PCIe® 2.0 Link Layer Test Concepts

Markus Zelleröhr

R&D Project Manager

Agilent Technologies

LTSSM Testing - Agenda

- The PCI Express® Standard
- How does PCI Express Link Training work?
- What the LTSSM challenges really are!
- What kind of test device and functionality you will need for efficient and effective testing
- How to force the LTSSM into various states so that the state transitions can be verified
- How previous and new test approaches and tools differ

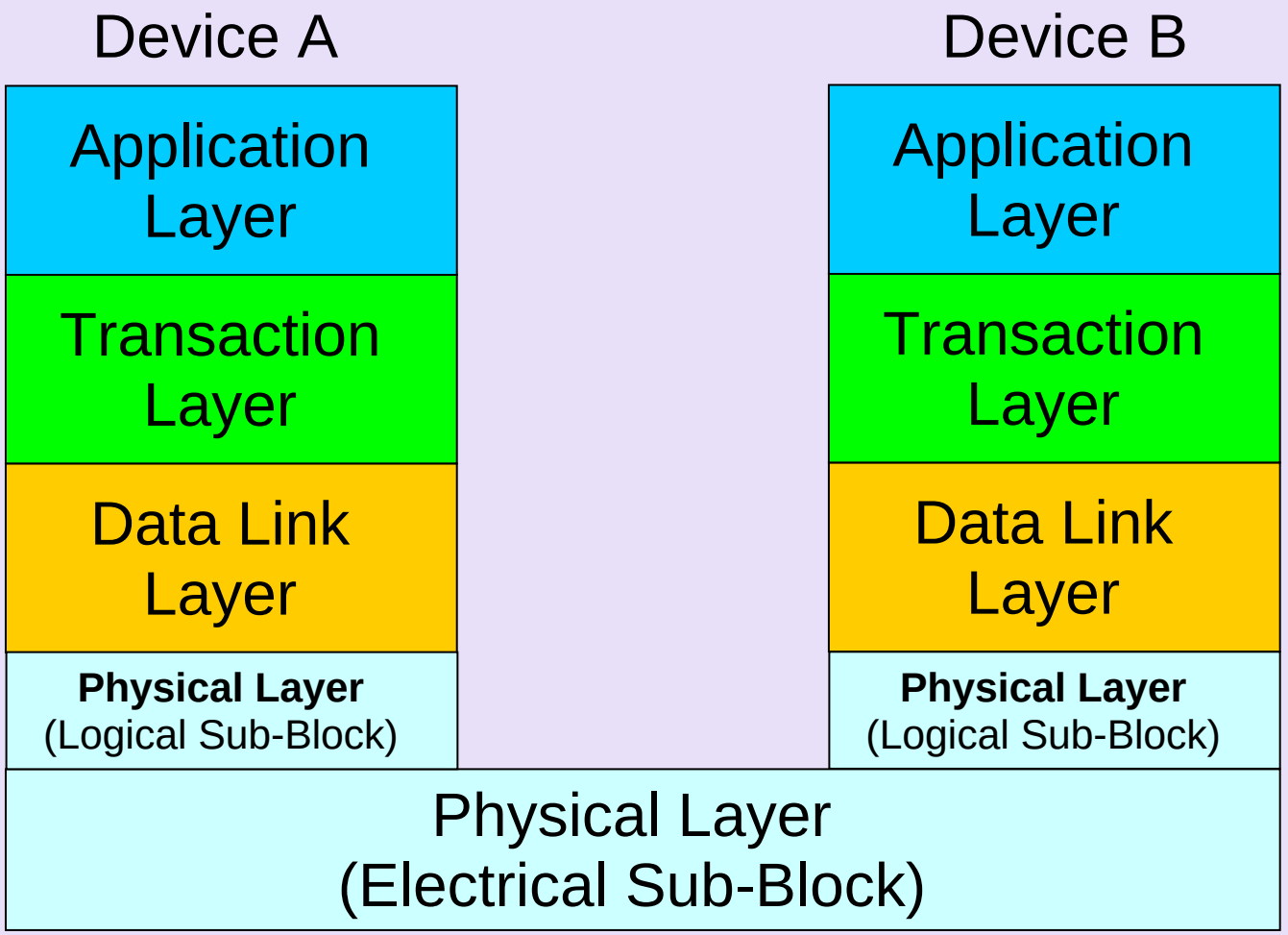
PCI Express – a standard is growing up!

- PCI maintained hardware backwards compatibility over 3 or 4 generations: PCI, PCI 66MHz, PCI-X™ 133, PCI-X 266/533
- PCI Express Standards evolved from 1.0a, 1.1 to 2.0 with now 5 GT/s, the latter one released in December 2006
- PCI Express has become more and more popular over the last three years
- Backward compatibility is a key success factor for the PCI standards

