PCI Express

User's Guide

October 2005
ipug25_03.0

Introduction

PCI Express is a high performance, general purpose Serial I/O Interconnect defined for a wide variety of future computing and communication platforms. The basic premise of PCI Express is that the host PCI software remains compatible with an endpoint device without new drivers or operating-system software. Salient PCI attributes, such as its usage model, load-store architecture, and software interfaces, are maintained, whereas its bandwidth-limiting and parallel bus implementation is replaced by a highly scalable, fully serial interface. PCI Express takes advantage of recent advances in point-to-point interconnects, switch-based technology, and packetized protocol to deliver new levels of performance and features.

Lattice's PCI Express IP Core is an endpoint device supporting a x1 link. It consists of three layers of the endpoint device namely the Physical, Data Link and Transaction layers. This IP core targets the programmable array of the ORCA Series 4 ORT42G5 FPSC. The complete solution supports up to 2.5Gbps data rate as specified in *PCI Express Specification 1.0a*. For more information on this and other Lattice products, refer to the Lattice web site at www.latticesemi.com.

This user's guide explains the functionality of the Lattice PCI Express core and how it can be implemented to provide endpoint interface to an application running in the Transaction layer.

The PCI Express core comes with the following documentation and files:

- Data Sheet
- User's Guide
- Lattice gate level netlist
- Model for simulation
- Core instantiation template
- Testbench and testbench coding template

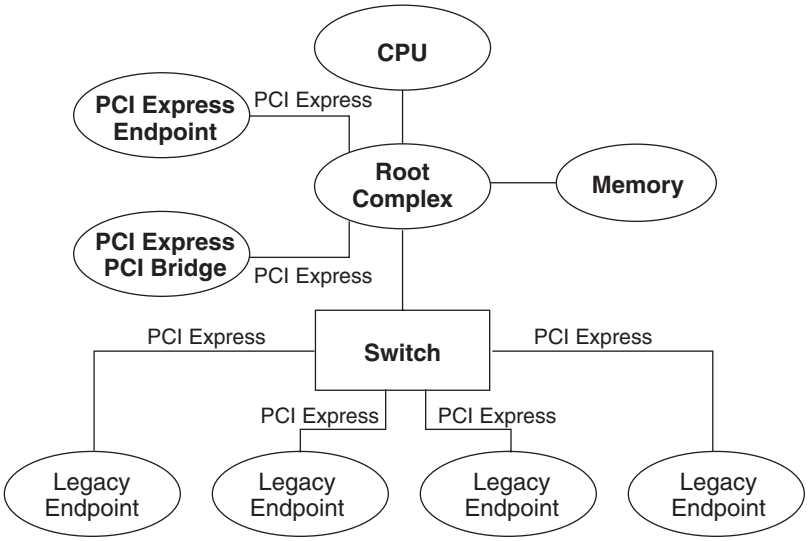
Features

- Lane width of x1 configuration.
- Effective raw data rate of 2.5Gbps/lane/direction (with target device support).
- 8b/10b encoding / decoding for symbols and special symbols (with target device support).
- Data scrambling / de-scrambling
- Link initialization and training.
- Flow control initialization.
- Data integrity checking for both Data Link Layer Packets and Transaction Layer Packets.
- Data Link Layer retry mechanism for transmitted Transaction Layer Packets.
- Acknowledgement and Timeout replay mechanisms.
- All error statuses are reported in the backend User Interface
- Credit availability calculation and reporting

General Description

PCI Express ties together various components within a system and provides connection points for future high-speed devices. The following diagram gives an overview of how PCI Express is used in a typical system.

Figure 1. Typical PCI Express Topology



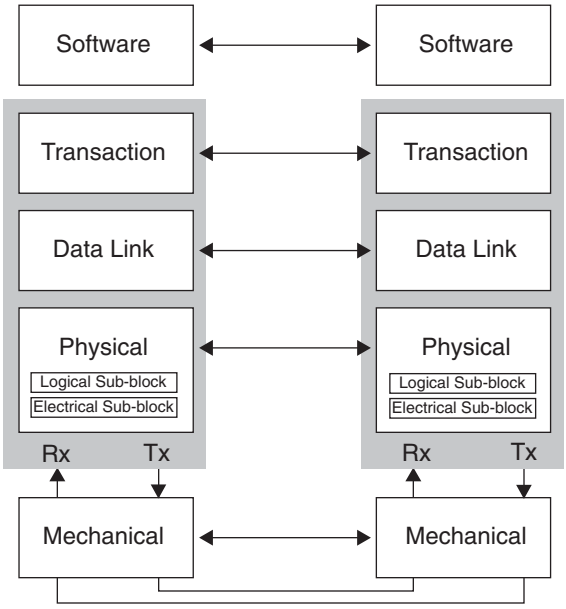
In the above environment a Root Complex denotes the root of an I/O hierarchy that connects the CPU/memory subsystem to the I/O.

