
PCI
SIG™

PCI-X Mode 2 to HyperTransport™ Bridge

Mike Lowe

Silicon Engineering Director

AMD

Objective

This presentation is intended to provide insight into the architecture, design, verification, and validation of a PCI-X to HyperTransport™ Bridge device

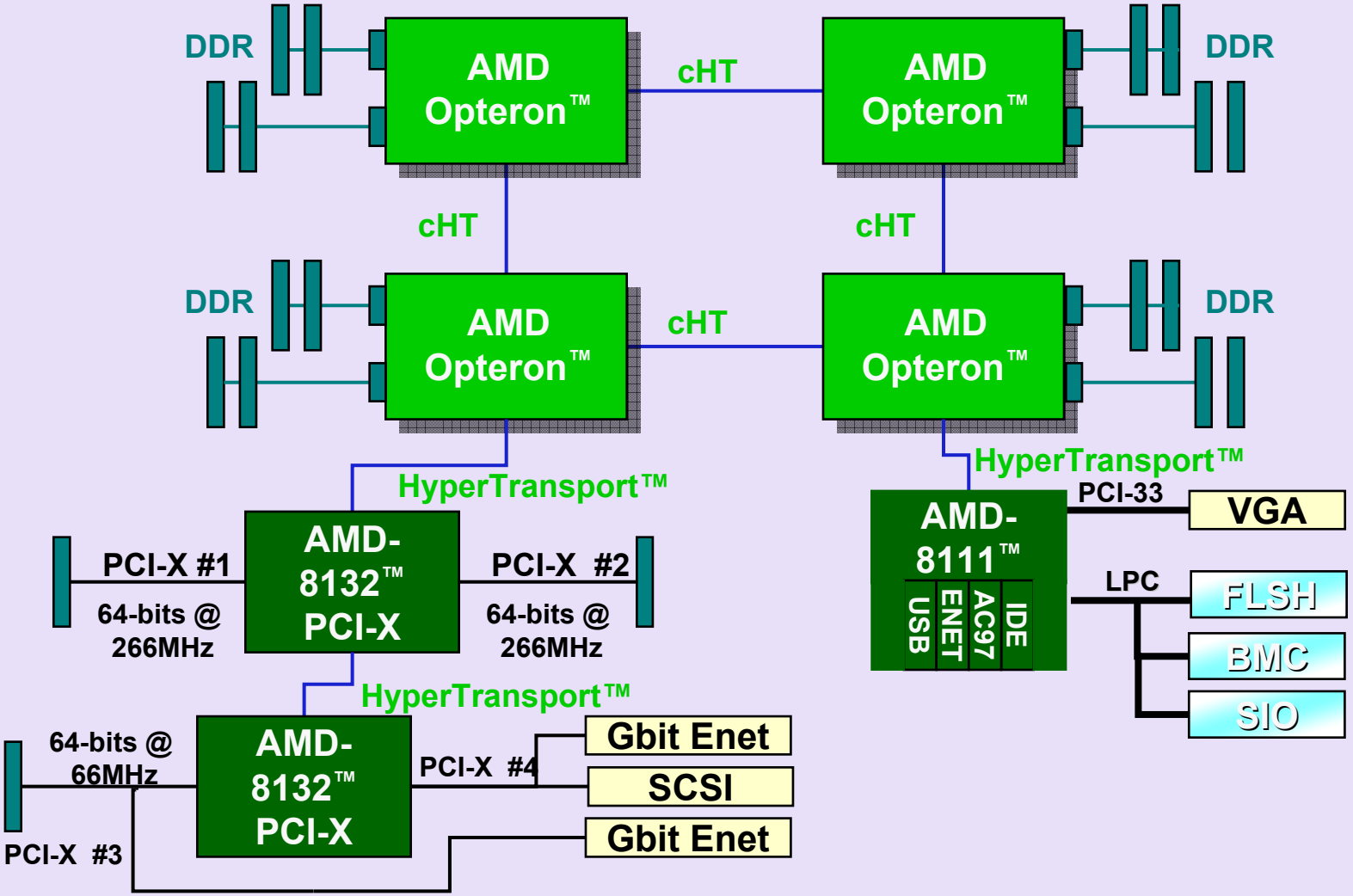
Agenda

- System Architecture Overview
  - ✓ Block Diagrams
- Device Architecture Overview
  - ✓ PCI-X Cycle Translation to and from HyperTransport™ technology
  - ✓ Transaction Example
- Verification Concerns
  - ✓ Methodologies
  - ✓ Deadlock Testing Example
- System considerations
  - ✓ Bridge Architectural Limitations
  - ✓ System Validation Methodologies and Concerns
  - ✓ Validating without PCI-X Mode2 cards

