PCIe® Core Verification Using Random Error Injection

Gene Saghi

Vice President

Astek Corporation

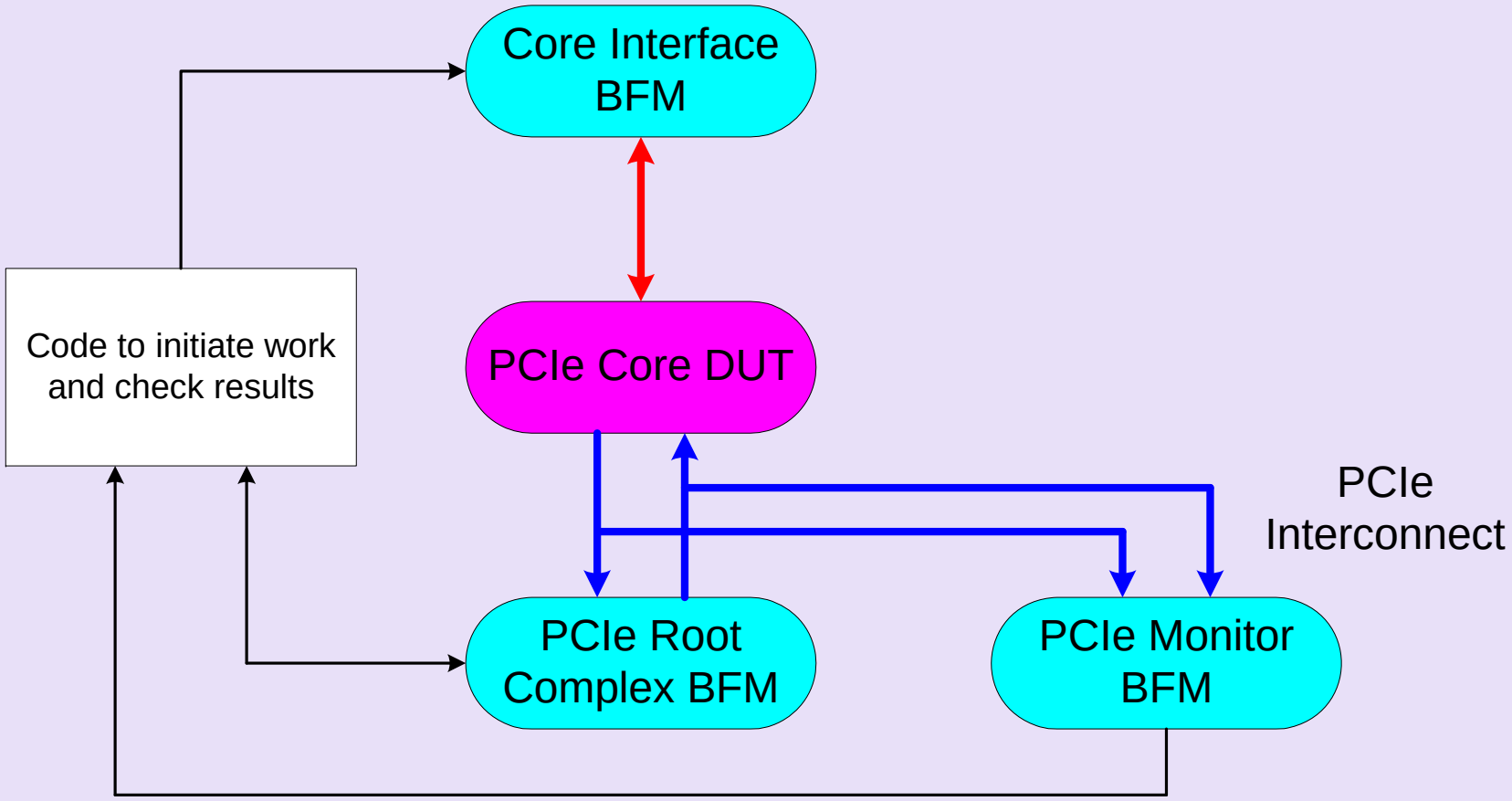
Disclaimer and Perspective

- All presented opinions, judgments, and recommendations are those of the presenter and do not necessarily reflect those of the PCI-SIG®
- The material in this presentation is derived from endpoint implementations, but much of it applies to other implementations
- Listed specification references are for PCI Express® Base Specification 2.0

