Chipsmall Limited consists of a professional team with an average of over 10 year of expertise in the distribution of electronic components. Based in Hongkong, we have already established firm and mutual-benefit business relationships with customers from,Europe,America and south Asia,supplying obsolete and hard-to-find components to meet their specific needs.

With the principle of “Quality Parts,Customers Priority,Honest Operation,and Considerate Service”,our business mainly focus on the distribution of electronic components. Line cards we deal with include Microchip,ALPS,ROHM,Xilinx,Pulse,ON,Everlight and Freescale. Main products comprise IC,Modules,Potentiometer,IC Socket,Relay,Connector.Our parts cover such applications as commercial,industrial, and automotives areas.

We are looking forward to setting up business relationship with you and hope to provide you with the best service and solution. Let us make a better world for our industry!

Contact us

Tel: +86-755-8981 8866 Fax: +86-755-8427 6832
Email & Skype: info@chipsmall.com Web: www.chipsmall.com
Address: A1208, Overseas Decoration Building, #122 Zhenhua RD., Futian, Shenzhen, China

PCI IP User's Guide

November 2010
IPUG18_09.2

Table of Contents

Introduction

Lattice's Peripheral Component Interconnect (PCI) Intellectual Property (IP) cores provide an ideal solution that meets the needs of today's high performance PCI applications. The PCI IP cores provide a customizable, 32-bit or 64-bit PCI Master and Target or Target only solution that is fully compliant with the *PCI Local Bus Specification, Revision 3.0* for speeds up to 66MHz. The PCI cores bridge the gap between the PCI Bus and specific design applications, providing an integrated PCI solution. These cores allow designers to focus on the application rather than on the PCI specification, resulting in a faster time-to-market.

PCI is a widely accepted bus standard that is used in many applications including telecommunications, embedded systems, high performance peripheral cards, and networking. The family of PCI IP core is one of the many in Lattice's portfolio of IP cores. For more information on these and other products, refer to the Lattice web site at: http://www.latticesemi.com/products/intellectualproperty/.

This document covers Target only, Master and Target, 64-bit, and 32-bit PCI IP cores implemented in a number of devices. Details of Master and 64-bit operation only apply to the appropriate cores. Pin assignments for specific variations of this core are described at the end of this document.

Quick Facts

Table 1-1 through Table 1-8 give quick facts about the PCI IP core for LatticeEC™, LatticeECP™, LatticeECP2™, LatticeECP2M™, LatticeECP3™, LatticeXP™, LatticeXP2™, LatticeSC™, MachXO™, MachXO2™, and LatticeSCM™ devices.

Table 1-1. PCI IP Core Quick Facts--PCI master/target 66MHz/64bit

		PCI IP configuration					
		PCI master/target 66MHz 64bit					
Core Requirements	FPGA Families Supported	LatticeEC LatticeECP	Lattice ECP2 Lattice ECP2M	LatticeXP	LatticeXP2	LatticeECP3	LatticeSC LatticeSCM
	Minimal Device Needed	LFEC10E-5F484C	LFE2-12E-6F484C	LFXP15C-5F388C	LFXP2-17E-6F484C	LFE3-35EA-7FN484CES	LFSC3GA15E-6F900C
Resource Utilization	Data Path Width	64					
	LUTs	2500					
	Registers	900					
Design Tool Support	Lattice Implementation	Lattice Diamond™ 1.0 or ispLEVER® 8.1					
	Synthesis	Synopsys® Synplify™ Pro for Lattice D-2009.12L-1					
		Mentor Graphics® Precision™ RTL					
	Simulation	Aldec® Active-HDL™ 8.2 Lattice Edition					
		Mentor Graphics ModelSim™ SE 6.3F					

Table 1-2. PCI IP Core Quick Facts--PCI master/target 66MHz/32bit

		PCI IP configuration					
		PCI master/target 66MHz 32bit					
Core Requirements	FPGA Families Supported	LatticeEC LatticeECP	Lattice ECP2 Lattice ECP2M	LatticeXP	LatticeXP2	LatticeXP3	LatticeSC LatticeSCM
	Minimal Device Needed	LFEC6E-5F256C	LFE2-6E-6F256C	LFXP6C-5F256C	LFXP2-5E-6FT256C	LFE3-17EA-7FN484CES	LFSC3GA15E-6F900C
Resource Utilization	Data Path Width	32					
	LUTs	1600					
	Registers	700					
Design Tool Support	Lattice Implementation	Diamond 1.0 or ispLEVER 8.1					
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1					
		Mentor Graphics Precision RTL					
	Simulation	Aldec Active-HDL 8.2 Lattice Edition					
		Mentor Graphics ModelSim SE 6.3F					

