**PCIe Logical Driver API**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Copyright (c) 2020, Vayavya Labs Pvt. Ltd. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted

provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions

and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of

conditions and the following disclaimer in the documentation and/or other materials

provided with the distribution.

Neither the name of Vayavya Labs Pvt. Ltd. nor the names of its contributors may be used to

endorse or promote products derived from this software without specific prior written

permission

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS

"AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT

LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A

PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL VAYAVYA LABS PVT.

LTD. BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY,

OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT

OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR

BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY,

WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR

OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF

ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

# Introduction

The Document describes the Generic APIs for the logical driver of PCIe RC and EP controller. These abstracted APIs will hide the datatype complexity defined at Physical level device driver(PDD) and helps users to build PCIe applications with reduced data type complexity and also aid in adopting the APIs in test automation framework like Open Hardware Software Interface standard(OpenHSI) .

# RC LDD WRAPPER API

## 2.1 pcieRcInit

**Synopsis:**

status\_t pcieRcInit(pcieRootcompDataHandle\_t \*pdata);

**Purpose:** Initialize RC

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

**Return Value:**

0 on success and -1 on failure

## 2.2 pcieRcCfg

**Synopsis**:

status\_t pcieRcCfg(pcieRootcompDataHandle\_t \*pdata,

unsigned long pcieAddr,

unsigned long cpuAddr,

unsigned long epCfgAddr,

unsigned long rcCfgAddr,

unsigned long addrSpaceSize,

unsigned char numRdDmaChnls,

unsigned char numWrDmaChnls,

unsigned char numObRegions,

unsigned char numIbRegions);

**Purpose:**  Configure RC

**Arguments:**

pcieAddr PCIe Address

cpuAddr CPU Address

epCfgAddr EP configuration Address

rcCfgAddr RC configuration Address

addrSpaceSize Address space size

numRdDmaChnls Number of read DMA channels

numWrDmaChnls Number of write DMA channels

numObRegions Number of outbound regions

numIbRegions Number of inbound regions

**Return Value:**

0 on success -1 on failure

## 2.3 pcieRcConfigAtuIbRegion

**Synopsis:**

status\_t pcieRcConfigAtuIbRegion(pcieRootcompDataHandle\_t \*pdata,

unsigned long baseAddr,

unsigned long trgtAddr,

unsigned long len,

unsigned int type,

unsigned int ibRgn,

unsigned char barNum);

**Purpose:**  Configure RC ATU inbound region

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

baseAddr Base address

trgtAddr Target address

len Length

type Type

ibRgn Inbound region

barNum BAR number

**Return Value:**

0 on success and -1 on failure

## 2.4 pcieRcConfigAtuObRegion

**Synopsis:**

status\_t pcieRcConfigAtuObRegion(pcieRootcompDataHandle\_t \*pdata,

unsigned long baseAddr,

unsigned long trgtAddr,

unsigned long len,

unsigned int type,

unsigned int obRgn);

**Purpose:**  Configure RC ATU outbound region

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

baseAddr Base address

trgtAddr Target address

len Length

type Type

obRgn Outbound region

barNum BAR number

**Return Value:**

0 on success and -1 on failure

## 2.5 pcieRcConfigReadWrite

**Synopsis:**

status\_t pcieRcConfigReadWrite(pcieRootcompDataHandle\_t \*pdata,

unsigned int regOffset,

unsigned int \*buffer ,

unsigned int capId,

unsigned char configSpaceSelect,

unsigned char accessSize,

readWriteFlag rw);

**Purpose:**  read or write RC configuration area

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

regOffset Offset to the register

\*buffer Output buffer into which data is read or whose content is used for write

capId Capability ID

configSpaceSelect Selection of configuration space

accessSize Access size

rw Read or write action specifier

**Return Value:**

0 success and -1 on failure

## 2.6 pcieRcDeinit

**Synopsis:**

**void pcieRcDeinit(pcieRootcompDataHandle\_t \*pdata);**

**Purpose:**  Deinitialize RC and release relevant memory

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

**Return Value:**

## 2.7 pcieRcDeinitprint

**Synopsis:**

void pcieRcDeinitprint(pcieRootcompDataHandle\_t \*pdata);

