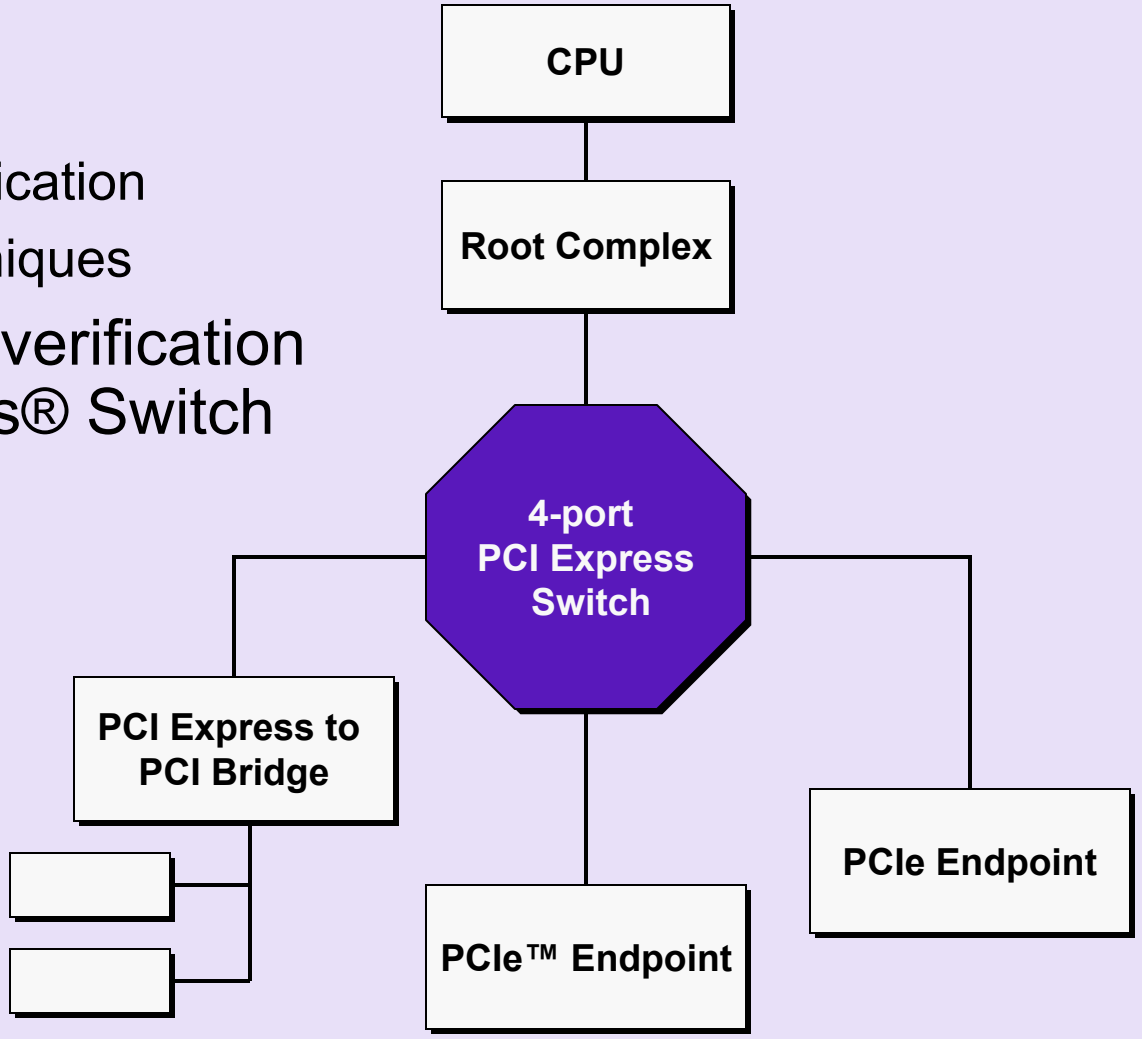

PCI
SIG®

Modeling Techniques for Efficient Verification of a PCI Express® Switch

Asad Khan
Roy Wojciechowski, MGTS (Instructor)
Scott Morrison (Instructor)
Pradip Thaker, PhD
Henry Angulo, SMTS

Purpose

- We will present
  - ✓ Design for Verification
  - ✓ Modeling Techniques
- For the efficient verification of a PCI Express® Switch



Outline

- PCI Express Switch ASIC Architecture
  - Micro-architecture: Ingress Port Logic (IPL)
  - Micro-architecture: Router
  - Reference Model Example: Ingress Port Logic
  - Reference Model Example: Router
  - Integration of Reference Models at chip-level
  - Conclusion

PCI Express Switch Features

- One upstream port
- Switch configurations supported for number of ports ranging from 2 to 8
- Link width options of x1, x2 and x4
- Support for different link widths on different ports
- Support for peer-to-peer traffic
- Transaction Layer Protocol IPs scalable to 10.0 Gb/sec link throughput

Switch Architecture Topologies

- Shared Bus Switch
- Shared Memory Switch
- Input Buffered Switch
- Output Buffered Switch
- Buffered Crossbar Switch
- Input Buffered Crossbar Switch
- Banyan Switch

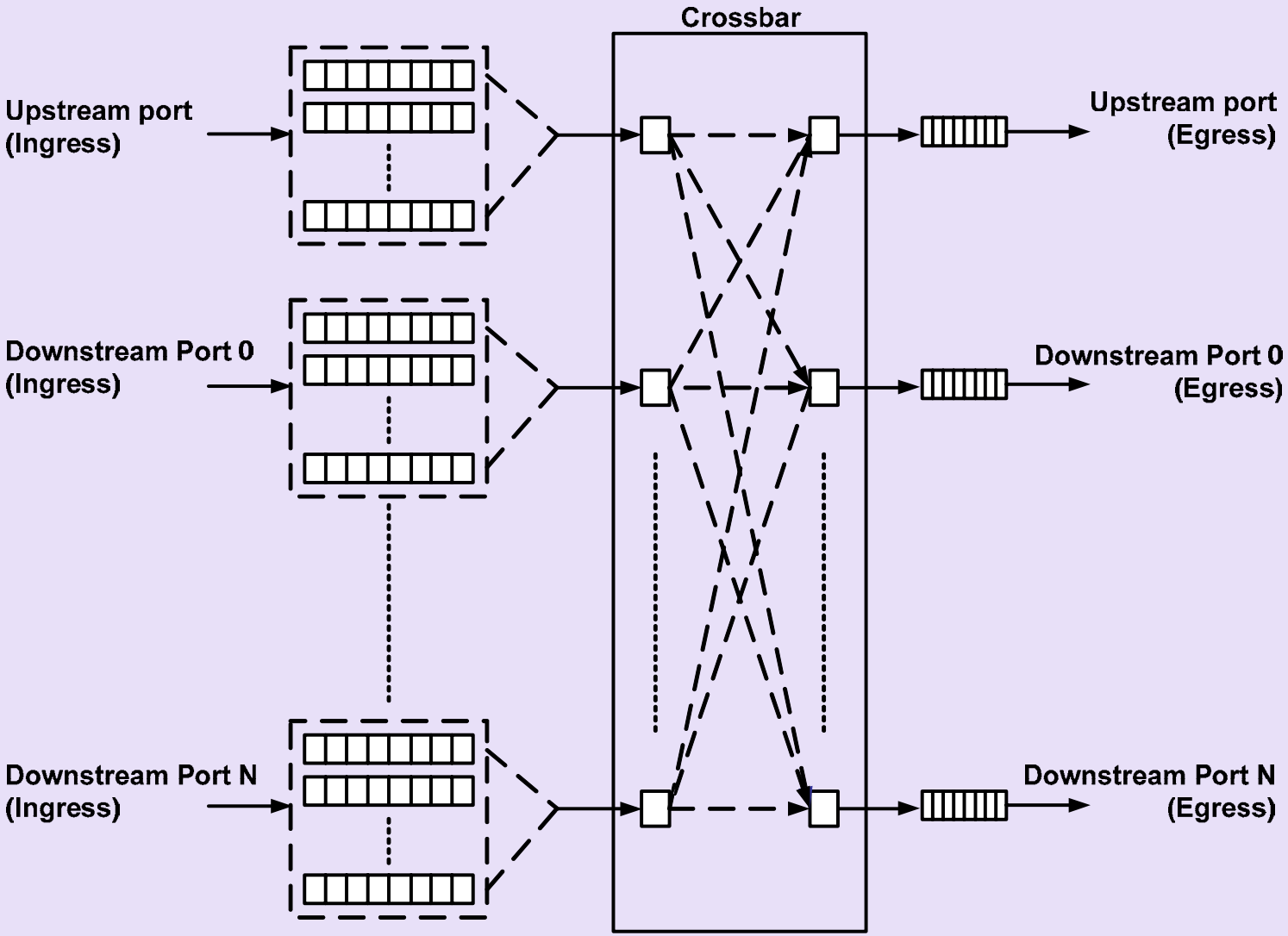
Architecture Concerns

- Flexible/Modular
  - ✓ Minimize die area used
  - ✓ Support changing final topology requirements
  - ✓ Support future topologies with minimal design effort and maximum reuse
  - ✓ Design for verification
    - Functional partitioning to reduce chip-level verification complexity
    - Standardized module interfaces to promote verification component reuse
    - Simulate complex functions at the module-level
      - Easier to set up stimulus
      - Smaller model/faster run time
      - Easier to debug problems

Architecture Chosen

- Input buffered crossbar switch
  - ✓ N:N topology
- Internal crossbar to make data path connections between ports
  - ✓ Support upstream to downstream traffic
  - ✓ Support downstream to upstream traffic
  - ✓ Support peer to peer traffic between downstream ports