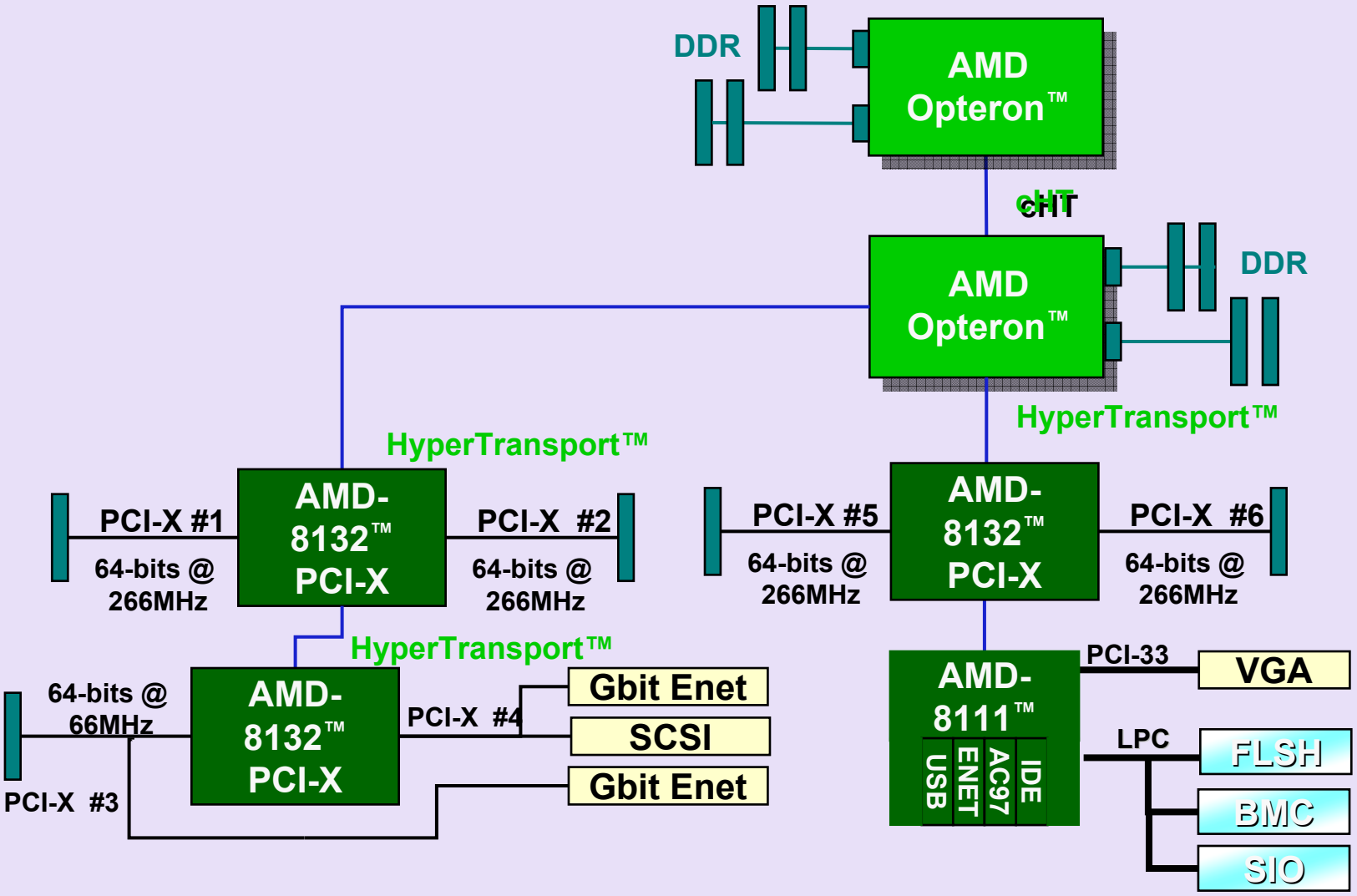
System Architecture

- Block Diagram for 4-CPU and 2-CPU systems with multiple PCI-X to HyperTransport™ Bridges is presented on the following slide
  - ✓ HyperTransport™ architecture allows chaining of PCI-X to HyperTransport™ bridge chips
  - ✓ 2, 3, or more Bridge chips can be utilized
- Transaction Routing
  - ✓ PCI-X transactions which target cards on the same bus do not enter the PCI-X to HyperTransport™ bridge
  - ✓ PCI-X transactions which target any other location flow to the CPU Host bridge for routing to their final destination
    - – Example: PCI-X card on bus #1 transactions to a PCI-X card on bus #2 will still flow through the CPU Host bridge within the AMD Opteron™ processor.

PCI-X System Example



PCI-X System Example



PCI-X HyperTransport™ Bridge Internal Architecture

- Architectural Goals
  - ✓ Minimize Latency when transferring from HyperTransport™ to HyperTransport™
  - ✓ Maximize Bandwidth when transferring to and from PCI-X
  - ✓ Provide 80% or more utilizable PCI-X bandwidth
  - ✓ Hot-Plug Support
  - ✓ Utilize ASIC flow for development and implementation
- Architectural Details