**Purpose:**  Deinitialize print API link

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

**Return Value:**

## 2.8 pcieRcEnumerate

**Synopsis:**

status\_t pcieRcEnumerate(pcieRootcompDataHandle\_t \*pdata);

**Purpose:**

PCIe enumeration of entire fabric under RC

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

**Return Value:**

0 on success and -1 on failure

## 2.9 pcieRcGetEpMemDetails

**Synopsis:**

status\_t pcieRcGetEpMemDetails(pcieRootcompDataHandle\_t \*pdata,

unsigned short vendorId,

unsigned short pcieDeviceId,

unsigned char barNum,

unsigned char bus,

unsigned char dev,

unsigned char fun,

unsigned int \*addr,

unsigned int \*size);

**Purpose:**

Get EP memory details. This will provide the base address and size of the memory relative to the asked BAR number of the device.

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

vendorId EP Vendor ID

pcieDeviceId EP Device ID

barNum BAR number

bus Bus number

dev Device number

fun Function number

\*addr Address pointer to EP memory

\*size Pointer to size value

**Return Value:**

0 on success and -1 on failure

## 2.10 pcieRcPreInit

**Synopsis:**

status\_t pcieRcPreInit(pcieRootcompDataHandle\_t \*\*tmp\_pdata);

**Purpose:**

Allocate private data structures and get driver API pointers

**Arguments:**

\*\*tmp\_pdata Pointer to RC private data structure of PCIe stack pointer

**Return Value:**

0 on success and -1 on failure

## 2.11 pcieRcProgConfigRegion

**Synopsis:**

status\_t pcieRcProgConfigRegion(pcieRootcompDataHandle\_t \*pdata);

**Purpose:**

Program RC configuration region

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

**Return Value:**

0 on success and -1 on failure

## 2.12 pcieRcProgPrefetchIoLimit

**Synopsis:**

status\_t pcieRcProgPrefetchIoLimit(pcieRootcompDataHandle\_t \*pdata,

unsigned int rcBarConfigReg ,

unsigned int type1BaseLmtCntrl,

unsigned int enableOrDisable);

**Purpose:**

Program prefetch IO limit

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

rcBarConfigReg RC BAR configuration register

type1BaseLmtCntrl type 1 base lmt cntrol

enableOrDisable Enable or Disable

**Return Value:**

Value on success and ERRORINVAL on failure

## 2.13 pcieRcReadEpConfig

**Synopsis:**

status\_t pcieRcReadEpConfig(pcieRootcompDataHandle\_t \*pdata,

unsigned int \*buffer,

unsigned short offset,

unsigned char size,

unsigned char bus,

unsigned char pcieDevice,

unsigned char function);

**Purpose:**

Read EP configuration area

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

\*buffer Pointer to output buffer into which data will be read

offset Register offset

size size

bus Bus number

pcieDevice Device number

function) Function number

**Return Value:**

0 on successful read and EINVAL on failure

## 2.14 pcieRcReadEpMem

**Synopsis:**

status\_t pcieRcReadEpMem(pcieRootcompDataHandle\_t \*pdata,

unsigned int offset,

unsigned char \*dataPtr,

unsigned int \*addrPtr,

unsigned int numOfBytes);

**Purpose:**  Read EP memory

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

offset Offset

\*dataPtr Pointer to output data buffer

\*addrPtr Pointer to memory address

numOfBytes Number of bytes to be read

**Return Value:**

0 on success and -1 on failure

## 2.15 pcieRcWriteEpConfig

**Synopsis:**

status\_t pcieRcWriteEpConfig(pcieRootcompDataHandle\_t \*pdata,

unsigned short offset,

unsigned char size,

unsigned char bus,

unsigned char pcieDevice,

unsigned char function,

unsigned int value);