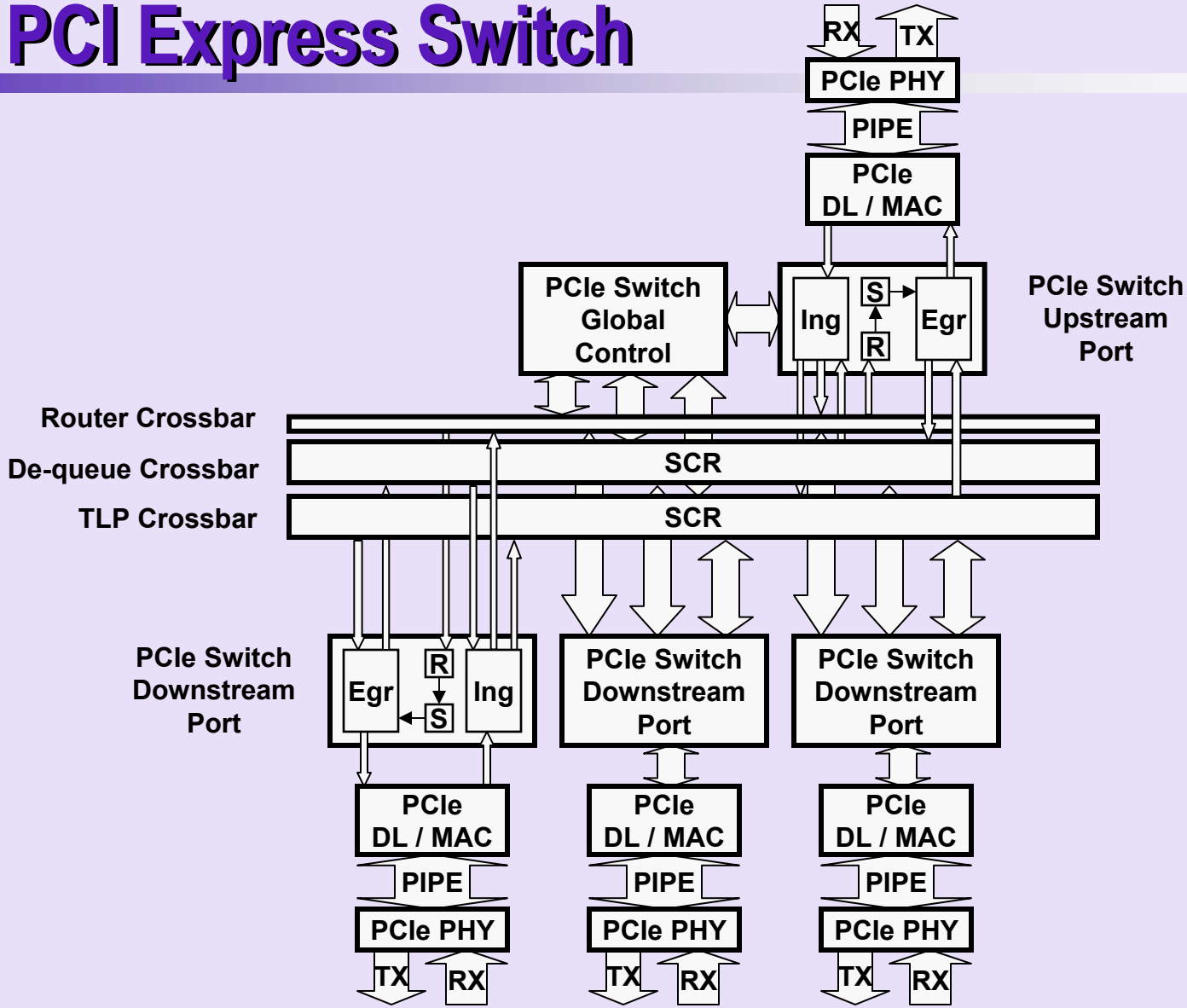
Input Buffered Crossbar Switch



Input Buffered Crossbar Switch (cont.)

- Input TLPs from each port are stored into their respective ingress buffer
- Ingress buffer holds TLPs in input queues for each egress buffer until they are scheduled to be forwarded via a switching crossbar
- Bandwidth requirement at memory port
  - ✓ N+1 for N:N matching (an Input port may have to schedule for all output ports simultaneously)
- Used for medium/large switches with medium/high throughput requirements and medium/high port count
- HOL blocking solved by N input queues for each egress port

PCI Express Switch



Outline

- PCI Express Switch ASIC Architecture
- Micro-architecture: Ingress Port Logic (IPL)
- Micro-architecture: Router
- Reference Model Example: Ingress Port Logic
- Reference Model Example: Router
- Integration of Reference Models at chip-level
- Conclusion

Ingress Memory Architecture Challenges

- Unpredictable TLP stream
  - ✓ Wide Range of TLP lengths
    - As short as 3 double words
    - As long as maximum payload length plus 4 double word header and one double word digest
  - ✓ Any type or length of TLP can be followed by any type or length of TLP
  - ✓ Error or Nullified TLPs
- Two sources of TLPs
  - ✓ External from the Link
  - ✓ Internally generated

IPL Architectural Tradeoff

- Fixed memory allocation
  - ✓ Requirements
    - Buffer per header credit advertised
    - Has to hold longest possible TLP that can be received.
  - ✓ Advantages
    - Reduced design and verification complexity
  - ✓ Disadvantages
    - Extremely large RAM size for total data that a port could hold
    - Large RAMs require too much die area
    - Not a cost effective solution

IPL Architectural Tradeoff (cont.)

- Dynamically linked list managed memory allocation
  - ✓ Requirements
    - Enough memory to hold all the data represented by the credit advertisements
    - Additional memory for “ragged” TLPs (TLPs that only fill part of a memory word and waste the rest of the memory word)
    - Memory lines dynamically allocated to a TLP as it is being received
  - ✓ Advantages
    - Minimizes size of TLP storage memory
  - ✓ Disadvantages
    - Linked list manager design and verification complexities

Outline

- PCI Express Switch ASIC Architecture
- Micro-architecture: Ingress Port Logic (IPL)
- Micro-architecture: Router
- Reference Model Example: Ingress Port Logic
- Reference Model Example: Router
- Integration of Reference Models at chip-level
- Conclusion

PCI Express TLP Routing

- Programming model
  - ✓ PCI Bridge at each port
    - TLP can be claimed and consumed by the bridge
    - TLP can be claimed and passed through the bridge
    - TLP can be ignored by the bridge
  - ✓ A TLP going from port to port will pass through two PCI Bridges
- Switch uses a crossbar connection
  - ✓ TLPs move directly from port to port
  - ✓ TLPs move directly from port to internal logic
  - ✓ Bridges are “virtual”

PCI Express Routing Challenges

- Modular ports require distributed routing to determine destination of incoming TLP
  - ✓ Routing information is broadcast from each port when a TLP is received
  - ✓ Each port has routing logic to claim TLPs destined for the port
  - ✓ Every TLP has to be claimed
    - Even error TLPs
    - Each TLP has to be de-queued from the IPL input buffer

Routing and Verification

- Routing implementation does not look like programming model
  - ✓ Extensive verification needed
  - ✓ Interaction between ports a key challenge
    - Daisy chain connections required between ports
    - Routing takes place in a single clock cycle
    - Routing is the key to implementing the “virtual” PCI bridges

Design for Verification (DFV)

- Minimize side band controls
- Use a standard bus for module interfaces
  - ✓ One BFM needed for busses in module simulations
- Add DFV signals to reduce verification complexity
  - ✓ IPL error interface includes a port indicator field
  - ✓ IPL port indicates when the TLP has an internal or external source
  - ✓ IPL indicates cut-through or store-and-forward state
- Limited variability of pipe-line timing

Outline

- PCI Express Switch ASIC Architecture
- Micro-architecture: Ingress Port Logic (IPL)
- Micro-architecture: Router
- Reference Model Example: Ingress Port Logic
- Reference Model Example: Router
- Integration of Reference Models at chip-level
- Conclusion

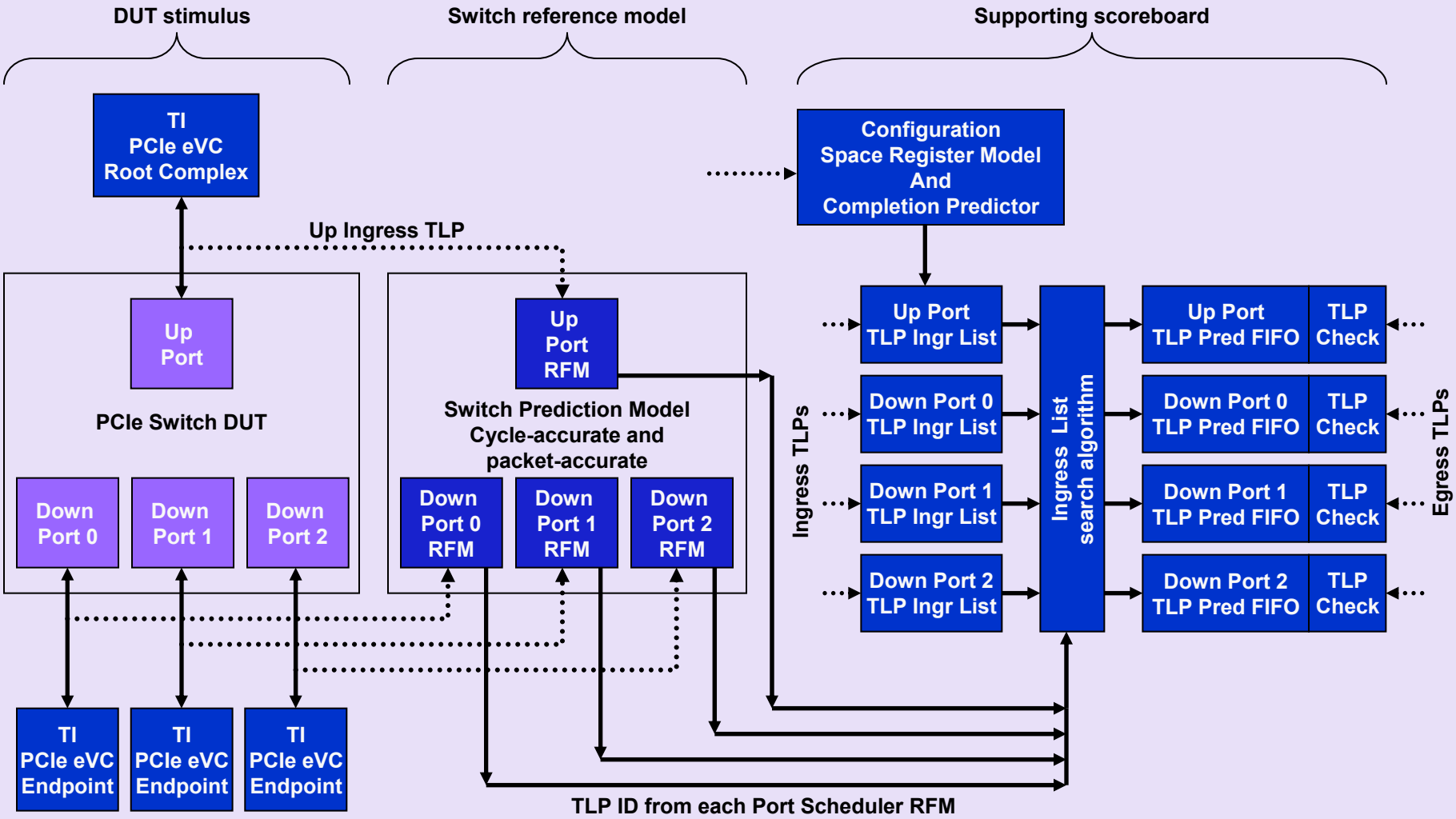
Module and Chip Verification Strategy

- Top-down Methodology
  - ✓ Reusable
  - ✓ Directed-random stimulus
  - ✓ Reference Models (RFM) for automated checking of DUT behavior
- Hybrid approach
  - ✓ Control paths: cycle-accurate modeling
  - ✓ Data paths: packet-accurate modeling
  - ✓ Integration of models for chip-level prediction
  - ✓ Rigorous testing of linked list management, data link layer, routing, and arbitration logic
  - ✓ Some directed testing required

What is a Reference Model?