  - ✓ PCI-X engine is based upon a licensed “Golden Master”
  - ✓ HyperTransport™ links (which run at up to 16-bits, 2 GTps externally) are converted to 128bits and up to 250Mhz internally
  - ✓ PCI-X logic domain runs internally at PCI/PCI-X bus frequency
  - ✓ High Performance PCI Bridge Capability also

PCI-X HyperTransport™ Bridge Internal Architecture

- Background Information
  - ✓ HyperTransport™ packets flow in three virtual channels – posted, non-posted, and response
  - ✓ HyperTransport™ data payload can be a maximum of 64 bytes per packet
  - ✓ PCI-X data payload can be a maximum of 4096 bytes per transaction
  - ✓ Therefore, there can be up to 64 HyperTransport™ 64-byte packets required to complete a maximum length PCI-X data transaction
  - ✓ “Cacheline” buffers are 64 bytes

PCI-X HyperTransport™ Bridge Internal Architecture

- PCI-X to HyperTransport™ Command Mappings

PCI-X	HyperTranport™
Posted Memory Write (PMW)	Write Posted
Splittable Write Request (SWR)	Write Nonposted
Splittable Read Request (SRR)	Read Nonposted
Immediate Write Completion (IWC) Response	TgtDone
Immediate Read Completion (IRC)	Read Response
Split Write Completion (SWC) Response	TgtDone
Split Read Completion (SRC)	Read

