

Post Silicon Functional Validation: PCIe

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE AWARD
OF THE DEGREE OF

MASTER OF TECHNOLOGY
IN



VLSI DESIGN AND EMBEDDED SYSTEMS

Submitted by

Suraj Singh 2K21/VLS/20

Under the supervision of
Dr. M S Choudhary

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering) Bawana Road, Delhi-110042

MAY, 2023

DECLARATION

I, **Suraj Singh** hereby declare that the work presented in this thesis entitled “**Post Silicon Functional Validation : PCIe**” in partial fulfillment of the requirement for the award of degree of **Master of Technology (VLSI Design and Embedded Systems)** submitted at **Electronics & Communication Department** , Delhi Technological University (DTU), Delhi . The matter presented in this has not been submitted either in part or full to any other university or institute for the award of any other degree.

Place : Delhi

Date : -05-2023

Suraj Singh

(2K21/VLS/20)

**DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING DELHI
TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)BawanaRoad, Delhi-110042**

CERTIFICATE

I hereby certify that the major project Dissertation titled “Post Silicon Functional Validation: PCIe” which is submitted by Suraj Singh, Roll No. 2K21/VLS/20 of Electronics & Communication Department, Delhi Technological University (DTU), Delhi in partial fulfillment of the requirement for the award of degree of Master of Technology, is a record of the project work carried out by the students under my supervision to the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi
Date: -05-2023

SUPERVISOR
Dr. M S Choudhary

ACKNOWLEDGEMENT

I take this opportunity to express my gratitude to my guide **Dr. M S Choudhary**, Professor, Department of Electronics and Communication Engineering, Delhi Technological University (DTU) for his guidance and constant encouragement throughout the course of this project.

I would also like to thank **Prof. O P Verma**, Head of Department, Electronics & Communication Engineering, Delhi Technological University for giving such an opportunity of internship at Intel Corporation Pvt. Ltd and rendering all kinds of supports throughout the project work.

Last but not the least I thank my family, friends, and colleagues for their timely help and valuable suggestions.

Place: Delhi
Date : -05-2023

Suraj Singh
(2K21/VLS/20)

ABSTRACT

Post Si validation is one of the most important and crucial step in the PRQ of the product into the market. As, in this era, there are a lot of IPs or peripherals are present on the SoC and also contains complex circuitry on in the chip. Major chunk of the silicon chip constitutes analog and digital circuits which contains high speed input output link interface called PCIe and other parallel links for example DDR. Validation of this link interface IPs are very hard to validate as the observability of the signals are very less in the post Si domain.

This thesis proposes functional validation of high speed serial links such as PCIe to improve the quality and Performance of the IP in the SoC with the help of correct debug tools. And, also describes the systematic approach to validate the IP in a correct way and to meet the market timing constraints

TABLE OF CONTENT

Sr No.	Name	Page No
	<i>Certificate</i>	<i>ii</i>
	<i>Declaration</i>	<i>iii</i>
	<i>Acknowledgement</i>	<i>iv</i>
	<i>Abstract</i>	<i>v</i>
	<i>List of Figures</i>	<i>viii</i>
	<i>List of Abbreviations</i>	<i>x</i>
1.	Introduction	7
1.1.	Major Steps in Post Silicon Validation	9
1.2.	Need of Post Silicon Validation.....	10
2.	Overview of PCIe	14
3.1	Introduction	14
3.2.	PCIe Topology	15
3.3.	Layered Architecture.....	16
3.	Errors Associated with PCIe	20
4.	PCIe: Link Training and Initialization.....	23
5.1	Different States of PCIe LTSSM... ..	24
5.2	Power States associated with PCIe Link	26
6.3	PCIe Configuration Register associated with LTSSM.....	28
5.	System Power State and Device State	31
7,	Statement Of Problem Based On Identified Research Gaps	33
8.	Post Silicon Validation Paradigm	34
8.1	Tools Used for Execution and Debugging in Post Silicon validation	36
9.	Work done, partial simulation/hardware implementation, partial Results and Discussion.....	38
9.1	Overview of Work Done	38
9.2	Test Cases implemented with OSBV.....	38
9.3	Test Cases implemented with Synthetic.....	39

10	Conclusion.....	44
10.1	Future Scope.....	44
11	References	47

Table of Figures

Sr No	Description	Page No
Fig1.	Intel SoC Architecture	7
Fig2.	Validation loop.....	9
Fig3.	Comparison between Domains.....	10
Fig4.	CEM and M.2 Slots	14
Fig5.	PCIe Gen Speed depending upon Lanes	14
Fig6.	PCIe Tree Topology	15
Fig7.	Layered Architecture	16
Fig8.	Types of Transaction	16
Fig9.	Transaction Layer and TLP	17
Fig10.	TLP Field	18
Fig11	DLLP	18
Fig12	PHY Layer	19
Fig13	Types of PCIe Error	21
Fig14	Message Packet.....	21
Fig15	Error Message.....	22
Fig16	Completion TLP.....	22
Fig17	LTSSM.....	23
Fig18	Link going from Detect to Polling.....	24
Fig19	Polling State.....	25
Fig20	Configuration State	25
Fig21	L0 State	26.

Fig22	L0s State.....	27
Fig23	L1 State	27
Fig24	Link Capability Register.....	28
Fig25	Link Status Register	29
Fig26	Link Control Register.....	29
Fig27	State and Link State.....	32
Fig28	D state and Link State.....	32
Fig20	Phases in Post Silicon Validation.....	34
Fig30	TLA GUI	36
Fig31	Protocol Analyzer GUI.....	37
Fig32	Perspec Tool	45
Fig33	Perspec tool Flow.....	45

ABBREVIATIONS

SoC: System on Chip

GPU: Graphics Processor Unit

CPU: Central Processing Unit

RVP: Reference validation platform

PCIe: Peripheral component Interconnect Express

PO: Power On

POE: Power on Exit

TLP: Transaction Layer Packet

DLLP: Data Link Layer Packet

PHY: Physical Layer

PRQ: Product Release Qualification

OSBV: OS based Validation

I/O: Input Output

DMI: Direct Media Interface

TLA: Tektronix Logic Analyzer

1. INTRODUCTION

A System of Chip (SoC) involves various kinds of IPs or integrated circuits which contains different memory controllers (SRAM, ROM, etc.) and other peripherals. It also contains various digital, analog and mixed signal circuitry which are responsible for intended functionality of the SOC.

The major advantage of using the SOC is that size of whole circuitry become small and also consumes less power.

The Basic architecture of Intel SoC contains a microprocessor and other supporting Intellectual property that enables a building block for variety of computing system. Intel architecture consists of important parts: a microprocessor chip and the companion chip which also known as Platform Control Hub. The microprocessor chip and the companion chip are connected via interface called DMI.

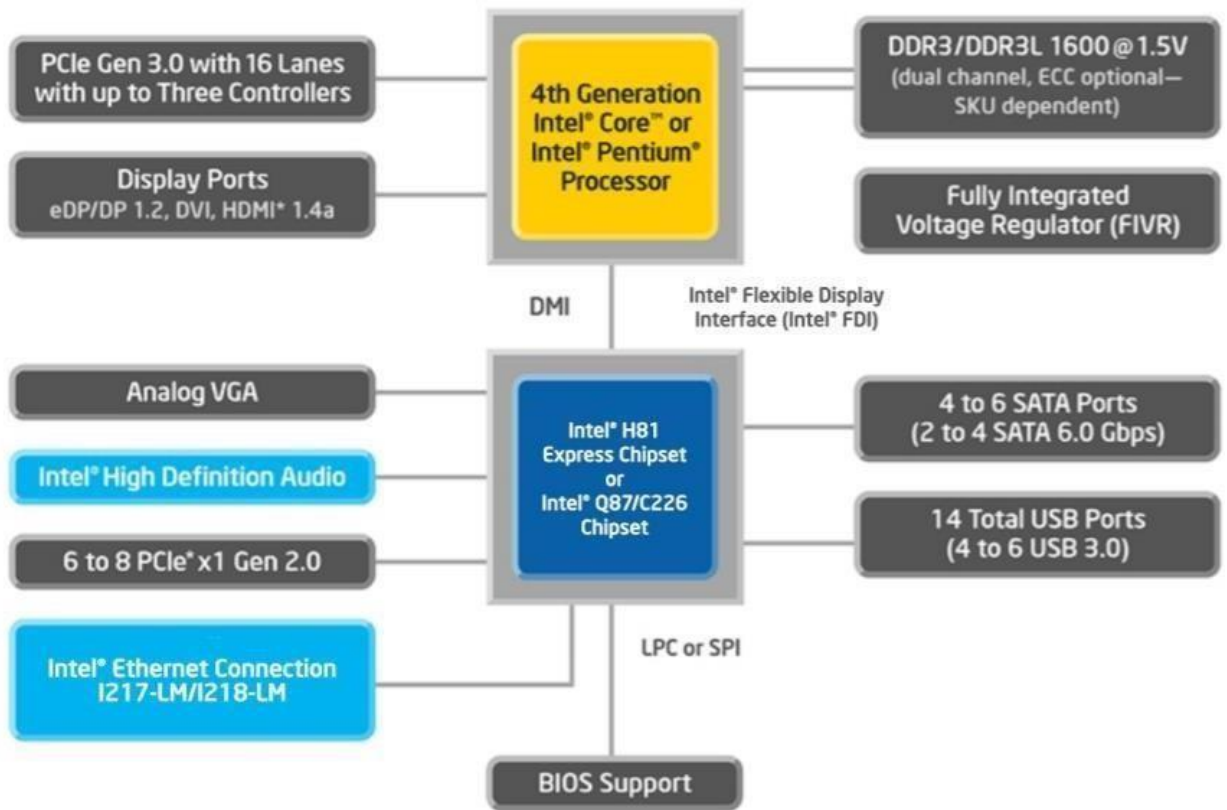


Figure 1: Intel SoC Architecture

The high speed I/O interface used in the Intel architecture is PCIe interface. The Number of PCIe lanes depends upon the type of processor used. The maximum common width of PCIe lane is 16 as it is supported by PCIe graphics card.

The PCIe uses differential signals as DMI uses. PCIe supports highest data rates up to Gen5 Speed nowadays.

Post silicon validation is the last step in semiconductor chip industry before launching the product into the market. Functional Validation (FV) is one of the most important part of post silicon validation. The goal of this step is to validate the functionality of the silicon chip with respect to the specification provided for the exact SoC (System on Chip) design. Thus making sure that the product is functional well by considering end user case scenario. This will make a product successful in the market.

Post-Silicon validation of a SoC comprises of various types of validation, which includes system validation (SV), functional validation (FV), and electrical validation (EV) and so on. Different category of validation focus on different parameters and execute in parallel. Ultimately, post-silicon validation aims to qualify a product over all process corners as well as operating conditions. At the end of post-silicon validation, a Product Release Qualification (PRQ) decision is made according to the risk assessment of how many systems fail according to the specifications. In a product's life cycle, the PRQ decision is an important milestone after which the product is released into the market.

The aim of Post Silicon Validation is to guarantee that the silicon chip works properly under real operating environment while executing real software, identify and correct the faults and errors that may have untouched due to the capability of Pre silicon Validation. Validation (Post Silicon) is a method in which the chip is made to undergo end user case scenario/tests for all accuracy and correctness with respect to its function in an experimental lab environment. This is done through using the actual silicon chip which is put together on a reference validation platform along with all other hardware which are part of the system for which, the chip has designed for.

Validation has become the major challenging task for companies validating IP and SoC products to meet the customer requirements. The complexity of modern SoC creates an enormous validation challenge which can only be met through the application of advanced validation techniques and optimal reuse. It is pointed out that post silicon validation stage is one of the important task or the level for a product before it comes out in the market because there are so many bugs which remain un addressed during the pre-silicon verification which can create huge problem to the end user.