- A model that is independent of the DUT implementation
- Coded in high-level, human-readable language
- Cycle-accurate prediction where necessary
  - ✓ Because of maintenance overhead
- Able to be co-simulated with the DUT to predict and check the runtime behavior
- The auto-checking RFM+DUT simulation environment makes use of directed-random stimulus
  - ✓ “Let the machine do the work”

Chip-level Prediction Model



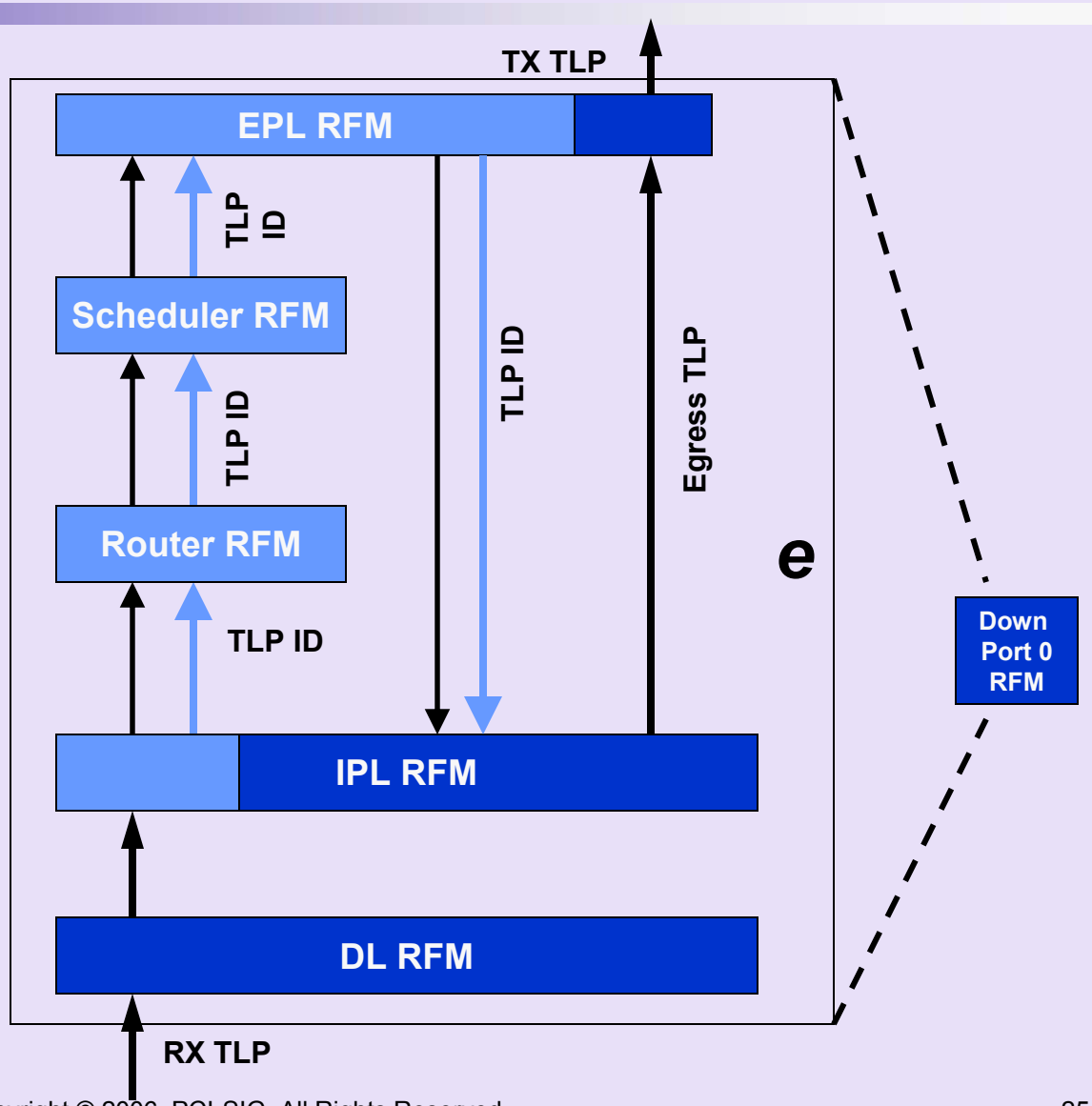
Chip-level Prediction Model: Closer look