**Purpose:**

Write EP configuration area

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

\*buffer Pointer to input buffer from which data will written

offset Register offset

size size

bus Bus number

pcieDevice Device number

function Function number

**Return Value:**

0 on successful read and EINVAL on failure

## 2.16 pcieRcWriteEpMem

**Synopsis:**

status\_t pcieRcWriteEpMem(pcieRootcompDataHandle\_t \*pdata,

unsigned int offset,

unsigned char \*dataPtr,

unsigned int \*addrPtr,

unsigned int numOfBytes);

**Purpose:**

Write EP memory

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

offset Offset

\*dataPtr Pointer to input data buffer

\*addrPtr Pointer to memory address

numOfBytes Number of bytes to be written

**Return Value:**

0 on success and -1 on failure

## 2.17 pcieRcAtuObCfgTlpInfo

**Synopsis:**

status\_t pcieRcAtuObCfgTlpInfo(pcieRootcompDataHandle\_t \*pdata,

unsigned int ob\_rgn,unsigned int format,

unsigned int tlp\_type, unsigned int traffic\_class,

unsigned int is\_id\_order, unsigned int tlp\_hints, unsigned int tlp\_digest,

unsigned int poisoned\_data, unsigned int attr, unsigned int address\_type,

unsigned int length, unsigned int tag, unsigned int processing\_hints);

**Purpose:**

Configure ATU outbound TLP information

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

ob\_rgn Outbound region

format Format

tlp\_type TLP type

traffic\_class Traffic class

is\_id\_order Indicates if the ID is ordered ID

tlp\_hints TLP hints

tlp\_digest TLP digest

poisoned\_data Indicates if data is poisoned data

attr Attribute

address\_type Address type

length Length

tag Tag

processing\_hints Processing hints

**Return Value:**

0 on success and -1 on failure

## 2.18 pcieRcRegWrite

**Synopsis:**

status\_t pcieRcRegWrite (pcieRootcompDataHandle\_t \*pdata, uint32\_t size, uint32\_t offset, uint32\_t data);

**Purpose:**  Register write

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

size Size

offset Offset address

data Data to be written

**Return Value:**

0

## 2.19 pcieRcRegRead

**Synopsis:**

status\_t pcieRcRegRead (pcieRootcompDataHandle\_t \*pdata, uint32\_t size, uint32\_t offset, void \*data);

**Purpose:**

Register write

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

size Size

offset Offset address

data Data to be written

**Return Value:**

0

## 2.20 pcieRcResizeLink

**Synopsis:**

status\_t pcieRcResizeLink(pcieRootcompDataHandle\_t \*pdata, int link\_width);

**Purpose:**  Alter the PCIe link size

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

link\_width Target link width

**Return Value:**

0 on success and -1 on failure

## 2.21 pcieRcUnusedTieOffLanes

**Synopsis:**

status\_t pcieRcUnusedTieOffLanes(pcieRootcompDataHandle\_t \*pdata, int lanes);

**Purpose:**  Tie off unused lanes

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

lanes Lanes

**Return Value:**

0 on success and -1 on failure

## 2.22 pcieRcSpeedChangeRequest

**Synopsis:**

status\_t pcieRcSpeedChangeRequest(pcieRootcompDataHandle\_t \*pdata, unsigned int target\_speed\_req);

**Purpose:**

Change PCIe link speed

**Arguments:**

\*pdata Pointer to RC private data structure of PCIe stack

target\_speed\_req Target link speed to be requested

**Return Value:**

0 on success and -1 on failure

# EP LDD WRAPPER API

## 3.1 pcieEpPreInitDev

**Synopsis:**

status\_t pcieEpPreInitDev(struct pcie\_dev\_ep\_prv\_data \*\*tmp\_drv\_data);

**Purpose:**

Allocate memory for device private data structure

**Arguments:**

\*\*tmp\_drv\_data Pointer to RC private data structure pointer of device

**Return Value:**

0 on success, -1 on failure

## 3.2 pcieEpCfgDev

**Synopsis:**

status\_t pcieEpCfgDev(struct pcie\_dev\_ep\_prv\_data \*drv\_data,

unsigned long ep\_base,unsigned long pcie\_axi\_addr,

unsigned int num\_of\_funcs,unsigned char num\_ob\_regions,

unsigned char num\_ib\_regions,

unsigned char num\_rd\_dma\_channels,unsigned char num\_wr\_dma\_channels );