A Switch device is defined as a logical assembly of multiple virtual PCI-to-PCI Bridge devices

An Endpoint device refers to a type of device that is either the Requester or Completer of a PCI Express transaction either on its own behalf or on behalf of a distinct non-PCI Express device (other than a PCI device or Host CPU), e.g., a PCI Express attached graphics controller or a PCI Express-USB host controller. Endpoints are either classified as legacy or PCI Express Endpoints. It is basically a device with a type 00h Configuration Space header.

PCI Express uses a packetized and layered protocol structure. The three main protocol layers implemented within the core are the Physical Layer, Data Link Layer and Transaction Layer. Packets are formed in the Transaction and Data Link Layers to carry the information from the transmitting component to the receiving component. As the transmitted packets flow through the other layer, they are extended with additional information necessary to handle packets at those layers. At the receiving side the reverse process occurs and packets get transformed from their Physical Layer representation to the Data Link Layer representation and finally to the form that can be processed by the Transaction Layer of the receiving device. The following figure shows the conceptual flow of packet information through the layers.

Figure 2. PCI Express Layers



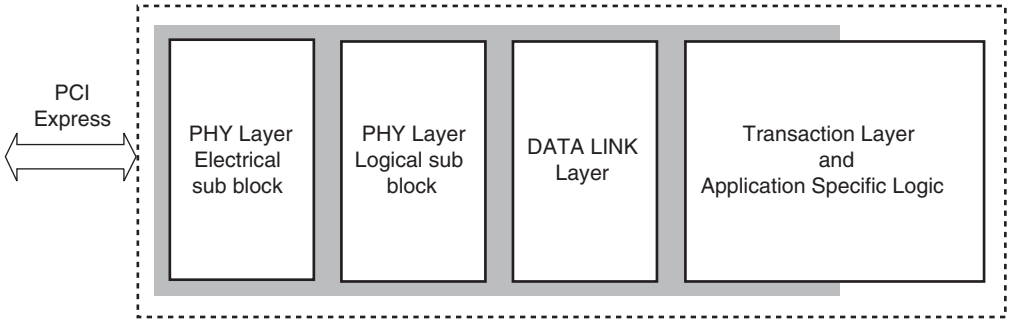
Functional Description

The Lattice PCI Express core is an implementation of the logical sub-block of Physical layer, Data Link Layer and part of the Transaction layer as specified in the PCI Express Base specification revision 1.0a.

- All configuration registers are left out of the core and all required status signals are made available at the User Interface side in order to facilitate an integrated easy implementation of configuration registers in the User Logic.
- The Lattice PCI Express core does not currently support L0s, L1, L2 and External Loop back states

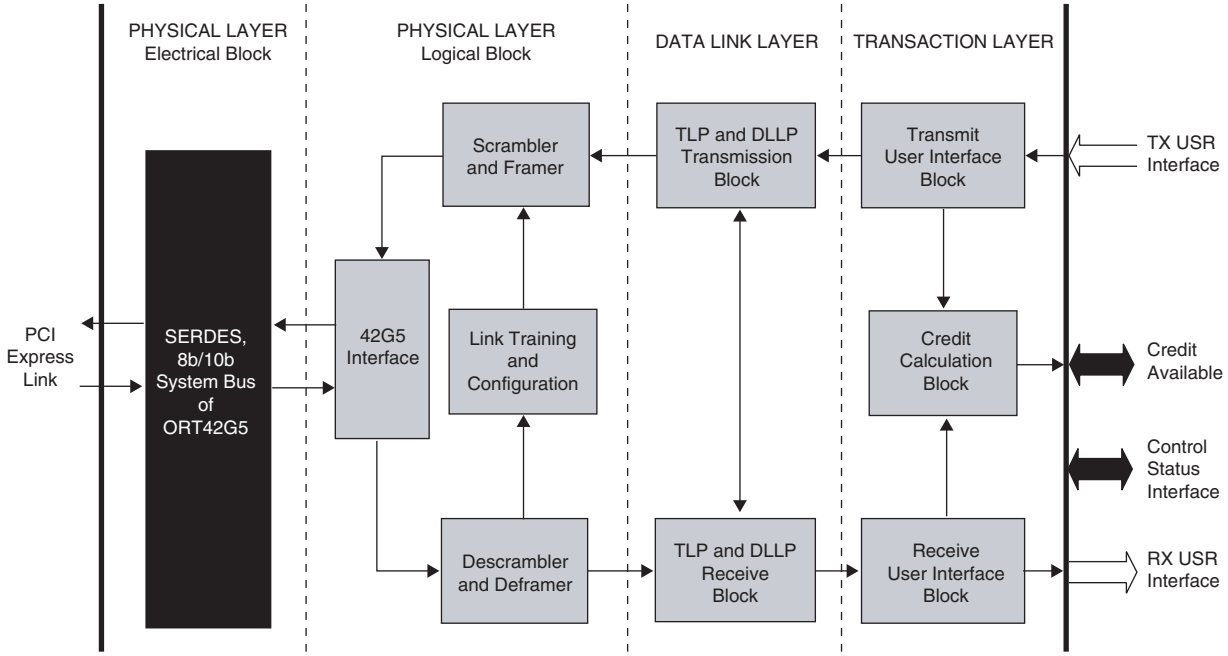
The following figure gives a graphical indication of the layers implemented (shaded blocks) in this PCI Express solution (Lattice IP on ORT42G5 FPSC device).