PCIe Switch Port Reference Model Predictor

Cycle accurate control path

Each TLP is tracked via a unique TLP ID

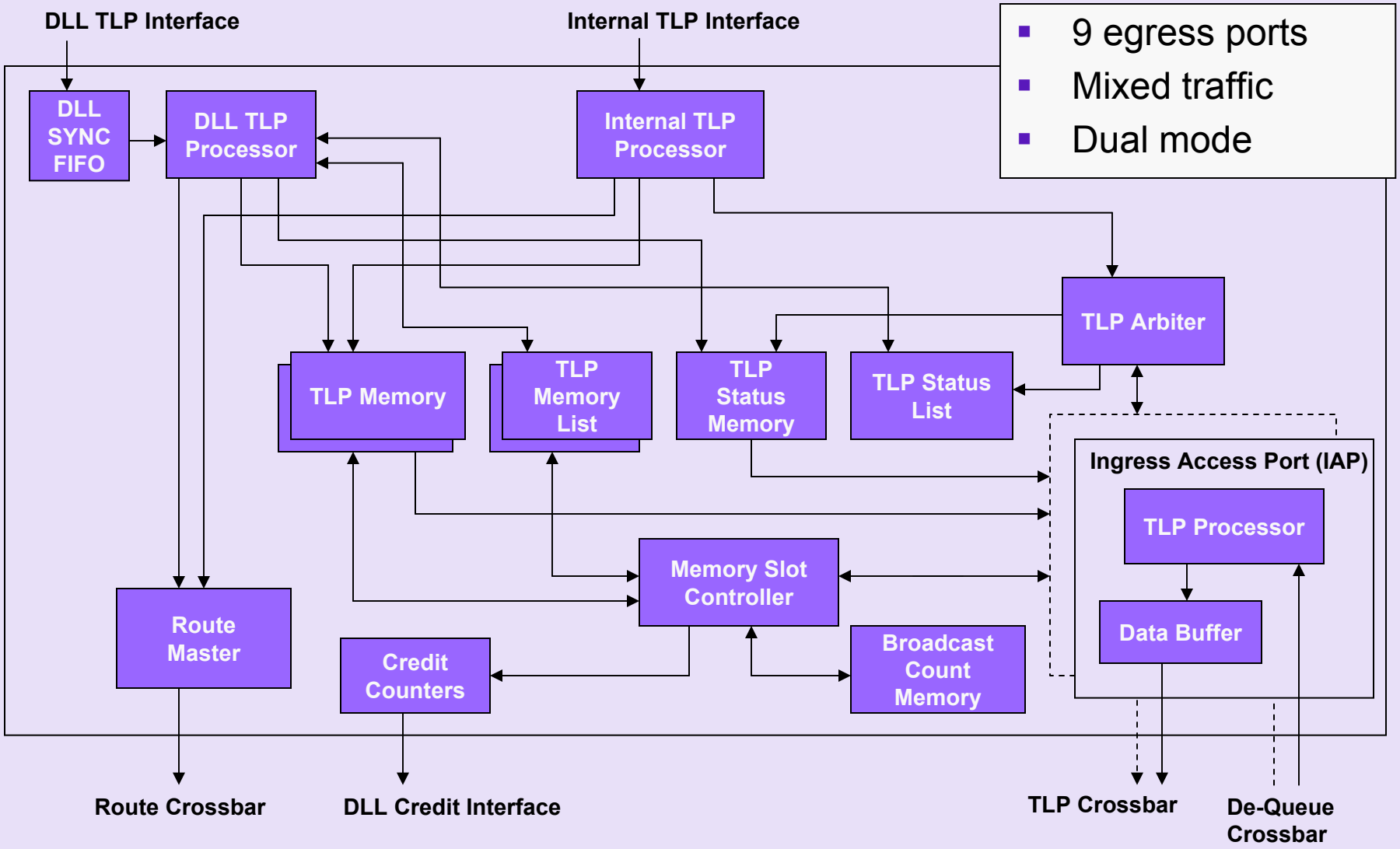
Packet-accurate data path



IPL: Verification Challenges

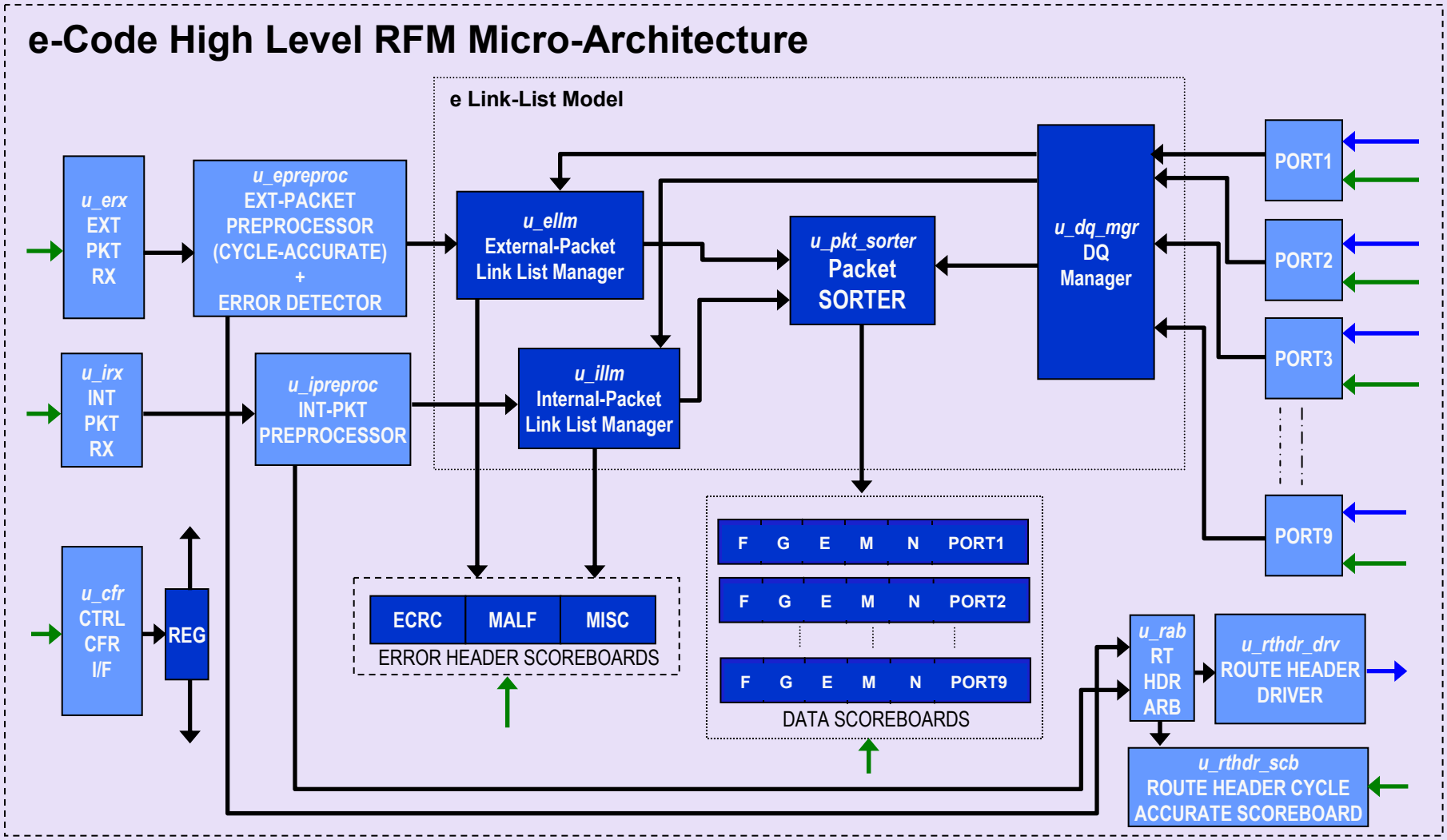
- Ingress packet architecture with packets buffered at input
- Infinite memory space challenge due to dynamic link list management for en-queue and de-queue of TLPs
- Dual mode support including dynamic switching between cut-through and store-and-forward behaviors
- Aggregation of multiple traffic flows without back-pressure for inbound traffic
- Support for mixed mode traffic among normal as well as error packets
- Scalability up to 9 simultaneous de-queues and all permutations of throughput combinations among available ports (x1, x2 and x4)

IPL DUT: How to verify?

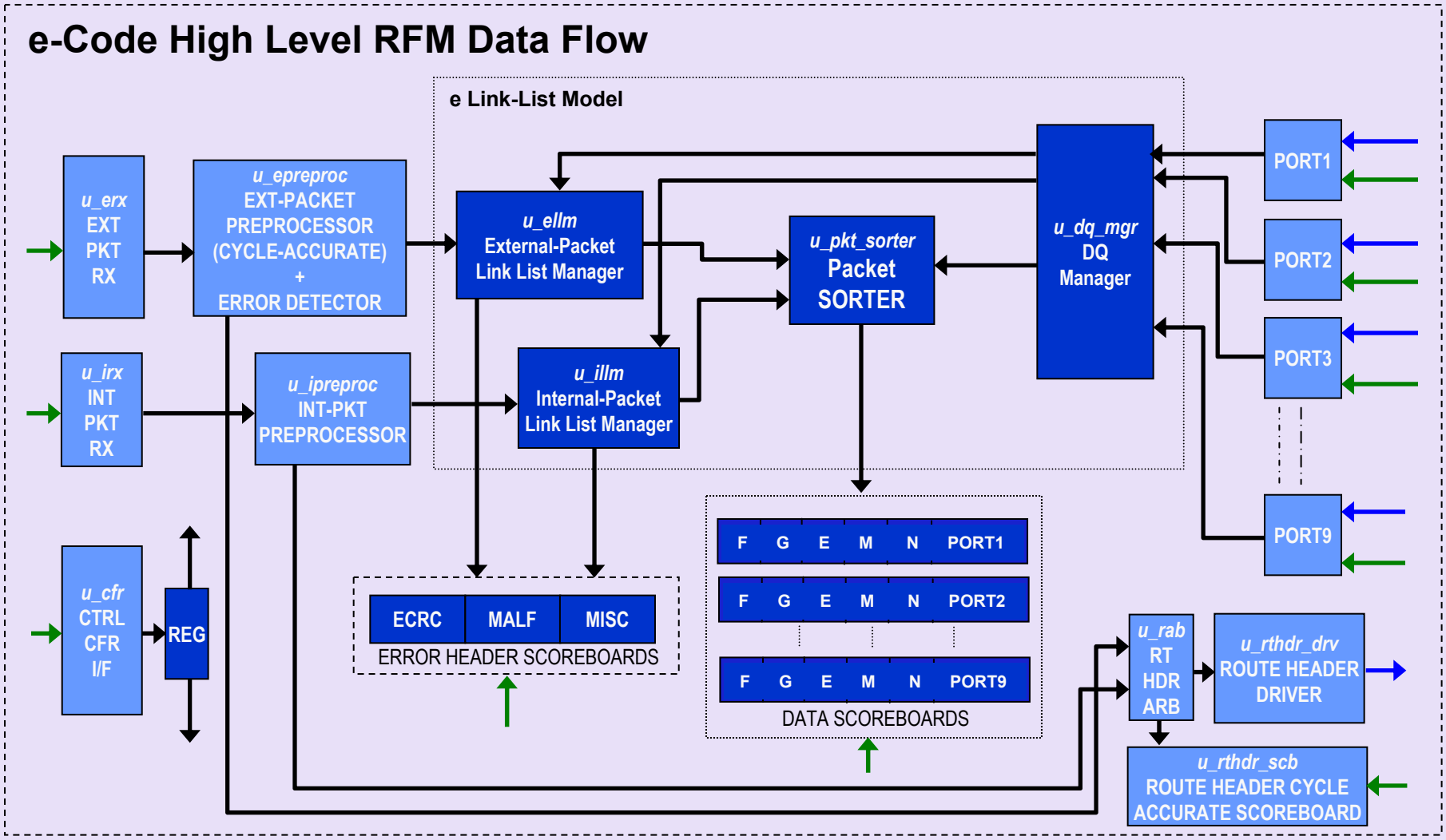


- 9 egress ports
- Mixed traffic
- Dual mode

IPL Reference Model Architecture



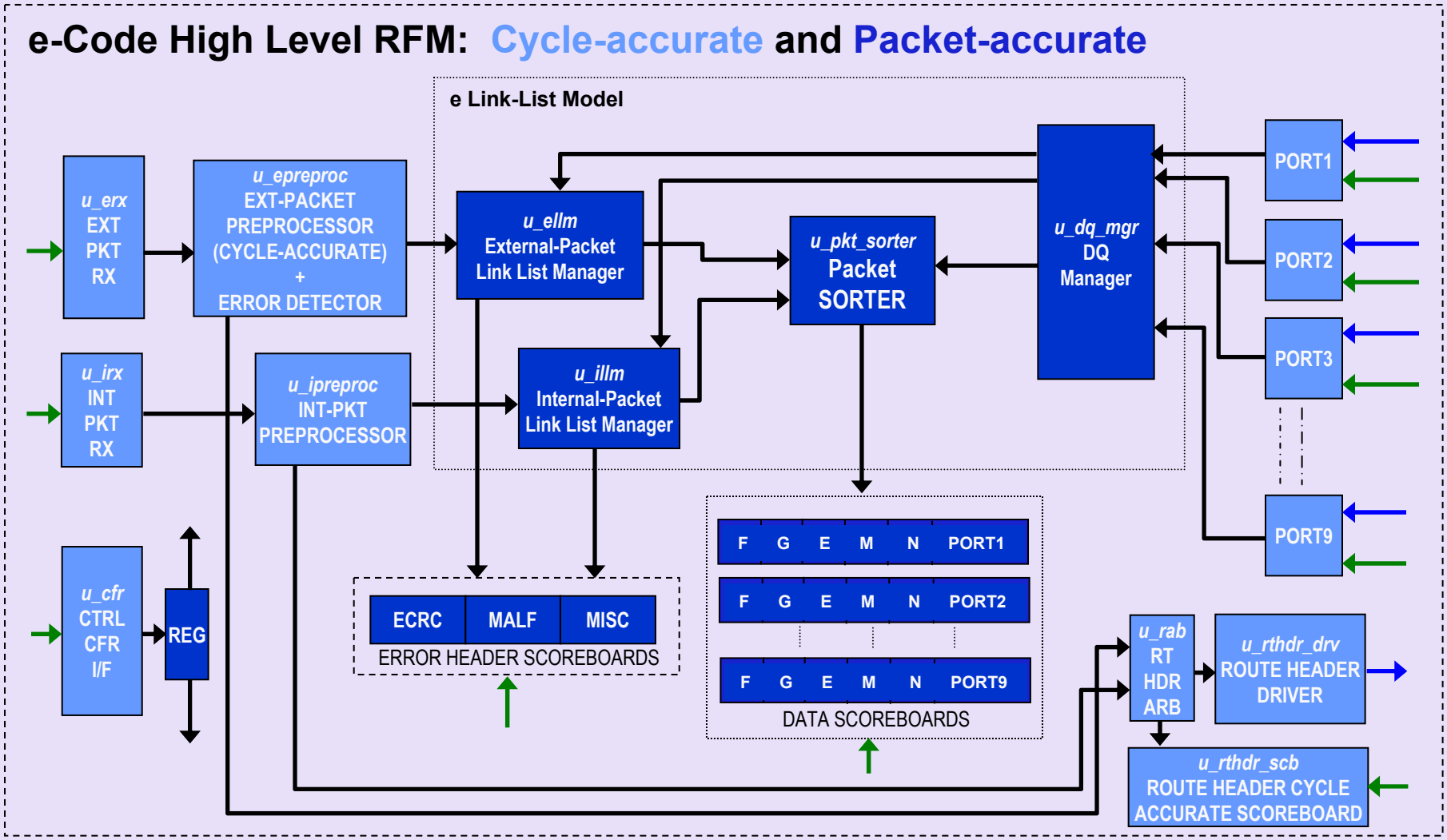
IPL Reference Model Architecture (cont.)