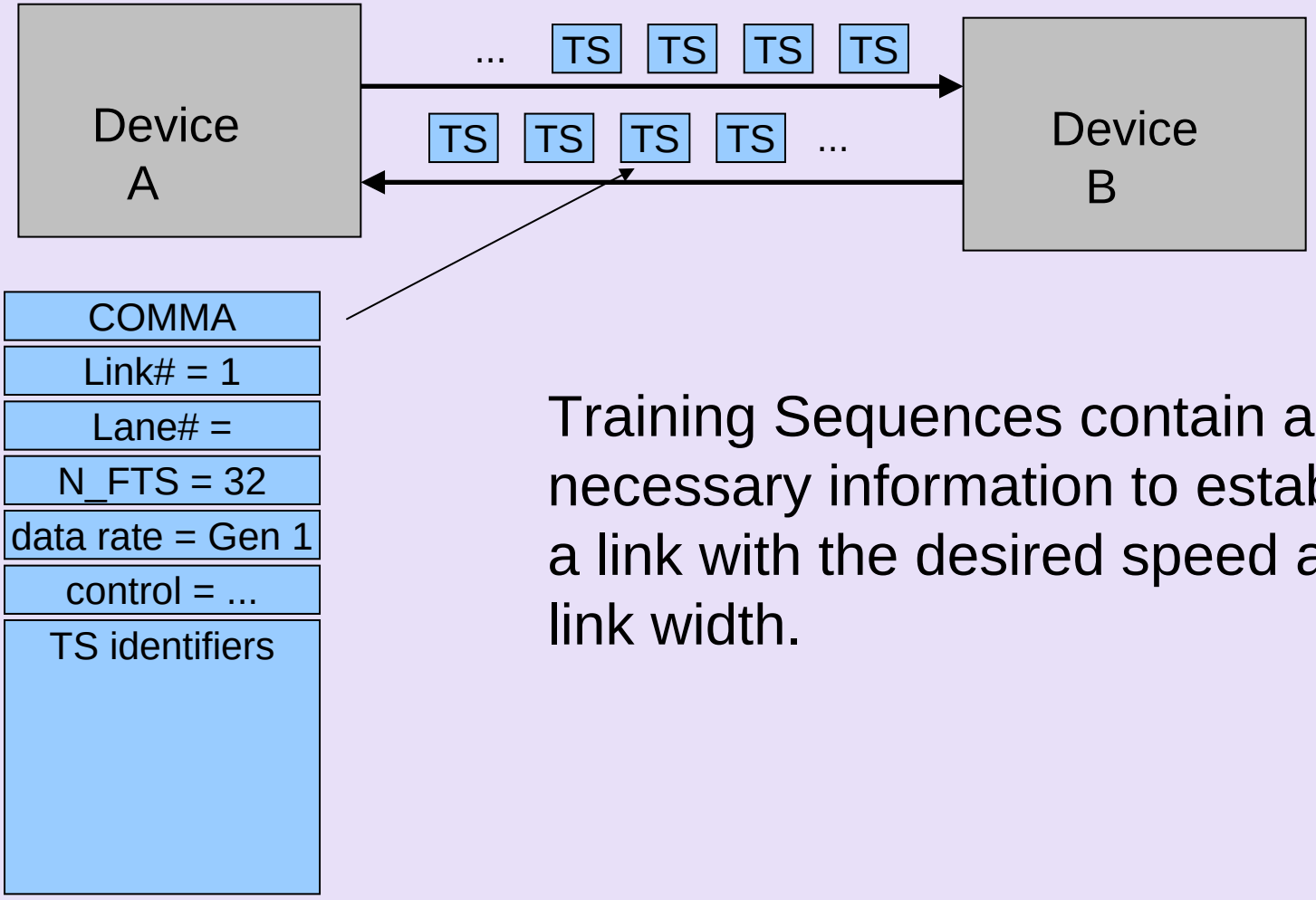
PCI Express – a standard is growing up!

- PCI Express 2.0 is an evolutionary step based on PCI Express 1.x
  - ✓ PCI 2.0 maintains hardware backwards compatibility with PCIe 1.x
- The link initialization process is reused from PCI Express 1.x. It has been enhanced to support the new link speed of 5 GT/s while maintaining backward compatibility.

How does link training work?

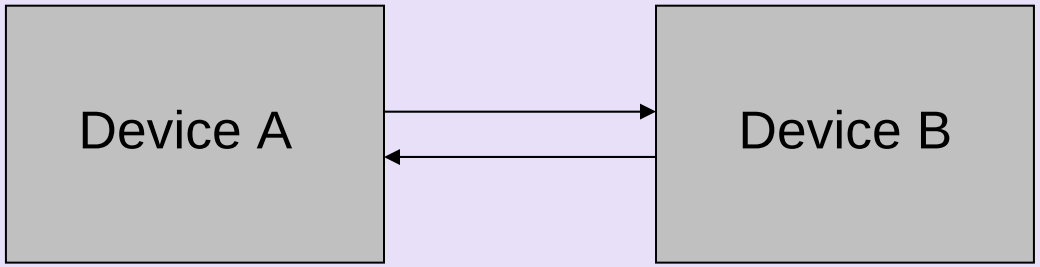


How does link training work?



Training Sequences contain all the necessary information to establish a link with the desired speed and link width.

How does link training work?