Figure 3. PCI Express Layers Implemented



A typical implementation of these layers is better illustrated in the following block diagram.

Figure 4. Block Diagram



Physical Layer Implementation

Electrical Sub Block

The Electrical sub block of the Physical Layer is implemented in the Embedded logic of the FPSC. It includes one channel of SERDES and also 8b/10b logic.

The SERDES circuitry consists of a receiver, transmitter, and auxiliary functional blocks. It supports serial data up to 3.7 Gbits/s. 8b/10b Decoder/Encoder logic follows 8b/10b transmission code as defined in the ANSI X3.230-1994 and IEEE 802.3z specifications.

Refer to the ORT42G5 and ORT82G5 data sheet for more details on SERDES and 8b/10b operation.

Logical Sub Block

The Logical sub block of the Physical Layer is implemented in the programmable logic portion of the FPSC. This includes the Scrambler/De-scrambler, Framer / De-Framer, LTSSM block and LWLSN block. These blocks are described below:

Scrambler/De-Scrambler

The Scrambler/De-scrambler function is implemented using Linear Feedback Shift Registers. This is performed by serially XORing the 8-bit character with the 16-bit output of the LFSR. The final output stage of the LFSR is XORed with the lower bit of the data character and then both the LFSR and data register are serially advanced.

The LFSR implements the polynomial:

$$G(S) = X^{16} + X^5 + X^4 = X^3 + 1$$

Framer/De-Framer

The Framer/De-Framer implements the mechanism which uses special symbols like K28.2 (SDP) to start a DLLP, K27.7 (STP) to start a TLP, and K29.7 (END) to mark the end of either a TLP or a DLLP. When no packet information or special ordered-sets are being transmitted, the transmitter will be sending idle data. The idle data consists of the data byte 0 (00h). During transmission of idle data the skip ordered set will continue to be transmitted.

TLPs are framed by placing a STP Symbol at the start of the TLP and an END Symbol or EDB Symbol at the end of the TLP. DLLPs are framed by placing an SDP Symbol at the start of the DLLP and an END Symbol at the end of the DLLP.

Link Training and Status State Machine (LTSSM)

The LTSSM is a set of interacting state machines implemented as per PCI Express base specification 1.0a section 4.2.5, which includes the following category of state functions:

Detect: To detect whether the far end receiver is powered on or not.

Polling: Transmission and reception of training ordered-sets to establish bit lock and symbol lock.

Configuration: Transmission and reception with negotiated data rate with configured link width.

Recovery: Transmission and reception of training ordered-sets to re-establish bit lock and symbol lock.

L0: Normal operation where data and control packets are transmitted and received.

L0s: Power saving state.

L1: Additional power saving state.

L2: State to conserve power aggressively.

External Loop Back: Loop back between receiver and transmitter for testing and validation purposes.

Disabled: Link is disabled.

Link Control Reset: Link is reset.

Note: The Lattice PCI Express IP core currently does not support L0s, L1, L2 and External Loopback states.

Link Width and Lane Sequence Negotiation (LWLSN)

The link width and lane sequence negotiation logic helps the LTSSM block to negotiate and finalize the link and lane number of the core as part of the training process. The core follows the steps as specified in the PCI Express spec 1.0a for an end point device.

Data Link Layer Implementation

The Data Link Layer tracks the state of the link. It communicates link status with the Transaction and Physical Layers, and performs Link Management through the Physical Layer.

Before starting normal operation following power-up or interconnect reset it is necessary to initialize flow control for the virtual channel. The TLP traffic starts only after this flow control initialization. This flow control initialization is implemented in the Data Link layer according to *PCI Express Specification 1.0a*.

Transmit Path

Transmit TLP Block

The Transmit TLP processing block implements the Data Link Layer requirements for TLPs; sequence number generation mechanism and LCRC calculation. It appends the information to the TLP that is getting transmitted. Also it implements the mechanism to resend the same TLP if a retry is requested from the receiver component. To support this mechanism it has to store the whole TLP that is being transmitted until this component receives an ACK for that particular TLP or until time-out indication occurs.

Transmit DLLP Block

The Transmit DLLP processing block, generates ACK or NAK DLLPS based on the instructions from the Receive TLP processing block. This block also generates InitFC DLLPs corresponding to FCI requirements. Appropriate power management DLLPs are generated as instructed by external power management logic. This block consists of a 16-bit CRC generation logic that adds the calculated CRC to the tail of all transmit DLLPs to meet data integrity requirements.

Receive Path

Receive TLP Block

Receive TLP processing block implements sequence number checking, data integrity checking, and through LCRC checking for received TLPs. It includes an appropriate retry mechanism to respond to any errors detected in received TLPs. This block is provided with a buffering mechanism to hold the whole TLP until its data integrity is checked and an appropriate instruction is generated to the Transmit DLLP block to send ACK and NAK DLLPs. Associated retry logic implements the time-out mechanism as specified in the PCI Express specification. After all these Data Link Layer requirements for a TLP are met the TLPs are sent out over Receive TLP interface to the transaction layer.