Topics

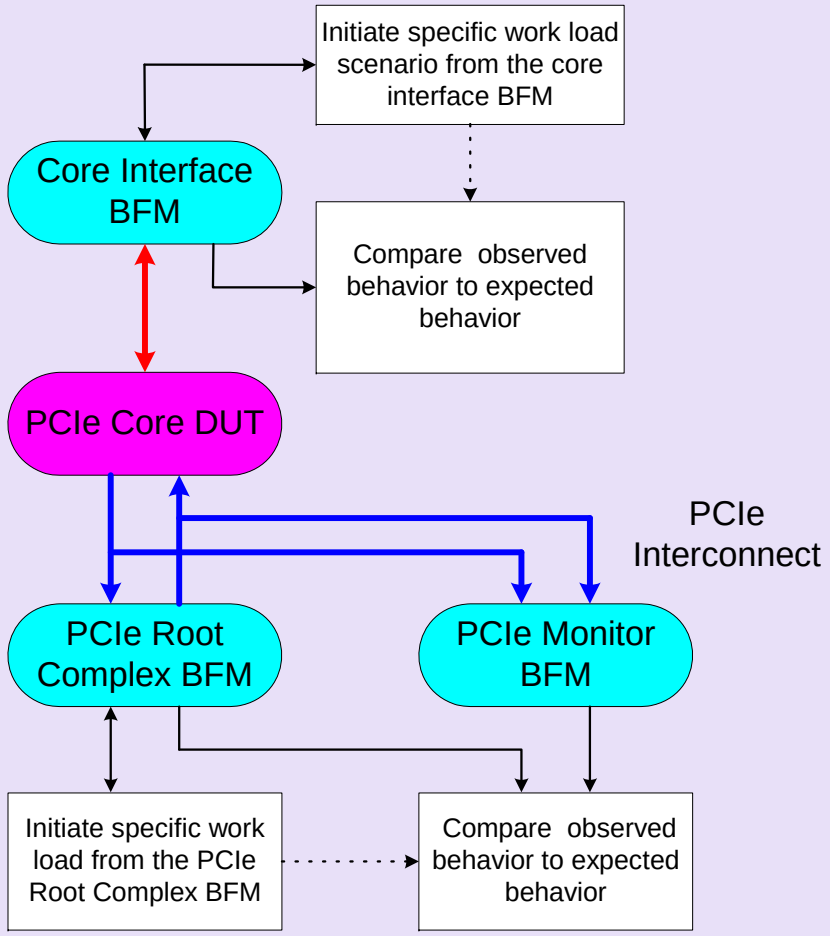
- Directed Simulation
- Random Simulation
- Determination of Variable Parameters
- Need for Error Injection
- Error Types
- Tips
- Conclusion

Verification Testbench



Directed Simulation

- Directed simulations
  - ✓ Verification engineer
    - Targets a specific function or scenario to be tested
    - Writes custom code to test that function/scenario
  - ✓ Examples
    - DUT receives a Configuration Write packet that contains an ECRC error
    - DUT transmits a large block of data that straddles the addresses from 32-bit address space into 64-bit address space.



