

Leveraging Robust 3rd Party VIP for Successful PCI Express Compliance Verification

By Shaw Yang, Rajat Mitra, and Chih-Neng Chung

Verification Intellectual Property (VIP) streamlines the path to compliance sign-off through a reusable, layered verification methodology supported by highly configurable and feature-rich transactors, protocol assertions, advanced debug support, and comprehensive functional and compliance checklist test suites. This paper considers the role of VIP compliance testing for core- and chip-level verification of PCI Express-based designs to ensure robust device-level and interface compliance. With the major revision of PCI Express 2.0 (Gen-2) specification, compliance verification to Gen-2 features and to cross mode interoperability becomes critically important in chip and system development.

Overview

For a PCI Express product to be successful, it must be compliant to the standard specification published by PCI Special Interest Group (PCI-SIG) and be able to inter-operate with designs from different vendors. PCI-SIG provides a Compliance Program to validate a design to a certain level of specification compliance. Devices that fail compliance testing can be costly to the product introduction with delays in time to market. To insure a higher chance of success in compliance and interoperability, users can verify their design early during the design phase by using verification IP with robust compliance test suites.

The PCI Express Base Specification Revision 2.0 (Gen-2) specification [1] is a major revision most notably an updated LTSSM for speed negotiation and lane reconfiguration. The number of PCI-SIG Compliance checklist items has also increased significantly with approximately 1000 new items from 1.0a to 1.1 revisions and perhaps several hundred more in 2.0. These revisions put significant demand on robust compliance testing of Gen-2 functionality and on cross revision compliance testing between 1.0a, 1.1, and 2.0.

Considerations for Verification IP of PCI Express have been described in "Advanced Functional Verification and Debug of PCI Express-based Designs" by Chris Browy [2] in *Embedded Computing Design Resource Guide* / August 2006. A key component highlighted is the comprehensive compliance test suite. This white paper describes the need for a mature robust 3rd party compliance test suite and illustrates some of the important attributes needed in such a test suite.

Avery's PCI Express Xactor (PCI-Xactor) Verification IP has demonstrated high coverage of the PCI-SIG Compliance Checklist on many customer designs. And by having tested with the most number of reference designs in the industry, Avery's PCI-Xactor demonstrates proven interoperability and provides a robust comprehensive compliance test suite for the end users. This paper quantifies some of the measured coverage of PCI-Xactor and describes some of its capabilities in achieving high compliance.

PCI-SIG Compliance Program

PCI-SIG has the Compliance Program to encourage compliance with the PCI specification. PCI-SIG sponsors the Compliance Workshop that provide sessions for each add-in vendor and system vendor to run tests on their products to ensure interoperability. In addition, the PCI-SIG offers testing sessions to confirm a given product's compliance with the latest version of the PCI specification. In the PCI-SIG-sponsored testing sessions, a "pass" or "fail" is issued for each test performed. To pass the Compliance Workshop, the product must score a minimum of 80 percent for interoperability testing, and pass the PCI-SIG-sponsored tests.

In order to obtain a higher assurance that a device can successfully pass the PCI-SIG Compliance validation test, pre-silicon verification of that design using a robust and mature compliance test suite in simulation is critical.

1. PCI-SIG Documentation on Test Considerations

PCI-SIG published the following documents to provide a set of basic functional tests to assist compliance testing.

- Electrical Test Considerations for PCI Express Architecture Revision 1.0
- Configuration Space Test Considerations for PCI Express Architecture Revision 1.0
- Platform Bios Test Considerations for PCI Express Architecture Revision 1.0
- Link Test Considerations for PCI Express Architecture Revision 1.0
- Transaction Layer Test Considerations for PCI Express Architecture Revision 1.0

These test documents contain a list of Test Assertions and a set of Test Definitions pertaining to that Protocol Layer. These documents do not describe a full set of PCI Express tests and assertions and is in no way intended to measure products for full design validation.

Avery's Compliance Test Suite contains all the tests listed in these Test Consideration documents. These PCI-SIG tests comprise between 20- 30% of Avery's complete test suite. Significantly more tests were added by Avery to increase compliance coverage.

2. Specification revisions (Gen-2) and cross revisions testing

Since its initial release in 2003, the PCI Express standard has been revised several times, and with major enhancements recently introduced in PCI Express 2.0 (Gen-2).

Among the changes for Gen-2 are:

- General Gen2 LTSSM changes supporting speed up/down negotiation and link up/down configuration (Recovery)
- Polling, Compliance Data Rate
- Training Sequences changes for Autonomous change /link upconfig/Selectable Deemphasis bit, Electrical Idle Ordered Set Configuration
- Inferring Electrical Idle
- N_FTS changes (number of FTS ordered sets required by the receiver to obtain reliable bit and symbol lock when operating with a common clock)
- Link Error Recovery for certain LTSSM states
- Lane-to-Lane De-skew timing parameters and LTSSM states
- Link Bandwidth Management Status and Link Autonomous
- Bandwidth Status bits
- N_FTS Due to Timeout in Rx_L0s.FTS 4.2.6.10 Loopback in Gen2
- Compliance SOS transmission and Link Control2

Updating the models, test-bench and compliance tests to be compliant to the new revisions as well as ensuring cross revision compliance requires sizable continuing engineering over long durations. By using a 3rd party VIP, this investment can be directly leveraged by the end user and thus free up key resources for user specific design and verification needs.

3. PCI-SIG Compliance Checklists

PCI-SIG published the official checklists to provide a common measure for coverage. The 1.0a checklist was published in 9/14/2004. The 1.1 Checklists was recently released on 10/19/2006. Checklist for PCI Express 2.0 is being developed. The following table (Table 1) shows the categories of the checklist and the number of items in each category.

Table 1: Checklist Items and Platforms

Categories	1.0a Items	1.1 Items
TPL	15	65
TXN	343	488
DLL	88	156
PHY	273	337
PMG	131	187
SYS	169	286
CFG	319	760
Total	1338	2279

In the PCI-SIG Specification, the main categories are Topology (TPL), Transaction (TXN), Data Link (DLL), Electrical (PHY), Management (PMG), System (SYS), and Configuration (CFG). There are also some additional motherboard and electrical checklist specified not listed here. This paper focuses on compliance checks related to the logical functionality that are covered in the main 7 sections mentioned above.

Each checklist item may be appropriate and can be applied to multiple platforms: End Point (EP), Motherboard (MB), Root Complex (RC), Switch (SW), Bridge (BR), as well as BIOS. Approximately 60%-70% of the checklist has overlaps among the platforms.

There are ~1300 checklist items covering the main sections of the 1.0a spec and ~2300 items for version 1.1. Version 2.0 would likely add hundreds more items when it becomes available. Observations of the 2.0 Specification indicate that many more configurations checklist items may be added. The numerous additions in these revisions demand that a capable verification framework with test-bench and BFM be designed well to incorporate the new additions in an efficient manner.