Table 1-3. PCI IP Core Quick Facts--PCI master/target 33MHz/64bit

		PCI IP configuration					
		PCI master/target 33MHz 64bit					
Core Requirements	FPGA Families Supported	LatticeEC LatticeECP	Lattice ECP2 Lattice ECP2M	LatticeXP	LatticeXP2	LatticeXP3	LatticeSC LatticeSCM
	Minimal Device Needed	LFEC10E-5F484C	LFE2-12E-6F484C	LFXP15C-5F388C	LFXP2-17E-6F484C	LFE3-35EA-7FN484CES	LFSC3GA15 E-6F900C
Resource Utilization	Data Path Width	64					
	LUTs	1400					
	Registers	800					
Design Tool Support	Lattice Implementation	Diamond 1.0 or ispLEVER 8.1					
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1					
		Mentor Graphics Precision RTL					
	Simulation	Aldec Active-HDL 8.2 Lattice Edition					
		Mentor Graphics ModelSim SE 6.3F					

Table 1-4. PCI IP Core Quick Facts--PCI master/target 33MHz/32bit

		PCI IP configuration							
		PCI master/target 33MHz 32bit							
Core Requirements	FPGA Families Supported	MachXO	MachXO2	LatticeEC LatticeECP	Lattice ECP2 Lattice ECP2M	LatticeXP	LatticeXP2	LatticeXP3	LatticeSC LatticeSCM
	Minimal Device Needed	LCMXO1200E-5FT256C	LCMXO-1200HC-6TG144CES	LFEC6E-5F256C	LFE2-6E-6F256C	LFXP6C-5F256C	LFXP2-5E-6FT256C	LFE3-17EA-7FN484CES	LFSC3GA15E-6F900C
Resource Utilization	Data Path Width	32							
	LUTs	900							
	Registers	600							
Design Tool Support	Lattice Implementation	Diamond 1.0 or ispLEVER 8.1							
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1							
		Mentor Graphics Precision RTL							
	Simulation	Aldec Active-HDL 8.2 Lattice Edition							
		Mentor Graphics ModelSim SE 6.3F							

Table 1-5. PCI IP Core Quick Facts--PCI target 66MHz/64bit

		PCI IP configuration					
		PCI target 66MHz 64bit					
Core Requirements	FPGA Families Supported	LatticeEC LatticeECP	Lattice ECP2 Lattice ECP2M	LatticeXP	LatticeXP2	LatticeXP3	LatticeSC LatticeSCM
	Minimal Device Needed	LFEC6E-5F484C	LFE2-12E-6F484C	LFXP10C-5F388C	LFXP2-8E-6FT256C	LFE3-17EA-7FN484CES	LFSC3GA15E-6F900C
Resource Utilization	Data Path Width	64					
	LUTs	1300					
	Registers	600					
Design Tool Support	Lattice Implementation	Diamond 1.0 or ispLEVER 8.1					
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1					
		Mentor Graphics Precision RTL					
	Simulation	Aldec Active-HDL 8.2 Lattice Edition					
		Mentor Graphics ModelSim SE 6.3F					

Table 1-6. PCI IP Core Quick Facts--PCI target 66MHz/32bit

		PCI IP configuration					
		PCI target 66MHz 32bit					
Core Requirements	FPGA Families Supported	LatticeEC LatticeECP	Lattice ECP2 Lattice ECP2M	LatticeXP	LatticeXP2	LatticeXP3	LatticeSC LatticeSCM
	Minimal Device Needed	LFEC3E-5Q208C	LFE2-6E-6F256C	LFXP3C-5Q208C	LFXP2-5E-6QN208C	LFE3-17EA-7FTN256CES	LFSC3GA15E-6F256C
Resource Utilization	Data Path Width	32					
	LUTs	900					
	Registers	500					
Design Tool Support	Lattice Implementation	Diamond 1.0 or ispLEVER 8.1					
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1					
		Mentor Graphics Precision RTL					
	Simulation	Aldec Active-HDL 8.2 Lattice Edition					
		Mentor Graphics ModelSim SE 6.3F					