Receive DLLP Block

The main function implemented in this block is related to the processing of all incoming DLLPs as listed below:

- Data integrity checking of all DLLPs through CRC checking.
- Detection of different types of DLLPs through DLLP type decode.
- Extraction of different fields of DLLPs based on the type of the DLLP decoded.
- ACK and NAK processing.

In case of CRC error the DLLP is discarded and the error is transmitted to the user logic.

If the DLLP passes data integrity checking it is processed as described below:

- **ACK DLLP:** TLP Sequence number received in this DLLP indicates successful receipt of TLPs up to this Sequence number. This information is passed to the transmit TLP processing block to manage its transmit retry mechanism.
- **NAK DLLP:** This TLP sequence number negative acknowledgement initiates a Data Link Layer retry by the transmit TLP processing block using its transmit retry mechanism.
- **InitFC1, InitFC2 and UpdateFC DLLPs:** Used for Flow Control Initialization and Credit Calculation.
- **Power Management DLLP:** PM request type is extracted from this DLLP and is made available in the User Interface side.

Receive User Interface

All Receive transactions to the user interface are synchronized with the rising edge of the `div2_pclk`. Signal transactions between the core and the user interface are carried out in the form of received TLPs. Apart from the received data the core also provides status of the received TLPs and DLLPs. In case of error during reception of a TLP or DLLP a corresponding error signal is asserted to notify the user interface.

Rx TLPs are output by the core through a 32-bit data bus, `rxtlpu_data`. The following handshake signals are provided to ensure a smooth and controlled interface between the core and the user interface.

`rxtlpu_st` – The core loads the first DWORD of the TLP and asserts this output signal to indicate the start of Rx TLP. All remaining parts of the TLP are loaded out of the core in subsequent clocks. The core at the positive edge of the `div2_pclk` clock, sends out each DWORD of the TLP.

rxtlpu_sd – The core asserts this signal when the first DWORD of the "data" within the TLP is sent out by the core.

rxtlpu_end – The core indicates the last DWORD of the TLP at the end of the transaction by asserting this output signal.

un_sup_req – This error signal for Unsupported request is driven high from start to end of the received TLP.

pois_tlp – This error signal for Poisoned TLP is driven high from start to end of the received TLP.

malf_tlp – This error signal for Malformed TLP is driven high at any time from start of the TLP and will be asserted till the end of the TLP.

Figure 6 shows the signal status for Receive Path User Interface.

PM DLLP Transmission

Apart from generating requests for transmitting TLPs the user interface logic can also initiate a request to transmit Power Management (PM) DLLPs.

Input signals `txpm_dllp_type` and `tx_pm` are used for transmitting PM DLLPs. The user interface logic places a valid PM DLLP type for transmission in `txpm_dllp_type` and asserts `tx_pm` as a strobing signal.

Output signals `rxpm_dllp_type` and `rx_dllp_val` are provided to notify the user interface logic about the received PM DLLP.

Figure 7 shows the signal status for PM DLLP Transmission.

Transaction Layer Implementation

Transmit Path

The Transmit Path acts as a bridge between the Data Link Layer and User Interface in the transmit path. This module provides handshake between the core and user logic request to transmit TLPs and PM DLLPs.

In addition, this module performs "Credit Available" calculation by monitoring the types of TLPs transmitted and the credit values received from the other node through Update FC DLLPs. It accumulates these credit values separately for PH, NPH, PD and NPD types and makes them available to the User Interface side based on the selected packet type as indicated in the input signal `pkt_type`. Refer to the Credit Available Calculation section for more details.

Receive Path

This module consists of all logic to provide handshake between the core and user interface for received TLPs. It checks for errors in the received TLPs and notifies the user interface side in case of errors like Poisoned TLP, Unsupported TLP type and Malformed TLP

In addition, credit consumed in received TLP is calculated in this module and Calculated Credit is sent as update information to Data Link Layer to send Update FC DLLP.

Credit Available Calculation

The core maintains the value of Available Credit for each packet type. User logic can fetch this Credit Available value from the core by appropriately selecting the packet type in the input signal `pkt_type`. The core outputs the Available Credit in the output signal `crdt_avail` for the selected packet type. Please refer to the Transmit User Interface section under Timing Diagrams for more detail.

The core stores the Credit Limit (CL) for each packet type that is extracted from the latest UpdateFC DLLPs received from the other side of the link. The core also calculates the Credit Consumed (CC) value for each packet type whenever a TLP is transmitted from the core into the PCI Express Link. The Credit Available (CA) value for

each packet is derived by subtracting the Credit Consumed from the Credit Limit (CA = CL - CC). The Credit Available value is calculated and stored separately for packet types P, NP and CPL and is made available to the user logic for the selected packet type.

It is expected that the user logic take into account the Available Credit before sending a request to the core for transmitting a TLP.

The core will not verify the credit availability while transmitting a TLP.