Of all the checklist items specified, about 20% of them are not verifiable or appropriate in digital logical simulations. They are typically electrical in nature (such as voltage specifications), declarative in nature with no actionable checking, or system level programming usage model related beyond core-level feature. The remaining ~80% of the checklist are related to the logical behavior of a functional feature and thus can be verified using digital simulation. This paper refers to these digital logic related checklist items as the "Digital" portions of the PCI-SIG Compliance Checklist and focuses on them as the target of VIP verification.

These checklists provide a formal measure of compliance coverage on user designs. A robust verification framework with test-bench, BFM, and compliance test suites should cover these checklist items. The checkers can be implemented in the BFM models or in the test scenarios that make up the compliance test suites. By running a simulation of any subset or full set of a test suite, coverage on which of these checklists have been checked "triggered" and its pass/fail (protocol violated) should be automatically logged at run time and a summary report generated once the simulation is complete.

A verification framework should also measure compliance checklist coverage while running random tests. Any compliance check should be logged with pass/fail status. The coverage result indicates the robustness of the random tests and its test-bench as well as the quality of the design.

Avery has developed a powerful verification framework and a comprehensive compliance test suite. Most of the PCI-SIG compliance checklist items have been implemented. Greater than 85% of the Digital portion of the 1.0a Compliance Checklist has been checked on several real customer designs.

Challenges of achieving compliance validation

Any PCI Express design project has the option to develop their own test-bench and test-suites and/or use a 3rd party VIPs. With the new PCI Express Revision 2.0 Specification, significant efforts are needed to verify that a design is compliant to the new specification. The following shows some of the key considerations in choosing a proper approach. Using a mature robust 3rd party compliance test suite to verify a design is an efficient and cost effective way to obtain high design quality and specification compliance.

1. Different abstractions with varied human interpretation

Software developers, architects, designers and design verifiers all interpret the PCI Express specification in their own way. Differences in human interpretation can be resolved early with detail design reviews and with design verification simulations, or late in post silicon HW lab debug. While having a separate in-house verification team to verify a design helps in double checking interpretations, the possible tainting of incorrect information between the design and verification engineers while working in the same environment can still occur. Given the huge expense in silicon fabrication and productizing costs, problems found and fixed earlier in the product design cycle is much more desirable than at a later stage. A 3rd party compliance test suite that tests all levels of abstraction, from the high application level to the low PHY layer signal level with maximum controllability and observability, provides the best way to bridge the possible human misinterpretations.

2. Different product segments with designer bias

PCI Express has a rich set of features that are applicable to a wide range of different product segments. Each product segment has its particular focus and terminology. Multi-CPU servers require reliable error system logging and recovery. Power sensitive mobile computing such as laptops requires advanced power management. Graphics engines require guaranteed bandwidth. A comprehensive verification suite should cover feature sets used in all application space. With the different product focus, designers may operate with their own product bias and terminology making spec compliance difficult to achieve. Testing with a VIP that has been used widely by different types of customer removes any product bias and provides the best assurance for compliant success.

3. Wide variety of system topologies and configurations

Designs may be compliant in one system configuration and not in another. Hidden configuration compliant problems may not be detected until the device is configured in that particular mode in the lab or worse in the market. A good compliance test suite should allow for a wide range of configurations (varying EPs, RCs, Switches, non-transparent bridging, etc) to be easily configured and tested.

Design-under-test (DUT) should be verified against all realistic system topologies including simple cross-link, multi-hosted embedded systems, and switch-based topologies. All types of PCI Express Transaction Layer Packets (TLPs) and Data Link Layer Packets (DLLPs) should be generated. Some types of compliance tests require robust controls over BFM operation at all protocol layers and back-end completion generation are needed. Host functions such as advanced PCI BIOS features such as bios enumeration and application driver operations including DMA are helpful.

4. Design Modifications

IP core design may be modified to add additional specification features. It may also be modified for better synthesis, timing, or testability. The synthesized gate level design may also be modified for better place and route, re-buffering, clocking, scan testability, or other engineering changes (ECOs). These changes may alter the core's compliance to PCI Express specification. By running full compliance test suites on these new design revisions would prevent non-compliant design bugs from slipping through at the late design stages before silicon fabrication.

5. Significant resource/effort needed to develop and maintain a full compliance test suite

The functional verification of PCI Express logic cores and the chips and systems utilizing them requires significant resource investment. The tasks include the development and maintenance of a verification framework comprised of BFM, assertions, test suites, DUT integration, checklist, and debug methods to isolate design bugs in different protocol layers. Avery Design's experience shows that the engineering effort to develop a capable test-bench with comprehensive test suite requires at least 15 staff-years.

6. Specification revisions and cross revision testing

PCI-SIG has produced several major revisions of the specification with 1.1 and 2.0. Models and tests must conform to multiple revisions and inter-operate in mixed-mode configuration to verify backward compatibility. This puts additional challenge to compliance testing. Test-bench and models must be architected and implemented to scale with the revisions and to provide mixed mode testing. Sizable effort is needed to update the functional verification models, test-bench and compliance tests to the latest specification errata, revisions and checklists.

In summary, choosing a verification strategy for PCI Express designs should address all the needs mentioned above. With the costs of multiple ASIC re-spins exceeding one million US dollars, higher assurance upfront that a design is fully compliant to standard specification is valuable.

Merits of Avery's PCI-Xactor Compliance Environment

Avery Design has developed a robust verification IP, PCI-Xactor, which addresses the needs and concerns listed in the previous section. The following sections describe some of the essential attributes of a capable compliance verification environment. The merits of Avery's PCI-Xactor that fulfill these requirements are described. Customers can choose to benefit from this mature and available technology to significantly increase assurance that their design is compliant to the latest PCI Express specification.

1. Used widely in Industry

By taking a proactive approach, Avery has become an industry leader with more IP design core vendors using the PCI-Xactor platform for their internal development than any other VIP vendor.

Avery's PCI Express Xactor test-bench and compliance test suite have been available for nearly 4 years since the beginning of 2003. It has been used to verify more than a dozen unique core controller designs by users from the graphics, storage, and networking industry. These 12 designs are comprised of mostly non-IP vendor core designs and 4 different third party commercial IP vendors' designs. Different companies' views and interpretations on the spec have been checked and resolved. By successfully verifying these designs, the interoperability and compliance are demonstrated and the experiences from this variety and depth of checking have been incorporated into Avery's VIP. In verifying with Avery's compliance test suite, the end user can benefit from the quality and maturity that is available in the product.

2. High Compliance Checklist Coverage

In testing with Avery's PCI-Xactor Compliance Test Suite on real customer designs, functional compliance test coverage of greater than 85% on the Digital logic portion of the PCI-SIG Compliance Checklist was achieved.

Avery took a proactive and customer driven approach to achieve this high level of compliance. Avery worked closely with multiple IP core vendors and early end users of PCI Express to verify that their design feature set completely comply with the Specification. Avery implemented the Digital portion of checklists in its BFM and in its test scenarios. Measured coverage of checklist items from simulation were rigorously reviewed with the customers. Additions to increase coverage were driven by customer's needs.