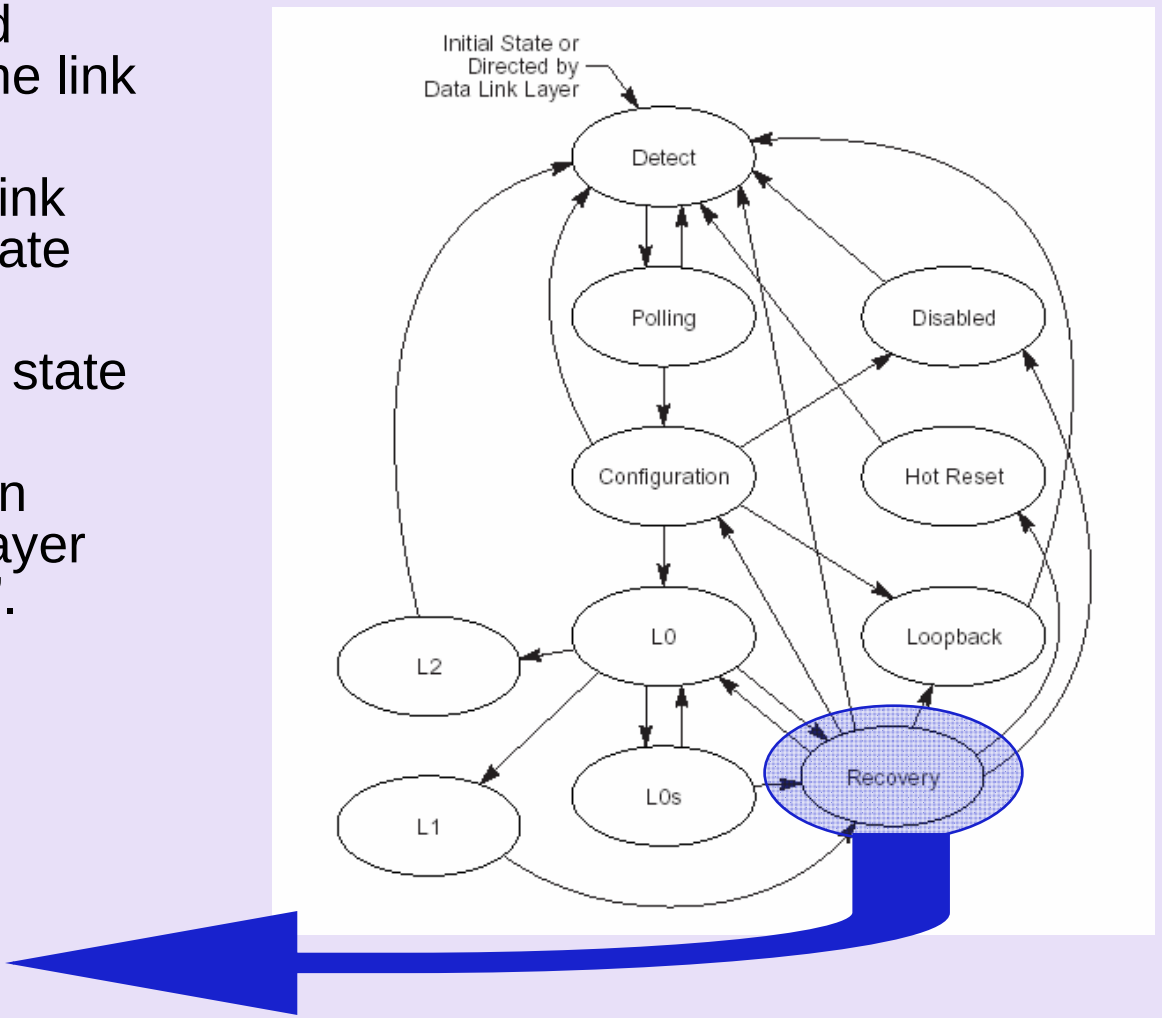
Two PCI Express devices exchange Training Sequences to negotiate link parameters like

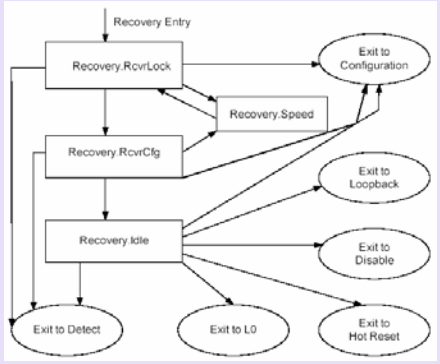
- lane polarity
- link number
- set of lanes that belong to the link
- lane numbers
- scrambler enabled or disabled
- link speed
- number of fast training sequences required
- ...

Training Sequences are also used to switch the link to low power states.

How does link training work?

- Each of the devices implements a so called LTSSM that controls the link training.
- LTSSM stands for “Link Training and Status State Machine”
- Link training starts in state Detect.
- An active link that can transport transaction layer packets is in state “L0”.





How does link training work?

- Example from the PCI Express 2.0 specification, Polling.Active state (p. 196):
  - ✓ “Transmitter sends TS1 Ordered Sets with Lane and Link numbers set to PAD (K23.7) on all lanes ... ”
  - ✓ “Next state is Polling.Configuration after at least 1024 TS1 Ordered Sets were transmitted, and all Lanes ... receive eight consecutive TS1 or TS2 Ordered Sets ... ”
  - ✓ “Otherwise, after a 24 ms timeout the next state is:
    - Polling.Configuration if, ...
    - Polling.Compliance if ...
    - Else Detect if the conditions to transition to Polling.Configuration and Polling Compliance are not met. ”

What are the LTSSM challenges?

- The LTSSM with all its states, substates, transitions and conditions is quite complex. The number of possible scenarios is immense.
- Link training is a dynamic process.
- The sequence and timing of state transitions is not fixed. The sequence differs with
  - ✓ different lane ordering
  - ✓ different timing behavior
  - ✓ signal integrity (occasional bit errors)
  - ✓ violations of the standard (error scenarios)
  - ✓ implementation specific behavior

Example: Transition coverage

- Timeout in Configuration.Idle:
  - ✓ Otherwise, after a minimum 2 ms timeout:
    - If the idle_to_rlock_transitioned variable is 0b, the next state is Recovery.RcvrLock.
    - The idle_to_rlock_transitioned variable is set to 1b upon transitioning to Recovery.RcvrLock.
    - Else the next state is Detect." (PCIe 2.0, p. 213)
- The transition to Recovery will nearly never occur. At the same time this transition creates a complex scenario with a link recovery during the link training.
- How do you find out whether a device implements the transition to Recovery.RcvrLock?
- How do you know whether the second time the transition to Detect is really used?

