PCIe Logical Device Driver

Verification Plan

Change History

|  |  |  |  |
| --- | --- | --- | --- |
| Version No | Date | Change Detail | Author |
| 0.1 | 02nd Nov 2020 | Initial version | Vayavya Labs |
|  |  |  |  |

Table of Contents

[**Introduction**](#_vis82ydut4uq) **3**

[**PCIe Logical Device Driver architecture overview**](#_wmqvnsr23pam) **4**

[**UFS Host Controller Validation using LDD**](#_fynxd4gacgj2) **5**

[**Test Scenarios**](#_zeudsuiqflkd) **6**

[4.1. Test scenario - Pcie\_enum\_scenario.xml](#_2et92p0) 6

[4.2. Test scenario - Pcie\_read\_ep\_config\_scenario.xml](#_1nqq3mys5yhi) 7

[4.3. Test scenario - Pcie\_rc\_register\_read\_write\_scenario.xml](#_3zf02w2xt4fi) 8

[4.4. Test scenario - Pcie\_rc\_read\_ep\_mem.xml](#_4d34og8) 9

[4.5. Test scenario - Pcie\_rc\_link\_management\_scenario.xml](#_2xcytpi) 10

[4.6. Test scenario - Pcie\_rc\_band\_width\_management\_scenario.xml](#_3whwml4) 11

[4.7. Test scenario - Pcie\_test\_msi\_intr.xml](#_2p2csry) 12

[4.8. Test scenario - Pcie\_test\_msix\_intr.xml](#_tzx617ymg52v) 13

[4.9. Test scenario - Pcie\_test\_msix\_properties.xml](#_1v1yuxt) 14

[4.10. Test scenario - Pcie\_rc\_read\_ep\_mem\_multi\_bytes.xml](#_4f1mdlm) 15

# **Introduction**

This document describes the verification of a PCIe host controller interface using a logical device driver. The verification will be achieved by having unit level test cases for

1. PCIe host
2. PCIe host connected to PCIe devices

The testing will happen in its respective environment

**References**

[1] PCie LLD driver architecture document

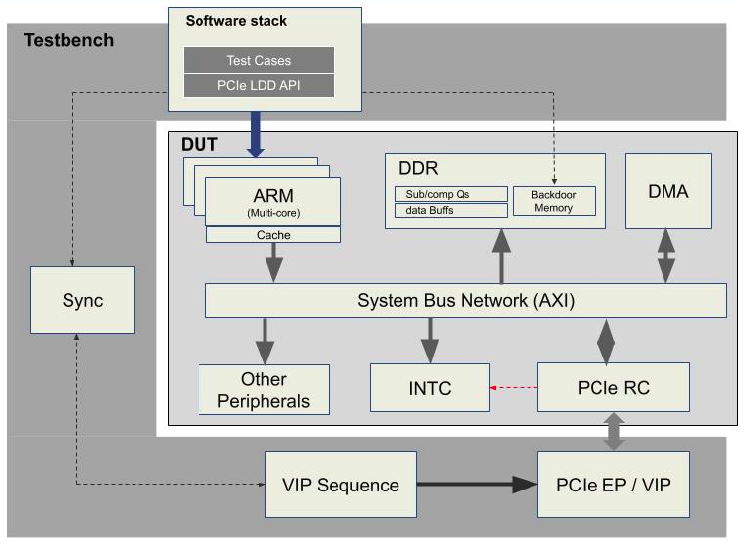
# **PCIe Logical Device Driver architecture overview**

PCIe is a high-speed serial computer expansion bus standard. It is the common motherboard interface for personal computers' graphics cards, hard drives, SSDs, Wi-Fi and Ethernet hardware connections. PCIe has numerous improvements over the older standards, including higher maximum system bus throughput, lower I/O pin count and smaller physical footprint, better performance scaling for bus devices, a more detailed error detection and reporting mechanism (Advanced Error Reporting, AER) and native hot-swap functionality. More recent revisions of the PCIe standard provide hardware support for I/O virtualization.



A logical device driver is a high level abstraction of how a device behaves. Physical device driver is a driver for a specific piece of hardware. The logical device driver talks to the physical device driver for you to keep you abstracted from the underlying hardware.