The result of this early and proactive customer engagement is the high compliance coverage of Avery's PCI-Xactor. On a typical end-user's EndPoint design, out of the 811 Digital checklist items Avery's compliance test suite has triggered 691 items (85%). Avery's revisions 1.1 and 2.0 checklist coverage should exceed 80%. Customers have reviewed Avery's compliance checklist coverage and are satisfied with the results, stating that they exceed protocol checking when compared with other alternatives they have employed.

Table 2: Summary Checklist Coverage

<i>Platform</i>	<i>Revision</i>	<i>Total</i>	<i>Digital Items</i>	<i>Checked</i>	<i>% of Digital Covered</i>
Endpoint	1.0a	967	811	691	85%
	1.1	1492	1200	905	75%
	2.0 (estimate)	1700	1300	1000	77%

The Digital coverage for Root Complex and Switch are typically 10% to 20% below that obtained on an EndPoint. There are practical limits that prevent coverage to reach near 100%. It is common for designs to choose not to implement the full feature sets specified in the spec.

For example, for an EndPoint design, the following optional features are often partially or not implemented:

- Limited number of VCs
- Active state power management
- Advanced error reporting
- Multi-function
- Crosslink
- Access control services

In typical root complex designs, further restrictions occur in implementation decisions. The following features sometimes are left out further reducing compliance coverage.

- RCRB
- Integrated endpoints
- Event Collector
- Multi-root ports
- Hot Plug
- Peer to peer forwarding

And for a Switch design, the following features are optional for implementation:

- Isochronous
- Multi-port arbitration

End Users desire an automated way of viewing checklist coverage from their simulation runs. Avery PCI-Xactor automatically generates a detail spreadsheet with dynamic coverage listing which checklist items were checked and a few sample test scenarios that triggered the item.

The following spreadsheet (Table 3) illustrates a snippet of a typical compliance checklist coverage report on an End point device design.

Table 3: Checklist Spreadsheet Report

Checklist	Checker in BFM	Checker in Testcase	Dynamic Coverage Report	Scenario	Description
PHY.02.06#01	Y		Y		Training sequence ordered-sets are never scrambled but are always 8b/10b encoded.
DLL.05.02#01		Y	Y	link_test521ep.vh	of its RETRY_BUF as soon as (after finishing the current TLP in progress) a NAK is received
TXN.02.05#01	Y		Y	s_enum_address.vh	ID routing is used with Configuration Requests
SYS.01.02#04		Y	Y	s_msi_test_ep.vh	produce interrupts must support the 64 bit version of the MSI (or MSI-X) capability structure.
PMG.03.02#02	Y				state whenever the Memory Space Enable, I/O Space Enable, or Bus Master Enable bits have been set.
CFG.08.08#02	Y	Y	Y	config_1_5.vh	The Negotiated Link Width field must correctly indicate the negotiated width of the PCI Express link.

Column A lists the PCI-SIG checklist items.

Column B shows whether the checklist item is implemented in the Avery BFM. One or more test cases (direct or random) can trigger this item.

Column C shows whether the checklist item is implemented in the Test case.

Column D shows the Dynamic Coverage Report: "Y" denotes that one or more test case(s) triggers checklist checker; "N" denotes that no compliance test triggers checklist checker.

Column E lists a few of the test scenarios that triggered the checklist checker.

Column F provides a description of that checklist item.

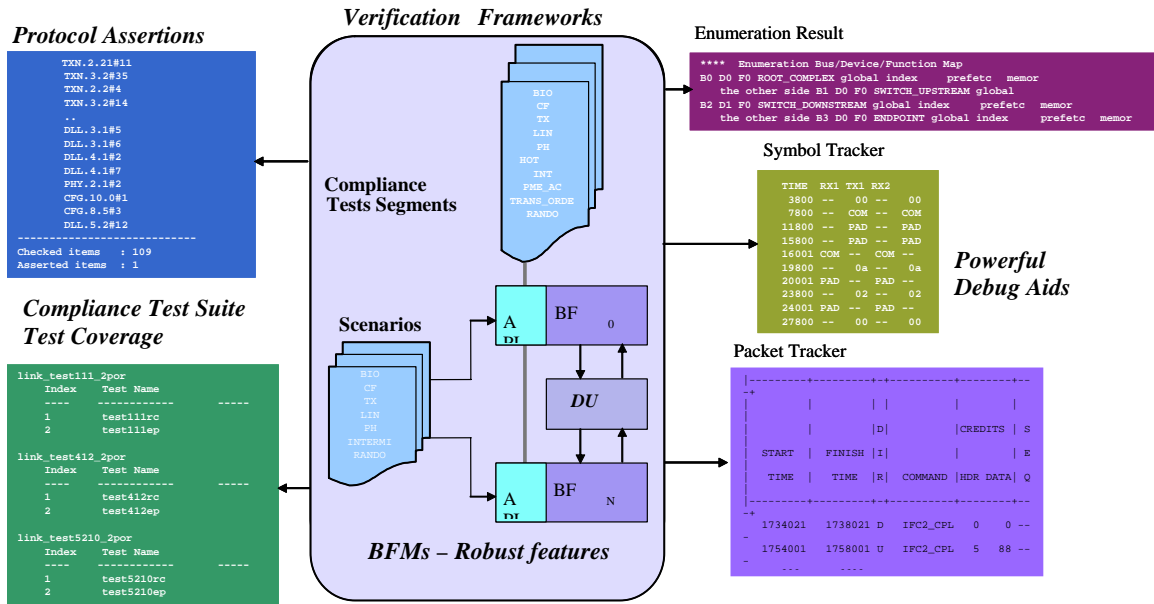
Avery continues to take the lead in Compliance testing with focused engagements with early adopters to implement the new features and checklists.

3. PCI-Xactor supports robust features enabling comprehensive testing

The PCI-Xactor test environment is a set of behavioral models (BFMs) and test suites that simulate the behavior of the PCI Express components. The models, provided in Verilog HDL, are tools for system designers to exercise and debug the design of components/systems based on the PCI-SIG standard. Designs under verification (DUTs) can be verified against all realistic system topologies including bridge and switch-based topologies. The test environment, provided in Verilog, provides a high-level test API to interact with the behavioral models supporting PCI Express root complexes, endpoints, and switches. It has a SuperMonitor model which passively monitors and reports PCI Express protocol violations, validates end-to-end transactions, and measures and reports transaction trace analysis of devices by address, bytes

transferred, and command types utilized. In addition, the test environment includes a full suite of compliance test scenarios that verify endpoint, switch, and bridge designs are comply fully with the PCI Express specification. Functional compliance test coverage and assertion/checklist coverage are also provided giving a formal measure of DUT compliance. Tests are highly reusable on any design/topology used in writing them based on an innovative scenarios based methodology. Both random and directed test-cases to stress the design are supported. Tests can be randomized at the TL/DLL/PHY layers as well at the test segment and scenario-level.