Example: More than one condition for the same transition

- There are cases in the LTSSM specification where two different conditions can cause the same state transition.
- Polling.Active to Polling.Configuration:
  - ✓ “Next state is Polling.Configuration after at least 1024 TS1 Ordered Sets were transmitted, and all lanes .. receive eight consecutive TS1 or TS2 Ordered Sets.”
  - ✓ “Otherwise, after a 24 ms timeout the next state is: Polling.Configuration if, …” (PCIe 2.0, p. 196)
- The second condition allows the LTSSM to progress even if there are signalling issues. A decision about link width is postponed to the next state.
- How do you make sure a device implements both conditions correctly?

Examples: Implementation specific behavior

- "At least a predetermined number of Lanes that detected a Receiver during Detect have detected an exit from Electrical Idle at least once since entering Polling.Active" (PCIe 2.0, p.197)
- "A device is permitted .. to infer Electrical Idle instead of detecting Electrical Idle using analog circuitry." (PCIe 2.0, p.182)
- "Note: It is recommended that any possible multi-Lane Link that received an error in a TS1 Ordered Set on a subset of the received Lanes delay the evaluation listed above by an additional two TS1 Ordered Sets so as not to prematurely configure a smaller Link than possible." (PCIe 2.0, p. 206)
- How do you make sure your device tolerates various behaviors of another PCIe device?

Example: Inferring Electrical Idle

- "A device is permitted ... to infer Electrical Idle instead of detecting Electrical Idle using analog circuitry."(PCI 2.0, p. 182)
- In Recovery.RcvrCfg electrical idle can be inferred by "Absence of a TS1 or TS2 Ordered Set in a 1280 UI interval." (PCI 2.0, p. 182)
- Some implementations support both
  - ✓ detecting Electrical Idle (analog circuitry)
  - ✓ inferring Electrical Idle
- How do you test whether the inferrence works as desired?

Example: Failing Speed Change

- When the speed of operation is increased one of the devices may fail to achieve symbol lock, e.g. due to marginal signal quality.
- PCIe 2.0 requires the devices to recover from such a scenario.
- Mechanism:
  - ✓ The device that can‘t achieve symbol lock transmits Electrical Idle.
  - ✓ The other device detects Electrical Idle and both devices revert back to 2.5G operation.
  - ✓ see Recovery.RcvrLock on p. 216
- How do you find out whether a device supports this mechanism?
- How do you test whether Electrical Idle inferrence works in this scenario?

Example: Testing robustness of an implementation

- In Configuration.Linkwidth.Accept the device decides which lanes participate in the link.
- The device looks for just 2 consecutive TSs on each lane.
  - ✓ What if the device sees lane-to-lane skew?
  - ✓ What if one lane has occasional bit errors?
- PCIe 2.0 recommends to delay the decision in order avoid a pre-mature decision.
  - ✓ It‘s a recommendation, not a requirement.
  - ✓ The amount of delay is implementation specific.
  - ✓ Implementations are different and one implementation is more robust than the other.
- How do you find out whether a device is robust in this sense or not?

What kind of test device do you need?

- Controllability
  The test tool needs to stimulate the DUT‘s LTSSM in order to perform certain state transitions and actions. Ideally, all state transitions can be reached. Some state transitions can be stimulated by changing static parameters, others require a certain dynamic behavior.
- Observability
  The test tool should be able to display training sequences on the link as well as the corresponding LTSSM states.

What kind of test device do you need?

- A test tool can control the DUT‘s state transitions
  - ✓ by using different static parameters
    - An off-the-shelf device typically has a fixed behavior. A test tool can vary parameters like link width and link speed capabilities.
  - ✓ by initiating link state changes under user control
    - Example: An off-the-shelf device performs a speed change at a certain point in time. With a test tool the user can control directly when a speed change takes place.
  - ✓ by emulating error scenarios
    - An LTSSM tester can create scenarios (e.g. timeouts) that an off-the-shelf device would only show if it was broken

What kind of test device do you need?