Reference Model Features

- Cycle and Packet Accuracy (Hybrid Modeling)
  - ✓ Cycle accurate modeling on header routing path to support seamless chip-level integration of block-level reference model
  - ✓ Packet accurate modeling for the packet transmission path
- Modular Verification Architecture
  - ✓ Scalability for larger and faster switches through reuse of building blocks
- Pointer Management Scheme
  - ✓ Independent of Hardware Implementation
  - ✓ Re-usable

Hybrid Reference Modeling



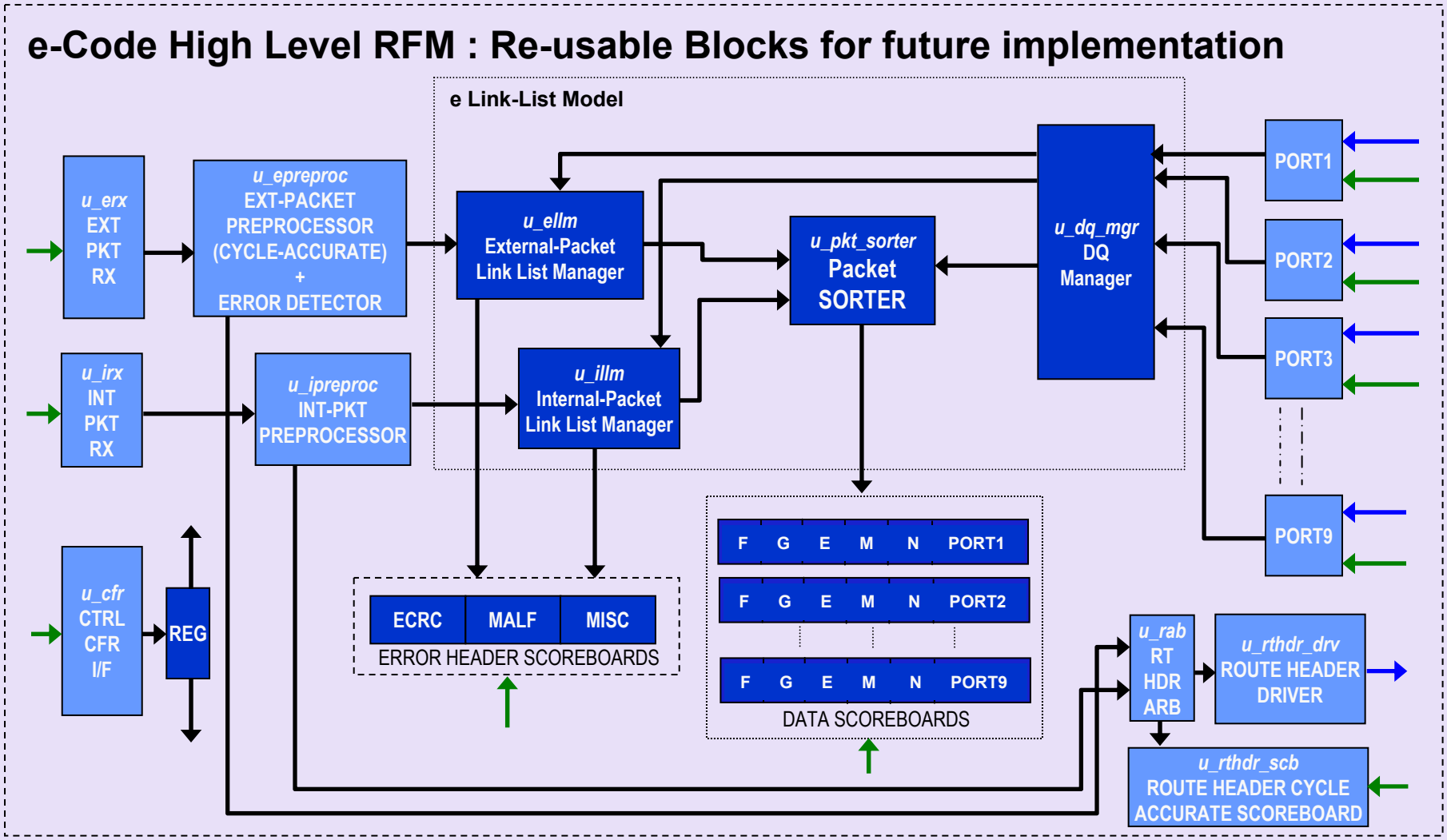
Reference Model Features

- Cycle and Packet Accuracy (Hybrid Modeling)
  - ✓ Cycle accurate modeling on header routing path to support seamless chip-level integration of block-level reference model
  - ✓ Packet accurate modeling for the packet transmission path

- Modular Verification Architecture
  - ✓ Scalability for larger and faster switches through reuse of building blocks

- Pointer Management Scheme
  - ✓ Independent of Hardware Implementation
  - ✓ Re-usable

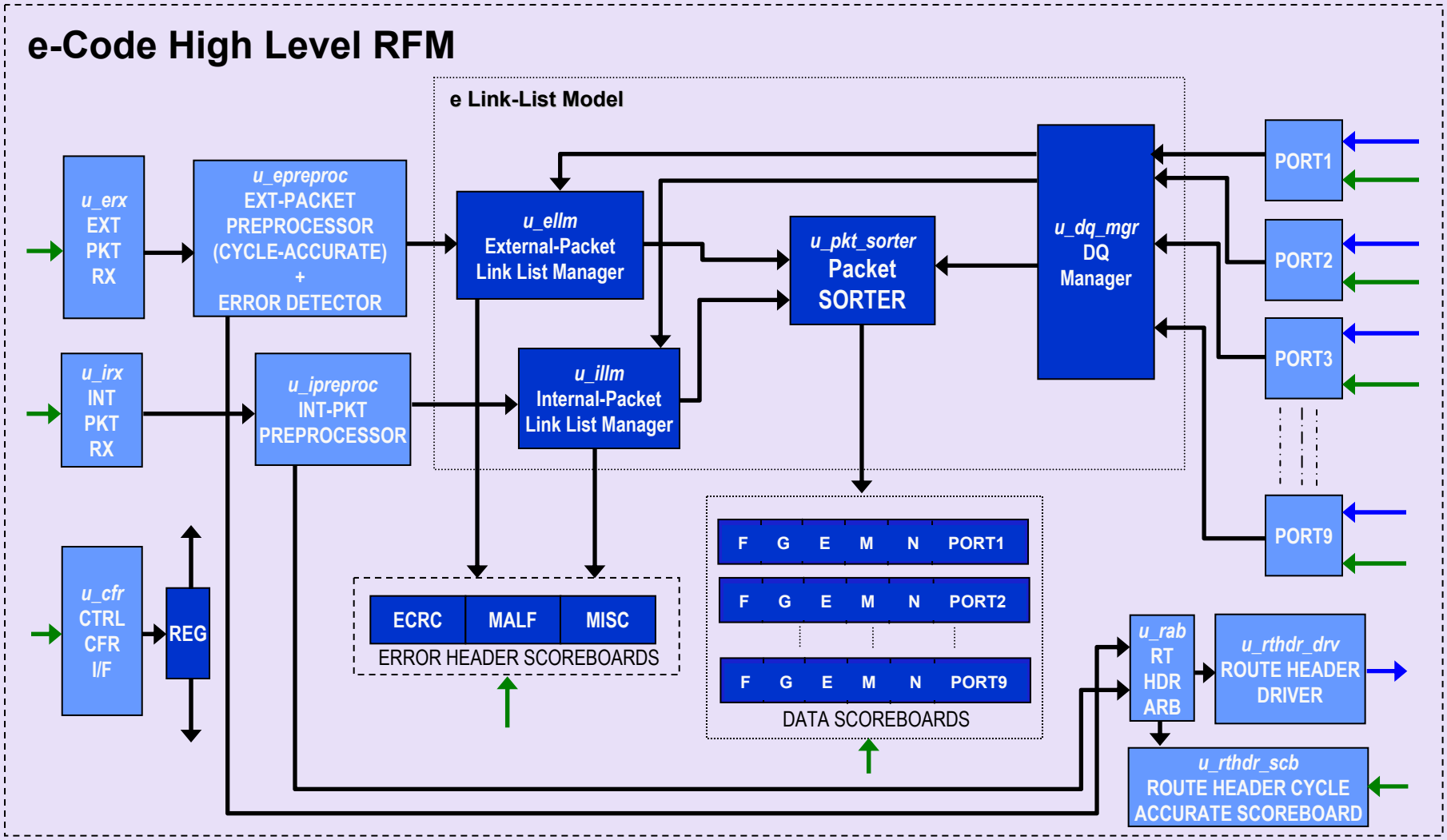
Modular Verification Architecture



Reference Model Features

- Cycle and Packet Accuracy (Hybrid Modeling)
  - ✓ Cycle accurate modeling on header routing path to support seamless chip-level integration of block-level reference model
  - ✓ Packet accurate modeling for the packet transmission path
- Modular Verification Architecture
  - ✓ Scalability for larger and faster switches through reuse of building blocks
- Pointer Management Scheme
  - ✓ Independent of Hardware Implementation
  - ✓ Re-usable