1.1. Major Steps in Post Silicon Validation

There are 4 major steps:

1. Reproducing the problem by executing various test cases and also by using real time end user application

Example: booting to OS, running games or any other heavy application, until the system failure reproduce (example: system hung, crash or any other machine error).

2. Identifying the reason/issue that cause a system failure.

Example: an application/system flow got break/crash might be due to any IP present on the SoC.

3. Addressing the root cause of the failure.

Example: Isolating the issue into a single reason.

4. Fixing of the issue/failure using various techniques like by providing different patches like BIOS, or firmware update.

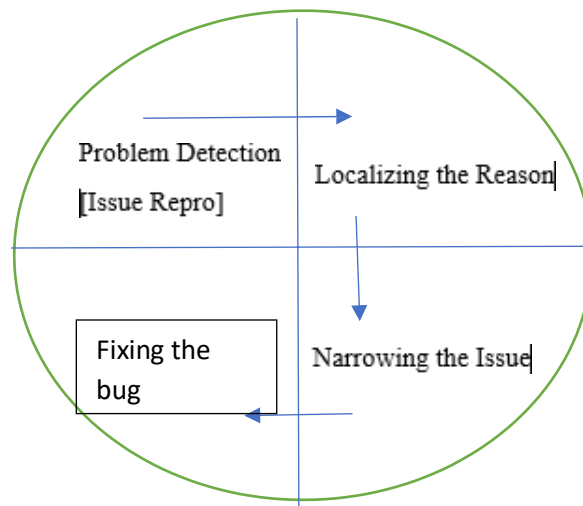


Figure 2: Validation loop

Each of these four phases poses significant challenges like reduce time to validate and improve robustness, improve the throughput, reduce time to debug.

To reduce the time of validation in post silicon can be done by structuring your strategy in an organized way can help you and in return you can deploy your product in to the market on time and quickly.

Hence, above mention above steps can help to achieve your goals easily.

1.2. Need of Post Silicon Validation

1. To deploy product into the market we can trust fully on the pre-si verification as there are lot of bugs which get un addressed in this domain. As, in pre-si is done various software/emulators which are way slower than a real silicon chip. Pre- si can give advantage in verifying individual logic of the chip but it reaches its limitation in terms of verifying full chip models. Hence, lots of issues get detected in post-si rather than the pre-si.

2. As the technology is increasing day by day therefore the complexity of the SoC is also increasing. Hence, the handshake between various components of design and electrical block of the system is become important, e.g., process variation, signal health (noise and cross talk) and thermal effects. Such handshake can lead to system malfunctioning/or not functioning properly and these bugs are considered as electrical bugs. Hence, handling of these errors become very difficult in the pre silicon verification.

3. Despite of manufacture defect, post-si errors may be reproduce by the repeatedly handshaking of the various IPs presents on the SoC with platform parameters which can result into logical errors or malfunction of the SoC. To create an error handling mechanism for these bugs become very difficult.

Pre-silicon verification	Manufacturing testing	Post-silicon validation
Excellent controllability and observability because any signal can be accessed	Controllability and observability primarily through scan DFT	Insufficient controllability and observability due to limited access to internal signals. Scan DFT useful for certain cases when a failure caused by a bug is repeatable.
Complex physical effects difficult to model	Several defect models exist	Accounts for signal-integrity, process variations, non-determinism
Simulation of full-chip designs very slow; formal verification only selectively applicable	Generally very fast (few seconds to minutes per chip)	Silicon speed orders of magnitude faster than simulation
Some metrics exist (e.g., code coverage, assertion coverage, mutation coverage)	Test Metrics (e.g., stuck-at, transition, N-detect coverage) widely used	Coverage metrics for post-silicon validation: Open research question
Bug fixing inexpensive	Bug fixing not the primary objective	Bug fixing can be expensive

Fig 3: comparison between the domain

2. OVERVIEW OF PCIE

2.1. Introduction

PCIe stands for Peripheral Component Interconnect Express. This IP is used for attaching peripheral devices such as (SSD, GFX Card, Memory Controllers, etc.) onto a computer motherboard. These are the hard wired links which are embedded onto a SoC platform. This can be also used as addressing mechanism that discovers and configures devices into a system. PCIe is a high performance, input output interconnect for the peripherals in computing and communication platforms.

It is a serial point to point interconnect between two devices. PCIe Slot comes in two configurations i.e., M.2 and CEM slot.



Fig 4: M.2 and CEM Slots

PCIe implements packet based protocol for information transfer. This IP also give scalable performance based on number of signal lanes implemented on the PCIe interconnect.

PCI Express Version	Transfer Rate	Throughput			
		x1	x4	x8	x16
1.0	2.5 GT/s	250 MB/s	1.0 GB/s	2.0 GB/s	4.0 GB/s
2.0	5 GT/s	500 MB/s	2.0 GB/s	4.0 GB/s	8.0 GB/s
3.0	8 GT/s	984.6 MB/s	3.94 GB/s	7.88 GB/s	15.8 GB/s
4.0	16 GT/s	1969 MB/s	7.88 GB/s	15.75 GB/s	31.5 GB/s

Fig 5: PCIe Gen Speed depending upon Lanes

The above fig demonstrates the PCIe gen speed upon the number lanes implemented upon the system/SoC.

2.2. PCIe: Topology

PCIe is a set of wires which can contain up to 32 PCIe devices and each device can have 8 functions. PCIe topology is an upside down tree model. Every device sits on a bus and these devices can have one or two functions and each function is a standalone function. Every function is independent instantiation of a PCIe device. PCIe can support 256 buses, 32 devices and 8 function.

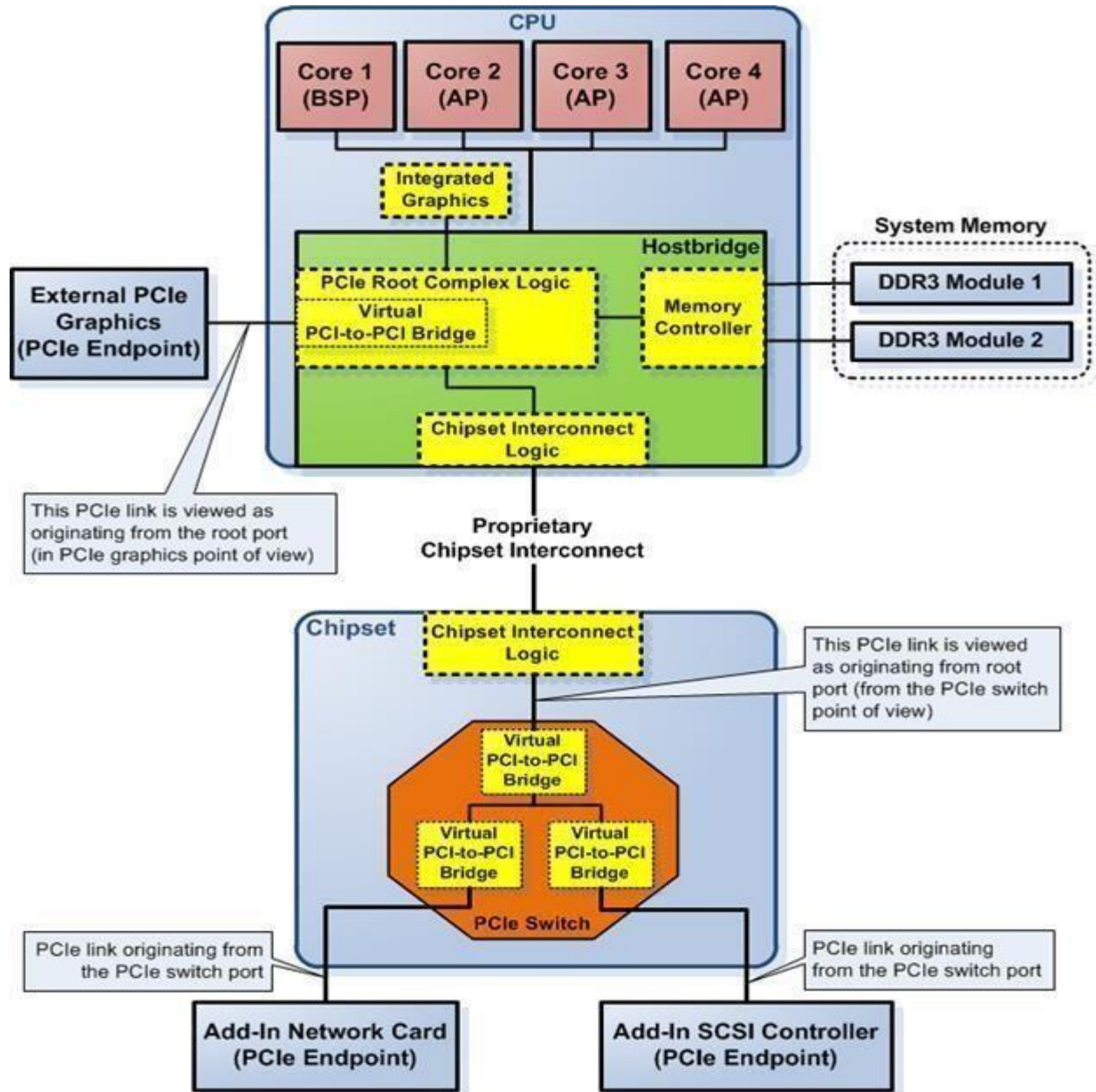


Fig 6: PCIe Tree Topology

2.3. PCIe: Layered Architecture

PCI Express has been implemented as a layered architecture. This layered architecture has been broken into 3 layers i.e., Transaction layer, Data Link Layer, Physical layer. Physical layer has been divided into 2 blocks: Logical and Electrical.

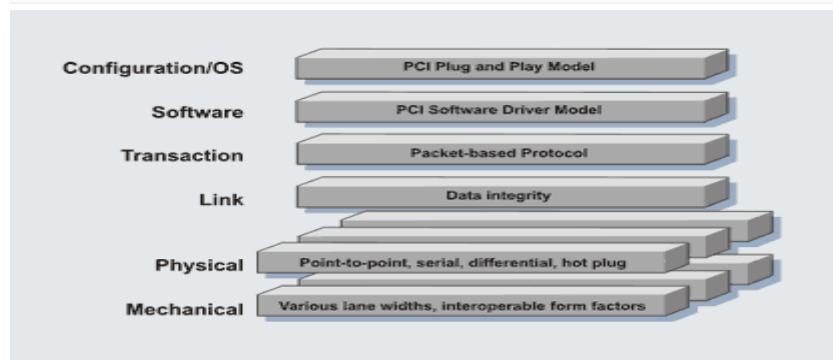


Fig 7: Layered Architecture

Software Layer

This layer is responsible for instantiating various different types of transaction such as Posted and Non posted transaction to the devices which are connected on the PCIe link. PCI Express maintains backward compatibility with PCI. This layer generates the address mechanism which helps to discover the devices which are connected onto the system.

Types of Transaction generated by the layer:

- Non Posted: These are the transaction where the requester expects to receive a completion TLP (Transaction Layer Packet) from the device completing the request.
- Posted: Transaction where the requester does not expect to and will not receive a completion TLP.

Transaction Type	Non-Posted or Posted
Memory Read	Non-Posted
Memory Write	Posted
Memory Read Lock	Non-Posted
IO Read	Non-Posted
IO Write	Non-Posted
Configuration Read (Type 0 and Type 1)	Non-Posted
Configuration Write (Type 0 and Type 1)	Non-Posted

Fig 8: Types of Transaction

Transaction Layer

This layer begins the process of building the transaction as a Packet which has to be send from one device to another. Types of transaction this layer request such as:

- Memory Read(MRd) or Memory Write (MRw): used to transfer data from or to a memory mapped location.
- I/O Read (IORd) or I/O Write (IOWr): used to transfer data from or to an I/O location. These transactions are restricted to support legacy end point device.
- Configuration Read (CfgRd) or Configuration Write (CfgWr): used to discover device capabilities, program features and check status in the 4KB PCI express configuration space.