Error Conditions
Receive logic of the core checks the received DLLP and TLPs for various error conditions. In case of error, associated signals are asserted to the user logic and the DLLP or TLP is processed as per the specification requirement. Details of the error conditions are described below:

Unsupported Request
The following types of TLPs are not supported by the core:

- Configuration Read Type 1
- Configuration Write Type 1
- I/O Read Request
- I/O Write Request
- Memory Read Request-Locked
- Completion for Locked Memory Read without Data - Used only in error case.
- Completion for Locked Memory Read - otherwise like CplD.

If the core receives any one of these types of TLPs, an error is transmitted by the user interface signal `un_sup_req`.

Poisoned TLP
There is no support to generate Data Poisoned TLP for transmission. But the core supports Data poisoning checks in the received TLP.

If the EP bit in a received Write Request (Posted or Non-Posted) or a Read Completion TLP (bit14 of header1) is set the core transmits the error in the user interface signal `pois_tlp`.

Malformed TLP
Malformed TLP detection is notified for the following cases:

- Size of TLP is more than `max_payload_size` (256 DWORD)
- Size of TLP exceeds length field
- Size of TLP doesn't match length field
- Undefined type of TLPs
- Default Traffic Class designator violation for Power Management message and Error Signaling messages

The core transmits the error in the user interface signal `malf_tlp`.

Bad TLP
This error signal is generated to indicate that the received TLP is erroneous when there is no `rx_err` signal from the Electrical Sub Block of the Physical Layer. The condition that leads to asserting this signal are listed below:

- CRC mismatch for Nullified TLP

- CRC error for Normal TLP
- If CRC is ok and received Sequence of the TLP violates the condition (NEXT_RCV_SEQ - Sequence Number) mod 4096 ≤ 2048

The core will discard this Bad TLP and reply with a NAK DLLP. Error signal `bad_tlp` will be asserted to notify the user logic.

Bad DLLP
This error signal is generated to indicate that the received DLLP is erroneous when there is no `rx_err` signal from the Electrical Sub Block of the Physical Layer. The condition that leads to asserting this signal is listed below:

- CRC error in the received DLLP

The core will discard this Bad DLLP and error signal `bad_dllp` will be asserted to notify the user logic.

Data Link Layer Protocol Error
The Data Link Layer Protocol Error is generated for the following conditions in a received DLLP:

- If ((NEXT_TRANSMIT_SEQ - 1) - AckNak_Seq_Num) mod 4096 > 2048, or
- If the above error condition is not noticed, but (ACKNak_Seq_Num - ACKD_SEQ) mod 4096 => 2048

Refer to Figures 3-17 in the *PCI Express Specification 1.0a* for more details.

The core will discard this DLLP and error signal `dll_perr` will be asserted to notify the user logic

Replay Number Rollover
The Replay Number Rollover status signal is a pulse of one clock duration and is generated when the internal replay number changes from 3 to 4.

The core asserts the user interface signal `rnum_rlor` to indicate this rollover.

Replay Timeout
The Replay time is the time between transmission of the last DWORD of a TLP and reception of the first DWORD of an ACK/NACK DLLP for that TLP.

The timer is restarted for any of the following conditions:

- When a DLLP is received and the retry buffer has some of the un-acknowledged TLPs
- At the beginning of a TLP transmission if the timer is not running
- At the expiration of the timer value.

The core asserts the user interface signal `rply_tout` to indicate this timeout.

More details on all these error signals are available in the Timing Diagram section.

User Logic Interface

Transmit User Interface

All transactions initiated from the User Interface side are synchronized with the rising edge of `div2_pclk`. Transactions between the core and the user interface logic are carried out in the form of TLPs.

Tx TLPs are loaded into the core through a 32-bit data bus, `txtlpu_data`. The following handshake signals are provided to ensure a smooth and controlled interface between the core and the user Interface logic:

`txtplu_req` – User interface logic asserts this input signal to request a transmission of TLP.

txtlpu_rdy – If the core is ready to service the request it asserts this output signal so that the user interface logic can start the transaction.

txtlpu_st – User ia nterface logic asserts this input signal to indicate the start of TLP and simultaneously loads the first DWORD of the TLP into the txtlpu_data bus. This signal should be active for only one clock period. All remaining parts of the TLP are loaded into the core in subsequent clocks, 32 bits per clock. Each DWORD of data is strobed in by the core at every positive edge of the div2_pclk clock.

txtlpu_end – User interface logic asserts this signal along with the last DWORD of the TLP in the txtlpu_data bus. This is to indicate the end of Tx transaction. This signal should be active for only one clock period.

txtlpu_nlfy – If the current Tx TLP is to be a nullified TLP the user interface logic asserts this to indicate the end of the nullified TLP. This signal should be active for only one clock period.

The user interface logic can hold the txtlpu_req asserted to continue to transmit TLPs as long as txtlpu_rdy is asserted the txtlpu_st, txtlpu_end and txtlpu_nlfy signals are required to be asserted at the appropriate time for each TLP as explained above.