PCI-X HyperTransport™ Bridge Internal Architecture

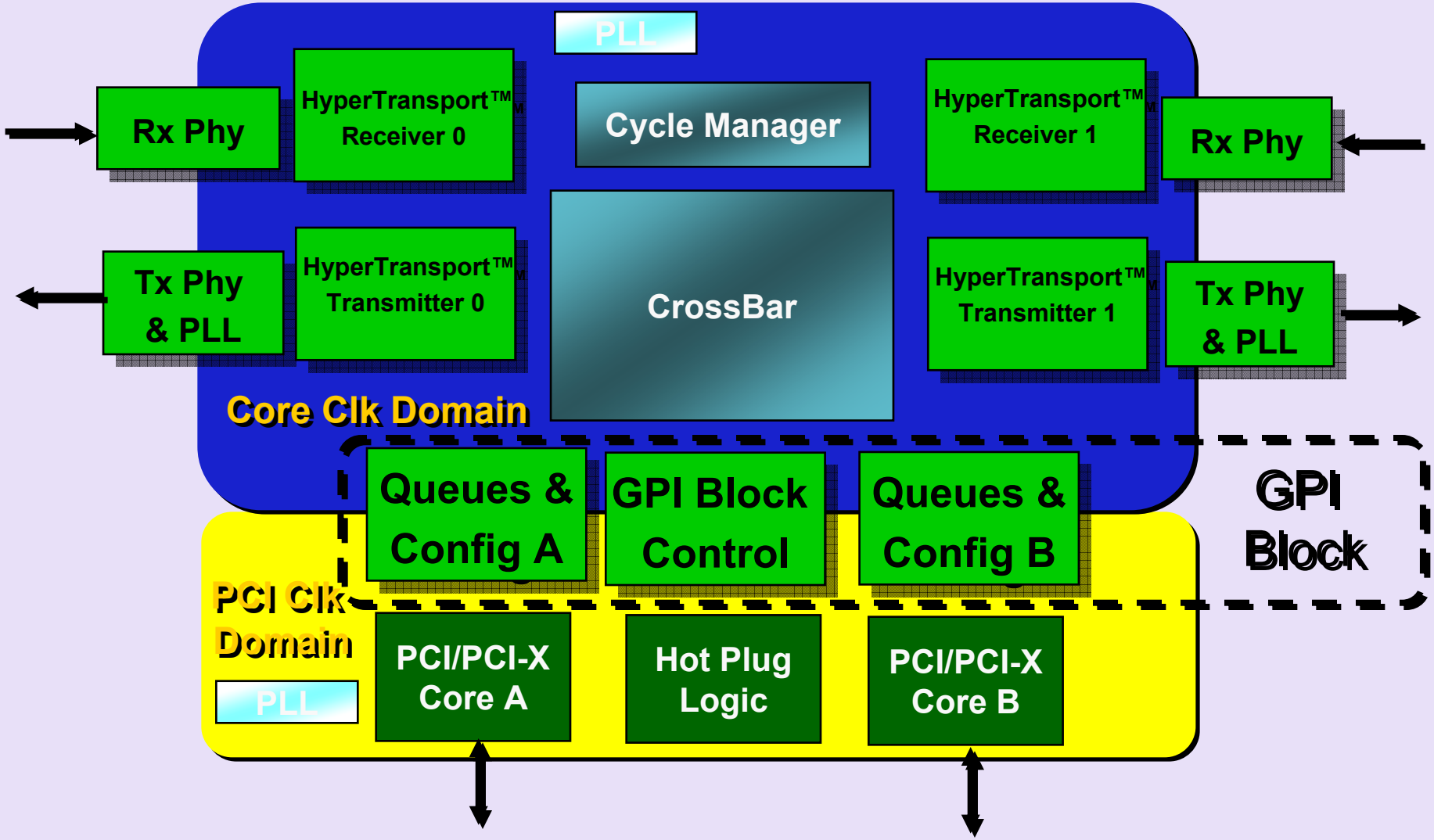
- HyperTransport™ to PCI-X Command Mappings

HyperTransport™ Packet Type	PCI-X Transaction Type
Write to Memory (Posted)	Posted Memory Write (PMW)
Write to Config or I/O (Nonposted) (SWR)	Splittable Write Request
Read to Memory (Nonposted) Request (SRR)	Splittable Read
Read to Config or I/O (Nonposted) (SRR)	Splittable Read Request
Read Response	Split Read Completion (SRC)
TgtDone Response Completion (SWC)	Split Write