Pointer Management Scheme



Reference Modeling Techniques

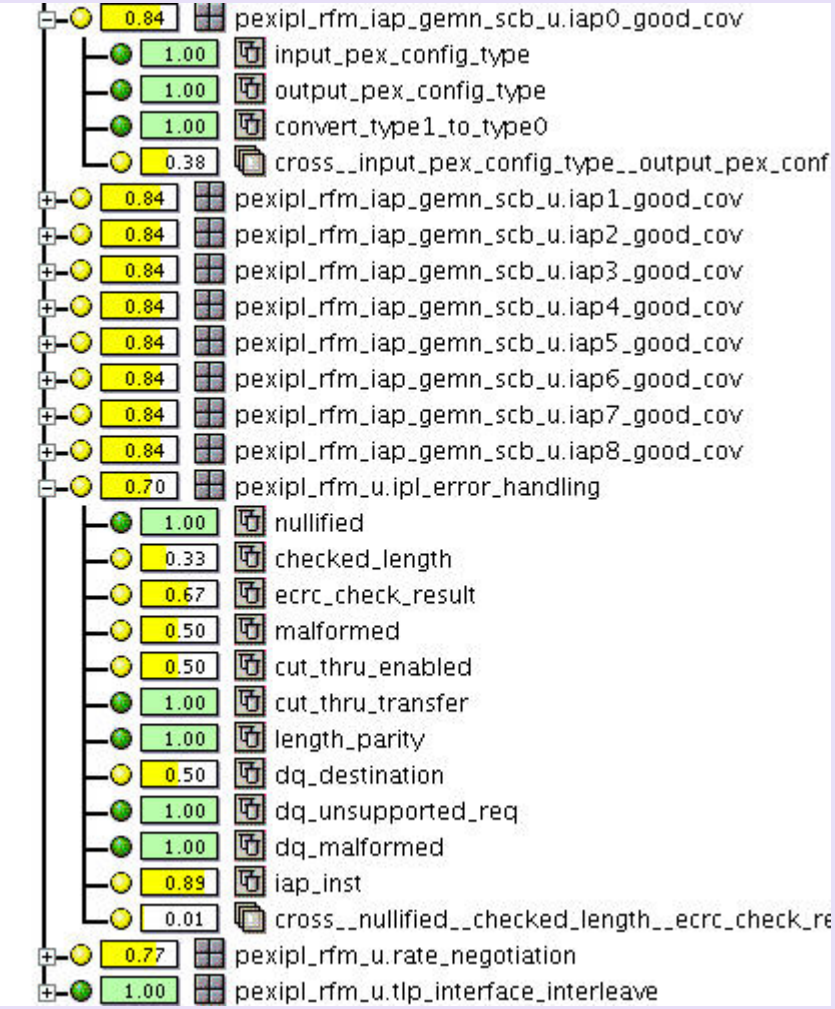
- Sub-function blocks coded as a high level units
- Communication between units done through ports
- A top-level unit binds all the sub-units through ports
- Data communicated using structs through ports
- Simulator callbacks reduced by using single event from top-level unit fanned out to sub-units
- Cycle-accurate RFM control information was driven through HDL signals: closed the pointer loop

Reference Modeling Techniques (cont.)

- Generic data collection monitors to support all speeds
- Units coded to follow eRM for multiple instantiations
- Design for verification signals used to avoid re-ordering problems during data compares to avoid cycle accurate modeling
- Hybrid modeling to reduce maintenance overhead

Coverage and Debug Messages

IPL RFM functional coverage



Coverage and Debug Messages (cont.)

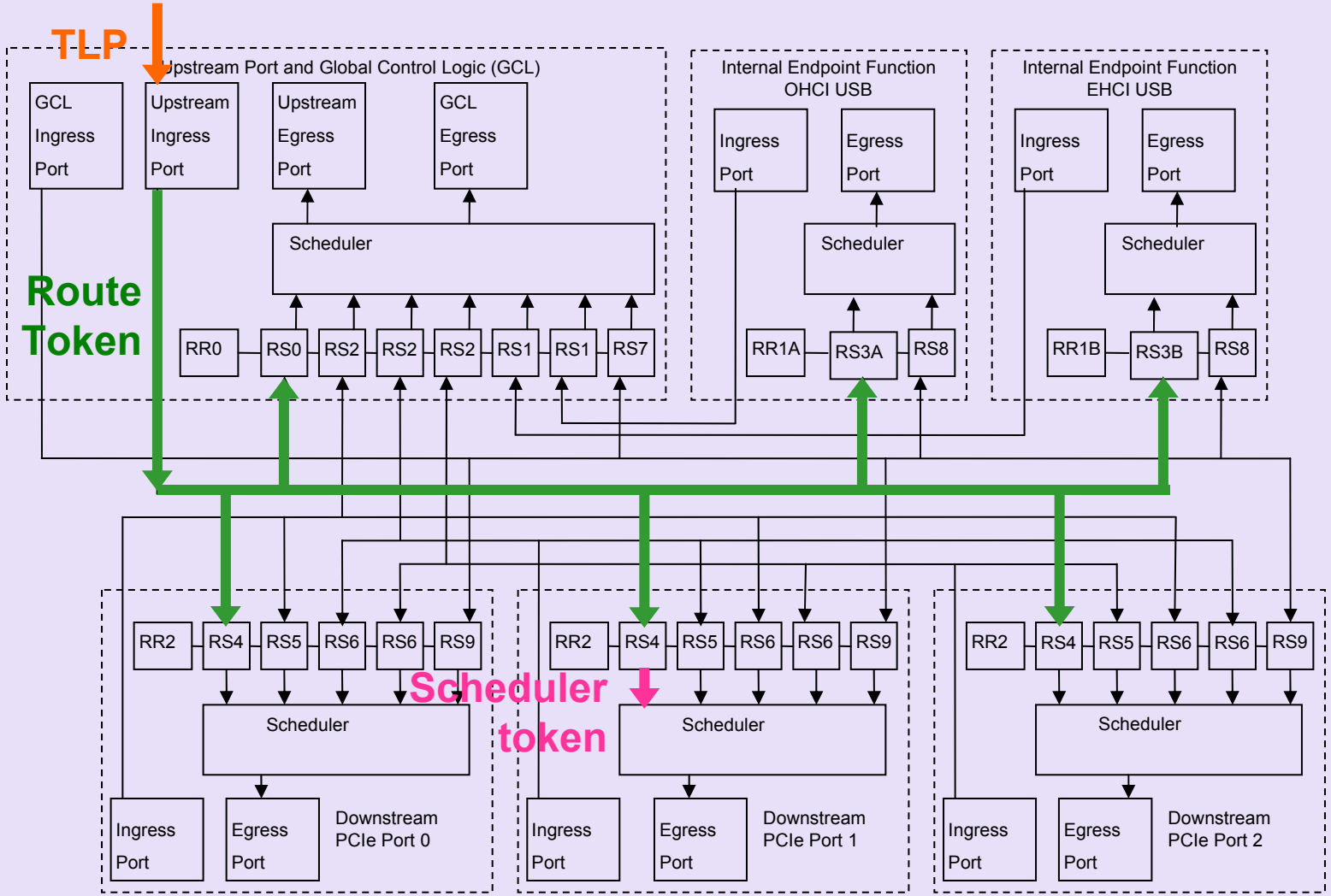
IPL RFM debug messages

```
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:    ================================
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:          DUT TLP : MWr DW4_WD
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:          RFM TLP : MWr DW4_WD
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:    ================================
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:      ============================
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:             RFM POINTER 2
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:      ============================
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:        IAP4 FULL LIST SCOREBOARD
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:      ============================
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:              RFM         DUT
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:              byte        byte
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:      ============================
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:               60           60
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:               0            0
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:              b0           b0
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:               a            a
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:               1            1
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:               0            0
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:               2            2
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:              ff           ff
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:              12           12
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:              34           34
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:              56           56
[2712 ns]  IPL_RFM_IAP_GEMN_SCOREBOARD:              78           78
```

Outline

- PCI Express Switch ASIC Architecture
- Micro-architecture: Ingress Port Logic (IPL)
- Micro-architecture: Router
- Reference Model Example: Ingress Port Logic
- Reference Model Example: Router
- Integration of Reference Models at chip-level
- Conclusion

Distributed Routers: "SuperRouter"

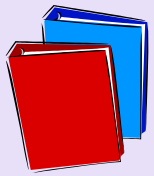


Router RFM: Why?

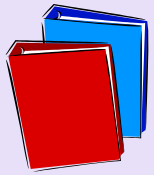
1724 pages of specifications

PCI Literature
PCI Local Bus Specification
PCI-to-PCI Bridge Specification
PCI Bus Power Management Specification

PCI Express Literature
PCI Express Base Specification
PCI Express to PCI Bridge Specification



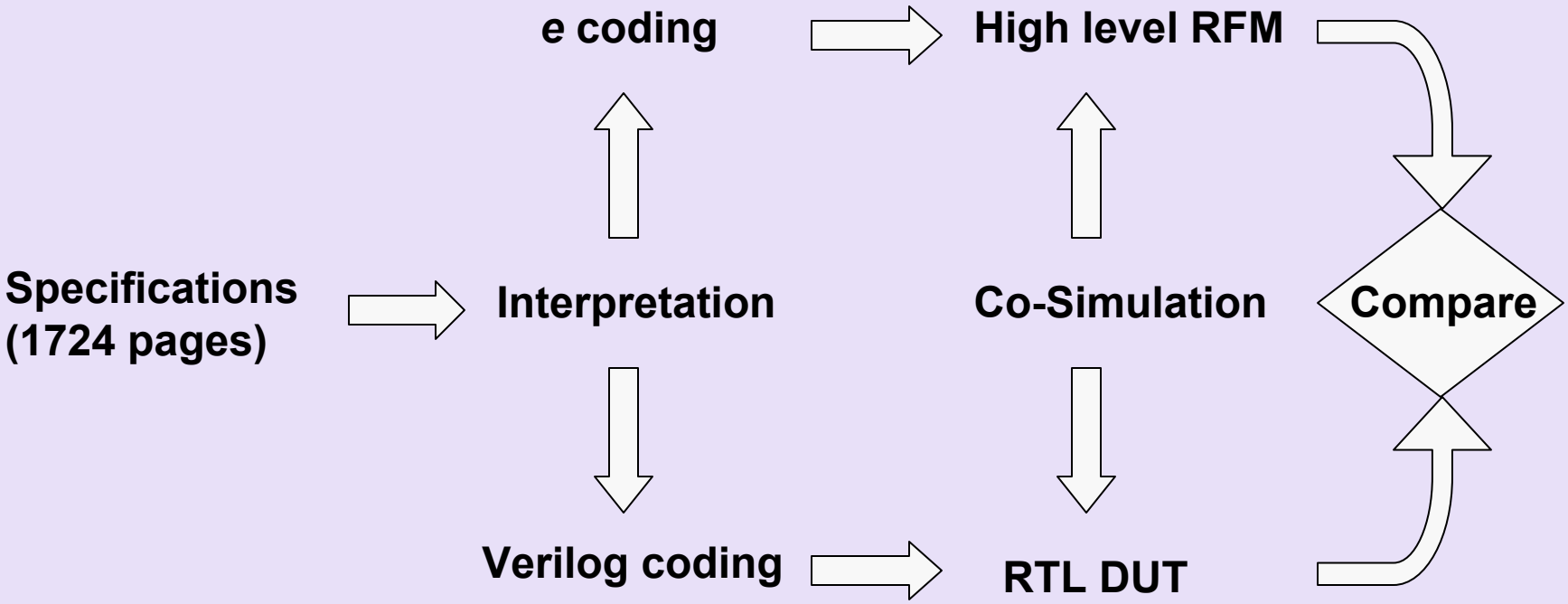
TI Functional Specifications
Functional Spec for PCIe Switch
Functional Spec for Multifunction PCIe Device