The core expects that the request from the user logic is always to transmit a full TLP. So the core will respond with txtlpu_rdy only if it has enough memory in the retry buffer to cater a TLP even with maximum payload.

The data transfer from the user logic and the core should be continuous without any break between the start of a TLP (as indicated by txtlpu_st) and the end of a TLP (as indicated by txtlpu_end) hence txtlpu_req should be held asserted till the end of a TLP. If the txtplu_req is deasserted by the user logic before the end of a TLP the core behavior will be unknown and the user logic should start a TLP afresh.

Based on the packet type selected by the user logic in the input signal pkt_type the core loads the corresponding Credit Available information in the output signal crdt_avail. In the above figure p_c, NP_c and cpl_c are the corresponding Credit Available values for the packet types p, NP and cpl. For more information about Credit Available Calculation please refer to the appropriate section under Functional Description.

Figure 5 shows the signal status for Transmit User Interface.

Register Description

PCI Express configuration space is not implemented in this core. But the core checks all TLPs related to the configuration access and appropriate action, as specified, is carried out in case of errors. It is left to the user logic to implement the necessary PCI Express registers as part of the design. All status and error signals are provided at the user interface to connect them to proper registers in the user logic.

User Configurable Parameters

The PCI Express core has all the three layers of the Endpoint device that is implemented in one configuration. Therefore there are no other configurations of this core.

Timing Diagrams

Figure 5. Transmit Path User Interface

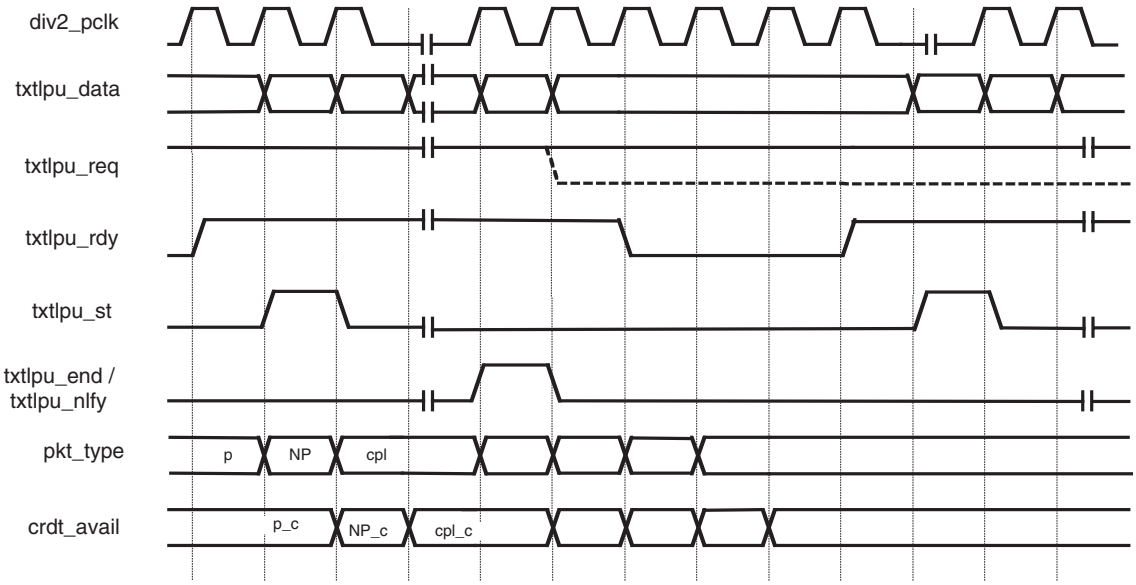


Figure 6. Receive Path User Interface

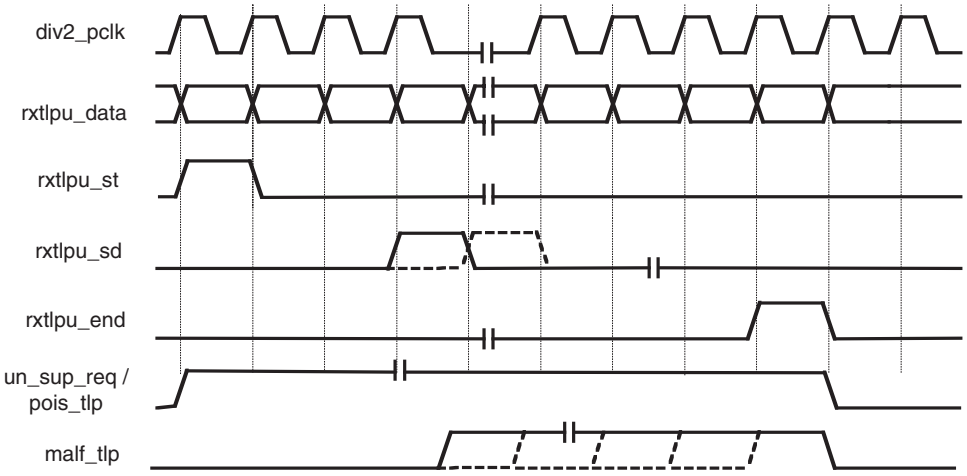
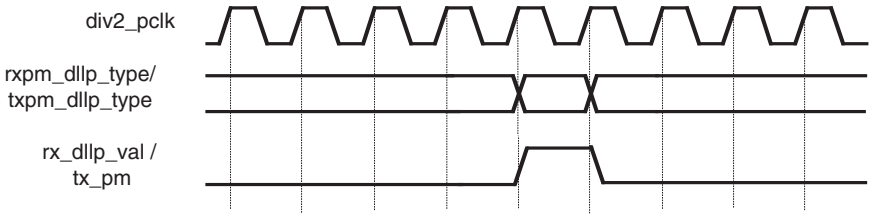