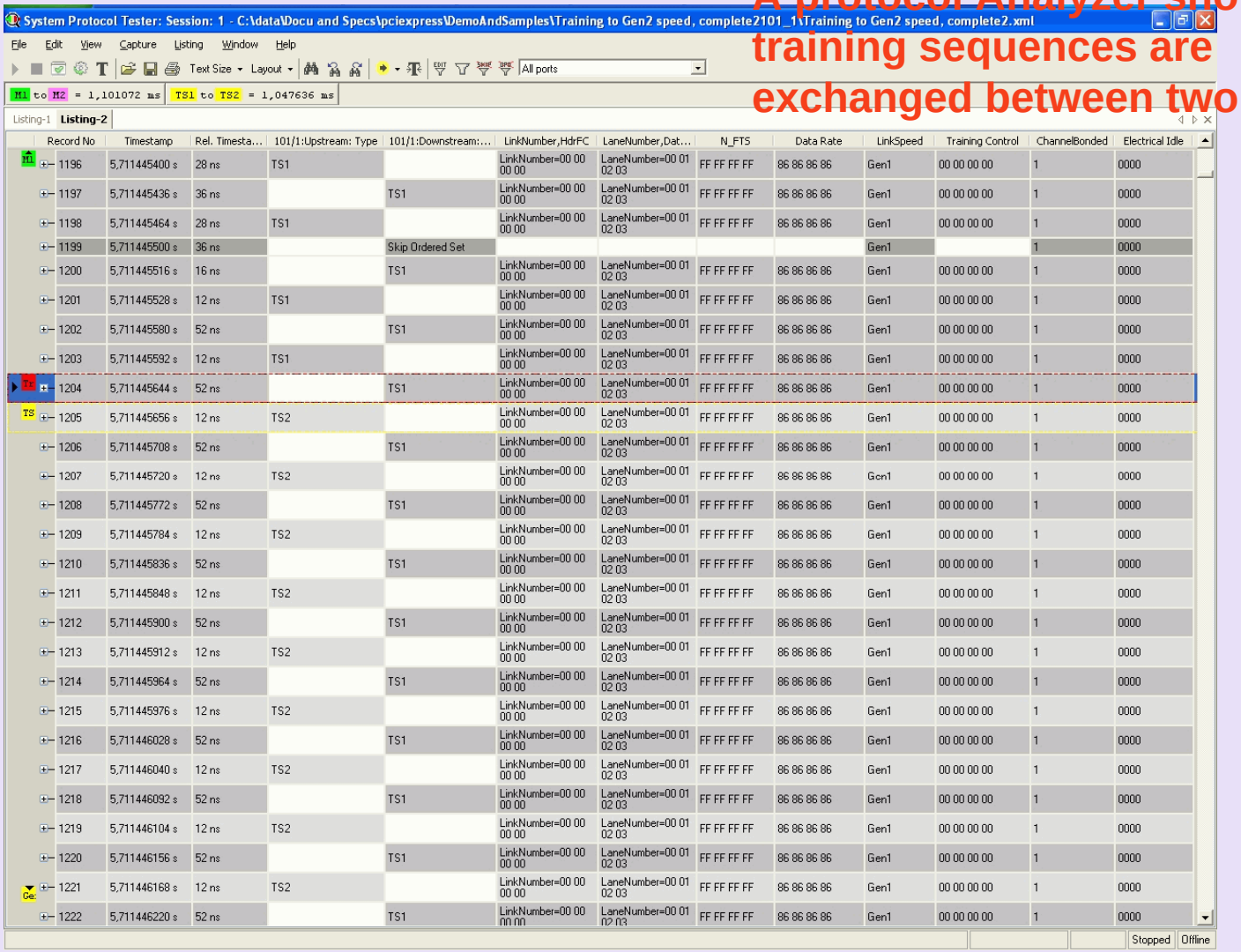
- A stimulus tool
  - ✓ should have flexibility to change its behavior on the fly
  - ✓ must be an interactive component, not a pattern generator that replays a recorded sequence
    reason: timing can change from run to run
  - ✓ should come with pre-canned tests so that a reasonable coverage can be achieved in a short period of time

What kind of test device do you need?

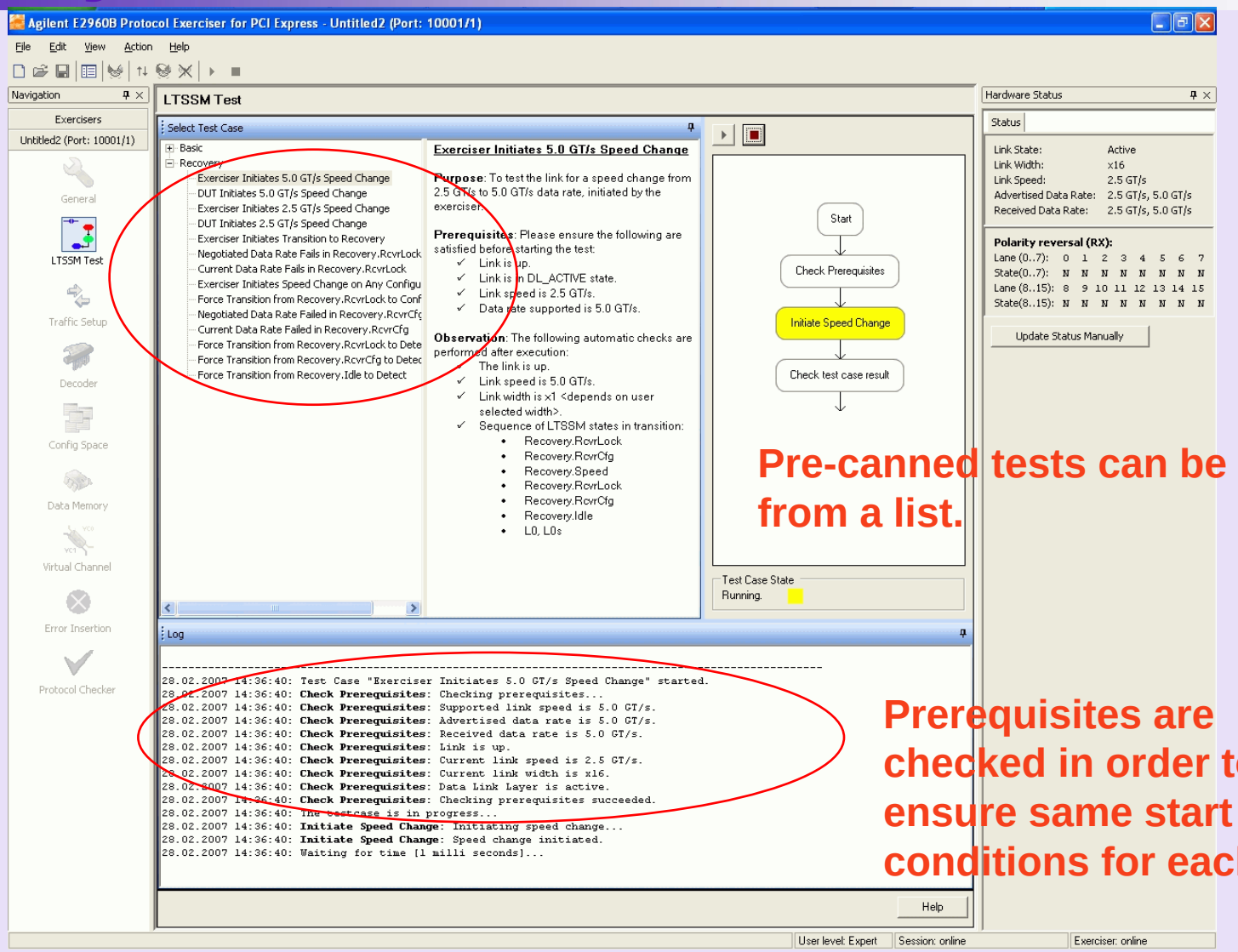
- A protocol analyzer can
  - ✓ observe and visualize training sequences
  - ✓ capture and display state transitions
  - ✓ detect and report incorrect state transitions