This layer is responsible for TLP creation on transmit side and TLP decoding on receiver side. This layer is responsible for Quality of service, Flow control, Transaction Ordering.

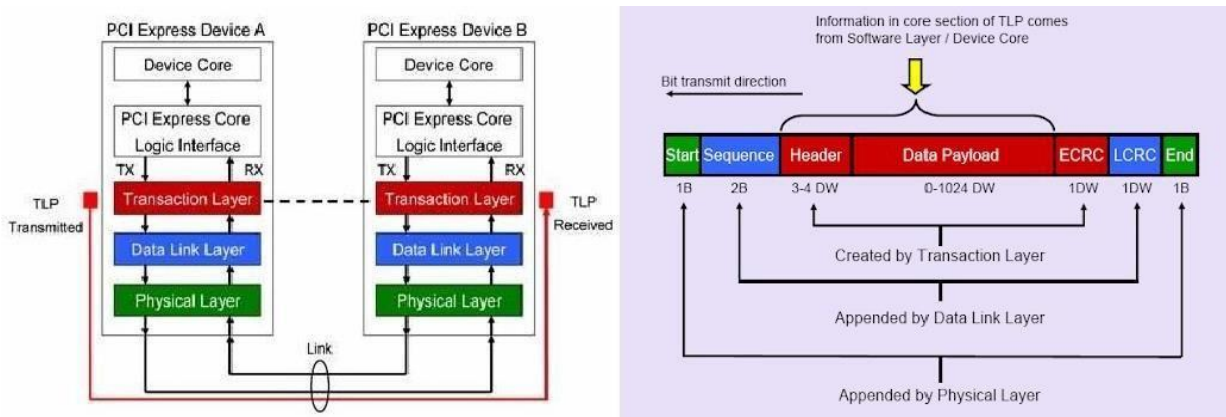


Fig 9: Transaction layer and TLP Assembly

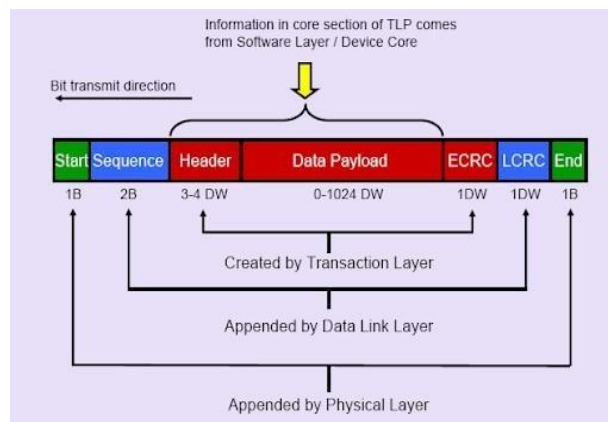


Fig10: TLP field

Date Link Layer

This Layer is responsible for reliable conveying the TLPs to the Physical layer. This layer appends transaction layer packet sequence number and error detection cyclic redundancy codes (CRCs) to the data packets to maintain the data integrity between the chip set and the I/O controller.

This layer is responsible for DLLP creation on transmit side and DLLP decoding on receiver side. This layer is responsible for link error detection and correction and for ack/nak protocol.

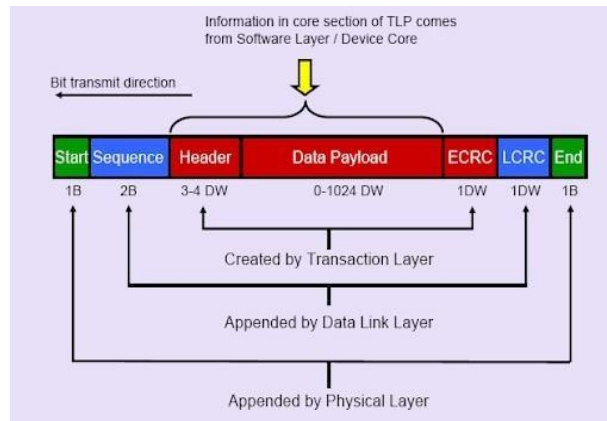


Fig 11: DLLP

This layer supports the link data integrity using ACK/NAK support and it also support low level power management.

Physical Layer

Implements the dual simplex PCI Express channels. The physical layer sits at the bottom of the interface between external physical link and data link layer. It converts the outbound packets into a serialized bit stream that is clocked onto all the lanes of the link. This layer also recovers all the bit stream from all the lanes of the link at the receiver side.

The contents of the layer are conceptual and don't precise logic blocks but to the extent that designers do partition them to match the spec because of increasing high data transfer rates. This layer is also responsible for the bit, byte and symbol lock on the link.

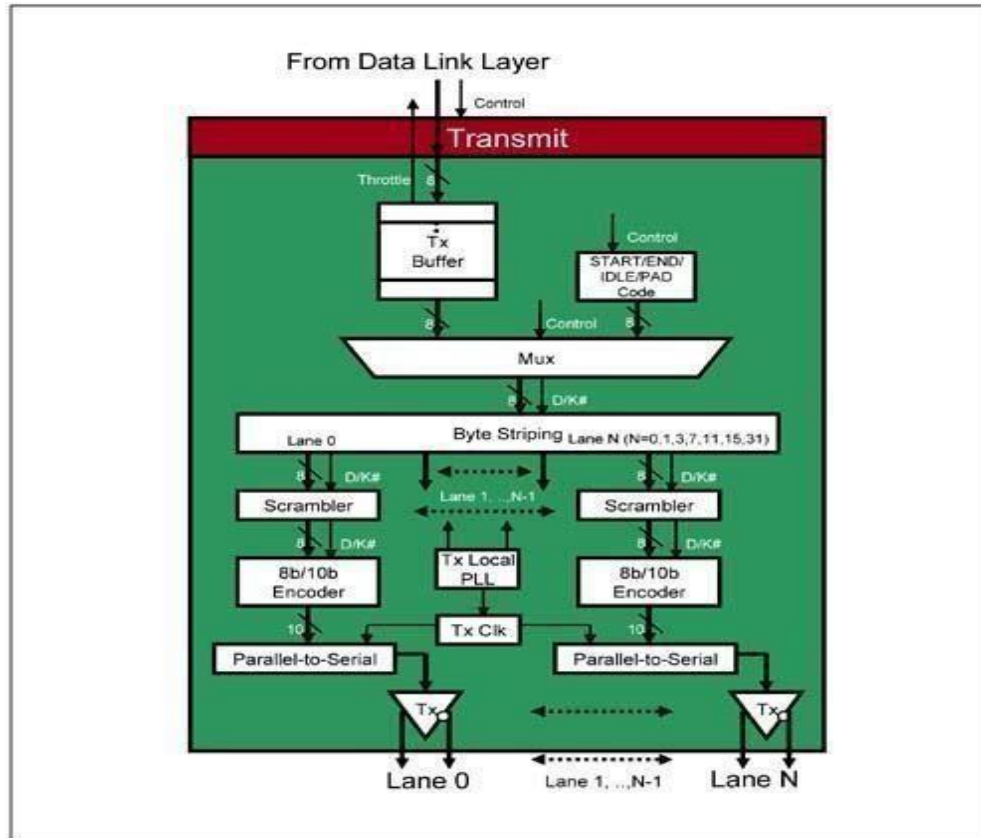


Fig 12: Phy layer

As mentioned above the physical layer contains two blocks i.e. Logical and Electrical Block. Below points explain the functionality of the blocks.

Logical Block:

1. Encoding/Decoding/Scrambling
2. Reset, initialization

Electrical Block:

1. Tx and Rx
2. Clks, PLL and Data Recovery
3. Low Level Link Power Control (L0, L0s, L1, L2)

4. ERRORS ASSOCIATED WITH PCIE LINK

PCIe IP provides great mechanism for error handling and logging. This feature provides great debugging method to identifying errors associated with the PCIe link, device specific errors and SoC failures which is caused by the PCIe IP.

There are 2 two types of errors associated with the PCIe:

1. Uncorrectable Errors: these errors are further divided into 2 categories i.e., Non – fatal errors which are handled by the device software and Fatal errors which are handled by the system software.
2. Correctable Errors: these errors are handled by hardware itself.

Uncorrectable Non-Fatal errors do not affect the PCIe fabric. PCIe fabric continues to work properly without any bug. In these type of errors, the data packet might lost or get corrupted. These type of error can't be fixed by the PCIe hardware.

Example of uncorrectable non-fatal error:

- Unsupported Request
- Unexpected Completion
- Completion Timeout
- Poisoned TLP

Uncorrectable Fatal Errors can be corrected by the PCIe link reset as these errors get generated when link become unreliable to send the data on the link.

Example of uncorrectable fatal errors:

- Link Training error
- Malformed TLP
- De Skew Buffer Full

Type of error	Errors examples	Pcie layer at which error found
Correctable	Receiver Error	Physical
Correctable	Bad TLP	Link
Correctable	Bad DLLP	Link
Correctable	Replay Time-out	Link
Correctable	Replay Number Rollover	Link
Uncorrectable - Non Fatal	Poisoned TLP Received	Transaction
Uncorrectable - Non Fatal	ECRC Check Failed	Transaction
Uncorrectable - Non Fatal	Unsupported Request	Transaction
Uncorrectable - Non Fatal	Completion Time-out	Transaction
Uncorrectable - Non Fatal	Completion Abort	Transaction
Uncorrectable - Non Fatal	Unexpected Completion	Transaction
Uncorrectable - Fatal	Training Error	Physical
Uncorrectable - Fatal	DLL Protocol Error	Link
Uncorrectable - Fatal	Receiver Overflow	Transaction
Uncorrectable - Fatal	Flow Control Protocol Error	Transaction
Uncorrectable - Fatal	Malformed TLP	Transaction

Fig 13: Types of PCIe errors

PCIe errors can be displayed by two methods:

1. By the Message transaction: which report the error to host.in this message packet there is a code field which signify the type of error.

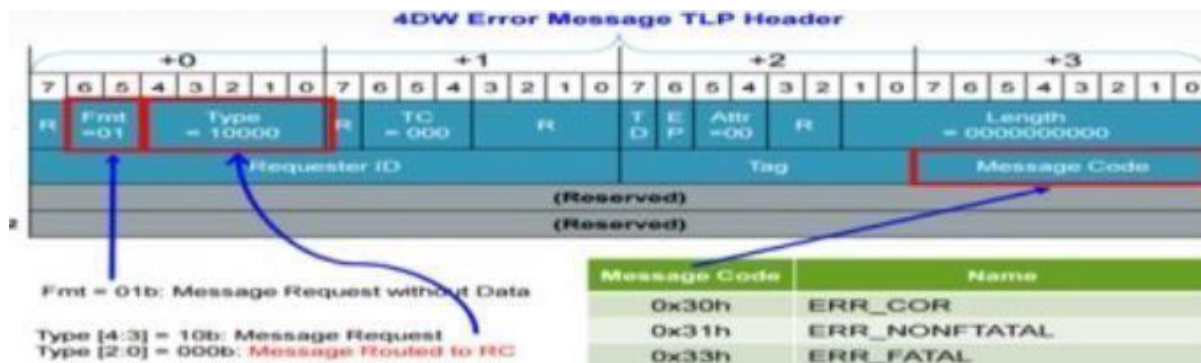


Fig 14: Message Packet

Above TLP is routed towards the Root Complex.

Message Code	Name	Description
30h	ERR_COR	used when a PCI Express device detects a correctable error
31h	ERR_NONFATAL	used when a device detects a non-fatal, uncorrectable error
33h	ERR_FATAL	used when a device detects a fatal, uncorrectable error

Fig 15: Error Message

2. With the help of Transaction Completion: the acknowledgment packet which receiver send to the transmitter also has an error field in the packet.



Fig 16: Completion TLP

All above mentioned errors can be fixed by firmware software/ other patches form phy/BIOS.