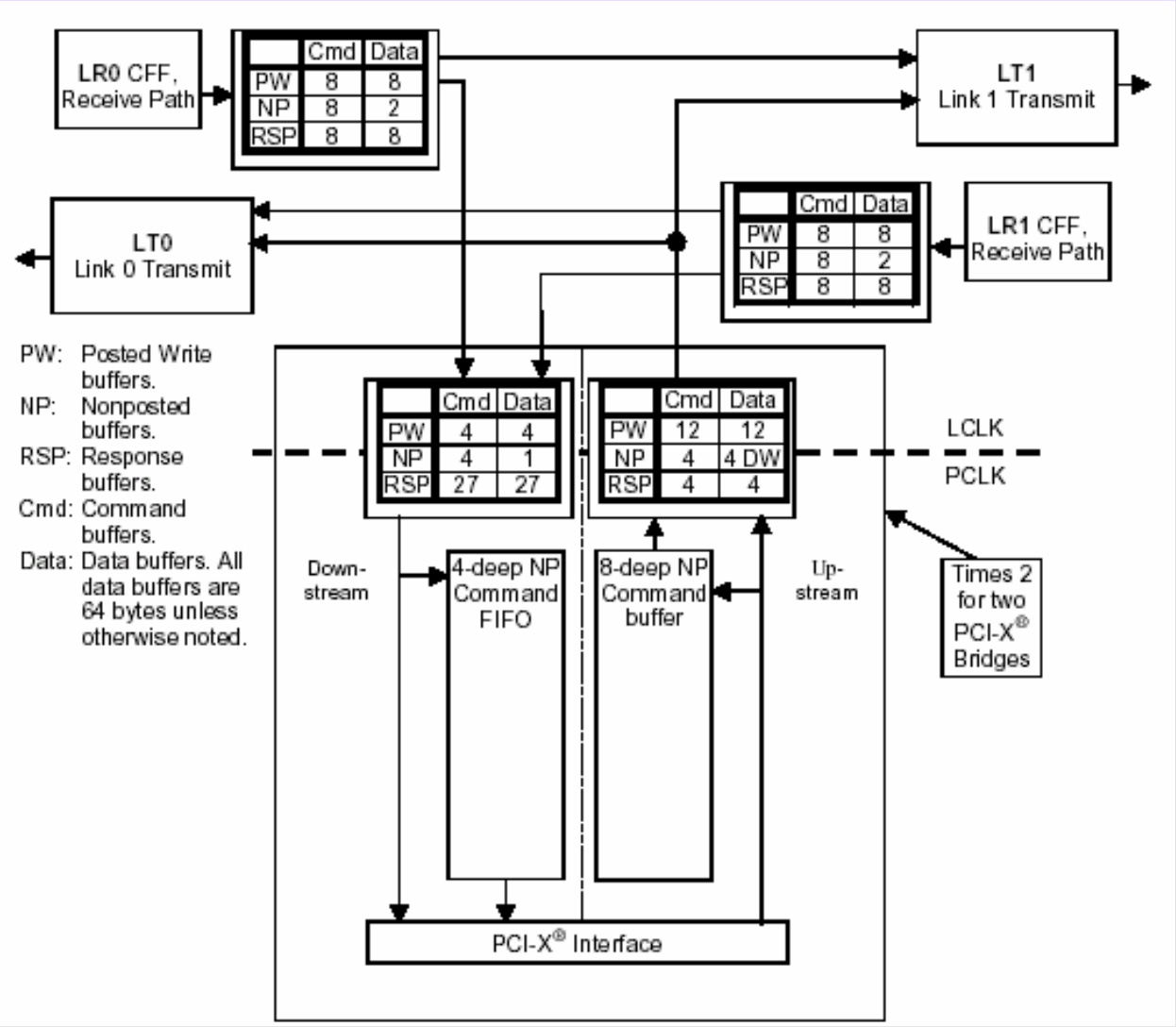
PCI-X HyperTransport™ Bridge Internal Architecture

- Error Reporting Strategy
  - ✓ Three Types of Error Responses
    - Generate an Interrupt (via HyperTransport™ messaging, Interrupt pins, and Virtual Wire APIC messages)
    - Error status in Response
    - Sync Flooding
  - ✓ Define the rules for classes of operation:
    - Sync Flood on Posted Serious Errors
    - Error Status in Response for Non-Posted Errors
    - Interrupts for some Errors
  - ✓ Or, allow user to select error response programmability :
    - Flood
    - Fatal Interrupt (in addition to Error Response where applicable)
    - Non-Fatal Interrupt (in addition to Error Response where applicable)
    - No Action

PCI-X HyperTransport™ Bridge Internal Architecture



PCI-X HyperTransport™ Bridge Internal Architecture



PCI-X HyperTransport™ Bridge Internal Architecture

- A Transaction Example
  - ✓ PCI-X upstream read of 4096 bytes is issued
  - ✓ The PCI-X Core terminates the cycle with a split response, and forwards the full data request to the Bridge's cycle manager
  - ✓ The cycle manager queues 64 HyperTransport™ read transactions upstream to the CPU and memory
  - ✓ Read Responses flow downstream to the Bridge's Cycle Manager
  - ✓ After the first cacheline has been retrieved, the GPI logic issues a split completion cycle to the PCI-X Core. This split completion returns the cacheline and the device continues to accumulate data via HyperTransport™. Any cachelines accumulated while the split completion is active will also be transferred.
  - ✓ The GPI logic monitors the split completion progress, and if data is delayed then the split completion is terminated at a disconnect boundary and another queued when more data arrives via HyperTransport™ read responses.