Directed Simulation

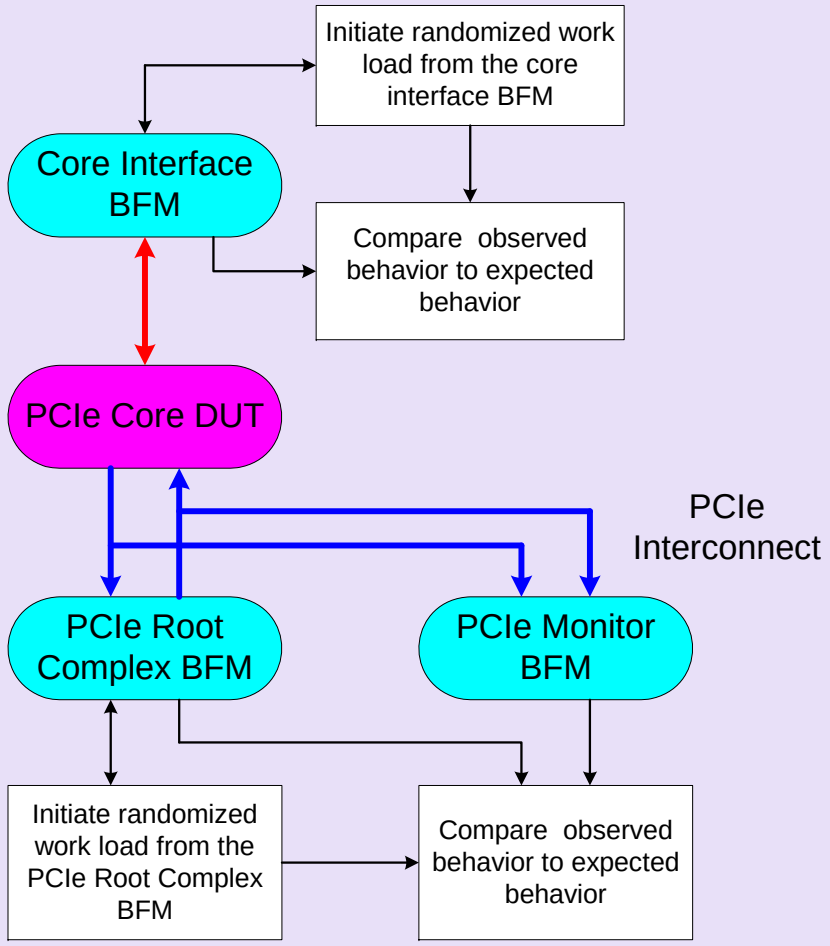
- Directed simulation strengths
  - ✓ Can target specific cases that need to be tested
    - Improve code coverage
    - Stress logic around known boundary areas
  - ✓ Are generally easy to debug
    - Simulation algorithm known/easily understood
    - Easy to determine when an error occurs
  - ✓ Can generally be developed quickly
  - ✓ Often the best approach for verification of resets and non-recoverable error handling

Directed Simulation

- Directed simulation weaknesses
  - ✓ Verification engineers can't be expected to anticipate and test every possible boundary condition
    - Multi-threaded scenarios
    - Timing between events infinitely variable
    - Event interactions difficult to predict
      - Two large packets received back-to-back may work fine
      - Two large packets with one small packet between them may cause a failure
  - ✓ A great number of simulations may be required to achieve adequate code coverage

Random Simulation

- Verification engineer
  - ✓ Writes work generator threads and scoreboarding code
  - ✓ Determines all simulation parameters that can be varied
  - ✓ Randomly generates values for those simulation parameters
    - At start of day
    - On the fly



Random Simulation

- Random simulation strengths
  - ✓ Once one simulation is working, many others can be put together in a short time
  - ✓ They can run for days at a time resulting in a greater number of test cases checked
  - ✓ Many combinations of workload and timing checked
  - ✓ Random simulations are designed to have no preconceived notions
    - – Beware of unintended restrictions

Random Simulation

- Random simulation weaknesses
  - ✓ Generally takes much longer to get the first simulation set up and working
  - ✓ Troubleshooting more difficult
    - Simulations often long
    - Simulations often multi-threaded
  - ✓ Fatal error handling techniques very difficult to test, especially for multi-threaded simulations
    - Scoreboarding code gets very involved when the scoreboard modules have to deal with numerous error conditions
    - Behavioral models have the same issues dealing with non-recoverable errors
  - ✓ Similar issues exist with resets
  - ✓ Simulations may not be as thorough as expected

