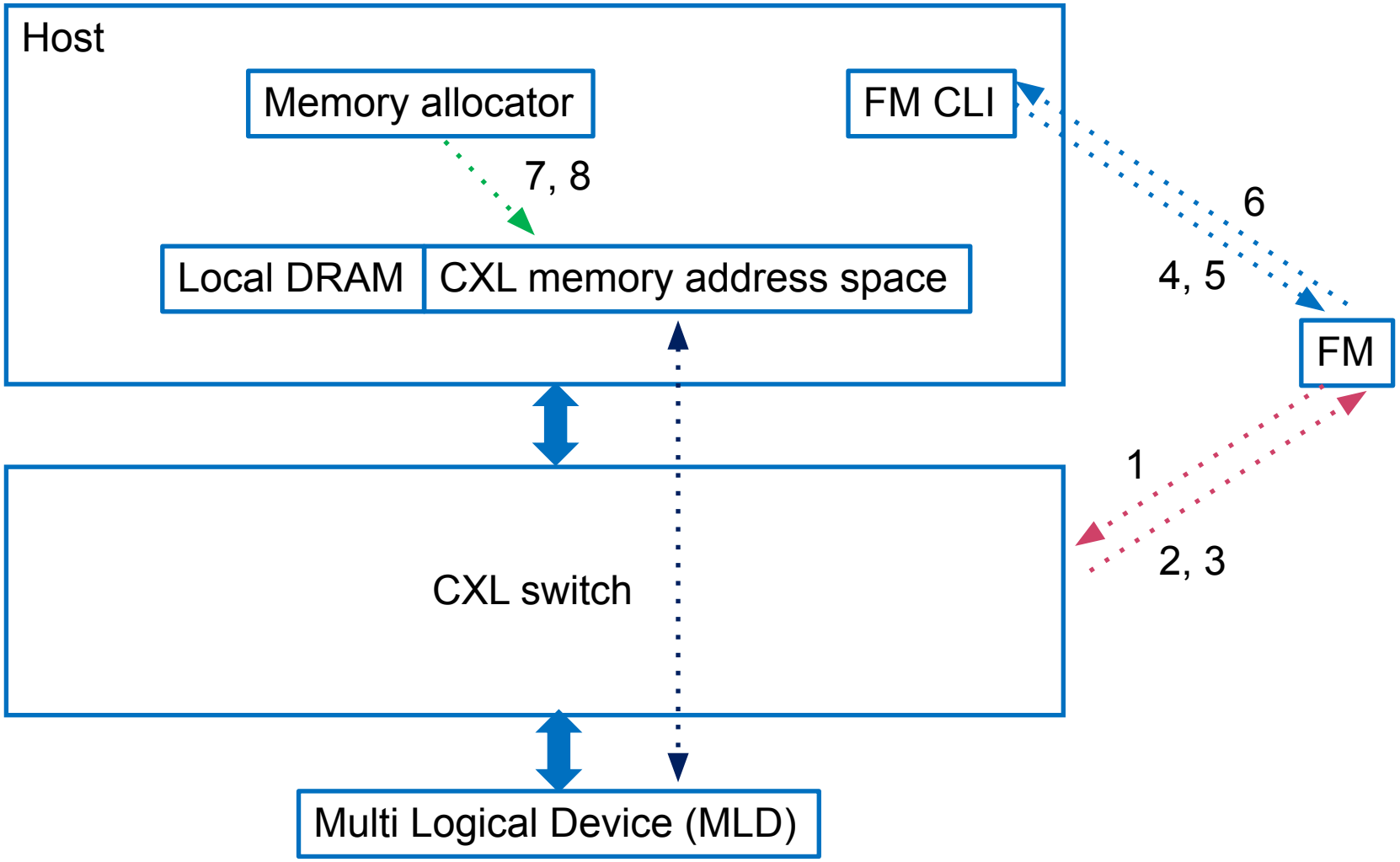
FM API Commands (defined by CXL Specification 3.0)

1. **Physical switch** (Identify Switch Device, Get Physical Port State, Physical Port Control, Send PPB (PCI-to-PCI Bridge) CXL.io Configuration Request).
2. **Virtual Switch** (Get Virtual CXL Switch Info, Bind vPPB (Virtual PCI-to-PCI Bridge), Unbind vPPB, Generate AER (Advanced Error Reporting Event).
3. **MLD Port** (Tunnel Management Command, Send LD (Logical Device) or FMLD (Fabric Manager-owned Logical Device) CXL.io Configuration Request, Send LD CXL.io Memory Request).
4. **MLD Components** (Get LD (Logical Device) Info, Get LD Allocations, Set LD Allocations, Get QoS Control, Set QoS Control, Get QoS Status, Get QoS Allocated Bandwidth, Set QoS Allocated Bandwidth, Get QoS Bandwidth Limit, Set QoS Bandwidth Limit).
5. **Multi-Headed Devices** (Get Multi-Headed Info).
6. **DCD (Dynamic Capacity Device) Management** (Get DCD Info, Get Host Dynamic Capacity Region Configuration, Set Dynamic Capacity Region Configuration, Get DCD Extent Lists, Initiate Dynamic Capacity Add, Initiate Dynamic Capacity Release).

FM daemon architecture



Memory management



1. Discovery
2. Get info
3. Get virtual switch info
4. Bind vPPB
5. Set region configuration
6. Success/failure notification
7. Allocate memory
8. Free memory

Orchestrator responsibilities

- Discover and keep knowledge about all existing FM instances in management network
- Manage FM instances
- Service client requests (FM CLI tool)
- Deliver client requests to FM instances

Current status

1. Initial state of FM CLI tool
    - Rust implementation
    - Accepts CLI commands
    - Send command to FM daemon through network (considering to use Redfish)
2. Initial state of FM daemon
    - Rust implementation
    - Receive commands (and do nothing, currently)
    - Communication with QEMU emulated CXL switch by means of Component Command Interface (CCI) is in progress (under implementation)

https://github.com/computexpresslink/cxl-fabric-manager.git

Open Questions

1. Does architecture make sense? Are we going in right direction?
2. Rust implementation?
3. Should Fabric Manager (FM) be represented as software daemon?
4. Which FM implementation makes more sense: (1) software on host machine, (2) embedded software on BMC, (3) embedded firmware on CXL switch, (4) state machine within CXL device?
5. Do we need open-source FM implementation with potential presence of proprietary implementation?
6. Any known efforts to implement Fabric Manager (FM)?
7. How reasonable is Redfish as network transport for Fabric Manager (FM)?
8. How probable will be the complex hierarchy of CXL switches?
9. Would anybody like to join this open-source implementation?


THANK YOU
QUESTIONS???