5. PCIE: LINK TRAINING AND INITIALIZATION

PCIe uses a Link training and initialization mechanism called LTSSM. This LTSSM process is control by the Physical layer. Therefore, link should get initialized properly before routing the data packets on the PCIe link. This process gets initialized automatically when the platform comes out of the reset.

PCIe port/link required to go through a series of link training before can be used to transfer or receiving the data.

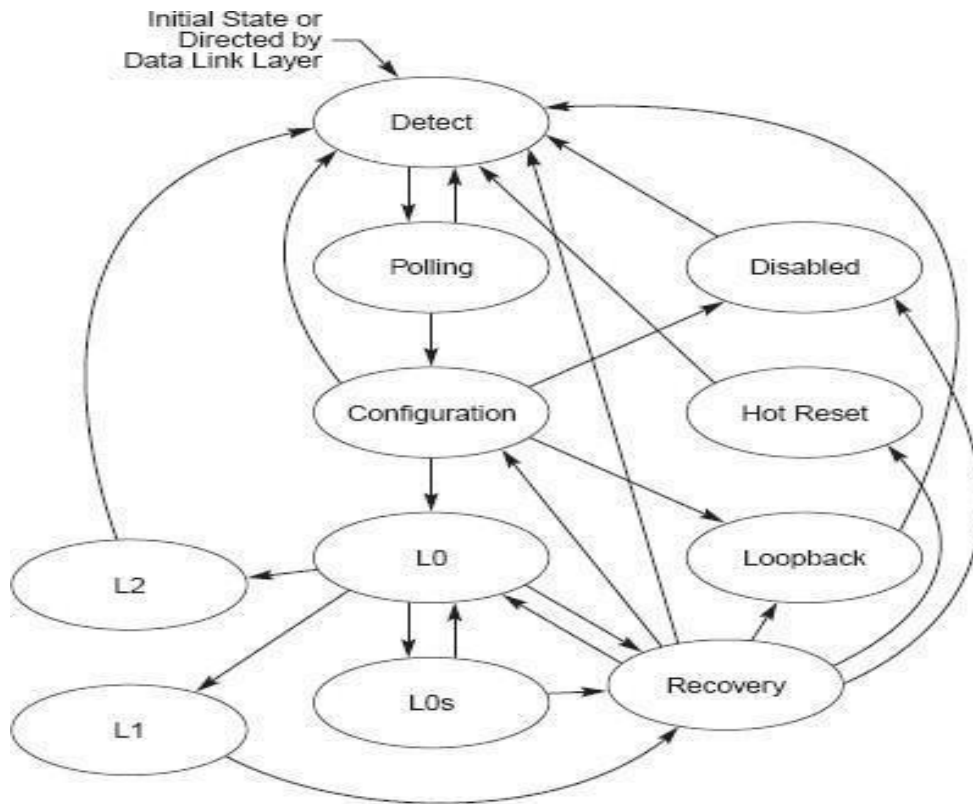


Fig 17: LTSSM

The above LTSSM process is also followed when link configuration/equalization happens between the device and the controller for synchronization to make link stable. This LTSSM link training can also be used to recover unstable link to stable link.

5.1 Different States of PCIe LTSSM

LTSSM – Detect State

This is the first state of the LTSSM FSM when the link comes out of the reset. When the link comes out of the reset it detects whether the other device is present or absent on the other side of the link. Detect state can be entered from other states also depending upon the scenario of the link.

Below figure explains the Detect State:

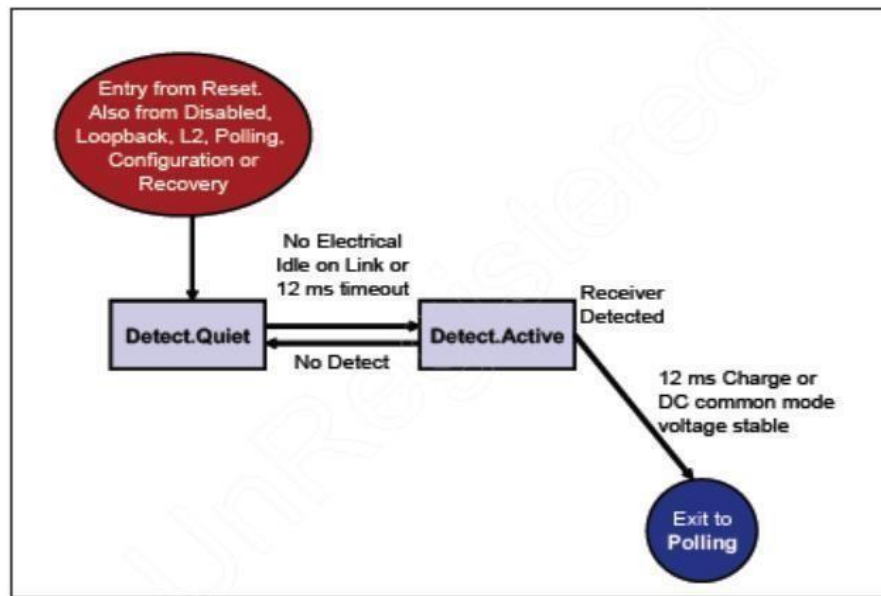


Fig 18: Link going from Detect to Polling

This is first state where PCIe link enters after the system comes out of the reset. Or, the host reset is executed by the software layer of the device/the root port controller. The link can be enter detect state from the following state:

- L2 State
- Disable State
- Recovery
- Polling

LTSSM – Polling State

Polling state is the first state in the process of link Initialization. In this state, training sequence ordered sets (TS1 and TS2) gets exchanged between devices i.e. transmitting device and the receiver device.

This ordered sets gets exchanged on all the lanes which are present between the devices. These ordered set get only exchanged when bit / symbol lock get achieved.

Below fig explains what happen inside the polling state:

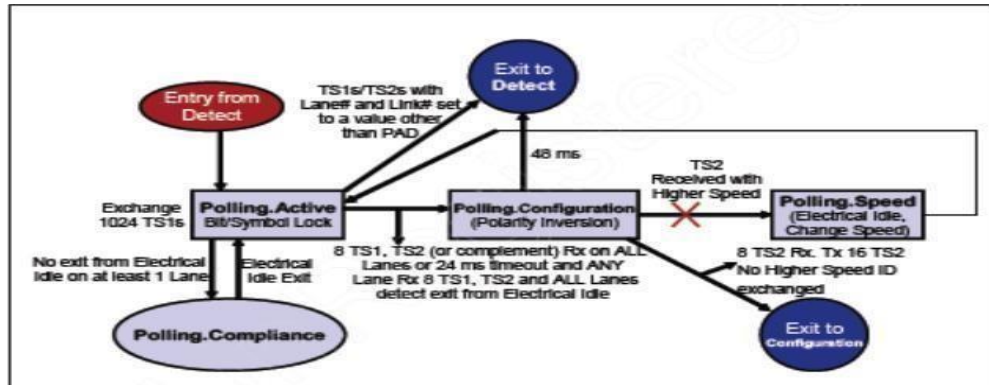


Fig 19: Polling State

LTSSM – Configuration State

This is the state where the lane number and link number get assigned between the different devices. In this state the upstream port (Root Complex) sends the TS1 ordered sets to the downstream device for begin the process of link and lane numbering. Once, this is done then TS2 packets get exchanged to determine the lane width.

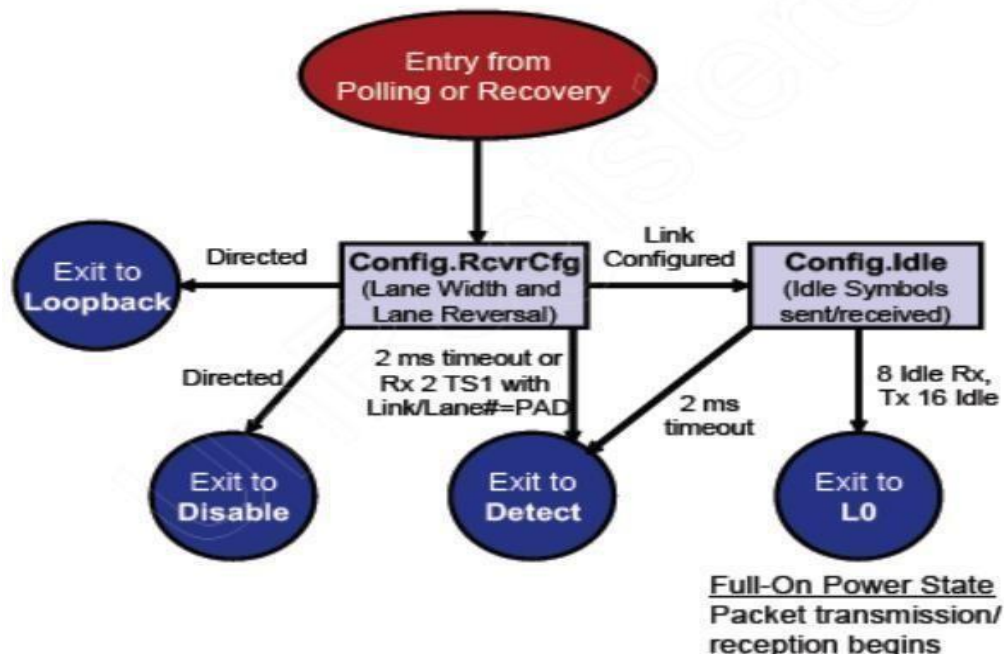


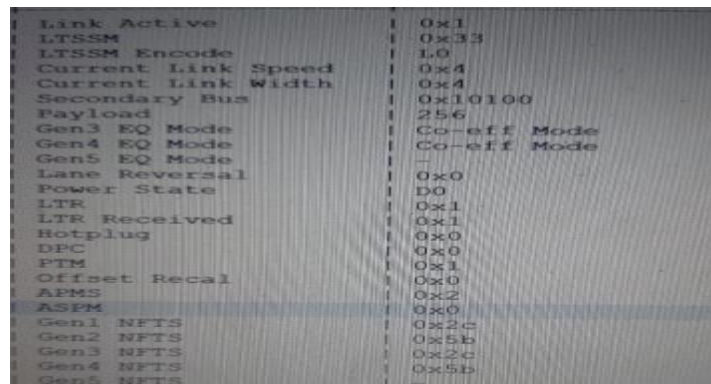
Fig 20: Configuration State

LTSSM – L0 State

L0 state is the active state of the PCIe which means the link is properly established between the devices. And, the link is ready to use for the transferring of data between the devices.

Below is the example when all the parameters get set properly when the link is in L0:

Work done:



Link Active		0x1
LTSSM		0x33
LTSSM Encode		L0
Current Link Speed		0x4
Current Link Width		0x4
Secondary Bus		0x10100
Payload		256
Gen3 EQ Mode		Co-off Mode
Gen4 EQ Mode		Co-off Mode
Gen5 EQ Mode		-
Lane Reversal		0x0
Power State		D0
LTR		0x1
LTR Received		0x1
Hotplug		0x0
DPC		0x0
PTM		0x1
Offset Recal		0x0
APMS		0x2
ASPM		0x0
Gen1 NPTS		0x2c
Gen2 NPTS		0x5b
Gen3 NPTS		0x2c
Gen4 NPTS		0x5b
Gen5 NPTS		-

Fig 21: L0

5.2 Power States associated with PCIe Link

There are different types of power states associated with the PCIe link as follows:

- L0s: L0s is the low power standby state of the link. Link can be entered in this state whenever it detects some idle time on the link by exchanging EIOS ordered sets. In this state both or either TX or RX can be in L0s. And, this is the state in which there is no need to tell the higher layer i.e., software layer.
- L1: L1 is the conventional low power state of the PCIe link. This state can be achieved when ASPM is enabled. ASPM refers to Automatic State Power Management module which is present on the SOC. Here, the link can only be go into L1 when both the TX and RX agrees to go in the low power because here we have intimate the higher layer of the device also. And, this intimation is done by sending the PM_Request_DLLP from the device to the other device who is sitting at the farther end of the link.
- L1.1 and L1.2: L1.1 and L1.2 is the lowest power substate of the PCIe link. This link state can only be achieved when below 2 mentioned conditions are met:

- The LTR value of the device should be greater than the LTR threshold value which is programmed by the BIOS.
- CLKREQ# should be de-asserted.