Determination of Random Parameters

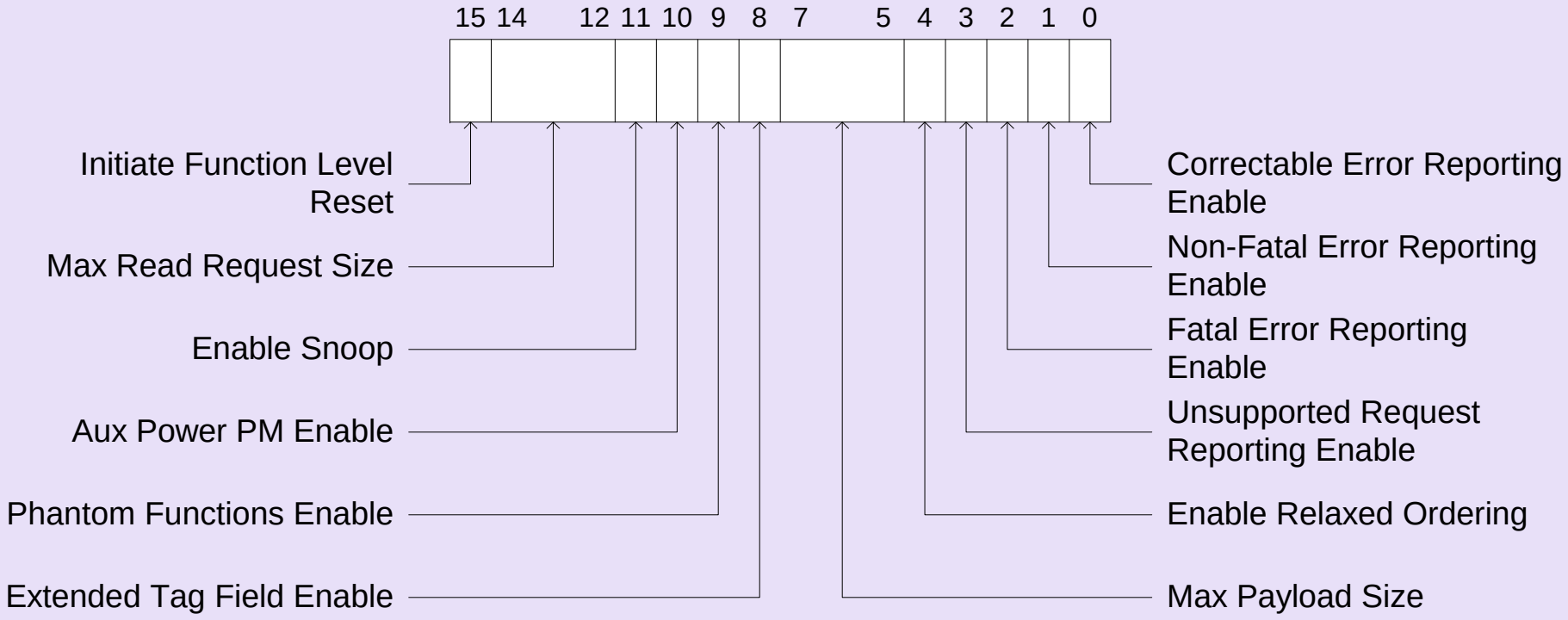
- Parameters can be varied
  - ✓ Under user control, generally at start of simulation
  - ✓ Randomly
    - At start of simulation
    - On the fly
- The list of parameters that can be randomized is very large
- Choosing which parameters to vary is a judgment call, but don't skimp

Determination of Random Parameters

- Places to look for possible parameters
  - ✓ PCI configuration space control registers
    - PCI Control Register
    - Device Control Register
    - Link Control Register
    - And so on
  - ✓ Other areas in the PCIe® specification
    - Physical Layer Specification
    - Link Layer Specification
    - Transaction Layer Specification
  - ✓ Specification for your core

Configuration Space Parameters

PCIe Device Control Register



Transaction Layer Parameters

- Examples of transaction-layer parameters
  - ✓ Section 2.3: values in TLP reserved fields must be ignored by the receiver
    - Randomize all transaction packet reserved fields to verify that the receiver ignores these fields
  - ✓ Section 2.2.8.7: Table 2-20 lists messages that should be ignored if they are received
  - ✓ Section 2.3.2: Receipt of an unexpected completion
    - The agent receiving the unexpected completion must discard the completion
  - ✓ Section 2.6: Vary the amount of initial credit granted to the DUT - including infinite credit