Figure 7. PM DLLP Transmit User Interface



User Interface Signals AC Characteristics

This section gives the I/O timings on all the signals on the user interface assuming these signals are coming out of the FPGA I/O pins.

Table 1. User Interface Signals for AC Characteristics

Signal Name	Type	Typical (ns)
All Input Signals	t_{SU}	7.0
All Output Signals	t_{CO}	7.0

Signal Descriptions

Table 2. PCI Express Core Signal Descriptions

Port Name	Active State	I/O	Signal Description
rst_n	Low	I	Asynchronous System Reset
div2_pclk	—	I	62.5MHz System Clock
PCI Express Line Interface (Inputs)			
REFCLKN_A	—	I	125MHz Differential Reference Clock for SERDES Block A
REFCLKP_A	—	I	125MHz Differential Reference Clock for SERDES Block A
REFCLKN_B	—	I	125MHz Differential Reference Clock for SERDES Block B
REFCLKP_B	—	I	125MHz Differential Reference Clock for SERDES Block B
USR_CLK	—	I	50 MHz clock for system bus
HDINN_AC	—	I	Rx Data input – SERDES Quad A, Channel C
HDINP_AC	—	I	Rx Data input – SERDES Quad A, Channel C
HDINN_BC	—	I	Rx Data input – SERDES Quad B, Channel C
HDINP_BC	—	I	Rx Data input – SERDES Quad B, Channel C
PASB_PDN	—	I	Active Low Power Down Input
PASB_RESETN	—	I	Active Low Reset for the Embedded Core
PASB_TESTCLK	—	I	Clock Input for BIST and Loopback Test
PASB_TRISTN	—	I	Active low 3-state for Embedded Core Output Buffers
PBIST_TEST_ENN	—	I	Selection of PASB_TESTCLK Input for BIST Test
PLOOP_TEST_ENN	—	I	Selection of PASB_TESTCLK Input for Loopback Test
PMP_TESTCLK	—	I	Clock Input for Microprocessor in Test Mode
PMP_TESTCLK_ENN	—	I	Selection of PMP_TESTCLK in Test Mode
PSYS_DOBISTN	—	I	Input to Start BIST Test
PCI Express Line Interface (Outputs)			
HDOUTN_AC	—	O	Tx Data input - SERDES Quad A, Channel C
HDOUTP_AC	—	O	Tx Data input - SERDES Quad A, Channel C
HDOUTN_BC	—	O	Tx Data input - SERDES Quad B, Channel C
HDOUTP_BC	—	O	Tx Data input - SERDES Quad B, Channel C
REXT_A	—	O	Reference Resistor – SERDES Quad A
REXT_B	—	O	Reference Resistor – SERDES Quad B
PSYS_RSSIG_ALL	—	O	Output result of BIST Test
External Circuit Interface			
start_rx_detect	—	O	Poll rx_detect. Equivalent to TxDetectRx/Loopback in PIPE Spec
rx_detect	High	I	Indicates PCI_Express Link presence is Detected. Equivalent to RxStatus=011b in PIPE Spec

Port Name	Active State	I/O	Signal Description
RX_USR Interface			
rxtlpu_data[31:0]	—	O	Received TLP Data to Transaction Layer
rxtlpu_st	High	O	Start of Received TLP on the "rxtlpu_data"
rxtlpu_sd	High	O	Start of Data in the Received TLP
rxtlpu_end	High	O	End of TLP on the "rxtlpu_data"
rxpm_dllp_type	—	O	Type of Received PM DLLP
rxpm_dllp_val	High	O	Valid Signal to Sample "rxpm_dllp_type"
pois_tlp	High	O	Poisoned TLP received indication
un_sup_req	High	O	Unsupported Request Received Indication
malf_tlp	High	O	Malformed TLP Received Indication
bad_tlp	High	O	Bad TLP Received Indication
bad_dllp	High	O	Bad DLLP Received Indication
dll_perr	High	O	Data Link Layer Protocol Error Indication
rnum_rlor	High	O	REPLY_NUM Rollover Indication
rply_tout	High	O	Replay Timer Timeout Indication
TX_USR Interface			
txtlpu_req	High	I	User Interface Request to Send TLPs
txtlpu_data[31:0]	—	I	TLP Data to be Transmitted on PCI Express Link
txtlpu_st	High	I	Start of TLP Signal that Indicates Start of New TLP on the txtlpu _data Bus
txtlpu_end	High	I	End of TLP Signal that Indicates End of TLP on the txtlpu _data Bus
txtlpu_nlfy	High	I	User Interface Request to Generate a Nullified TLP This Signal is to be Asserted Along With End of TLP
no_pcie_training	High	I	No LTSSM Training to be performed
pkt_type	—	I	Packet type for credit info
txpm_dllp_type	—	I	PM packet type to transmit
tx_pm	High	I	Send PM DLLP, with PM data in "tx_data"
l_retrain_link	High	I	Initiate Link Retraining
l_ext_sync	High	I	Extended syn
loc_lnk_cntl[3:0]	High	I	Link Control bits from User Interface **Bit 0** – Set Hot_Reset bit in TS1/TS2 Will switch the Core to Reset state if current state is Recovery **Bit 1** – Set Disable_Link bit in TS1/TS2 Will switch the Core to Disable state if current state is Recovery **Bit 2** – Set Loopback bit in TS1/TS2 Will switch the Core to Tx_Loopback state if current state is Recovery **Bit 3** – Set Disable_scramble bit in TS1/TS2
l_go_config	High	I	Direct the Core to Switch to Configuration State
txtlp_rdy	High	O	The Data Link Layer is Ready to Accept TLPs from Transaction Layer
crdt_avail	—	O	Credit Available for the Packet Type