Table 1-7. PCI IP Core Quick Facts--PCI target 33MHz/64bit

		PCI IP configuration					
		PCI target 33MHz 64bit					
Core Requirements	FPGA Families Supported	LatticeEC LatticeECP	Lattice ECP2 Lattice ECP2M	LatticeXP	LatticeXP2	LatticeXP3	LatticeSC LatticeSCM
	Minimal Device Needed	LFEC6E-5F484C	LFE2-12E-6F484C	LFXP10C-5F388C	LFXP2-8E-6FT256C	LFE3-17EA-7FN484CES	LFSC3GA15E-6F900C
Resource Utilization	Data Path Width	64					
	LUTs	800					
	Registers	600					
Design Tool Support	Lattice Implementation	Diamond 1.0 or ispLEVER 8.1					
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1					
		Mentor Graphics Precision RTL					
	Simulation	Aldec Active-HDL 8.2 Lattice Edition					
		Mentor Graphics ModelSim SE 6.3F					

Table 1-8. PCI IP Core Quick Facts--PCI target 33MHz/32bit

		PCI IP configuration							
		PCI target 33MHz 32bit							
Core Requirements	FPGA Families Supported	MachXO	MachXO2	LatticeEC LatticeECP	Lattice ECP2 Lattice ECP2M	LatticeXP	LatticeXP2	LatticeXP3	LatticeSC LatticeSCM
	Minimal Device Needed	LCMXO1200E-5FT256C	LCMXO-1200HC-6TG144CES	LFEC3E-5Q208C	LFE2-6E-6F256C	LFXP3C-5Q208C	LFXP2-5E-6QN208C	LFE3-17EA-7FTN256CES	LFSC3GA15E-6F256C
Resource Utilization	Data Path Width	32							
	LUTs	600							
	Registers	500							
Design Tool Support	Lattice Implementation	Diamond 1.0 or ispLEVER 8.1							
	Synthesis	Synopsys Synplify Pro for Lattice D-2009.12L-1							
		Mentor Graphics Precision RTL							
	Simulation	Aldec Active-HDL 8.2 Lattice Edition							
		Mentor Graphics ModelSim SE 6.3F							

Features

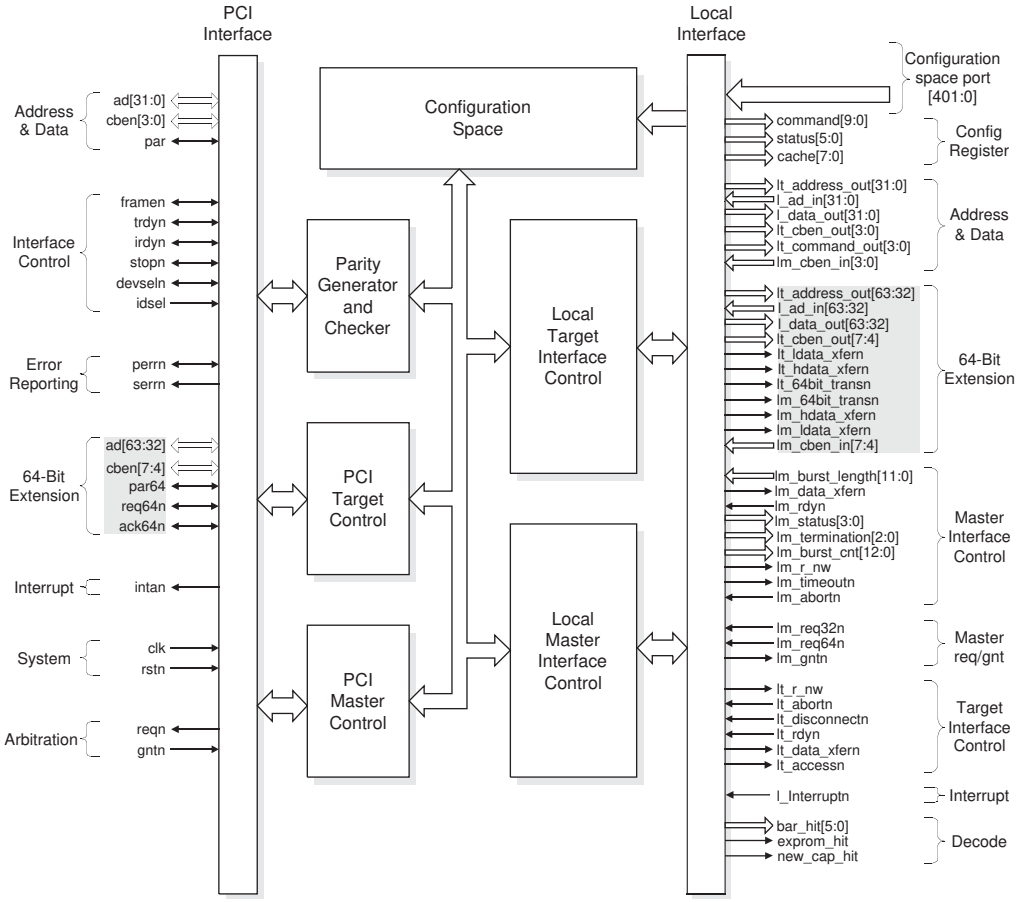
- Available as 32/64-bit PCI bus and 32/64-bit local bus
- PCI SIG Local Bus Specification, Revision 3.0 compliant
- 64-bit addressing support (dual address cycle)
- Capabilities list pointer support
- Parity error detection
- Up to six Base Address Registers (BARs)
- Fast back-to-back transaction support
- Supports zero wait state transactions
- Special cycle transaction support
- Customizable configuration space
- Up to 66MHz PCI
- Fully synchronous design