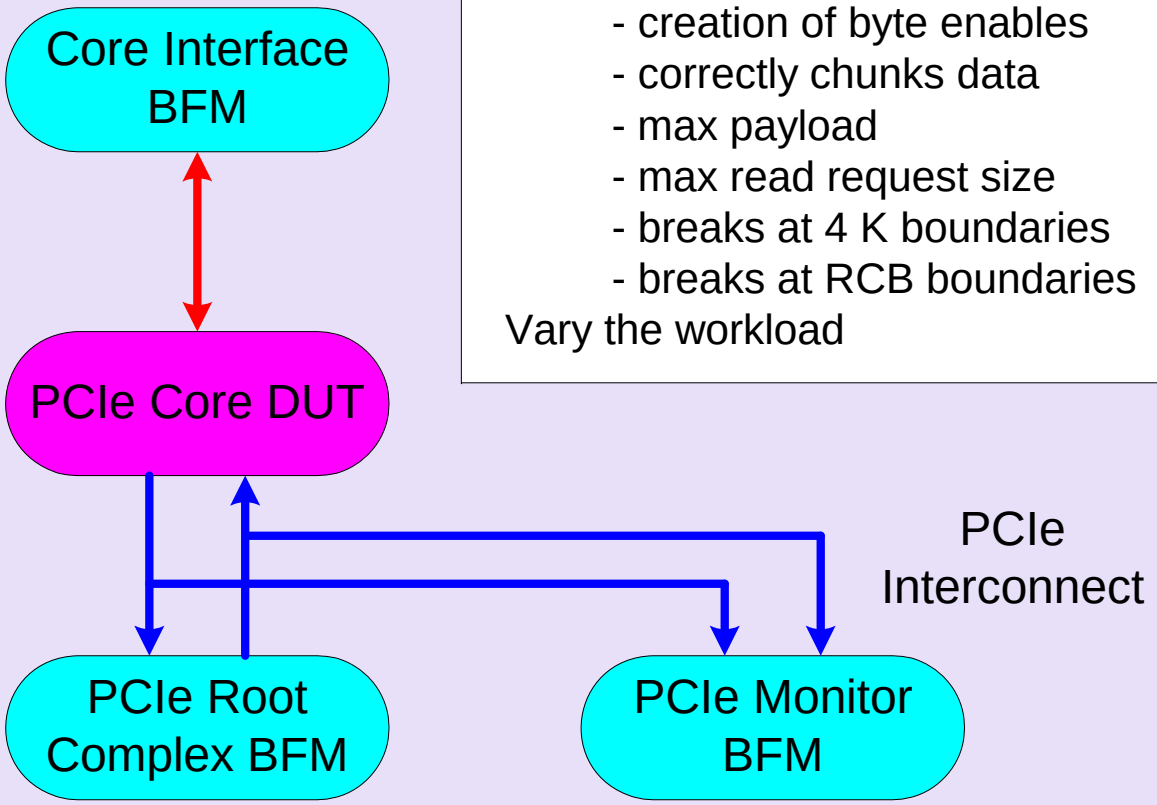
Transaction Layer Parameters

- ✓ Vary spacing between packets sent to DUT
- ✓ Send both 3 DW header packets and 4 DW header packets
- ✓ Send packets with ECRC and packets without
- ✓ Section 2.5: Components that do not support the Virtual Channel Capability structure must accept requests with non-zero TC and must return that TC in any completions it generates
- ✓ Send Vendor-defined messages
  - Type 0 are reported as unsupported if unimplemented
  - Type 1 are silently discarded if unimplemented

Transaction Layer Parameters

If applicaple, send work to core that tests it's ability to build packets
- creation of byte enables
- correctly chunks data
- max payload
- max read request size
- breaks at 4 K boundaries
- breaks at RCB boundaries

Vary the workload



Link Layer Parameters

- Examples of Link-Layer Parameters
  - ✓ Section 3.4.1: values in DLLP reserved fields must be ignored by the receiver
    - Randomize all DLLP reserved fields to verify that the receiver ignores these fields
  - ✓ Section 3.5.2.1: Ack/Nak latency
    - Acks and Nak transmission to the DUT can be delayed up to the replay timeout value (tables 3-4 and 3-5)
  - ✓ Section 3.5.2.2: A received DLLP that is not corrupt, but that specifies an unsupported DLLP type, is discarded without error
  - ✓ Section 3.5.2.2: Discard Ack/Nak DLLP if sequence number specified does to match an unacknowledged TLP or the value of ACK_SEQ

Link Layer Parameters

- ✓ Section 3.5.3.1: If the TLP sequence number is not equal to the expected value
  - Discard the TLP
  - Free storage allocated for the TLP
- ✓ Section 3.5.3.1: If the end symbol is EDB and the LCRC is the logical NOT of the calculated value
  - Discard the TLP
  - Free storage allocated for the TLP

Physical Layer Parameters

- Examples of Link-Layer Parameters
  - ✓ Section 4.2.1.3: 8b/10b decoding error – received symbol invalid
  - ✓ Section 4.2.2: Framing error
  - ✓ Section 4.2.4.1: Reserved bits in training sequences must be ignored
  - ✓ Section 4.2.4.6: If implementation specific number of 8b/10b errors encountered, symbol lock must be re-established ASAP
  - ✓ Switch back and forth between 2.5GT/s data rate and 5GT/s data rate