**Purpose:**

configure EP device

**Arguments:**

\*drv\_data Pointer to RC private data structure of device

ep\_base EP base address

pcie\_axi\_addr AXI address

num\_of\_funcs Number of functions

num\_ob\_regions Number of outbound regions

num\_ib\_regions Number of inbound regions

num\_rd\_dma\_channels Number of read dma channels

Num\_wr\_dma\_channels Number of write dma channels

**Return Value:**

0 on success, -1 on failure

## 3.3 pcieEpConfigObDev

**Synopsis:**

status\_t pcieEpConfigObDev(struct pcie\_dev\_ep\_prv\_data \*drv\_data,

unsigned long base\_addr,

unsigned long trgt\_addr,unsigned long len,unsigned int type,

unsigned int ob\_rgn);

**Purpose:**

configure device outbound region

**Arguments:**

\*drv\_data Pointer to RC private data structure of device

base\_addr EP base address

trgt\_addr Target address

len Length

type Type

ob\_rgn Outbound region

**Return Value:**

0 on success, -1 on failure

## 3.4 pcieEpInitDev

**Synopsis:**

status\_t pcieEpInitDev(struct pcie\_dev\_ep\_prv\_data \*drv\_data);

**Purpose:**

EP device initialize

**Arguments:**

\*drv\_data Pointer to RC private data structure of device

**Return Value:**

0 on success, -1 on failure

## 3.5 pcieEpInitSetUpLdd

**Synopsis:**

int pcieEpInitSetUpLdd(struct ep\_prv\_data \*\*pdata\_pro,

struct pcie\_dev\_ep\_prv\_data \*\*drv\_data,

unsigned long ep\_base,

unsigned long pcie\_axi\_addr,

unsigned int num\_of\_funcs,

unsigned char num\_rd\_dma\_channels,

unsigned char num\_wr\_dma\_channels,

unsigned long base\_addr,

unsigned long trgt\_addr,

unsigned long len,

unsigned int type,

unsigned long atu\_ob\_base\_addr,

unsigned long atu\_ob\_trgt\_addr,

unsigned long atu\_ob\_len,

unsigned int atu\_ob\_type);

**Purpose:**

Configure PCIe stack ldd layer values

**Arguments:**

\*\*pdata\_pro Pointer to EP private data structure of device

\*\*drv\_data Pointer to RC private data structure of device

ep\_base EP base address

pcie\_axi\_addr AXI address

num\_of\_funcs Number of functions

num\_rd\_dma\_channels Number of read dma channels

num\_wr\_dma\_channels Number of write dma channels

base\_addr Base address

trgt\_addr Target address

len Length

type Type

atu\_ob\_base\_addr ATU outbound base address

atu\_ob\_trgt\_addr ATU outbound target address

atu\_ob\_len ATU outbound length

atu\_ob\_type ATU outbound type

**Return Value:**

0 on success, -1 on failure

## 3.6 pcieEpDeinitDev

**Synopsis:**

status\_t pcieEpDeinitDev(struct pcie\_dev\_ep\_prv\_data \*drv\_data);

**Purpose:**

Deinitialize device

**Arguments:**

\*drv\_data Pointer to RC private data structure of device

**Return Value:**

0 on success, -1 on failure

## 3.7 pcieEpDeinitialize

**Synopsis:**

status\_t pcieEpDeinitialize(struct pcie\_dev\_ep\_prv\_data \*drv\_data,struct ep\_prv\_data \*pdata );

**Purpose:**

Deinitialize memory allocated for device private data structures

**Arguments:**

\*drv\_data Pointer to RC private data structure of device

\*pdata Pointer to the PCIe EP stack

**Return Value:**

0 on success, -1 on failure