This chapter provides a functional description of the Lattice PCI IP core.

The PCI IP cores bridge the PCI bus to the back-end application. They decode transactions and pass PCI requests to the Local Interface. The back-end applications then send or receive the proper data associated with the PCI Interface via their Local Interface to respond to the PCI transactions. In the case of master versions the core executes PCI bus transactions based on back-end requests. Figure 2-1 illustrates the functional modules and internal bus structure used in the PCI IP core.

Block Diagram

Figure 2-1. PCI IP core Block Diagram



Note: Signals in shaded boxes are used for 64-bit PCI Cores.

The PCI Master Target IP Core consists of multiple blocks, as shown in Figure 2-1. This section provides a detailed description of these blocks.

PCI Master Control

The PCI Master Control interfaces with the PCI bus. It supports all of the address and command signals required to execute transactions on the PCI bus for both 32-bit and 64-bit PCI applications. A list of the supported PCI signals is available in the PCI Interface Signals section of this document. Once the Local Master Interface Control is granted the bus, it passes the transaction information to the PCI Master Control using the internal bus. The PCI Master Control then requests and executes the transaction on the PCI bus. The PCI IP cores support all of the

commands specified in the *PCI Local Bus Specification, Revision 3.0.* Table 2-1 lists the supported PCI commands.

Table 2-1. PCI IP Core Command Support

cben[3:0]	Command	Support
0000	Interrupt Acknowledge	Yes
0001	Special Cycle	Yes
0010	I/O Read	Yes
0011	I/O Write	Yes
0100	Reserved	Ignored
0101	Reserved	Ignored
0110	Memory Read	Yes
0111	Memory Write	Yes
1000	Reserved	Ignored
1001	Reserved	Ignored
1010	Configuration Read	Yes
1011	Configuration Write	Yes
1100	Memory Read Multiple	Yes
1101	Dual Address Cycle	Yes
1110	Memory Read Line	Yes
1111	Memory Write and Invalidate	Yes

The PCI Master control supports data transfer requirements for both high and low throughput back-end applications. It maintains up to the maximum 528 MBytes per second (MBps) burst data transfer rate when operating at 66MHz with a 64-bit data bus. The Advanced Master Transactions section of this document describes burst data transfers in further detail. For slower applications, single data phase transactions can also be easily implemented. The Basic PCI Master Read and Write Transactions section describes these basic transactions in detail.