Figure 1 Verification Framework



4. Open Verilog source fully available

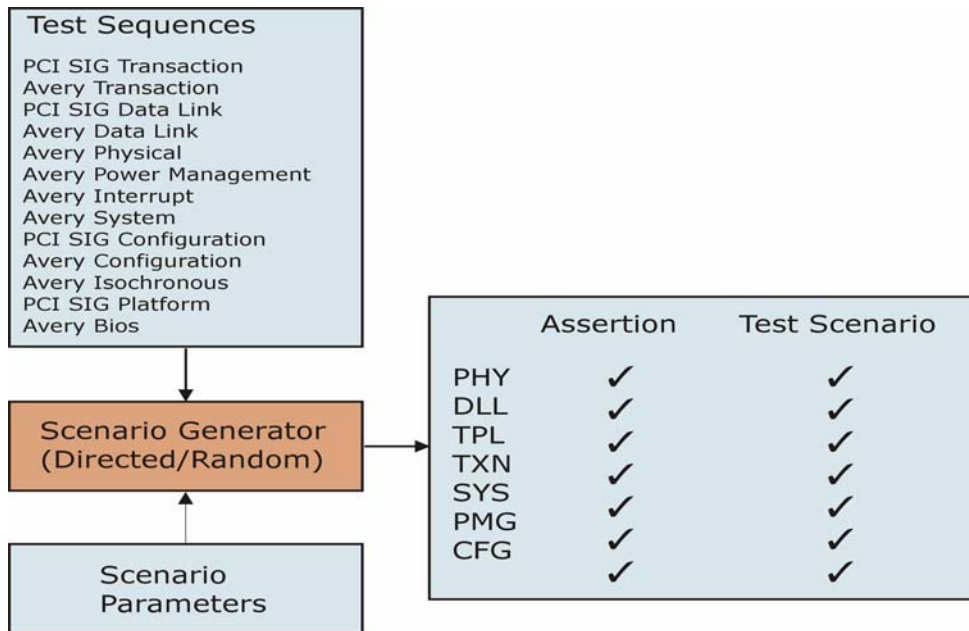
For maximum flexibility and ease of understanding and debug, the BFM and Compliance Tests are in Verilog source and provided to the user. Full visibility into actual model and test functionality is available. While verification may be based on a black-box approach, debugging is fully transparent for higher efficiency. Tracing through tasks provides thorough understanding of how a test sequence is executed. Test sequences are documented in html format with URL pointers so that the user can traverse through the test descriptions via web browser. With all the code available in Verilog, the user does not have to go through the complex interfacing between Verilog and other languages along with multiple environments to debug their design.

Checklist items are also fully documented and coverage reports on what's triggered, passed or failed are accessible to the user.

5. Layered verification with re-usable test sequences

Compliance tests can be used to assist in the verification of each PCI Express component type and design. Portable tests with parameterized test sequence libraries supporting a rich set of transaction sequences for each of the PCI Express protocol layers and major functions best exercises the DUT. As shown in Figure 2, it is desirable to have test sequences that can be combined under directed or random modes to create a complete set of tests to verify a design against the expansive PCI-SIG compliance checklists. End users can reuse these parameterized test sequences as building blocks to create a custom device-level test that meet their specific test needs and as a result lowers development and debug time.

Figure 2 Scenario generation linked with compliance coverage



6. Avery's enhanced test suites

Using the advanced layered testing methodology described, Avery has enhanced numerous tests in every functional category far beyond those tests suggested by PCI-SIG. These tests are readily applicable to various platforms: RC, EP, and SW. Sample test scenarios are shown in the following table. Source “PCISIG” denotes tests specified in the PCI-SIG Test Consideration documents. Source “AVY” denotes Avery's enhanced tests to cover specific functional features or design corners.

Table 4: Sample Compliance Scenario Tests

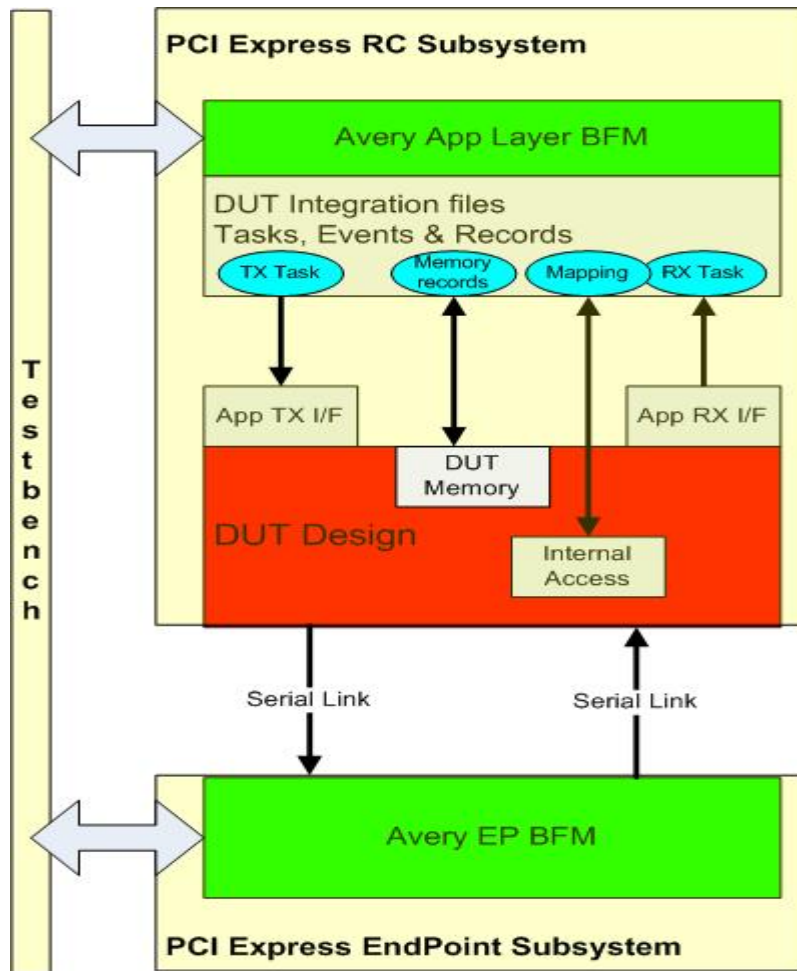
Source	Type	Document or Sub-Type	Scenario
PCISIG	TXN	PCI_Express_Test_Spec_Transaction_Layer_1.0.pdf: - 7:TXN_BFT_LegacyInt [Test 2.1] [Sec 5.2.1] - See PCI SIG Transaction Test Considerations ver 1.0	txn_2_1.vh
AVY	TXN	DMA READ/WRITE-5: DMA Write and Read-Verify -7: Perform 4KB write and read-verify use random TLP data payload will use max_payload_size for length	s_region_lengthd_ep.vh
AVY	TXN	ADDRESS BASED ROUTING-11: TLP Forwarding of DMA for Pre and Non-Pre Memory Requests or IO Requests from Between Designated Endpoints Using Various Port Arbitration - verify switch forwards all DMA style memory and IO address types between specified peers using WRR128	s_ep_memory7.vh
AVY	TXN	INTERRUPT SIGNALLING-3: INTx Interrupt Signaling Rules TXN.2.13#N - verify INTx messages of all types are forwarded and that switch collapses messages properly	avery_txn_2_13.vh
AVY	TXN	LOCKED TRANSACTIONS-4: Locked Transactions - Generate memory read locked transactions with errors and verify unlock message	avery_locked0.vh
AVY	TXN	ERROR SIGNALLING-1: Error Signaling Messages TXN.2.15#1~4 - EP issues all message types	avery_txn_2_15_ep.vh
AVY	TXN	TRANSACTION ORDERING -4: Transaction Ordering P passes NP - Verify Table 2-23 Row c Col 2	tor_c2.vh