What kind of test device do you need?

A protocol Analyzer shows how training sequences are exchanged between two devices.



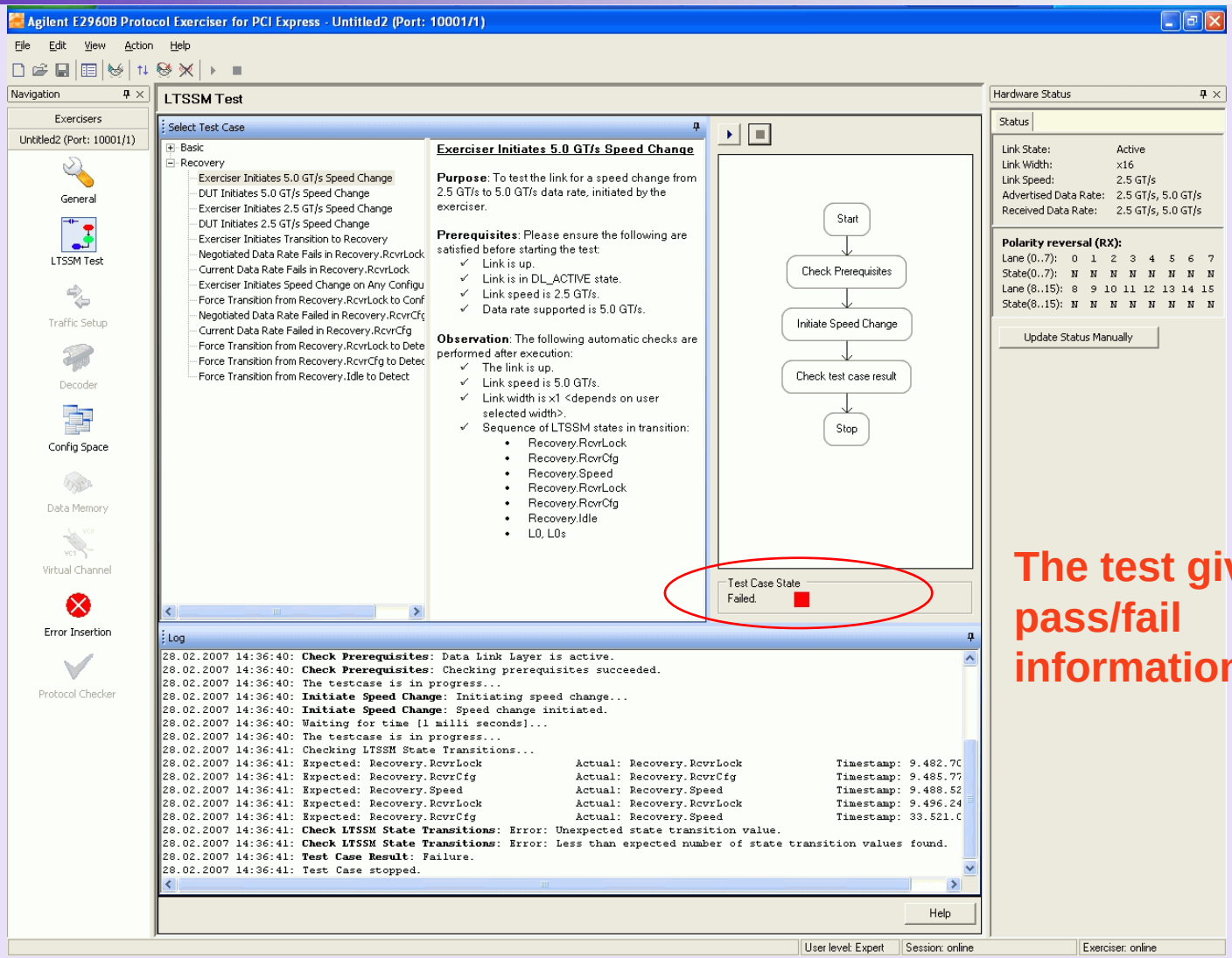
What kind of test device do you need?



Pre-canned tests can be selected from a list.

Prerequisites are checked in order to ensure same start conditions for each run.

What kind of test device do you need?



The test gives a pass/fail information.

What kind of test device do you need?