PCI Target Control

The PCI Target control interfaces with the PCI bus. It processes the address, data, command and control signals to transfer data to and from the PCI IP core for both 32-bit and 64-bit PCI applications. A list of the supported PCI signals is available in the PCI Interface Signals section. Once the PCI Target control detects a transaction, it passes the transaction information to the Local Interface control using the internal bus. It also responds to most Configuration Space accesses with no intervention from the Local Interface. The PCI IP core supports all of the commands specified in the *PCI Local Bus Specification, Revision 3.0*. Table 2-1 lists the supported PCI commands.

When designing for a particular target application, the back-end target design may not support all the commands listed in Table 2-1. As a result, the PCI IP core does not transfer data using those commands. For cases where the back-end target application does not support all the commands, it must issue the proper termination as described in the Target Termination section of this document.

The PCI Target control supports the data transfer requirements for both high and low throughput back-end applications. It can maintain a 528 MBps transfer rate during burst transactions when operating at 66MHz with a 64-bit data bus. The Advanced Target Transactions section describes the Burst transactions in further detail. For slower applications, single data phase transactions can also be easily implemented. The Basic PCI Target Read and Write Transactions section describes the these basic transactions in detail

Local Master Interface Control

The Local Master Interface facilitates master transactions on the PCI Bus with the commands listed in Table 2-1. The Local Master Interface Control passes the local master transaction request from the user's application to the PCI Master Control which then executes the PCI bus transaction.

Local Target Control

The Local Target Control responds to target transactions on the PCI bus. Fully decoded BAR select signals (bar_hit) and new capabilities select signal (new_cap_hit) are provided by the Local Target Control to indicate that the PCI IP core has been selected for a transaction. Registered address and command signals are available at the Local Interface from the Local Interface Control for the back-end application to properly handle the core's request. Additionally, the Local interface also supplies Configuration Space Register signals and a local interrupt request (l_interruptn) for users' applications. A full list of Local Interface signals and descriptions is available in the Local Interface Signals section.

Configuration Space

The Configuration Space implements all the necessary Configuration Space registers required to support a single-function PCI IP core. It provides the first 64 bytes of header type 0, which is used for all device types other than PCI-to-PCI and CardBus bridges. The first 64 bytes of the predefined header region contain fields that uniquely identify the device and allow the device to be generically controlled. This predefined header portion of the Configuration Space is divided into two parts. The first 16 bytes of the header are defined the same way regardless of the type of device. The remaining bytes have different definitions depending on the functionality that the PCI IP core supports. These bytes include six Base Address Registers (BARs), the Capabilities Pointer (Cap Ptr), and the registers that control the interrupt capability. Refer to the Configuration Space Set-up section for additional information on the Configuration Space.

Accesses to the first 64 bytes of the Configuration Space are completed by the PCI IP core control with no intervention from the Local Target Interface control. Access beyond the first 64 bytes, such as the Capabilities List, is left to the Local Target Interface control. These transactions are described in the Advanced Configuration Accesses section.

Parity Generator and Checker

Parity checking must occur on every PCI address and data cycle to be compliant with the *PCI Local Bus Specification, Revision 3.0.* The PCI IP core's Parity Generator and Checker module does all parity checking for the PCI device. The Parity Generator and Checker determines if the master is successful in addressing the desired target. It also verifies that data transfers occur correctly between the master and target devices. The address and byte enable signals are included in every calculation to ensure accuracy. Each address and data cycle that occurs on the PCI bus is checked for errors.

The parity check signals perrn and serrn are enabled or disabled using bit 6 and bit 8 of the PCI Command Register, which is part of the Configuration Space.

Signal Descriptions

Pin Assignments for the evaluation configurations are shown "Pin Assignments For Lattice FPGAs" on page 161. Final selection of the pinouts is left to the designer to allow for maximum flexibility in the design. Pinouts are defined in the HDL source code, or as follows:.

- In Diamond, choose **View > Show Views > File List**, double-click the.lpf file, and edit the file to add pin location preferences.
- In ispLEVER, double-click **Edit Preference (ASCII)** in the Processes window, and edit the file in the Text Editor to add pin location preferences.

Refer to the Diamond or ispLEVER software help for additional information.

There are five types of signals defined in Table 2-2.