AVY	TXN	VIRTUAL CHANNEL -1: Virtual Channel (VC) Mechanism TXN.5.0#2 #3 #4 #7 - Generate a series of NP requests for all TCs and verify completion TC used	avery_txn_5_0.vh
AVY	TXN	DATA POISONING-4: Nullify TLP - RC nullifies completion	compl_nullify_ep.vh
AVY	PHY	SYMBOL ENCODING-25: Symbol Encoding Disparity in Endpoint - Changes the lane disparity for each lane after enumeration before sending TLPs	slane_disparity_rc.vh
AVY	PHY	LINK INITIALIZATION AND TRAINING-3: Lane Polarity Inversion - Changes the lane polarity for each lane after enumeration before sending TLPs	slane_polarity_ep.vh
AVY	PHY	LTSSM TRANSITIONS-13: Polling.Config => Detect - Transmit < 8 TS2 within 48 ms of entering state to force timeout	ltssm_Polling_Config_Detect1_ep.vh_
AVY	PHY	CLOCK TOLERANCE-1: Skip Order Set Frequency - Change the skip order set frequency to exceed limit	avery_skip_ep.vh
AVY	PMG	L-STATES-12: L0 => Recovery.RcvrLock => Recovery.RcvrConfig => Recovery.Idle => L0 in Endpoint - Change Recovery.RcvrLock lane negotiation states tx_count use TIMEOUT_PERCENTAGE	recov_rcvrlock_0_rc.vh
AVY	PMG	D-STATES-3: Verify Power Management D2 state - Direct DUT D-state to D2 check that link states enters L1 within timeout range	pm_d2_ep0.vh
AVY	SYS	ERROR REPORTING-1: Flow control update error reporting - Generate flow control update for unmapped VC check error reporting	avery_fcpe.vh
PCISIG	CFG	PCI_Express_Test_Spec_Config_Space_1.0.pdf-21: Device Response to Different Bus and Device Numbers [Test 2.5] [Section 3.2.5] - See PCI Express Configuration Space Test Considerations Rev1.0D	config_2_5.vh

7. Maximum Control with DUT Integration

The functional verification of PCI Express logic cores and the chips and systems utilizing them requires significant investment to develop and maintain a layered test bench environment comprised of transactors, assertions, test suites, and debug methods to isolate design bugs in different protocol layers. The test-bench should be capable of core-through-chip verification including full control over the Device Under Test's application logic interface. To obtain highest coverage, maximum controllability and observability of the DUT, Avery has developed a sophisticated DUT integration scheme, as illustrated in Fig. 3, to provide the high levels of test control. These added controls are needed to raise the compliance coverage significantly beyond the coverage from PCI-SIG test cases.

Fig.3 DUT-1 Integration



Avery BFM can control the application interface of DUT using two degrees of increasing control. The compliance test suite is grouped in categories according to the level of DUT integration. The two levels of integration are:

1) DUT-0 level:

In the DUT-0 integration level, the user's DUT is the complete backend design which directly implements the initiator, completer and target logic (like a network card EP for EP DUT). The test-bench connects to the DUT's external IO interface without direct access (probe) into DUT. To test an EP DUT, Avery RC BFM sends all the commands while EP DUT just responds to the commands. There is no access/control the internals/behaviors of a DUT.

2) DUT-1 level:

For more control to enhance testing effectiveness and completeness, the test-bench provides the application layer of the RC or EP and controls the backend application interface of the controller. Test-bench can drive signals for initiator, completer, and target as well as poke/control the internal of EP DUT design. Avery's testbench provides the application layer that interfaces to the DUT. It converts pins using task and event to transfer across the interface. It captures relevant information

into Avery records, maps Avery memory and records to DUT's memory and records, and performs handshakes to achieve asynchronous interface.

Sample template driven integration codes are provided for end users to customize to their designs. All the codes are in Verilog making it easy to integrate a DUT into the testbench. A sample code segment of the transmit task "TX_Task" of Fig. 3 is shown below.

```
// Transmit Application I/F (DUT as initiator)
task tx_record_insert;
input [2:0] vchan2;
input [`PTR_LENGTH-1:0] rec2;
output reject2;
begin
  $tb_var_record("t1p_record4_1024", rec2);
  @(posedge `DUT_TOP.aclk250M);
  if (vchan2==0) begin
    //header phase
    if (format2[0]==1) begin
      `DUT_TOP.tx_desc_vc0={h1,h2,h3,h4};
      header_count=3;
    end
    else begin
      header_count=2;
      `DUT_TOP.tx_desc_vc0={h1,h2,h3,32'b0};
    end
    fork:tx_packet_frame_vc0
      begin
        `DUT_TOP.tx_req_vc0=1;
        if (format2[1])
          `DUT_TOP.tx_dfr_vc0=1;
        end
      begin
        wait(`DUT_TOP.tx_ws_vc0==1)
        @(posedge `DUT_TOP.aclk250M);
        wait(`DUT_TOP.tx_ws_vc0==0);
        end
      begin
        wait (`DUT_TOP.tx_ack_vc0==1)
        `DUT_TOP.tx_req_vc0<=0;
        disable tx_packet_frame_vc0;
        end
    join
  end
endtask
```

A sample code segment of the receive task "RX_Task" of Fig. 3 is shown below.

```
// Receive Application I/F (DUT as target)
always begin:blk_received_req_vc0
  reg [31:0] req_1_dw, req_2_dw, req_3_dw, req_4_dw;
  fork
    begin
      wait(`DUT_TOP.rx_req_vc0==1)
      @(posedge `DUT_TOP.aclk250M);
      @(posedge `DUT_TOP.aclk250M);
      @(posedge `DUT_TOP.aclk250M);
      req_time=$time;
      req_1_dw = byte_reverse(`DUT_TOP.rx_desc_vc0[127:96]);
      req_2_dw = byte_reverse(`DUT_TOP.rx_desc_vc0[95:64]);
      req_3_dw = byte_reverse(`DUT_TOP.rx_desc_vc0[63:32]);
      req_4_dw = byte_reverse(`DUT_TOP.rx_desc_vc0[31:0]);

      $tb_record_new(rt1p_packet);
      $tb_save_record_field(req_1_dw, rt1p_packet, "header");
      $tb_save_record_field(req_2_dw, rt1p_packet, "data", 0);
      $tb_save_record_field(req_3_dw, rt1p_packet, "data", 1);
      tl.rt1p_status = `NO_ERROR;
      tl.rt1p_packet_ready = 1;
    end
  end
```