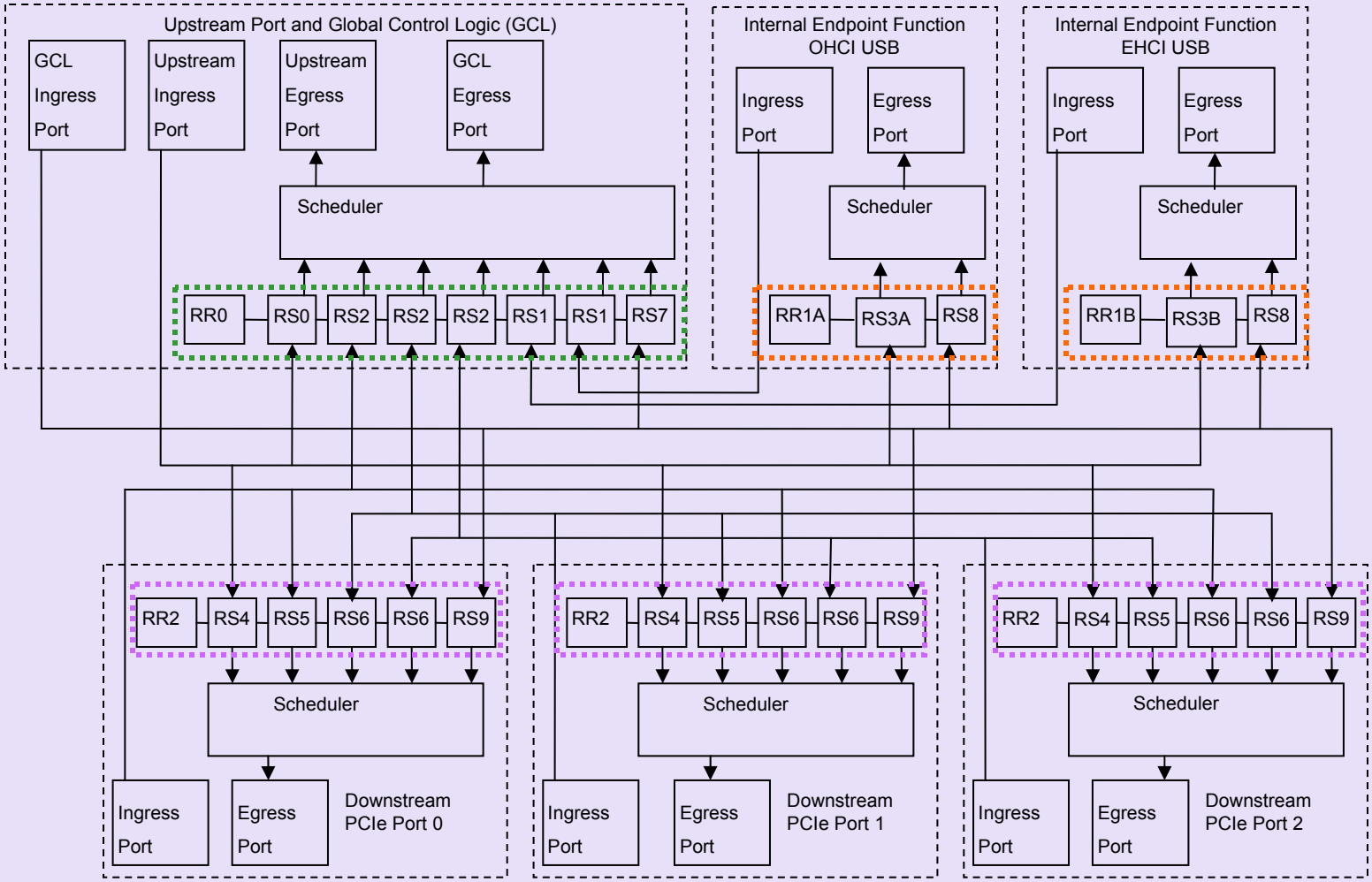
RTL Specifications
PCI Express Switch Implementation Specification
Router Implementation Specification

Directed test overload!

Router RFM: Why? (cont.)

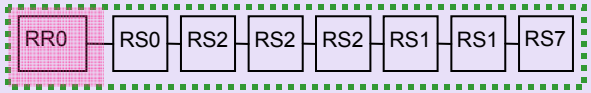


Router eRM Architecture



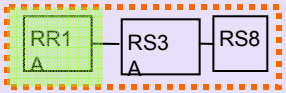
Router eRM Architecture (cont.)

UPSTREAM'mode
pexrtr_env_u

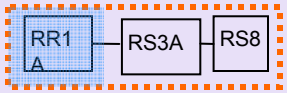


TYPE1'hdr_type

MULTIFUNCTION'mode
pexrtr_env_u



TYPE0_EHCI'
hdr_type

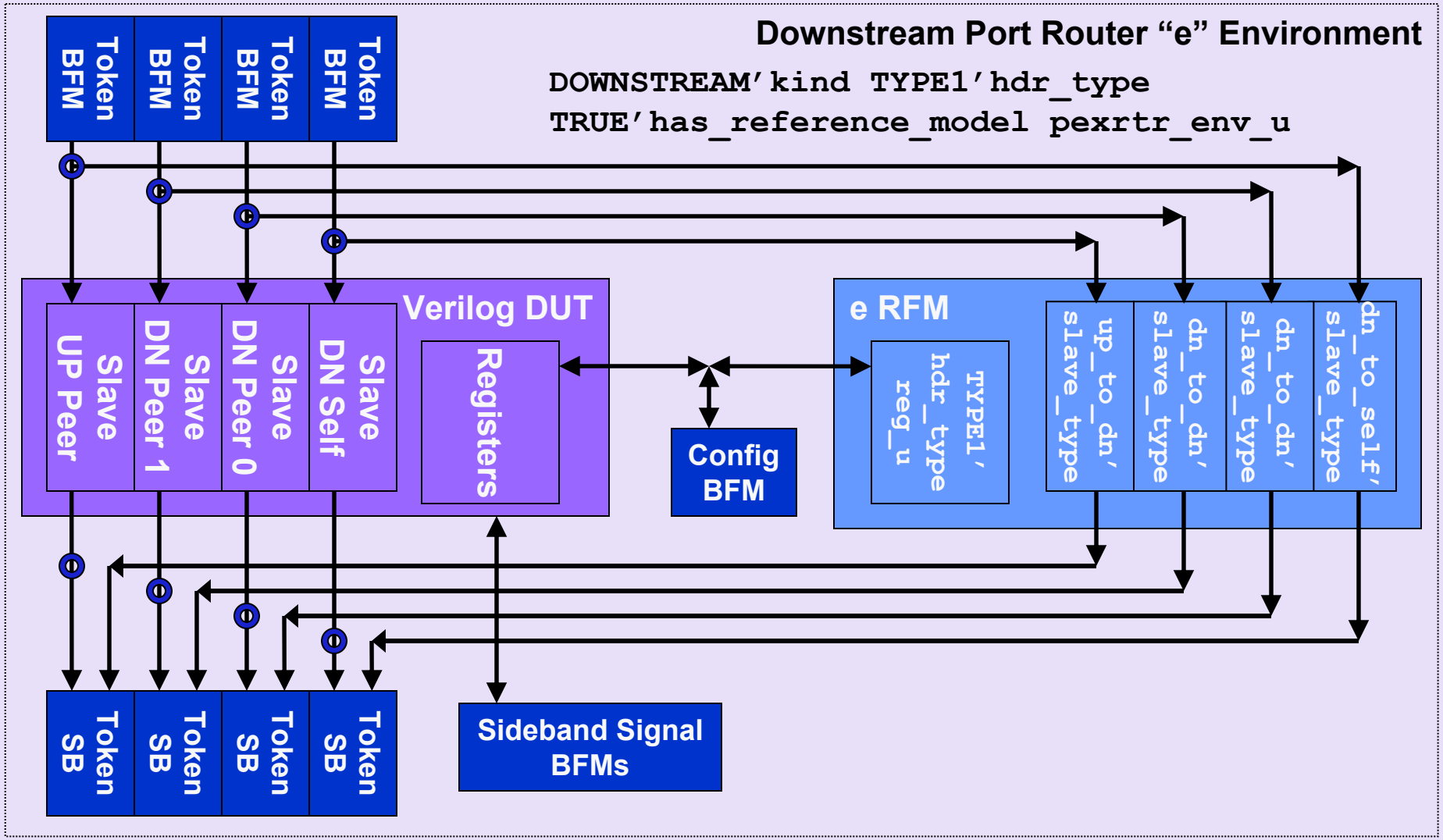


TYPE0_OHCI'
hdr_type

DOWNSTREAM'mode pexrtr_env_u

TYPE1'hdr_type

Router eRM Environment



Router RFM Comparison Logic

```
env        : pexrtr_env_u;
bus_name   : pexrtr_bus_name_t;
slave_type : pexrtr_rfm_slave_t;
```

"e" RFM Token Comparison Unit
pexrtr_rfm_compare_u

From input monitor

```
in_token : pexrtr_token_s;
event in_token_done_e;
```

```
1) Daisy chain pre-processing sequential logic.
```

Plug-in rule sets

```
2) Call compare_<type>_token() based on Memory, IO, etc.
```

```
3) Daisy chain post-processing sequential logic.
```

To output scoreboard

```
rfm_token : pexrtr_sch_token_s;
event rfm_token_done_e;
```