Below are some the captures which are taken by the Protocol analyzer which demonstrates low power states:

- **L0s:**

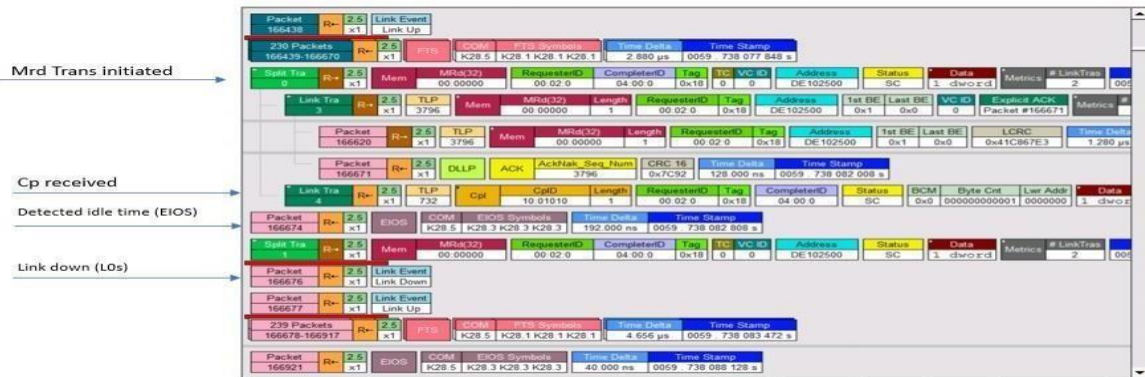


Fig 22: L0s State

- **L1:**



Fig 23: L1 State

5.3 PCIe Configuration Register related to LTSSM

This section describes some of the registers which are get configured during the link initialization and training of the PCIe link.

- Link Capability Register
- Link Status Register
- Link Control Register

Link Capability Register

Below fig describes the structure of link capability register.

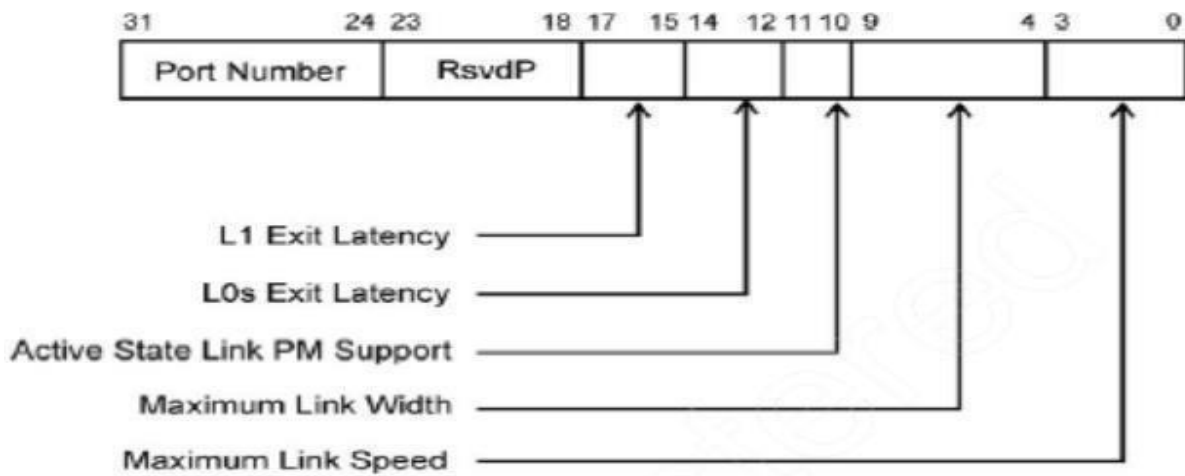


Fig 24: Link Capability register

Maximum link speed field specifies the maximum speed of a PCIe link which it can provide for data transmission. For example: if the field is set to 0010b that means the link set at Gen2 speed.

Maximum link width field specifies the width of the link. And, this register is updated by hardware when the ltssm start from the detect state or it is hard-wired. For example: 010000b: x16 width

Link Status Register

Below fig describes the structure of link status register.

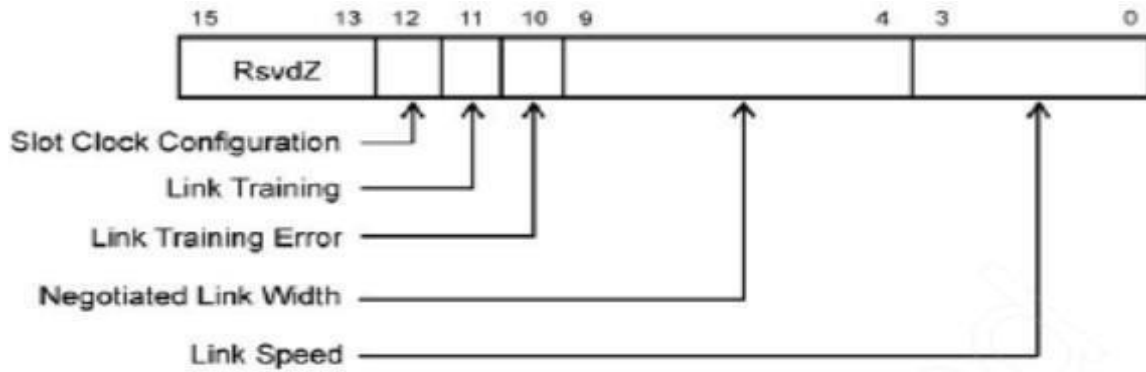


Fig 25 : Link Status Register

Link status register describes the status of the link after the polling state when all the negotiation between the controller and the device.

Link speed field is updated during the polling state on which both the Tx and Rx has agreed upon.

Negotiated Link width field represented the actual width of the link. Example: if controller exhibits **x8** and device exhibits **x4** then the nlw will be **x4**.

Link training error field specifies the training errors which can be occurred during link initialization or training. Example: if the link gets stuck in any of the LTSSM says(Recovery) then this field will get updated.

Link Control Register

Below fig describes the structure of link control register.

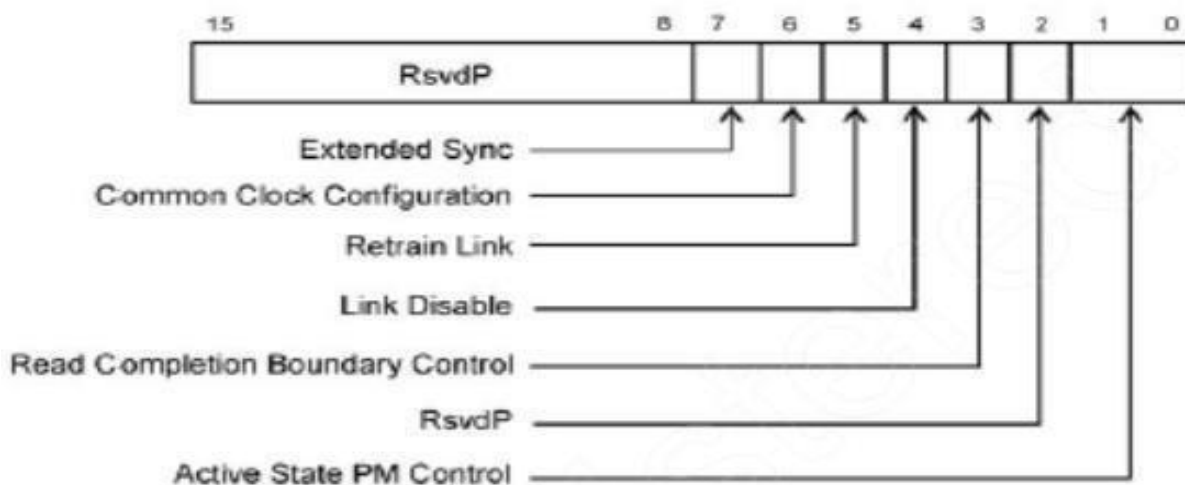


Fig 26: Link Control Register

Link Control Register is responsible for the controlling the state of the link like want to disable the link or retrain the link at appropriate speed or configuration.

Link Disable field is both read and write enable. This field is used to disable the link or check if the link is enable or disable.

Retrain Link field this field is very helpful in training the link from any error if it occurred on the link.

Extend Sync field is used for sending the fast training sequence ordered sets (FTS) in L0s state of the link following by the SKIPOS in L0.

6. SYSTEM POWER STATE AND DEVICE POWER STATE

In the desktop/mobile segment SoC provide several system state and device states in order to save the power. Advanced Configuration and Power state (ACPI) provides to save the power.

ACPI system state varies from S0(full operational state) to S5 (lowest power saving state). The transition of system state is totally controlled by the operating system; based on the input given by the user as Sleep/Hibernate or enter into low power state when there is inactivity timeout. When the system is in S0 considered as active operational and when it is in S5 considered as shutdown mode.

The system can only be transits from S5 to S0 only by the wake event from the OS. All these S- states have their different latencies for entry and exit. Below mention items relates the possible link states of PCIe:

- S0(operational state): this state is the fully operational working state of the system. In this state system can enter into low power mode. From here the system can enter C2/3 with the support of PCIe IP through the flow credit mechanism. The downstream devices consume the flow credits which indicates to BME on the system. Then the control logic of the power management decides to go into low power state.
- S1 Standby State: this state is considered as the stand by state and the entry and exit latency of this state is very less. In this state all the clocks and PLLs are on but depending on the system parameters devices can go into lower power device state and PCIe link will be in either L1 or L0s state.
- S3 (Sleep State) and S4 (Hibernate State): S3 is the low latency state of the system. In this state only memory contents get saved and other device contents go for a toss. S4 is lowest deep sleep state of the system with longest latency of entry/exit flow. In this state all the system contents get saved into main memory of the system. All the clocks and PLLs get shutdown except the wake logic. The PCIe link can enter into L2 or L3 depending upon the endpoint capability.
- S5 (Shutdown State): S5 is the complete shutdown state of a system in which no content of the system gets saved. This state requires full boot process to wake up.

Below table explains the relationship between the system and PCIe link state:

S-state	Permissible L-state
S1	L0s or L1
S1/POS	L2
S3	L2 or L3, where L2/L3 Ready is a transitional state
S4	L2 or L3, where L2/L3 Ready is a transitional state
S5	L2 or L3, where L2/L3 Ready is a transitional state

Fig 27: State and L State

Device Power State

ACPI also specifies the Different Power or D states of the device. Below mentioned points will explain the relation between the PCIe link state and Device D state:

- D0: this state is the full ON state of the device in which it is fully operational. In this state PCIe link can be in L0s or in L0. For example, link is configured to go in L0s then ASPM will transit the TX, RX of both the side into L0s. The link can go into L1 either through hardware ASPM or by the OS make device to go into D1, D2 State.
- D1, D2: In both D1 or D2 state the PCIe link should be in L1 state. The only difference between D1 and D2 state is power saving percentage. D2 save more power than D1.
- D3: D3 is the lowest deep sleep D state of the device in which device become totally un operational. As all the power supply to the device get cut off and it is also known as run time D3 state. And in this state the PCIe link will be in L2. And, device can only be exit to D0 by the Wake signal or the PERST signal on the PCIe link.