PCI-X HyperTransport™ Bridge Internal Architecture

- Going Deeper into the Transaction Example
  - ✓ PCI-X core transfers request by entering request into a NonPosted Queue Command Buffer in the GPI block
  - ✓ NonPosted Queue Command Buffer generates HyperTransport™ requests to fulfill PCI-X request and queues them to the Cycle Manager as response buffer space is available (max 27 outstanding at once)
  - ✓ The Cycle Manager routes them to the appropriate transmitter as HyperTransport™ buffer space is available
  - ✓ HyperTransport™ read responses are returned, and routed to the Response Cacheline buffers in the GPI block.
  - ✓ GPI logic detects the presence of responses, and initiates the split completion cycle into the PCI-X core

PCI-X HyperTransport™ Bridge Verification

- Areas of Focus
  - ✓ Proper Transaction Resolution
    - Intelligent C++ Object Oriented Transaction Tracker
  - ✓ Directed and Controlled Random Testing
    - Random testing via HyperTransport™ and PCI-X Bus Functional Models and via CPU Northbridge Unit stimulation
    - Directed tests to supplement Random
  - ✓ Compliance to Ordering Rules
    - Checked by the C++ Object Oriented Transaction Tracker
  - ✓ Deadlock Avoidance
    - Controlled Random Testing combined with the ability to block specific channels in a given direction
  - ✓ Hot-Plug Compliance
    - Particularly reset sequences and switching back and forth between PCI and PCI-X modes and speeds
  - ✓ Performance
    - Bandwidth Maximization, Fair Bandwidth Allocation, and Starvation Avoidance.

Deadlock Avoidance Guidelines

- **Deadlock avoidance strategy**
  - ✓ Do not make acceptance of a posted request dependent upon the ability to issue another request.
  - ✓ Do not make acceptance of a nonposted request dependent upon the ability to issue another nonposted request.
  - ✓ Do not make acceptance of a request dependent upon receipt of a response.
  - ✓ Do not make issuance of a response dependent upon the ability to issue a nonposted request.
  - ✓ Do not make issuance of a response dependent upon receipt of a response.
- **Ordering Rules (Row Pass Column?)**
  - ✓

	Posted	Non-Posted	Response
Posted	**No**	**Yes**	**Yes**
Non-Posted	**No**	**Yes/No**	**Yes/No**
Response	**No**	**Yes**	**Yes/No**

NOTE: this table changes if the PassPW bit is set in HyperTransport™ request packets.

Deadlock Avoidance Test Strategy

- ✓ Block a specific outgoing virtual channel
  - Denial of Flow Credits for HyperTransport™
  - Continuous Retries from downstream targets for PCI-X
- ✓ Direct traffic from all incoming interfaces toward all outgoing interfaces including the blocked one. Requests should be issued to all virtual channels to further stress the blocked channel and also to allow a complete ordering rule check.
- ✓ Verify that forward progress can be made on all interfaces per the Ordering Rules table on the previous slide
- ✓ Verify that the blocked interface can accept incoming traffic and transmit outgoing traffic along other virtual channels in accordance with the ordering rules
- ✓ Fail if:
  - Unable to issue a request in a “Yes” channel within a user defined timeframe
  - Requests in a “No” channel pass the blocked requests
  - All requests do not complete after the block is removed within a user defined timeframe

Deadlock Avoidance Test Strategy

- Deadlock Traffic Test Strategy Chart
  - ✓ B = Blocked
  - ✓ NB = Not Blocked
  - ✓ X = Either blocked or not blocked
  - ✓ Yes = virtual channel must make forward progress
  - ✓ No = virtual channel cannot make forward progress
- Outgoing Channel ||| Incoming Progress ||| Outgoing Progress

Post	NP	Resp ||| Post	NP	Resp ||| Post	NP	Resp
B	X	X ||| Yes	No	No ||| No	No	No
NB	B	NB ||| Yes	Yes	Yes ||| Yes	No	Yes
NB	X	B ||| Yes	No	Yes ||| Yes	No	No