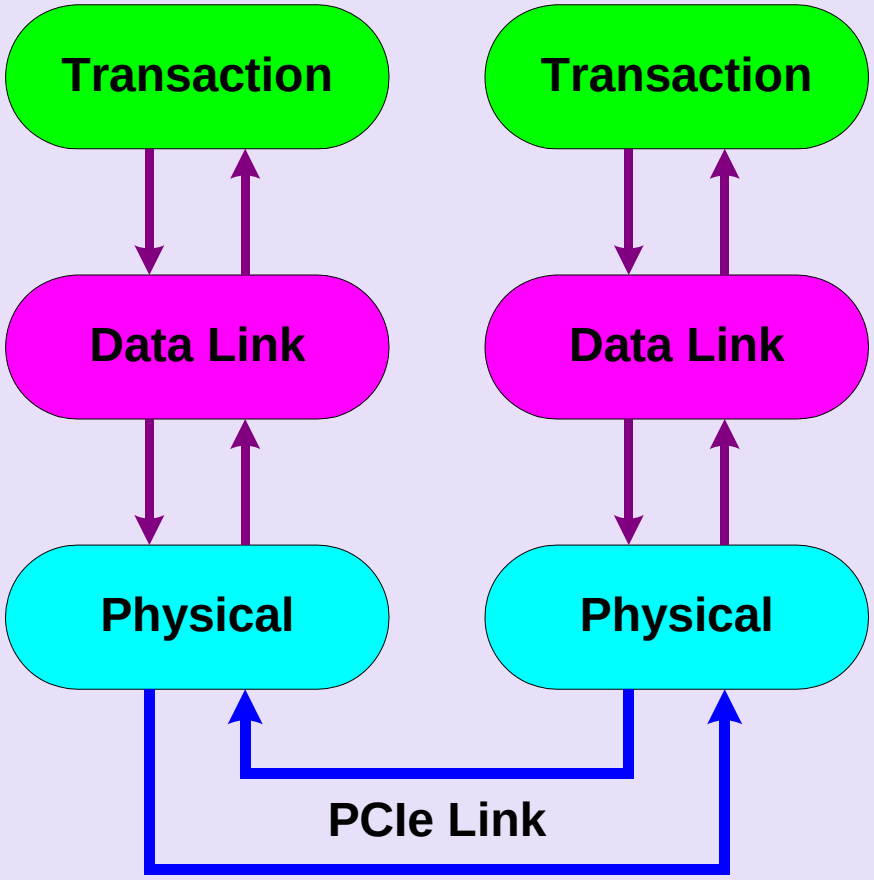
Need for Error Injection

- PCIe signals are subject to random and deterministic jitter even under ideal conditions
- Under less than ideal conditions, the BER can become much higher
- PCIe has built-in error detection and correction capability – it must be tested to insure it works
  - ✓ LCRC, ECRC
  - ✓ 8B/10B encoding
  - ✓ Sequence number
  - ✓ Unexpected completion
- Some applications cannot tolerate even one corrupted data bit.

Need for Error Injection

- Examples of problems uncovered with random error injection
  - ✓ Endpoint receives 3 back-to-back completions
    - Worked great when no errors present
    - Data corruption could occur if the middle packet had an LCRC error
  - ✓ On a system with a high BER,
    - The link would lose sync every so often and would then go through a re-train sequence
    - Under certain load conditions the PCIe core would start corrupting data after the retrain
  - ✓ When an EOF character was corrupted, found that data corruption could occur