Table 2-2. Signal Types

Signal Type	Description
in	**Input** is a standard input only signal.
out	**Output** is a standard output only signal.
t/s	**Tri-state** is a bidirectional, tri-state input/output pin.
s/t/s	**Sustained Tri-State** is an active low tri-state signal owned and driven by one agent at a time.
o/d	**Open Drain** allows multiple devices to share as a wire-OR. A pull-up is required to sustain the inactive state until another agent drives it and must be provided by the central resource.

PCI Interface Signals

The PCI Interface signals correspond to the PCI bus specification. Table 2-3 shows the input and output signals for the PCI IP core. These are the signals required by the PCI IP core to handle PCI bus side transactions. Table 2-3 describes each signal.

In addition to the signals required by the PCI IP core, there are some signals on the PCI Bus, referred to as "Additional Signals" in the PCI specifications, which must be handled appropriately to insure proper PCI IP core functions in a system. Refer to the relevant PCI specifications for a description of those Additional Signals (which are beyond the scope of this document). Examples of this type of signal are `M66EN` and `PRSNT[1:0]`.

Table 2-3. PCI IP Core Signals[1]

Name	I/O	Polarity	Description
PCI System			
`clk`	in	—	The PCI system clock provides timing for all transactions. The clock frequency operates up to 66MHz. This clock is also used to provide timing to the Local Interface.
`rstn`	in	low	The asynchronous PCI system reset is used to set the PCI device to a starting known and stable state.
PCI Address and Data			
`ad[31:0]`	t/s	—	The multiplexed address and data bus.
`cben[3:0]`	t/s	—	Multiplexed command and byte enable signals.
`par`	t/s	—	The `par` signal generates even parity for `ad[31:0]` and `cben[3:0]` signals
PCI Interface Control			
`framen`	s/t/s	low	The `framen` signal is driven by the current master and used to indicate the start of cycle and the duration of the cycle.
`irdyn`	s/t/s	low	The initiator ready signal indicates that the current master is ready for the data phase.
`trdyn`	s/t/s	low	The target ready signal indicates that the current target is ready for the data phase.
`stopn`	s/t/s	low	The PCI IP core, as a target, drives this signal low requesting to stop the current transaction.
`idsel`	in	—	The initialization device select is used to select a target for configuration reads and writes.
`devseln`	s/t/s	low	Device select is actively driven by the PCI IP core to indicate that it is the target of the bus transaction.

Table 2-3. PCI IP Core Signals[1] (Continued)

Name	I/O	Polarity	Description
PCI Error Reporting			
perrn	s/t/s	low	Data parity error is used to report parity errors in the data phase.
serrn	o/d	low	System error is used to indicate catastrophic errors.
PCI Interrupt			
intan	o/d	low	Interrupt A is used to request an interrupt.
PCI Bus Arbitration			
reqn	out	low	Request for the use of PCI bus.
gntn	in	low	Grant the master's access to PCI bus.
PCI 64-Bit Extension			
ad[63:32]	t/s	—	The upper 32 bits of multiplexed address and data bus.
cben[7:4]	t/s	—	The upper, multiplexed command and byte enable signals for 64-bit applications.
par64	t/s	—	The par64 signal generates even parity for ad[63:32] and cben[7:4] signals.
req64n	s/t/s	low	Used by the master to request a 64-bit data transaction.
ack64n	s/t/s	low	Signal used to indicate the acknowledgement of a request for 64-bit data transaction.

1. Shaded rows apply to 64-bit applications.

Local Interface Signals

The Local Interface provides all the necessary address and control signals to respond to and initiate transactions associated with the PCI bus. Command and status information are also available at the Local Interface, so the back-end application logic can essentially monitor the PCI bus. Table 2-4 contains the Local Interface signals that are divided into three different categories: Local Bus Signals, Local Target Bus signals and Local Master Bus signals.

The Local Bus Signals are shared between the Local Master Interface and Local Target Interface. These signals are typically denoted with an "l_". The Local Target Bus signals are used by the Local Target Interface and are denoted using "lt_". The Local Master Bus signals are used by the Local Master interface and are denoted using "lm_".