System Considerations

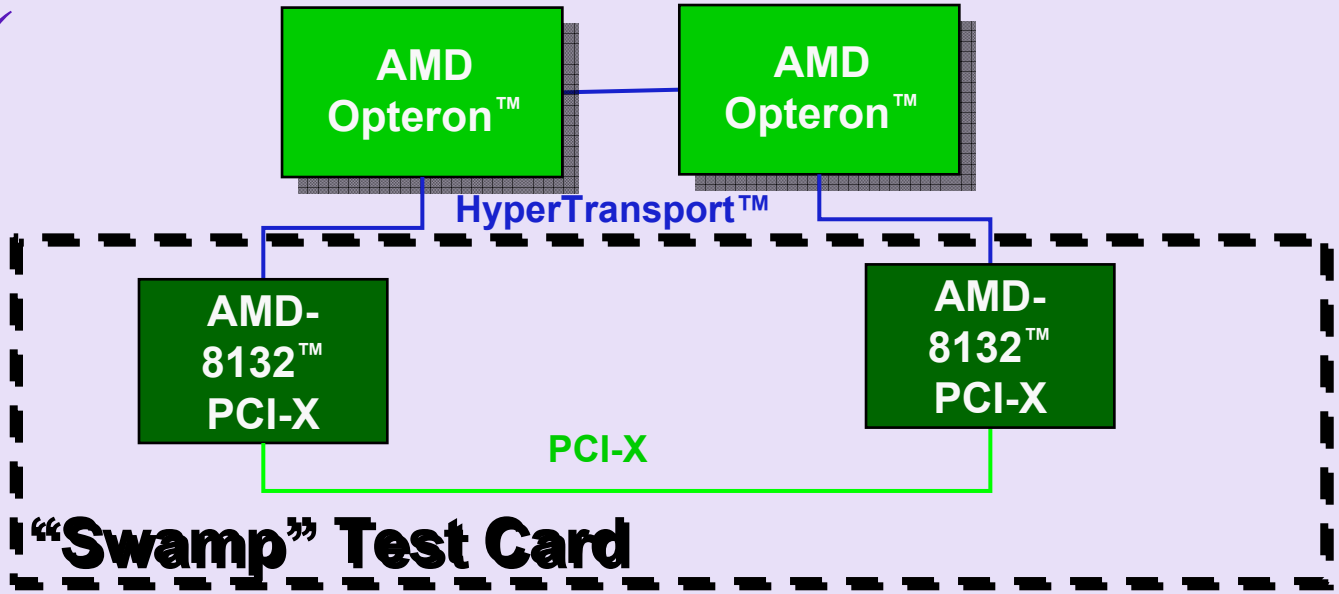
- Number of Bridges supported in a given system
  - ✓ The number of bridges that can be placed into a system effectively is governed by the bandwidth that can be supplied to them.
    - 1Ghz 16-bit HyperTransport™ (2GTps) links can support 4GBps in each direction
    - Dual PCI-X 133Mhz bridges can source/sink a maximum of 2GBps
    - Dual PCI-X 266Mhz bridges can source/sink a maximum of 4GBps
  - ✓ There are typically 2 HyperTransport™ links dedicated to IO devices in server and workstation systems.
  - ✓ Typically 2 AMD-8131™PCI-X 133 bridges and an IO Hub (aka Southbridge) are designed to reside on a single HyperTransport™ link
  - ✓ Up to 4 AMD-8132™ PCI-X 266 bridges are expected to perform well on a single HyperTransport™ link

System Validation

- Areas of Focus
  - ✓ General Wide Area Validation
    - Months of testing with various commercially available cards and software
    - Directed efforts to maximize system utilization during software/hardware testing
  - ✓ Electrical Characterization
    - Assures electrical specification compliance in a real-world environment
  - ✓ System Diagnostics
    - Focus on both system and device specific areas
    - Deadlock and Ordering Diagnostics
    - Hot Plug Diagnostics
    - Wide library of diagnostics also run for general device/system validation

Special Test Considerations

- PCI-X Mode 2 Testing
  - ✓ No PCI-X Mode 2 cards available for validation
  - ✓ Special Test Configuration used
    - Allow PCI-X bus to be connected back-back for signaling and protocol analysis
  - ✓



Questions and Answers

Thank you for attending the PCI-SIG Developers Conference 2004.

For more information please go to www.pcisig.com

PCI-X Mode 2 to HyperTransport™ Bridge

Mike Lowe

Silicon Engineering Director

AMD


PCI
SIG™