Techniques of Router RFM

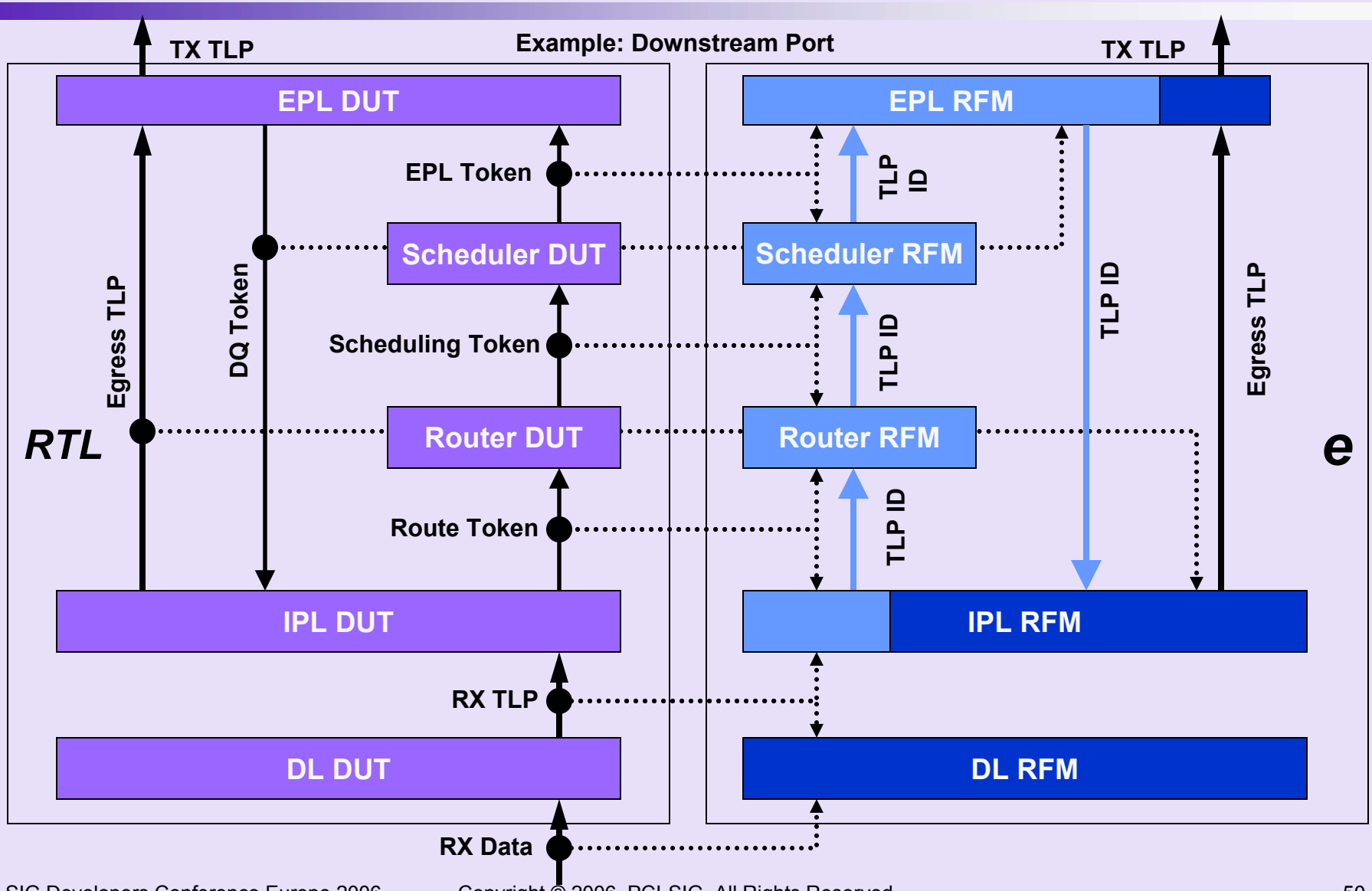
- *when* inheritance → plug-in rule sets
- Simulation of pseudo chip-level SuperRouter
- Detailed log files of each transaction
- RFM includes text comments explaining expected behavior:

```
Bus mastering disabled, so ignore all Memory TLPs.
Previously deemed Malformed. Ignore.
In IO window. Blocked.
Previously deemed UR.
Completion is outside sec-to-sub window, and sec!=0, so claim it.
Ignore all internally generated Vendor_Type1 MsgD messages.
```

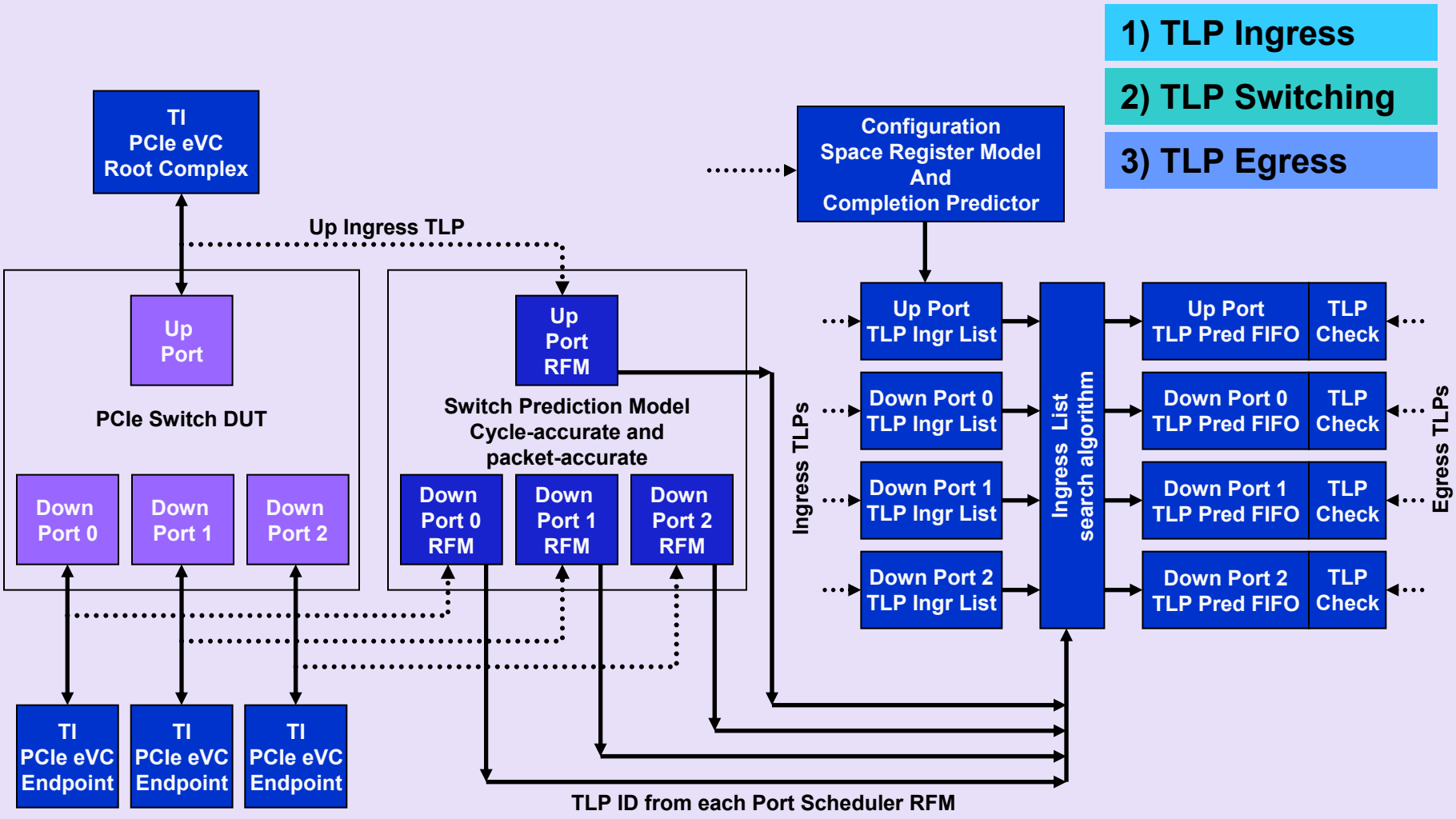
Outline

- PCI Express Switch ASIC Architecture
- Micro-architecture : Ingress Port Logic (IPL)
- Micro-architecture : Router
- Reference Model Example: Ingress Port Logic
- Reference Model Example: Router
- Integration of Reference Models at chip-level
- Conclusion

PCIe Port Reference Model



Finished Chip-level Prediction Model



Outline

- PCI Express Switch ASIC Architecture
- Micro-architecture: Ingress Port Logic (IPL)
- Micro-architecture: Router
- Reference Model Example: Ingress Port Logic
- Reference Model Example: Router
- Integration of Reference Models at chip-level
- Conclusion

Conclusion

- Architecture definition to facilitate Design for Verification
- RFM methodology requires dedicated and specialized Verification Engineer resources
- Rigorous block-level simulation permitted 2 days from integration to first chip-level simulation
- 2-weeks of chip-level simulations put robust FPGA build in the lab for emulation
- “SuperRouter” simulations eliminated lost/misdirected packets at chip-level
- Rigor of IPL simulations paved the way to stable FPGA performance at industry Plug Fests

Conclusion (cont.)

- RFM will provide dual, independent interpretation of specification
- RFM styles: Choose wisely
  - ✓ cycle accurate, packet accurate, hybrid
- Muscle of Specman random generation is only as good as the auto-checking features of testbench
- RFM hookup at chip-level:
  - ✓ Identify hookup issues, interface violations
  - ✓ Check for chip-level stimulus that was missed at module-level
  - ✓ Chip-level debug acceleration

About the Authors

- **Asad Khan (a-khan1@ti.com)**
  - ✓ Design Verification Engineer for PCI Express Switch
- **Roy Wojciechowski, MGTS (roywojciechowski@ti.com)**
  - ✓ Design Engineer for PCI Express Switch
- **Scott Morrison (scott@ti.com)**
  - ✓ Design Verification Engineer for PCI Express Switch
- **Henry Angulo, SMTS (h-angulo@ti.com)**
  - ✓ Design Verification Lead for PCI Express Switch and Multifunction Bridge
- **Pradip Thaker, PhD (p-thaker2@ti.com)**
  - ✓ Technical Lead for PCI Express Switch and Multifunction Bridge

Additional contributions from

- Paul Howard
- Beth Richard
- Wenmu He, PhD
- Srinadh Madhavapeddi
- Jim Skidmore, SMTS
- Sumit Das
- Sue Vining, MGTS
- Scott Kim
- Kim Clark

Thank you for attending the PCI-SIG Developers Conference Europe 2006.

For more information please go to www.pcisig.com