# **UFS Host Controller Validation using LDD**



# **Test Scenarios**

A test scenario is created using a tool called Platform Architecture Ultra (PAU) by connecting together tasks from the composite, protocol or driver layers to test the functionality of the PCIe subsystem. Each test scenario has a unique name and is saved in the repository as an XML file. To run the test scenario, it needs to be compiled successfully on the PAU platform. This will create an “.elf” file that can be executed on the VDK environment by specifying the executable file path. The test result will be displayed on the console window of the VDK.

The different test scenarios implemented are:

1. Pcie\_enum\_scenario
2. Pcie\_read\_ep\_config\_scenario
3. Pcie\_rc\_register\_read\_write\_scenario
4. Pcie\_rc\_read\_ep\_mem
5. Pcie\_rc\_link\_management\_scenario
6. Pcie\_rc\_band\_width\_management\_scenario
7. Pcie\_test\_msi\_intr
8. Pcie\_test\_msix\_intr
9. Pcie\_test\_msix\_properties
10. Pcie\_rc\_read\_ep\_mem\_multi\_bytes

**Note: Color coding** of the boxes in the flow diagrams has the following meaning:

## **4.1. Test scenario - Pcie\_enum\_scenario.xml**

**Brief description**: This test scenario is to test the PCIe enumeration.

It configures the PCIe controller driver and performs the PCIe enumeration by scanning the PCIe fabric to discover the machine topology and learn how the fabric is populated. Once the enumeration is completed, it is reported as a test passed.

**Flow diagram of tasks:**



## **4.2. Test scenario - Pcie\_read\_ep\_config\_scenario.xml**

**Brief description**: This test scenario is to test the PCIe endpoint configuration read. The read can be done only after enumeration is done.

It configures the PCIe controller driver and performs the PCIe enumeration. Once the enumeration is completed, It reads the configuration area (i.e.,Vendor ID & Device ID) of the EP & then read value is compared with the expected value to confirm configuration read is successful.

**Flow diagram of tasks:**







## 

## 

## **4.3. Test scenario - Pcie\_rc\_register\_read\_write\_scenario.xml**

**Brief description**: This test scenario is to test the PCIe Root complex Register read and write.

It writes a value into the root complex register and then reads the same root complex register. Theread value will be compared with the written value to confirm if the root complex register read and write happened successfully.

**Flow diagram of tasks:**

## **4.4. Test scenario - Pcie\_rc\_read\_ep\_mem.xml**

**Brief description**: This test scenario is to test the PCIe endpoint memory read.

It will read the memory region associated with the BAR registers of the EP.

**Note**: In order to ensure a EP memory read & write we have restricted the enumeration to one device.

MAX\_DEV\_CNT - Macro to change the device count .

**Flow diagram of tasks:**



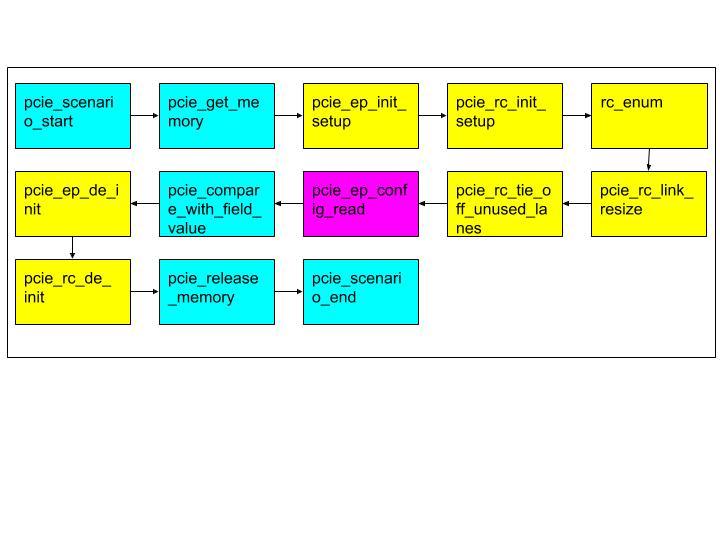
**Note:**

1. Please refer to the color coding information for type of tasks
2. Details of the different tasks executed in this test scenario as per the flow diagram of tasks are given in the subsequent chapters on “Test scenario tasks”, “Composite tasks”,“Protocol layer tasks” and “Driver layer tasks”.