end

A sample code segment of the Mapping function of Fig. 3 is shown below.

```
// Access DUT internals
function [`PTR_LENGTH-1:0] direct_data_read;
input [11:0] type2;
begin
  case (type2)
    `COUNT_LTSSM_TX: begin
      case (`DUT_TOP.test_out[329:327])
        3'b000: begin
          count2=`LTSSM_TX_NONL0;
          end
        3'b001: begin
          count2=`LTSSM_TX_L0S_ENTRY;
          end
        3'b010: begin
          count2=`LTSSM_TX_L0S_IDLE;
          end
        3'b011: begin
          count2=`LTSSM_TX_L0S_FTS;
          end
        3'b100: begin
          $display("DUT got to L0 ");
          end
        default: begin
          $display("DUT have error TX_LTSSM ");
          end
      endcase
    endcase
  end
  direct_data_read= count2;
end
endfunction
```

DUT-1 integration's higher controllability and observability expands the test suite utilization and increases coverage on the design. This tight level of access is crucial in achieving the high checklist compliance coverage.

8. Advanced testing

Avery's PCI-Xactor compliance test suite contains hundreds of directed target tests to exercise specific functionality and corner cases in the standard specification. All protocol layers and key state machines are covered. The test suite covers all functional features of the specification including advanced testing of

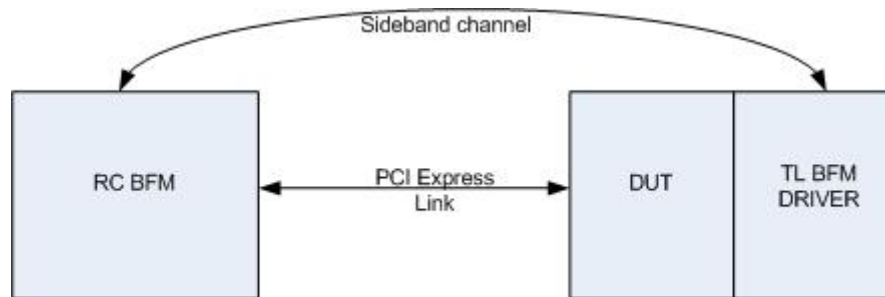
- isochronous traffic
- power management modes
- error logging and reporting
- switch - transaction order
- corner case and random stress testing
- LTSSM state transition test

Avery's compliance testing has high controllability and observability into the models. The error and completion sequences are powerful and easy to program. Error and completion/response control are provided from single errors to complex intermittent sequences. The transaction-oriented nature of defining multiple completion behaviors for several different PCI COMMAND/TARGET address adds significant control to stress the design under various error conditions.

The following two examples show in more detail the capabilities of the test suite.

1) Test example: Gen 2 Link Width Up / Speed Down Test

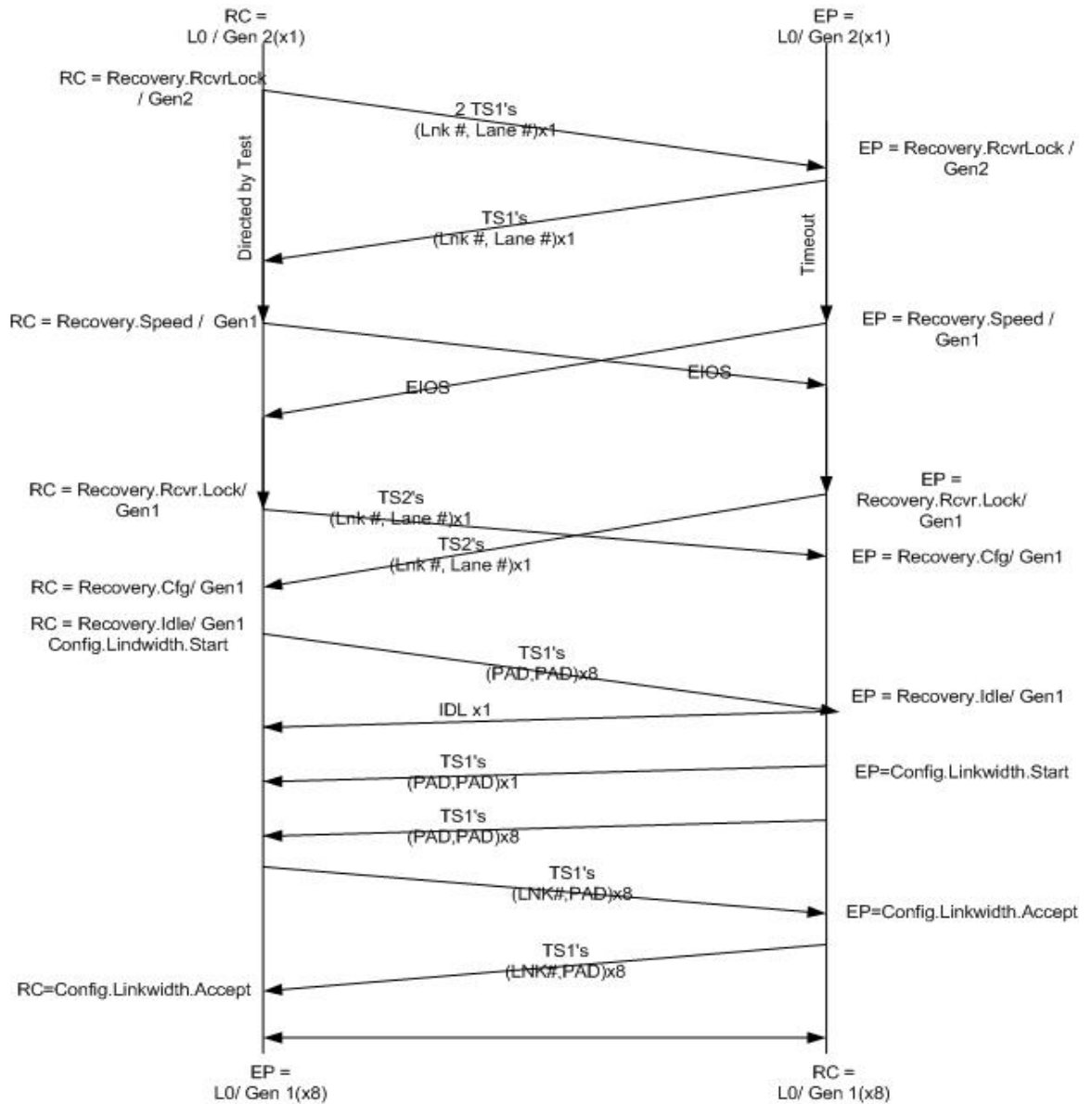
The Gen 2 PCI Express Specification allows for noisy links operating at Gen 2 rates to quickly retrain to Gen 1 rate and up shift the number of lanes to maintain data flow bandwidth. The verification of this feature is fairly complicated. Assuming that this feature is to be tested in the user's DUT which in this case is the EP (Endpoint), the RC BFM can verify correct functionality via the following mechanism. The diagram below shows the testbench setup.