Table 2-4. Local Interface Signals[1]

Name	I/O	Polarity	Description
Local Address and Data			
l_ad_in[31:0]	in	—	Local address/data input. The address input is used in Master Read/Write transactions, and the data input is used for master write/target read transactions
l_data_out[31:0]	out	—	Local Data output. Local side lower DWORD data output for a master read or a target write.
lt_address_out [31:0]	out	—	The local address bus for target read and write. This bus indicates the start address of the transaction. The bus, lt_address_out [31:0], is latched one clock after the framen signal is asserted on each transaction and remains unchanged until the next transaction.
lt_cben_out[3:0]	out	low	The local byte enables for target read and write. The lt_cben_out[3:0] determine which byte lanes of l_data_out[31:0] or l_ad_in[31:0] carry meaningful data.
lt_command_out[3:0]	out	—	The lt_command_out[3:0] latches the command information during the address phase of a PCI cycle. It indicates the PCI bus command for the current cycle (refer to Table 2-1).
lm_cben_in[3:0]	in	low	Local master command and byte enables.

Table 2-4. Local Interface Signals[1] (Continued)

Name	I/O	Polarity	Description
Local 64-Bit Extension			
l_ad_in[63:32]	in	—	Local address/data input. The address input is used in Master Read/Write transactions, and the data input is used for master write/target read transactions.
l_data_out[63:32]	out	—	Local Data output. Local side upper DWORD data output for a master read or a target write.
lt_address_out [63:32]	out	—	The local address bus for target read and write. This bus is valid only for 64bit address bar. The 64-bit combined signal lt_address_out [63:0] indicates the start address of the transaction. The high 32bit of the bus, lt_address_out[63:32], is latched two clock cycles after the framen signal is asserted on each transaction (only for dual address cycle) and remains unchanged until the next transaction.
lt_cben_out[7:4]	out	low	The local byte enables for 64-bit target read and write. The lt_cben_out[7:4] determine which byte lanes of l_data_out[63:32] or l_ad_in[63:32] carry meaningful data.
lt_ldata_xfern	out	low	This signal works same as lt_data_xfern. It applies to lower DWORD when local bus is 64bit.
lt_hdata_xfern	out	low	This signal works same as lt_data_xfern. It applies to upper DWORD when local bus is 64bit.
lt_64bit_transn	out	low	Signal to the local target that a 64-bit read or write transaction is underway on pci bus.
lm_ldata_xfern	out	low	This signal works same as lm_data_xfern. It applies to lower DWORD when local bus is 64bit.
lm_hdata_xfern	out	low	This signal works same as lm_data_xfern. It applies to upper DWORD when local bus is 64bit.
lm_64bit_transn	out	low	Signal to the local master that a 64-bit read or write transaction is underway on PCI bus.
lm_cben_in[7:4]	in	low	Local master byte enables.
Local Interrupt			
l_interruptn	in	low	The local side interrupt request indicates that the Local Interface is requesting an interrupt. This signal asserts the PCI side interrupt signal, intan, if interrupts are enabled in the Configuration Space.
Config Register			
cache[7:0]	out	—	The cache signal indicates the cache length in the cache registers defined in the Configuration Space
command[9:0]	out	—	Command register bits from the Configuration Space. Bit 0 - I/O space enable, Command[0] Bit 1 - Memory space enable, Command[1] Bit 2 - Master enable, Command[2] Bit 3 - Special cycles enable, Command[3] Bit 4 - Memory write and invalidate enable, Command[4] Bit 5 - VGA Palette Snoop, Command[5] Bit 6 - Parity Error Response, Command[6] Bit 7 - Reserved Bit 8 - SERR# enable, Command[8] Bit 9 - Fast back-to-back enable, Command[9]
status[5:0]	out	—	Status register bits from the Configuration Space. Bit 0 - Master Data Parity Error, Status[8] Bit 1 - Signaled Target Abort, Status[11] Bit 2 - Received Target Abort, Status[12] Bit 3 - Received Master Abort, Status[13] Bit 4 - Signaled System Error with SERR#, Status[14] Bit 5 - Detected Parity Error, Status[15]

Table 2-4. Local Interface Signals[1] (Continued)