Phy Layer and Link Layer Errors

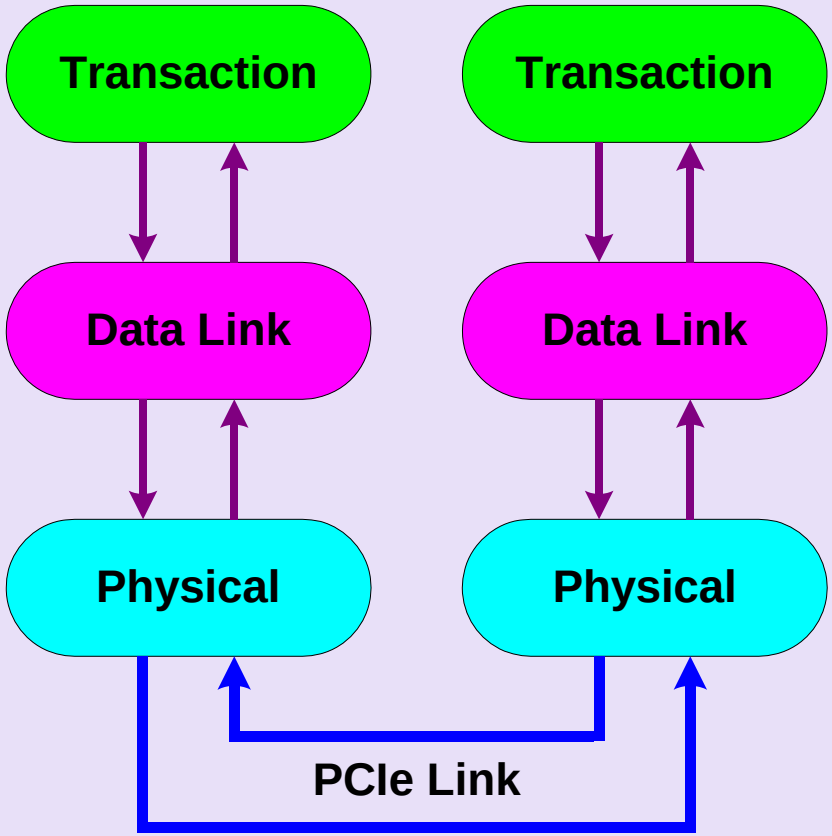


8b/10b
Framing
Lane Deskew
Loss of Symbol Lock
Elasticity Buffer
Overflow/Underflow

Sequence Number
LCRC
Nullified TLP
Data Link Protocol

Transaction Layer Errors

Unsupported Request
ECRC
Malformed TLP
Receiver Overflow
Unexpected Completion
Completer Abort
Completion Timeout
Poisoned TLP
Flow Control Protocol



Error Types

- Correctable errors are generally easy to insert
  - ✓ PCIe protocol designed to recover from such errors
  - ✓ Receiver responds to many Link and Phy layer errors by returning a NAK DLLP
  - ✓ Transmitter re-sends the packet from its replay buffer
- Examples
  - ✓ LCRC error
  - ✓ 8B/10B encoding error
  - ✓ Framing error

Error Types

- Some transaction layer errors can be easily inserted
  - ✓ Non-zero reserved fields
  - ✓ Completer receives a packet with a traffic class it does not support
  - ✓ Nullified TLPs
- Switching between PCIe data rates can also be inserted
  - ✓ To switch data rates, the link must retrain
  - ✓ No work in progress should be lost

Error Types

- Other transaction layer errors may require special error recovery procedures
  - ✓ ECRC error
    - Verify no data was written
    - Other actions are application specific
  - ✓ Unsupported request
  - ✓ Poisoned packets
    - Data should be discarded?
    - FC credits are consumed and freed
  - ✓ Malformed packets
    - Verify no data written
    - FC credits are not consumed or freed

Tips

- When building a random simulation environment, build in hooks for error injection
- Get the environment up and running without errors first
- Parameterize error injection
  - ✓ Allow errors to be selectively injected
  - ✓ Allow the frequency of errors to be controlled
- Certain errors (and resets) may be nearly impossible to add to the random simulation
  - ✓ Write good directed simulations to cover these errors

Tips

- Assertions are a valuable tool used with random or directed simulation
  - ✓ Insure that a desired condition is encountered
  - ✓ Insure that a protocol violation does not occur
    - Very useful for module I/O
    - Were any assumptions made by the designer violated?
  - ✓ Assertion examples
    - FIFO becomes full
    - FIFO becomes empty
    - FIFO never overflows or underflows
    - Signal X never on for more than one clock at a time
    - Signal X and signal Y are never simultaneously active

Conclusions

- The use of random simulations will often uncover “hidden” problems
  - ✓ Design engineer unlikely to test a mode of operation that was not foreseen during the design phase
  - ✓ Verification engineer not likely to know where the design weak points are
- Injecting errors into the random simulation helps insure a robust design
- Random simulation rule-of-thumb is to run simulations until no new errors have been uncovered for the last 2 weeks

QUESTIONS?

Thank you for attending the
PCI-SIG Developers Conference 2007.

For more information please go to
www.pcisig.com