## **4.5. Test scenario - Pcie\_rc\_link\_management\_scenario.xml**

**Brief description**: This test scenario is for link management. Here RC is requesting for the lane resize & tying off the unused lanes. If EP supports the requested lane size it automatically does lane resize to the requested lane & tie off the unused lanes automatically at EP side else it will reject the lane resize request of the RC. Once this is done test will read & compare EP register field value (PCIE\_CAP\_NEGO\_LINK\_WIDTH of register LINK\_CONTROL\_LINK\_STATUS\_REG) with requested lane size to confirm lane resize request has been done or not.

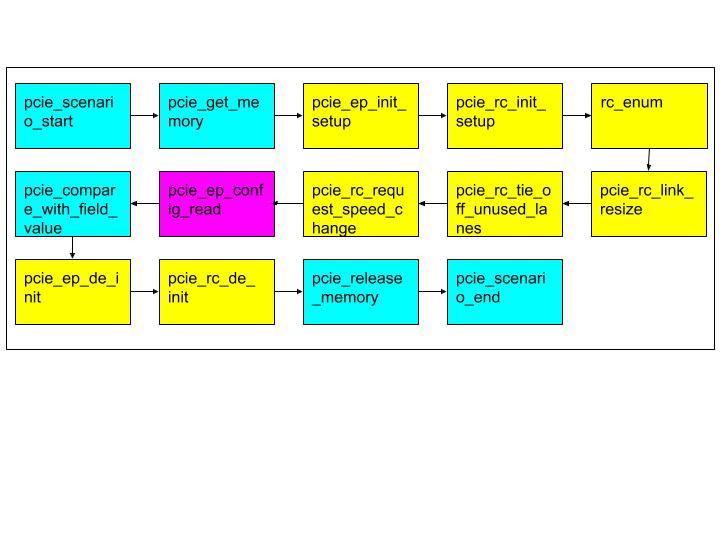
**Flow diagram of tasks:**

****

## **4.6. Test scenario - Pcie\_rc\_band\_width\_management\_scenario.xml**

**Brief description**: This test scenario is for bandwidth management. Bandwidth changes with change in the lane size & speed. Here RC is requesting for the lane resize & speed change. If EP supports the requested lane size & speed it automatically does lane resize to the requested lane & switches to that speed else it will reject the request of the RC. Once this is done test will read & compare RC register field value (PCIE\_CAP\_NEGO\_LINK\_WIDTH & PCIE\_CAP\_LINK\_SPEED of register LINK\_CONTROL\_LINK\_STATUS\_REG) with requested lane size & speed to confirm lane resize & speed request has been changed or not.