Log

```
--------------------------------------------------------------------------------------------------
28.02.2007 14:36:40: Test Case "Exerciser Initiates 5.0 GT/s Speed Change" started.
28.02.2007 14:36:40: Check Prerequisites: Checking prerequisites...
28.02.2007 14:36:40: Check Prerequisites: Supported link speed is 5.0 GT/s.
28.02.2007 14:36:40: Check Prerequisites: Advertised data rate is 5.0 GT/s.
28.02.2007 14:36:40: Check Prerequisites: Received data rate is 5.0 GT/s.
28.02.2007 14:36:40: Check Prerequisites: Link is up.
28.02.2007 14:36:40: Check Prerequisites: Current link speed is 2.5 GT/s.
28.02.2007 14:36:40: Check Prerequisites: Current link width is x16.
28.02.2007 14:36:40: Check Prerequisites: Data Link Layer is active.
28.02.2007 14:36:40: Check Prerequisites: Checking prerequisites succeeded.
28.02.2007 14:36:40: The testcase is in progress...
28.02.2007 14:36:40: Initiate Speed Change: Initiating speed change...
28.02.2007 14:36:40: Initiate Speed Change: Speed change initiated.
28.02.2007 14:36:40: Waiting for time [1 milli seconds]...
28.02.2007 14:36:40: The testcase is in progress...
28.02.2007 14:36:41: Checking LTSSM State Transitions...
28.02.2007 14:36:41: Expected: Recovery.RcvrLock          Actual: Recovery.RcvrLock          Timestamp: 9.482.704 ns
28.02.2007 14:36:41: Expected: Recovery.RcvrCfg           Actual: Recovery.RcvrCfg           Timestamp: 9.485.776 ns
28.02.2007 14:36:41: Expected: Recovery.Speed             Actual: Recovery.Speed             Timestamp: 9.488.528 ns
28.02.2007 14:36:41: Expected: Recovery.RcvrLock          Actual: Recovery.RcvrLock          Timestamp: 9.496.240 ns
28.02.2007 14:36:41: Expected: Recovery.RcvrCfg           Actual: Recovery.Speed             Timestamp: 33.521.040 ns
28.02.2007 14:36:41: Check LTSSM State Transitions: Error: Unexpected state transition value.
28.02.2007 14:36:41: Check LTSSM State Transitions: Error: Less than expected number of state transition values found.
28.02.2007 14:36:41: Test Case Result: Failure.
28.02.2007 14:36:41: Test Case stopped.
```

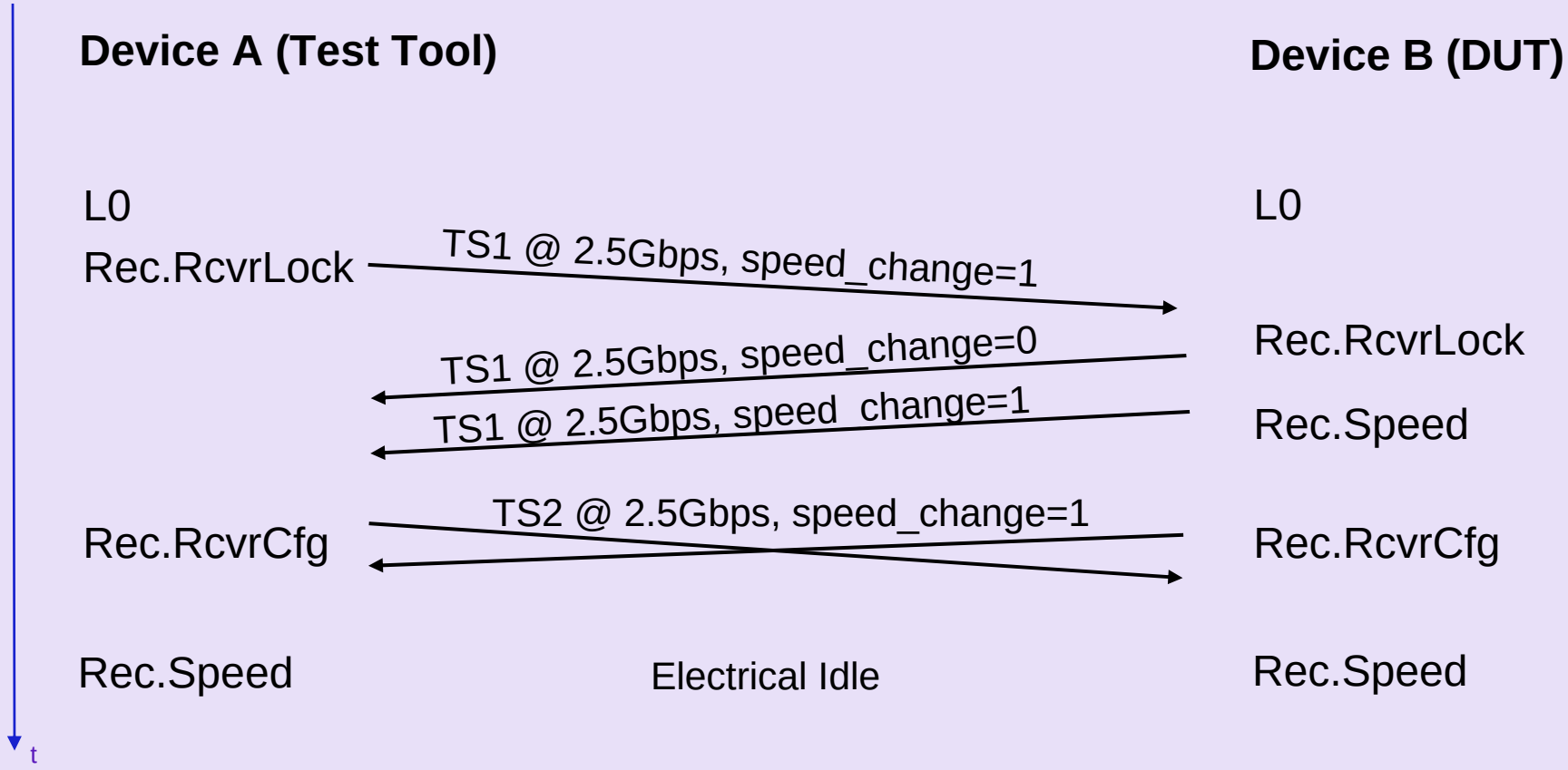
Information from the log file can be used to start the root cause analysis.