1. The RC BFM and the EP (DUT + Transaction Layer (TL) BFM Driver) initially train to L0 in Gen 1 mode as required by the PCI Express Specification. During this stage, the RC records the maximum number of lanes the EP DUT is capable of using. The RC BFM further records that the EP is capable of Gen 2 operation.
2. The RC BFM initiates a speed change to Gen 2 rate. During this time, it instructs the TL BFM Driver to train with a lower number of lanes, say x1. So at this stage both the RC and the EP links establish a Gen 2 / x1 link up.
3. The BFM then tries to revert to Gen 1 speed through one of the following ways –
 - a. Hot Reset
 - b. BFM sends only 2 TS1 to transit link entering Recovery.RcvrLock. Since there are only 2 TS1's transmitted, DUT should timeout and then transition to Recovery.Speed.
4. When the LTTSM (Link Training Status State Machine) transitions to Recovery.Idle, DUT directs LTTSM to Configuration State and increases lane number to the maximum possible.
5. The 2 links then must establish themselves at Gen 1 speed with all lanes activated. There are assertions in the testbench to verify this.

The following diagram (Fig. 4) shows details of how the Gen 1 / maximum link width is established. This assumes that revert back to Gen 1 is caused through the Recovery.Speed state (step 3b. above).

Fig. 4 Sequencing of state transitions to affect speed down and link width up



Using similar mechanism the following are required to be tested where Device A / Device B can be Upstream / Downstream and vice versa.

Case	Device A	Device B
1	Speed up	no up-down
2	Speed up	down configure width
3	Speed up	up configure width
4	Speed down	no up-down
5	Speed down	down configure width
6	No speed	up configure width
7	No speed	down configure width

Avery provides a comprehensive test suite to vary the above features which are a requirement of the PCI Express Gen 2 Specification.

2) Test example: Transaction ordering

Transaction ordering is a key feature of the PCI Express protocol. A comprehensive explanation of ordering rules can be found in [3]. Avery's PCI Express Verification IP offers a comprehensive set of tests to verify all possible ordering rules specified by the most current PCI Express specification.

The following PCI Express 1.0a table 2-23 [4] is a summary of which transactions can bypass the other.

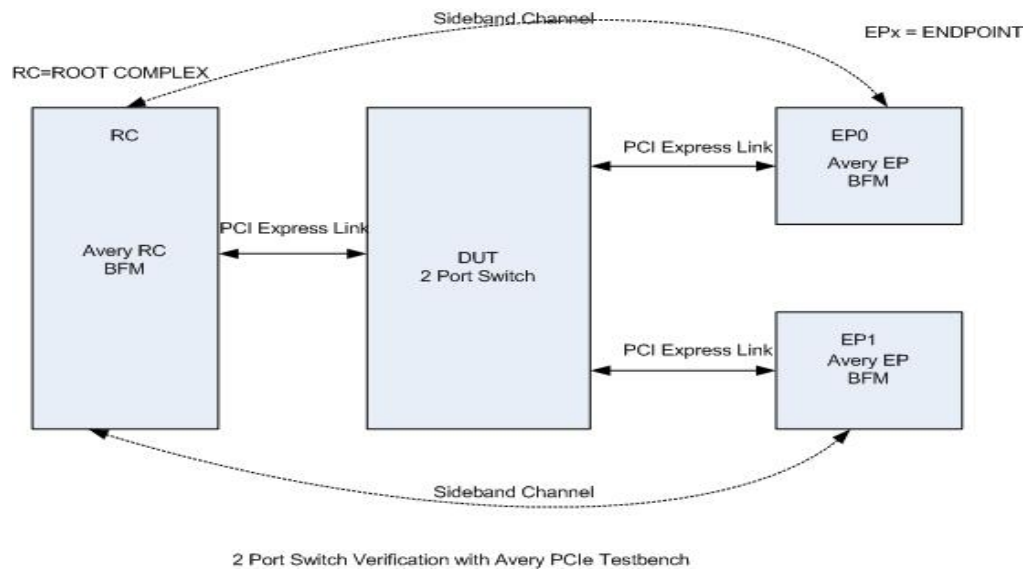
Table 5: Ordering Rules

Table 2-23: Ordering Rules Summary Table

Row Pass Column?		Posted Request	Non-Posted Request		Completion	
		Memory Write or Message Request (Col 2)	Read Request (Col 3)	I/O or Configuration Write Request (Col 4)	Read Completion (Col 5)	I/O or Configuration Write Completion (Col 6)
Posted Request	Memory Write or Message Request (Row A)	a) No b) Y/N	Yes	Yes	a) Y/N b) Yes	a) Y/N b) Yes
	Read Request (Row B)	No	Y/N	Y/N	Y/N	Y/N
Non-Posted Request	I/O or Configuration Write Request (Row C)	No	Y/N	Y/N	Y/N	Y/N
	Read Completion (Row D)	a) No b) Y/N	Yes	Yes	a) Y/N b) No	Y/N
Completion	I/O or Configuration Write Completion (Row E)	Y/N	Yes	Yes	Y/N	Y/N

In the following example we discuss how one such ordering rule is verified. Assume that the verifier wanted to verify whether Row A could pass Column 3. This would mean that a Memory Write or a Message Request would bypass an earlier launched Read Request. Further assume that the verifier is attempting to verify this property through a DUT that happens to be a 2 port switch. The following diagram shows the test-bench setup.

Fig. 5 2 Port Switch Verification with Avery PCIe Test-bench



Initially each PCI Express Link between the DUT and RC, EP0 and EP1 train up. That is each link enters the L0 state. This happens without user intervention. Thereafter the Data Link Layer for each PCI Express Link is established with initial flow control credits. Once flow control is established for VC0, the Root Complex enumerates each port on the downstream components. Then test executes to verify the desired ordering which in this case is whether a Memory Write or a Message can bypass a previously scheduled Memory Read that is held up in the switch. The following is the algorithm implemented.

1. Each EP via the sideband sets up within the RC the following transactions –
 1. Memory Read to a region within it local memory, followed by
 2. Memory Write / message Write to a local region within its local memory.
2. The Memory Read and Memory Write is set up to use the same Traffic Class.
3. The Memory Read is set up such that the RO (Relaxed Ordering) bit is set.
4. Each EP internally sets up the “ordering expects” which are the expected completion order.
5. Each EP advertises the following flow control–
 1. Non Posted Header insufficient to consume pending Memory Read.
 2. Posted Header and Posted Data enough to consume the pending Memory Write.
6. Each EP signals the RC to launch the Memory Read followed by Memory Write.
7. Each EP monitors the completion order; since the Memory Read does not have sufficient credits to complete, the Memory Write is expected first. If Memory Write is seen, credits are updates to allow the Memory Read to complete.
8. The Memory Read is then expected.

The important thing to note here is that any of the following conditions can cause a failure of the test.