Note2:

1. For all PCI Express Line Interface signals (both input and output) refer to the ORT42G5 and ORT82G5 data sheet for their usage.
2. External Circuit Interface: rx_detect signal to be generated by an external circuit whenever the start_rx_detect signal is asserted. Refer to the Receiver Detection section in "PHY Interface for the PCI Express Architecture".
3. For usage of loc_lnk_cntl and l_go_config signals refer to the Link Training and Status State Machine section of the PCI Express specifications.

PCI Express Core Design Flow

Lattice has created a detailed software IP tutorial available on the Lattice web site at www.latticesemi.com. Both a simple IP module evaluation and tutorial and a more detailed ispLeverCORE™ tutorial are available for download. Type "tutorial" in the Lattice web site search engine.

References

- PCI Express Specification 1.0A
- ispLEVER® Software Online Help Manual

Technical Support Assistance

Hotline: 1-800-LATTICE (North America)
+1-503-268-8001 (Outside North America)
e-mail: techsupport@latticesemi.com
Internet: www.latticesemi.com

Appendix for ORCA® Series 4 FPSCs – ORT42G5

Table 3. Performance and Resource Utilization[1]

Name of Parameter File	ORCA 4 PFUs	LUTs	Registers	EBR	PIO	f_{MAX} (MHz)
pci_exp_t42g5_1_001.lpc	937	3,406	3,078	10	116	66.7

1. Performance and utilization characteristics are generated using an ORT42G5-2BM484. When using this IP core in a different density, package, speed, or grade within the ORCA 4 family, performance and utilization may vary.

Supplied Netlist Configurations

The Ordering Part Number (OPN) for all configurations of this core is PCI-EXP-T42G5-N1. Table 3 lists the netlists available as Evaluation Packages for the ORCA Series 4 devices, which can be downloaded from the Lattice web site at www.latticesemi.com.

You can use the IPexpress software tool to help generate new configurations of this IP core. IPexpress is the Lattice IP configuration utility, and is included as a standard feature of the ispLEVER design tools. Details regarding the usage of IPexpress can be found in the IPexpress and ispLEVER help system. For more information on the ispLEVER design tools, visit the Lattice web site at www.latticesemi.com/software.

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for Development Software ***category:***

Click to view products by Lattice ***manufacturer:***

Other Similar products are found below :

SRP004001-01 SW163052 SYSWINEV21 WS01NCTF1E W128E13 SW89CN0-ZCC IP-UART-16550 MPROG-PRO535E AFLCF-08-LX-CE060-R21 WS02-CFSC1-EV3-UP SYSMAC-STUDIO-EIPCPLR LIB-PL-PC-N-1YR-DISKID 1120270005 MIKROBASIC PRO FOR FT90X (USB DONGLE) MIKROC PRO FOR FT90X (USB DONGLE) MIKROBASIC PRO FOR AVR (USB DONGLE LICEN MIKROBASIC PRO FOR FT90X MIKROC PRO FOR DSPIC30/33 (USB DONGLE LI MIKROPASCAL PRO FOR ARM (USB DONGLE LICE MIKROPASCAL PRO FOR FT90X MIKROPASCAL PRO FOR FT90X (USB DONGLE) MIKROPASCAL PRO FOR PIC32 (USB DONGLE LI SW006021-2H ATATMELSTUDIO 2400573 2702579 2988609 SW006022-DGL 2400303 88970111 DG-ACC-NET-CD 55195101-101 55195101-102 SW1A-W1C MDK-ARM SW006021-2NH B10443 SW006021-1H SW006021-2 SW006022-2 SW006023-2 SW007023 MIKROE-730 MIKROE-2401 MIKROE-499 MIKROE-722 MIKROE-724 MIKROE-726 MIKROE-728 MIKROE-732