unexpected transition to Recovery.Speed

timestamps indicate a 24 ms timeout

How a test tool controls the DUT's LTSSM

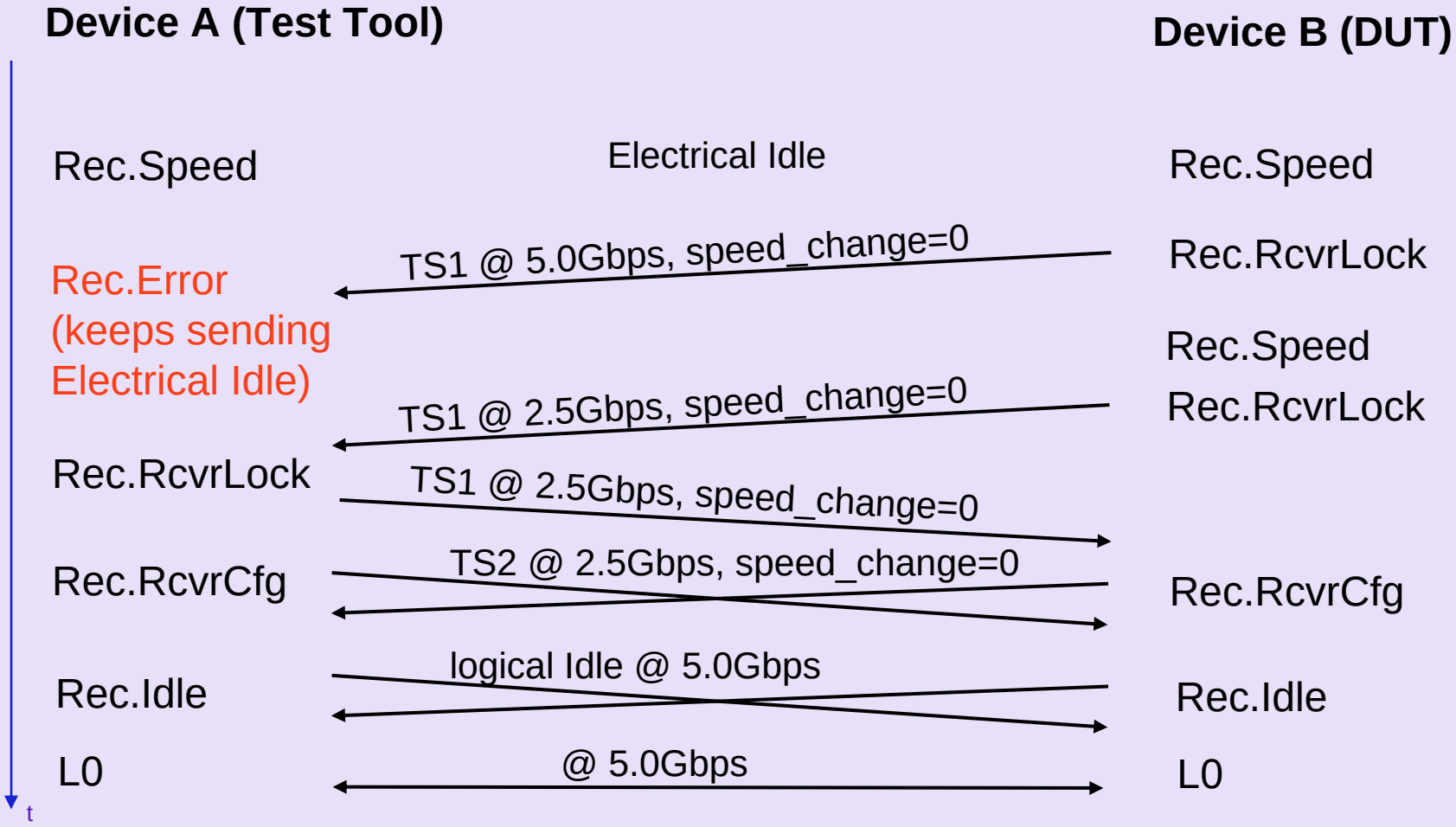
Example: Forcing a failing speed change



continued on next page

How a test tool controls the DUT's LTSSM - continued

continued from previous page



How previous and new test approaches differ!

- Previous protocol exercisers focused on the Transaction Layer and the Data Link Layer.
- Link training testing was limited to
  - ✓ using as many off-the-shelf devices as possible
  - ✓ using a test tool to change static link parameters
  - ✓ using pattern generators
- New test tools address the high complexity and the dynamic nature of link training and close the validation gap.

What to Remember about LTSSM Testing

- **What the LTSSM challenges really are!**
  - ✓ Complexity of the LTSSM. The devil is in the details.
- **What kind of test device and functionality you will need for efficient and effective testing.**
  - ✓ A test tool with the capability to create and check various link training scenarios.
  - ✓ An analysis tool that visualizes the link training.
- **How to force the LTSSM into various states so that the state transitions can be verified.**
  - ✓ The test tool can show various different behaviors, e.g. by injecting protocol violations.
- **How previous and new test approaches and tools differ**
  - ✓ The new concept tests features that were previously not even tested.

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

For more information please go to www.pcisig.com

PCIe 2.0 Link Layer Test Concepts

Markus Zelleröhr

R&D Project Manager

Agilent Technologies