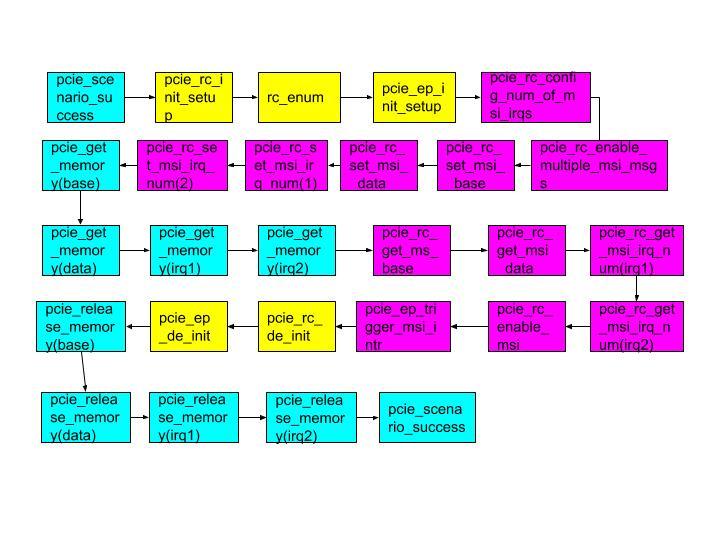
**Flow diagram of tasks:**

****

## **4.7. Test scenario - Pcie\_test\_msi\_intr.xml**

**Brief description**: Test scenario for MSI Interrupts. Configure the EP from RC and trigger the interrupt on EP.

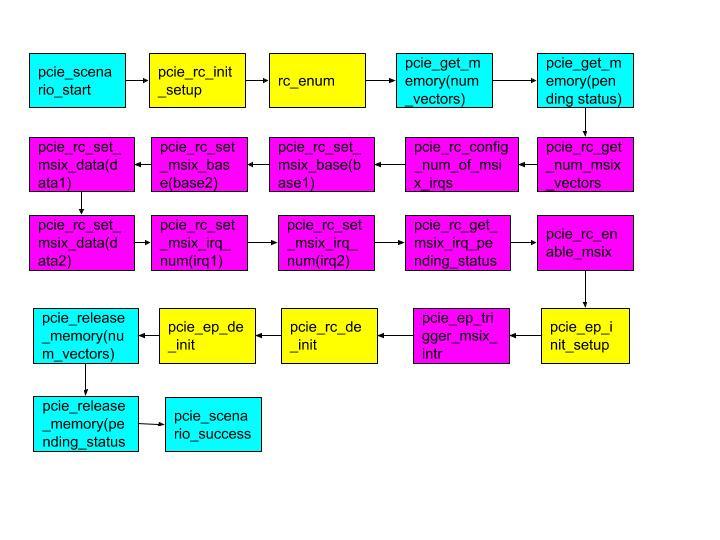
**Flow diagram of tasks:**

****

## **4.8. Test scenario - Pcie\_test\_msix\_intr.xml**

**Brief description**: Test scenario for MSIX Interrupts. Configure the EP from RC and trigger the interrupt on EP.

**Flow diagram of tasks:**

****

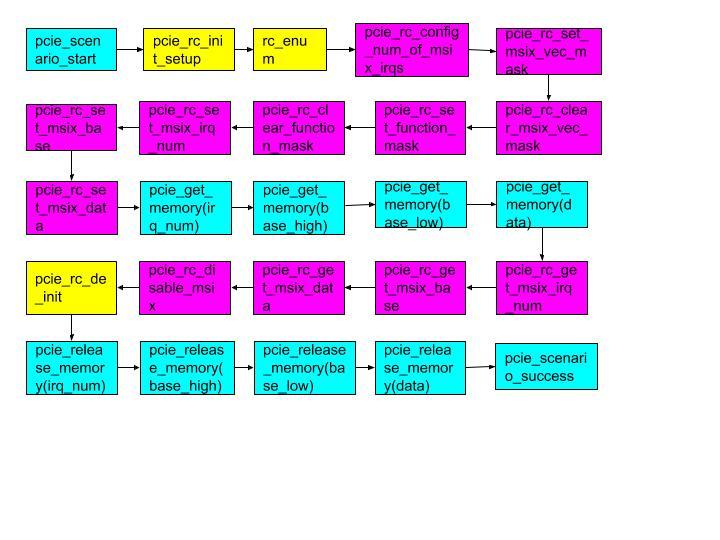
## **4.9. Test scenario - Pcie\_test\_msix\_properties.xml**

**Brief description**: Test msix functions.

Set and clear msix vector mask , function masks.

Set and get base, data and irq number.

**Flow diagram of tasks:**

****

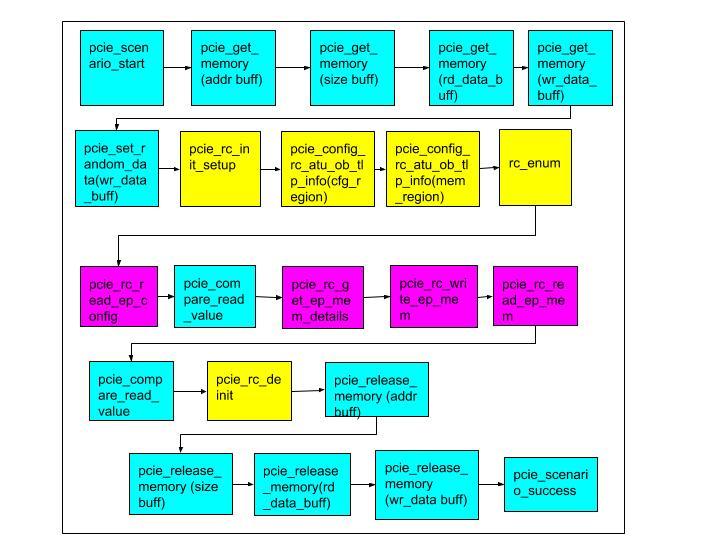
## **4.10. Test scenario - Pcie\_rc\_read\_ep\_mem\_multi\_bytes.xml**

**Brief description**: This test scenario will read the memory region associated with the BAR registers of the EP. This is an extension of the test scenario Pcie\_rc\_read\_ep\_mem to write and read multiple numbers of bytes.

**Note**: In order to ensure a EP memory read & write we have restricted the enumeration to one device.

MAX\_DEV\_CNT - Macro to change the device count .

**Flow diagram of tasks:**

****