Below table explains the relationship between the system and PCIe link state:

Downstream Component D-state	Permissible Upstream Component D-state	Permissible L-state
D0	D0	L0, L0s, or L1
D1	D0-D1	L1
D2	D0-D2	L1
D3 _{hot}	D0-D3 _{hot}	L1 or L2/L3 ready
D3 _{cold}	D0-D3 _{cold}	L2 or L3

Fig 28: D state and Link State

7. STATEMENT OF PROBLEM BASED ON IDENTIFIED RESEARCH GAPS

There are increasingly number of digital/analog/mixed signal circuits in microprocessors and SOCs. A major chunk of mixed-signal circuits are high-speed I/O links, including serial buses such as PCIe and parallel buses such as DDR. Post-si validation is a major step for any product manufacturing industry who is engaged in releasing the products like mobile or desktop into the market. Post-silicon validation of high-speed input/output (HSIO) links can be crucial for making a product release qualification. Peripheral component interconnect express (PCIe) is a high-performance serial interconnect architecture widely used in the computer industry, and one of the most complex HSIO IP. PCIe data rates or throughput increases on every new generation.

PCI-Express (PCIe) is mainstream interface of today's system of chip which requires high speed data transmission with high performance and throughput. It is being used immensely in different applications like computer cards, graphic cards, automotive, networking, and industrial and consumer applications. In automobile world or automotive applications, PCIe is useful for processing of data coming at a very high speed from real-time graphics and video processing. And hence, all these reasons make PCIe as an important part of today application.

Therefore, PCIe IP should be validated in all the possible scenarios like:

1. The link should always be stable and be in L0 State.
2. The link should work properly even with the power management flows enabled.
3. The link should work in all operating condition i.e.; it should be independent of process variation (PVT).

Thus, post si validation boundaries have exponentially increased in latest multi-core projects, not only due to their complex architecture, but also because the PRQ (Product release Qualification) demands keep getting smaller, as measured from first available prototype units until high-volume manufacturing approval. CPU validation complexity increases are due to the need to support existing legacy features while adding new features like new macro-instructions, new micro-architectural functionality and new power management mechanism.

Therefore, this report gives the solution of this problem that how using these debugging tools in effective way leads to good validation in terms of performance and accuracy.

8. POST SILICON VALIDATION PARADIGM

Post Silicon Validation Paradigm

Post silicon validation paradigm is a process which is followed for validating the IPs on the soc. And, this process is the base on which we will design our validation strategy for an IP which present in the soc.

The post silicon validation paradigm involves 6 phases such as:

- Power On Ready
- Power on Exit
- Volume Validation
- Tested
- Proven
- PRQ Ready

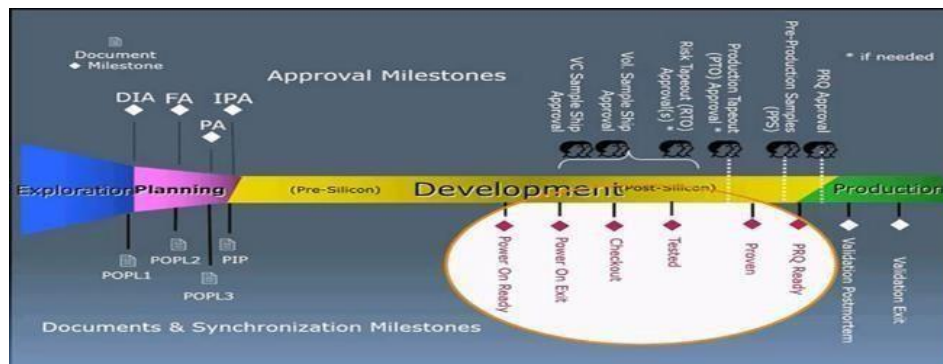


Fig29: Phases in Post Silicon Validation

Power on Ready

Power on is a crucial event and very first step in a project towards the feature enablement and validation. Power On enables volume deployment of silicon and platform for further detailed validation. Power on event helps to bring out early critical issues with silicon and helps to judge the initial health of the silicon.

Power on Exit

Once Power on scope and schedule is defined. Power on exit criteria (Goals) needs to communicate to all the stakeholders by power on lead. Power on exit goals are measured mainly on four vectors i.e., functional, stress and stability.

Volume Validation

After Power On exit phase then Checkout happens i.e., the volume validation starts by the all stake holders that means the SUT remains under test for long no test iteration cycle.

Tested and Proven

In tested and proven phase we identify no of bugs that are detected and the no of bugs that get fixed in the silicon. A report is made by the project owner and to represented to the management. And from this efficiency and quality of validation gets measured.

PRQ Ready

PRQ ready refers to the Production Ready Quality i.e., the quality silicon is ready and can be go into the market and can be used by the end user. This silicon is free from all hardware and software bugs.

8.1 Tools Used for Execution and Debugging in Post Silicon Functional Validation

The functional validation is carried out by two types of validation i.e. windows based validation and other is using the synthetic content.

Windows based Validation (OSBV) involves some OS based application which run SUT which keep the system under test for a long time to find the bugs. The OSBV application that are used for e.g.: SOLAR, RW etc.

SOLAR: This application is used to Sx cycling that means to test PM flows on the RVP and keeps the SUT under stress. PM flows involves: warm reset, cold reset and SX cycling such as S3, S4, and S5.

RW: This is a RW tool which is used to check the device enumeration in the PCIe which gives the information about the devices connected to the PCIe slot on the system. This application gives the information about the BDF of the device that is connected on the system.

Synthetic Content Validation refers to the test cases which are written in a high level language like Python. This type of content is used to validate the link status, features and registers of the PCIe IP. And, also validates the functionality of the IP in low power states. This type of content involves the features like link enable/disable, speed change, hot reset etc. And to run this type of content we need hardware probes like DCI, XDP which connects on to the system and gives the access to the register of the IP.

There is another tool called Status Scope Dump which is used to analyze and collect the logs when the system gets unexpectedly shut down or getting a blue screen error. Now there are some hardware tools which are used for debugging such as:

TLA: Tektronix Logic Analyzer

This tool used to detect and analyze the link state of the PCIe link and also can be used to measure the latencies of the different states of the link. Below fig explains the output window of the TLA.



Fig30 : TLA GUI

Oscilloscope : this tool is used to measure different voltage signal or the power to the PCIe slot.

And to debug at the traffic level / the packets which are getting exchanged between the Tx and Rx then we use Lecroy Protocol analyzer

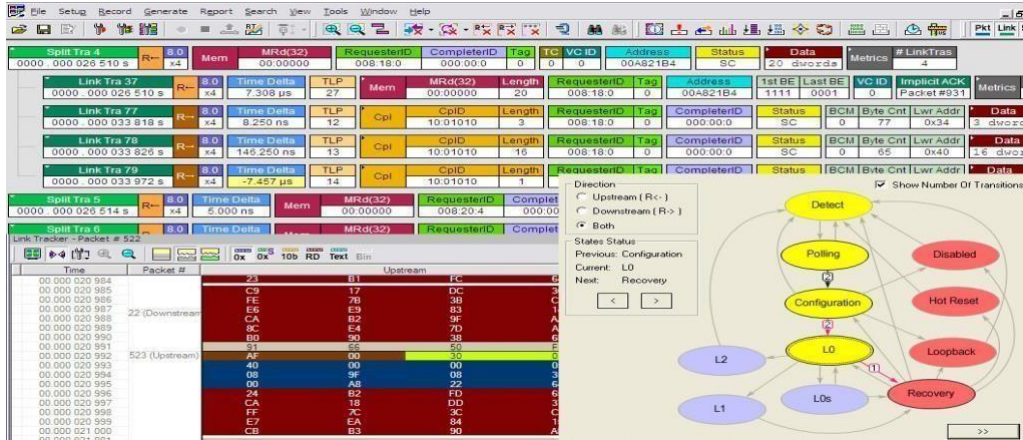


Fig 31: Lecroy Protocol Analyzer

9. Work done, partial simulation/hardware implementation, partial Results and Discussion.

9.1 Overview of work done

In functional validation of PCIe it requires three types of work to do as:

- Implementing PCIe and System validation test cases which have dependency on PCIe.
- Creating all the end user case scenario before the product got PRQ.
- Analyzing and debugging of the cases to resolve the bug.
- All these test cases

9.2 Test cases implemented with OSBV

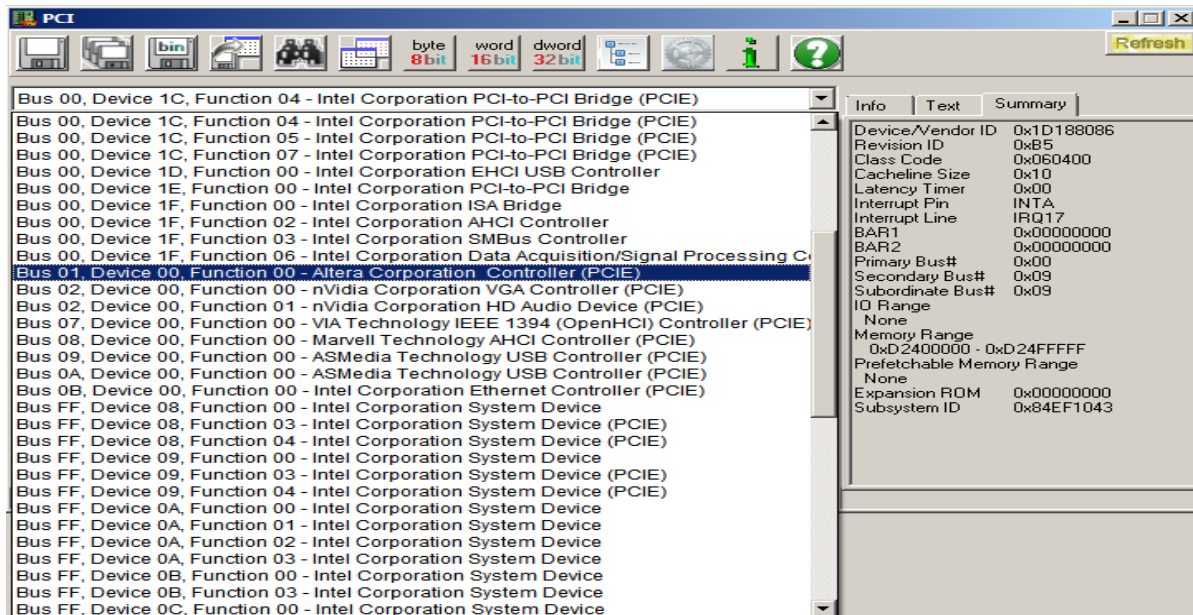
Scenario 1: To test PCIe devices get enumerated when connect on the SoC/System

Passing Criteria : All the devices should get there unique BDF (Bus,Device,Function) that means no device should get unaddressed.

To check this scenario we have a tool called Read Write which gives all the information about BDF assigned to each device.

RW Tool: This is an application which is used for verifying the enumeration of the device such as BDF of the device and also to verify the device Id.

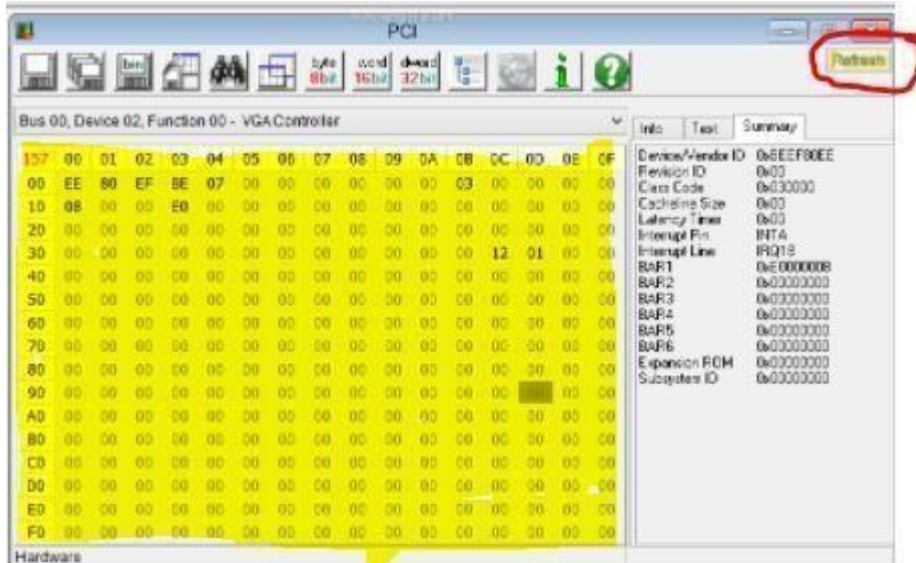
Output:



Scenario 2 : To test the link transits properly from low power state(L1 to L0) without degrading the quality of link

Passing Criteria: Link should properly transit without any speed and width drop.