- Timeout - none of the scheduled transactions are received by one or either EP.
- The Memory Read is received before the Memory Write indicating Flow Control Credit Rules were violated.
- The Memory Write is received but the Memory Read is not received and times out even after credits are updated so as to complete the Memory Read.
- All is well with one EP but not the other.

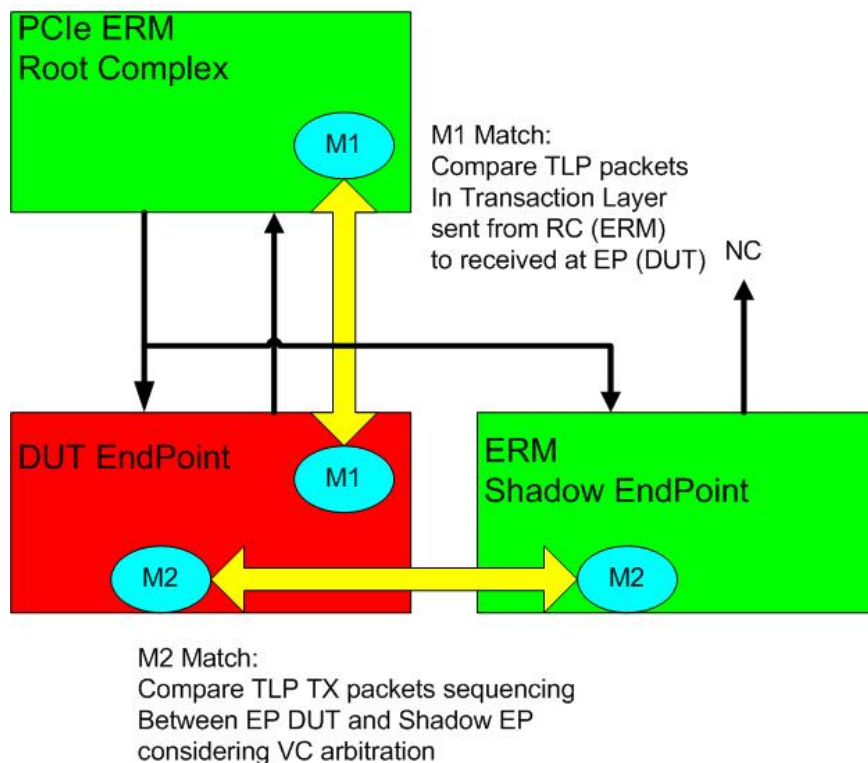
The example presented is for a two port switch. If the switch presents more ports, then the test-bench will automatically scale the number of endpoint BFMs to match the number of downstream ports and run the algorithm on each EP.

Using similar procedures, Avery PCI Express test suite offers directed tests to verify the entire ordering table depicted above for both strong and relaxed ordering cases. Verification of compliance to PCI Express ordering rules is critical to the proper implementation of a PCI Express Switch. The ordering rule verification process is one of the more tedious tasks in the verification effort.

9. Advanced Debugging with Design Match

An advancement in verification over traditional verification methods (such as bus and transaction monitoring), PCI-Xactor co-verification has an executable reference model (ERM) and innovative auto-debugging methods for bug detection and isolation. Using co-verification the DUT and ERM run concurrently, applying the compliance and systems tests to both models. Transaction and sequential consistency is verified using pre-configured design match points, which track transaction flow between protocol layers of the DUT and ERM (see Figure 6). Architectural visible state and transactions (match points) are analyzed applying relaxed time, ordering, and content rules defined by the PCI Express protocol to ensure meaningful sequential consistency checking.

Fig. 6 Design Match checking Sequential Consistency



When a mismatch occurs, auto-debugging is then used to perform causal analysis of the implementation model and ERM. Auto-debugging utilizes enhanced behavior traversal and transaction views added to advanced behavioral debugging systems such as Novas Verdi for better visualization of the behavior of the DUT and shadow ERM. Co-verification is also especially useful in the context of random testing where expected device operation is too complex to predict or when assertions are too complex to write. Here, match points verify the architectural state of the models on-the-fly under random input sequences.

Some example matchpoints from PCI_Express designs are:

- Transition (control)
 - Link Training State Machine FSM (LTSSM)
 - Retry buffer using saved records and retry trigger
 - VC's flow credits, rxfc_allocated_m, rxfc_infinite_m
 - Data link layer control/management FSM
 - Configuration data, including various capabilities
 - Power management FSM
- Transport (data)
 - Transaction packets at various points, including rx and tx side
 - For memory/IO/config access, address/word/read-write
 - UR packet to be returned

For example, design match capability isolated an incorrect LTSSM state transition to lower power state in a multifunction DUT under random test sequences. The white paper "Design Match and AutoDebug to verify and debug complex ASIC and SOC Designs" [5] describes more in detail on this advanced capability.

Summary

Achieving PCI Express device compliance is getting more challenging with the new revision of the specification (Gen-2). The need for a highly capable Verification IP having a robust compliance test suite to verify the design is becoming critical. Avery's PCI-Xactor having the following capabilities has demonstrated successes on many customers' designs.

- Complete solution for core, chip, and device-level verification
 - Closed-end core-level verification
- Open solution – Verilog source models and tests
- More reference designs & comprehensive compliance testsuites
 - Parameterized test macro library provides powerful test building blocks that lowers development/debug time
 - Advanced tests (isochronous traffic, power modes, error logging and reporting)
 - Complete documentation of tests and checklist coverage
- Powerful bug finders and debug features
 - Transaction and Symbol tracking files
 - Advanced sequential consistency matchpoint verification
- Simple and Efficient BFM API
 - Robust command layer
 - Common host support routines:
 - PCI BIOS routines (search config space for CAPID present)
 - App-level driver routines (isochronous DMA read-verify of 64KB)
 - Multiple task execution semantics: Blocking and non-blocking
- Powerful error and completion/response control
 - From single errors to complex intermittent sequences
 - Random and directed
- Closed-end core-level verification
 - Robust core application interface integration methodology
 - Supports application interface to DUT/core
 - command layer
 - transaction layer
 - link layer
 - Supports proprietary and standard application interfaces: ARM AXI/AHB
 - Maximizes compliance testing using DUT as initiator and completer

Avery's PCI-Xactor provides robust compliance test suites proven with multiple vendors' design cores. High coverage on real customer designs measured against the PCI-SIG checklist has been achieved. Customers are now actively using the PCI-Xactor for their verification on their Gen 2 designs.

References

- [1] PCI-SIG PCI Express Base Specification Revision 2.0
- [2] "Advanced Functional Verification and Debug of PCI Express-based Designs" by Chris Browy, Embedded Computing Design, August 2006
- [3] PCI Express System Architecture, Mindshare Inc., Budruk, Anderson, Shanley, Addison Wesley 2003.
- [4] PCI-SIG PCI Express Base Specification Revision 1.0a
- [5] "Design Match and AutoDebug to verify and debug complex ASIC and SOC Designs" by Shaw Yang and Yi-Hui Lin www.avery-design.com