For this scenario we can use the same above mentioned RW tool to stress the link



If you see in above gui of the tool there is one refresh tab. If set the refresh time it will send the Rd transaction and the link will go in L0 and after that again it will come in L1 state. This in turn we can check the status of the link through synthetic content as below

Output:

Link Active : 0x1	→	Link Active : 0x1
Link State : 0x33		Link State : 0x40
LTSSM : L0		LTSSM : L1
Link Speed : 0x4		Link Speed : 0x4
Link Width : 0x16		Link Width : 0x16

9.3 Test Cases implemented with Synthetic Content

Synthetic Content: This type of content validates the features of PCIe such as speed change, hardware equalization etc. by executing the test cases which are written in a high level language.

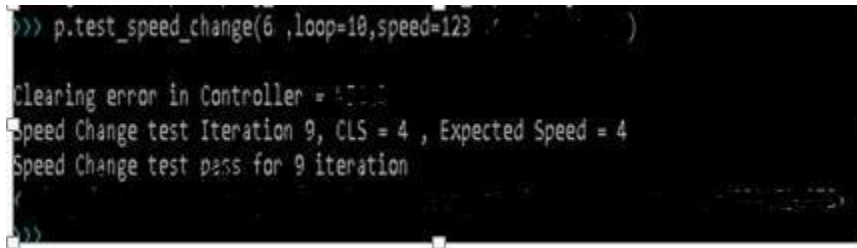
Scenario 1: Speed Change; this test case validates the link is working properly in every Gen speed.

Passing Criteria : link should get up in every gen speed without any errors. As, PCIe exhibits previous compatibility parameters.

Test Code:

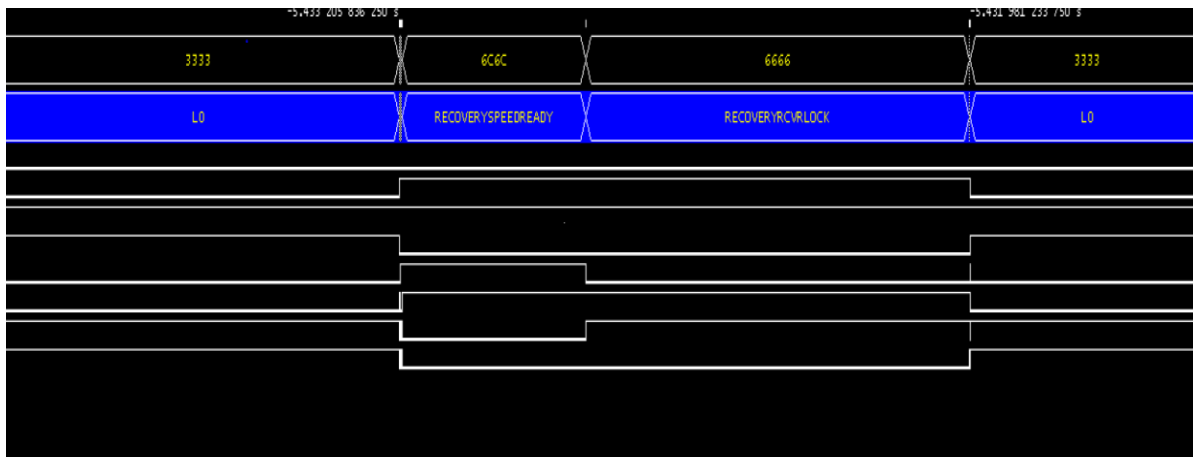
```
def test_speed_change (pcieslotnum, loop, diff_speed=1, 2, 3)
    slot = cpu_slot (pcieslotnum)
    if (diff_speed == 1, 2, 3):
        seq=[1, 2, 3]
    elif (diff_speed == 1, 2):
        seq=[1, 2]
    elif (diff_speed == 2, 3):
        seq = [2, 3]
    elif (diff_speed == 1, 3):
        seq = [1, 3]
    count = 0
    for count in range(loop):
        expectedspeed = random.choice (seq)
        currentlinkspeed= int (pcieslotnum.linkstatus.currentlinkspeed)
        while (currentlinkspeed == expectedspeed):
            expectedspeed=random.choice (seq)
        if (currentlinkspeed!=expectedspeed)
            print ("Error: test failed")
        elif:
            print ("speed change pass for %s iteration" % (count))
```

Output:



```
>>> p.test_speed_change(6, loop=10, speed=123)
Clearing error in Controller = 4000
Speed Change test Iteration 9, CLS = 4, Expected Speed = 4
Speed Change test pass for 9 iteration
>>>
```

TLA Capture:



Scenario 2: Hardware Equalization; this test case is used to validate the PCIe link quality with the help of pre-set and coefficient values that link should be stable in particular state. This preset and coefficient values are given by the electrical validation team.

Passing Criteria: link should be stable in particular state without any recoveries.

Test Code:

```
def test_hwequalization(pcieslotnum,loop,speed=4)
    slot = cpu_slot(pcieslotnum)
    if (slot.slotstautus=1 and slot.linkstatus=1):
        test_hwequalization(pcieslotnum,speed,loop)
        print("hweq on the pcie slot")
    if(slot.linkstatus!=1)
        print("hweq fails")
    count=0
    for count in range(loop):
        expectedspeed = 4
        currentlinkspeed= int(pcieslotnum.linkstatus.currentlinkspeed)
        print_hwequalization(pcieslotnum) #print_hweq is another func called inside main function to print equalization coeff.
```

Output:

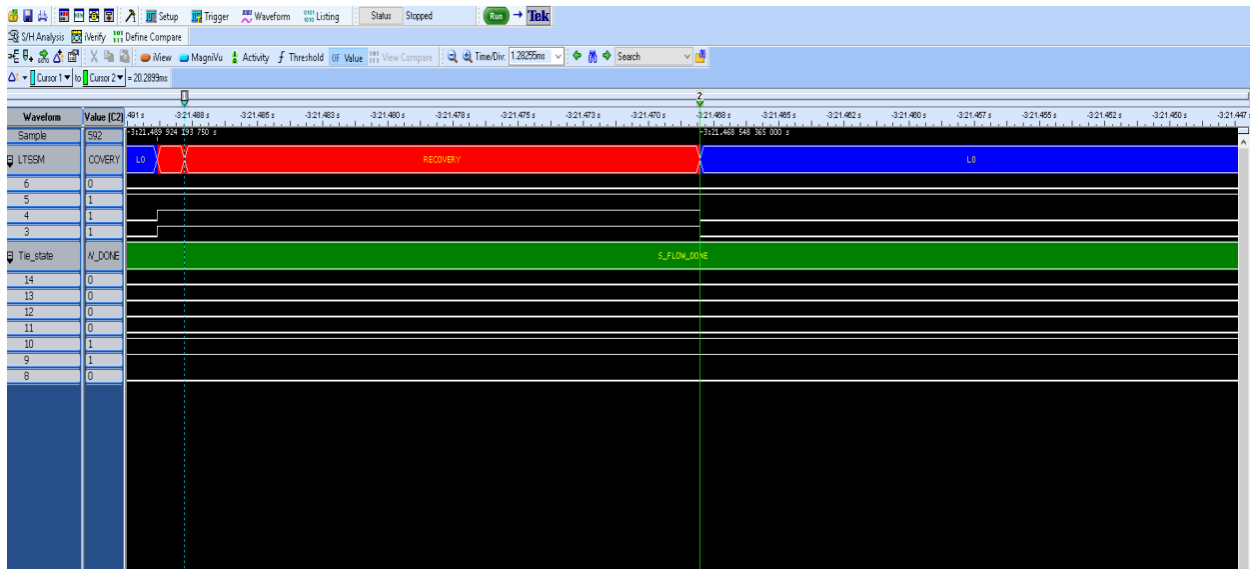
```
>>> p.test_hweq(ctrl_num=1, speed=4, loop=10000, t

HWEQ test on Controller
HWEQ Test iteration= 0

HWEQ Preset/Co-eff in Controller
Speed: Gen3
HWEQ: Preset Mode
+-----+
| Lane0 | Lane1 | Lane2 | Lane3 |
+-----+
| 0      | 0      | 0      | 0      |
+-----+

HWEQ Preset/Co-eff in Controller
Speed: Gen4
HWEQ: Preset Mode
+-----+
| Lane0 | Lane1 | Lane2 | Lane3 |
+-----+
| 8      | 9      | 9      | 9      |
+-----+
Speed = Gen4
+-----+
| List  | Lane0 | Lane1 | Lane2 | Lane3 |
+-----+
| 0x0   | 91    | 92    | 92    | 97    |
| 0x7   | 94    | 89    | 90    | 102   |
| 0x8   | 101   | 99    | 101   | 101   |
| 0x9   | 94    | 102   | 104   | 106   |
+-----+
```

TLA Capture :



Scenario 3: Hot Reset; this test case is used to validate because if any reset occur on the link : link should come to its operational state.

Passing Criteria: link should come in L0 (operational state)

Test Code:

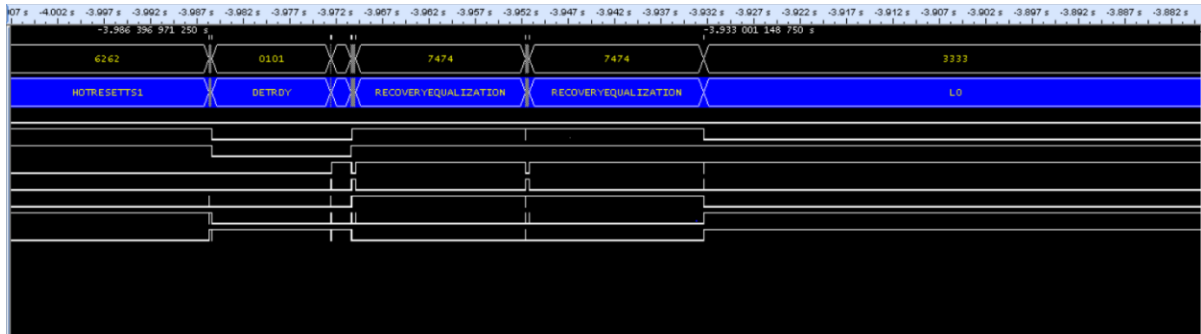
```
def test_hot_reset(pcieslot,speed,pcielinkstatus):
    pcieslot.configregister.sbr=1
    time.sleep()
    ltssmstate = int(pcieslot.configregisterfg.ltssmstate)
    pcieslot.configregister.sbr=0
    time.sleep()
    currentlinkspeed = int(pcieslot.configregisterfg.linkstatus.currentlinkspeed)
    linkwidth = int(pcieslot.configregisterfg.linkstatus.linkwidth)

    if ( debug == 1 ):
        print("Hot Reset test pass")
```

Output:

```
>>>p.test_hot_reset(6,speed=4,loop=5)
Hot_Reset passed for 5 iterations
```

TLA:



10. CONCLUSION

Post Si validation is one of the most important and crucial step in the PRQ of the product into the market. As, in this era, there are lot of IPs or peripherals are present on the SoC and also contains complex circuitry on the chip. Major chunk of the silicon chip constitutes analog and digital circuits which contains high speed input output link interface called PCIe and other parallel links for example DDR. Validation of this link interface IPs are very hard to validate as the observability of the signals are very less in the post si domain. This thesis proposes functional validation of high speed serial links such as PCIe to improve the quality and Performance of the IP in the SoC with the help of correct debug tools. And, also describes the systematic approach to validate the IP in a correct way and to meet the market timing constraints.

10.1 Future Scope

As of now, the golden recipe is needed as a base to validate the IP to improve the quality of the post silicon validation. So, that a systematic approach to be deployed and the bugs can be removed easily and we can produce a quality product. An automation script / software can be used to analyze the log dumps of test cases which are produced by the tools which are used in post silicon like the synthetic and OSBV content as in post silicon lot of scenarios comes into the consideration. We can also try to bridge the gaps between pre and post silicon with respect to the test case execution content or methodology which can reduce the time of verification and validation and also in turn improve the deliverability of the product into the market. Therefore, there is a tool which we are using right now to bridge the gap and we have started to go into this direction. The tool is called Perspec which is developed by the cadence. Every domain folks can use this product be it Pre silicon and Post Silicon just that implementation environment will be different for post silicon it will be on actual hardware and for pre silicon it will be on FPGA or emulators. This Perspec tool give advantage to simulate same test content and allow to write various types of concurrency test in Pre – Silicon or Post Silicon domain.

Below figure explains usage of Perspec in multiple domain and below points explains the feature of tools.

- Same test case content across domain
- Debug Scenarios are easy
- Able to find coverage bugs easy
- Test randomization content are easy to implement
- Generates the test report for every test case in a systematic way.

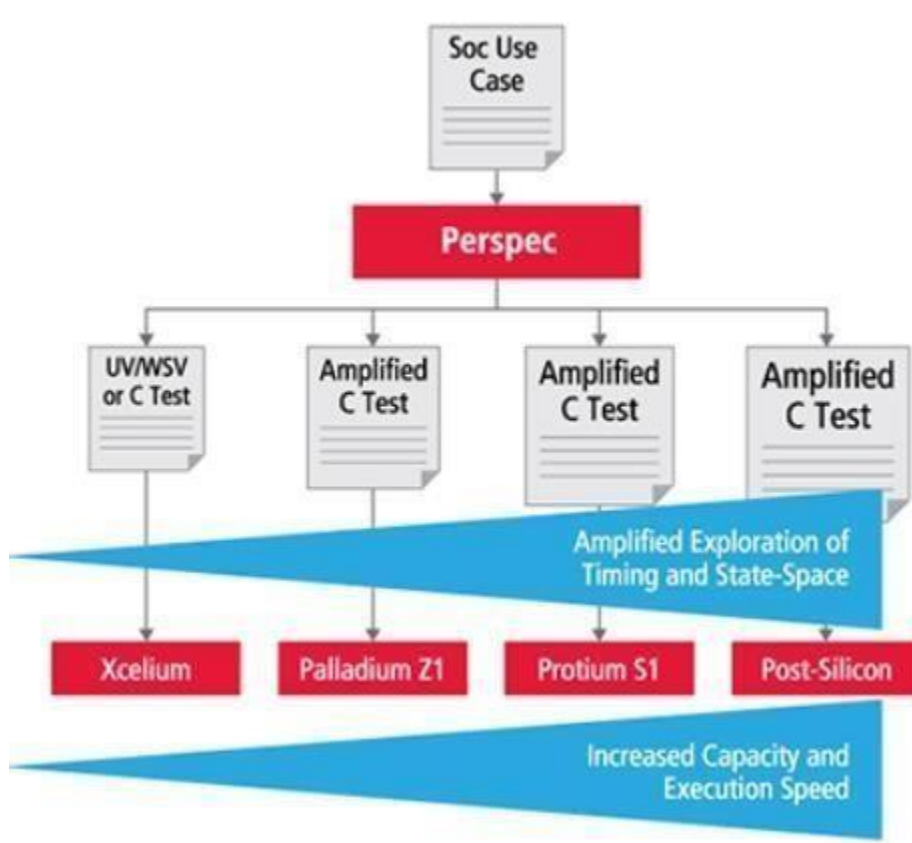


Fig 32: Perspec

Below flow chart explain the flow of the Perspec tool:

Perspec usage Flow

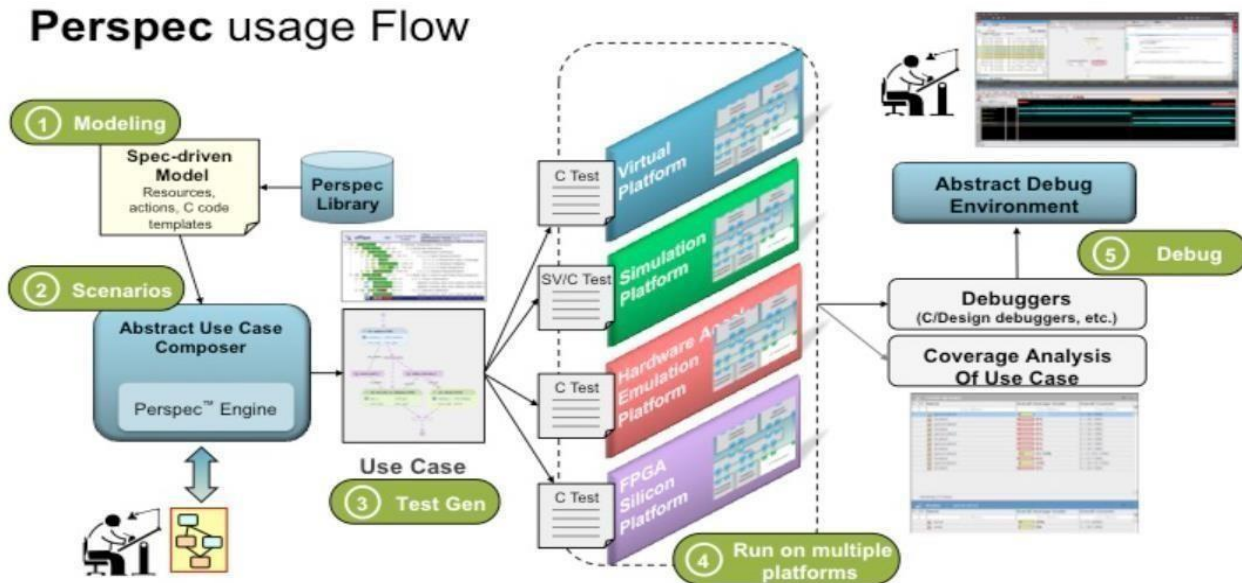


Fig : Perspec Tool Flow

For summarizing above flow chart in below points:

- Model the test case according to IP Spec
- Create multiple scenario at a time
- Run same content across different domains
- Provide additional window for debug

11. REFERENCES

- [1] Anysilicon, "What is a system on chip SoC," [Online]. Available:
<https://anysilicon.com/what-is-a-system-on-chip-soc/>
- [2] Chenjie Gu, "Challenges in post-silicon validation of high-speed I/O links", in IEEE, 2012.
- [3] Subhasish Mitra, Sanjit A. Seshia, Nicola Nicolici, "Post-Silicon Validation", Opportunities, Challenges and recent advances
<https://people.eecs.berkeley.edu/~sseshia/pubdir/postSi-dac10.pdf>
- [4] Sam Fleming "Accessing PCI Express Configuration Registers Using Intel® Chipsets", White Paper December 2008, Intel Corporation.
- [5] Jason Lawley, "Understanding Performance of PCI Express Systems", in Xilinx white paper,
https://www.xilinx.com/support/documentation/white_papers/wp350.pdf
- [6] PCI Express Base Specification, v2.0, <http://www.pcisig.com/specifications/pciexpress>.
- [7] Mike Jackson, Ravi Budruk, "PCI Express Technology", Mindshare Inc,
https://www.mindshare.com/Learn/PCI_Express
- [8] F. E. Rangel-Patino, J. E. Rayas-Sánchez and N. Hakim, "Transmitter and Receiver Equalizers Optimization Methodologies for High-Speed Links in Industrial Computer Platforms Post-Silicon Validation," 2018 IEEE International Test Conference (ITC), 2018, pp. 1-10, doi: 10.1109/TEST.2018.8624794.
- [9] P. Iyer, S. Jain, B. Casper, and J. Howard, "Testing High-Speed IO Links using On-Die Circuitry," in VLSI Design, 2006. Held jointly with 5th International Conference on Embedded Systems and Design., 19th International Conference on. IEEE, 2006, pp. 4–pp.
- [10] H. Nakamura, H. Takayama, Y. Yamaguchi and T. Boku, "Thorough analysis of PCIe Gen3 communication," 2017 International Conference on ReConFigurable Computing and FPGAs (ReConFig), 2017, pp. 1-6, doi: 10.1109/RECONFIG.2017.8279824.
- [11] Intel Corp, "PHY Interface for PCIe Architecture", in Intel white paper,
<https://www.intel.in/content/dam/doc/white-paper/phy-interface-pci-express-sata3-specification-v09.pdf>.
- [12] Verma, Anuj & Dahiya, Pawan, "PCIe BUS: A State-of-the-Art-Review" in IOSR Journal of VLSI and Signal Processing (IOSR-JVSP). 7. 24-28. 10.9790/4200-0704012428.

- [13] Neugebauer, Rolf & Antichi, Gianni & Zazo, Jose & Audzevich, Yury & López-Buedo, Sergio & Moore, Andrew, "Understanding PCIe performance for end host networking". 327-341. 10.1145/3230543.3230560,in SIGCOMM, Budapest Hungary.
- [14] Makowski, Dariusz, "Application of PCI express interface in high-performance video systems". 141-143. 10.1109/MIXDES.2015.7208498
- [15] Manikandan, P. & Manikandan, Juhee & Murugan s, Bala. (2013). Post Silicon Functional Validation from an Industrial Perspective. Middle East Journal of Scientific Research. 15. 1570-1574. 10.5829/idosi.mejsr.2013.15.11.11284
- [16] . B. Tommy, F. Igor and M. Robert, "Intel's Post Silicon functional validation approach," *2007 IEEE International High Level Design Validation and Test Workshop*, Irvine, CA, 2007, pp. 53-56, doi: 10.1109/HLDVT.2007.4392786.
- [17] Prabhat Mishra,Farimah Faramandi,"Post Silicon Validation and Debug". Available: <https://link.springer.com/book/10.1007/978-3-319-98116-1#about>
- [18] Prof. A.A.Shinde, Varsha Karambelkar,"Logic Analyzer",in International Journal of Engineering Research & Technology (IJERT).
- [19] David J. Rodgers," Protocol Solutions Group - The Protocol of the PHY High-speed Ethernet designs require protocol knowledge", in protocol analyzer white paper.
- [20]A. Kansal, R. Sinha and R. Jaiswal, "Fully automated regression tool for post silicon validation," 2017 8th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2017, pp. 1-4, doi: 10.1109/ICCCNT.2017.8203924..

PAPER NAME

THESIS_MAJOR_PROJECT-SURAJ-M.TE
CH.pdf

WORD COUNT

7308 Words

CHARACTER COUNT

42538 Characters

PAGE COUNT

49 Pages

FILE SIZE

2.3MB

SUBMISSION DATE

May 27, 2023 8:35 PM GMT+5:30

REPORT DATE

May 27, 2023 8:35 PM GMT+5:30

● **20% Overall Similarity**

The combined total of all matches, including overlapping sources, for each database.

- 17% Internet database
- 8% Publications database
- Crossref database
- Crossref Posted Content database
- 11% Submitted Works database

● **20% Overall Similarity**

Top sources found in the following databases:

- 17% Internet database
- 8% Publications database
- Crossref database
- Crossref Posted Content database
- 11% Submitted Works database

TOP SOURCES

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	dspace.dtu.ac.in:8080 Internet	3%
2	doku.pub Internet	2%
3	Mendoza-Bonilla, Jesus-Andres, Alejandro Cortez-Ibarra, Edgar-Andrei ... Crossref	1%
4	rei.iteso.mx Internet	<1%
5	Bojan Tommy, Frumkin Igor, Mauri Robert. "Intel&#x2019;s Post Silico... Crossref	<1%
6	tudr.thapar.edu:8080 Internet	<1%
7	hdl.handle.net Internet	<1%
8	design-reuse.com Internet	<1%