Name	I/O	Polarity	Description
Local Target Interface			
lt_abortn	in	low	Local target abort request is used to request a target abort on the PCI bus.
lt_disconnectn	in	low	Local target disconnect (or retry) is used to request early termination of a bus transaction on the PCI bus.
lt_rdyn	in	low	Local target ready signal indicates that the Local Interface is ready to receive or send data.
lt_r_nw	out	—	Read/Write (read/not write) to signal whether the current transaction is a read or write. 1 = read, 0 = write
lt_accessn	out	low	Local target can access local interface if lt_accessn is active. Once lt_accessn active, local target needs to be ready for next process based on lt_command_out. lt_accessn is active during either of active bar_hit, exprom_hit or new_cap_hit. It is also active during special cycle command.
lt_data_xfern	out	low	This signal indicates local input data (l_ad_in) being read or local output data (l_data_out) being available at current clock cycle. When lt_data_xfern is active, if core reads data from l_ad_in, back-end can update l_ad_in for next data at next clock cycle. If core writes data on l_data_out, back-end can get valid data from l_data_out. It is only used when the local bus is 32 bits.
Local Target Address Decode			
bar_hit[5:0]	out	high	The bar_hit signal indicates that the master is requesting a transaction that falls within one of the Base Address register ranges.
new_cap_hit	out	high	New Capabilities List hit. new_cap_hit indicates that the master is requesting a Configuration Space register out of internal registers (00h-3fh), that is 40h-FFh., Although the hardware associated with the New Capabilities reside in the back-end logic, logically they are part of the PCI Configuration Space.
Local Master req/gnt			
lm_req32n	in	low	Local master 32-bit data transaction request.
lm_req64n	in	low	Local master 64-bit data transaction request.
lm_gntn	out	low	Signal to the local master that gntn is asserted.
Local Master Interface Control			
lm_rdyn	in	low	Local master is ready to receive data (read) or send data (write)
lm_burst_length [11:0]	in	—	Local master burst length determines the number of data phases in the transaction. For single data phase, it should be set to 1. lm_burst_length set to 0 means the burst length is 13'b1,0000,0000,0000.
lm_data_xfern[3]	out	low	This signal indicates local input data (l_ad_in) being read or local output data (l_data_out) available at current clock cycle. When lt_data_xfern is active, if core reads data from l_ad_in, back-end can update l_ad_in for next data at next clock cycle. If core writes data on l_data_out, back-end can get valid data from l_data_out. It is only used when the local bus is 32 bits. In a single data phase, it should be set to 1. lm_burst_length set by 0 means the length is 13'b1,0000,0000,0000.
lm_r_nw	out	—	(Read/Write) to signal whether the current transaction is a read or write. 1 = read, 0 = write
lm_timeoutn	out	—	Indicates that the transaction has timed out.
lm_abortn	in	—	Local master issues an abort to terminate a cycle that can not be completed.

Table 2-4. Local Interface Signals[1] (Continued)

Name	I/O	Polarity	Description
lm_status[3:0]	out	—	Indicate the master operation status. 0001 - Address loading 0010 - Bus transaction 0100 - Bus termination 1000 - Fast_back_to_back
lm_termination[2:0]	out	—	Indicate the master termination status. 000 - Normal termination Normal termination occurs when the master finishes and completes the transaction normally. During a multi-data phase transfer, a condition can occur where the master's latency timer expires on the last data phase and master's gntn has been de-asserted. In this case, lm_timeoutn is asserted and the master also indicates this as Normal termination. 001 - Timeout termination Timeout termination occurs when, during a multi-data phase transfer, the master's latency timer expired before the last data phase and the master's gntn is de-asserted on or before the last data phase. lm_timeoutn is also asserted in this case. 010 - No target response termination. This is also known as Master Abort termination. 011 - Target abort termination 100 - Retry termination 101 - Disconnect data termination 110 - Grant abort termination 111 - Local master termination
lm_burst_cnt[12:0]	out	—	Local master burst transaction "down counter" value. When the local master requests a 32-bit transaction, the initial value of lm_burst_cnt is equal to lm_burst_length. When the local master requests a 64-bit transaction and lm_64bit_transn is active, the initial value of lm_burst_cnt is equal to lm_burst_length. When the local master requests a 64-bit transaction and lm_64bit_transn is inactive, the initial value of lm_burst_cnt is double of lm_burst_length.

1. Shaded rows apply to 64-bit applications.
2. A Memory Write and Invalidate transaction is not governed by the Latency Timer except at cacheline boundaries. A master that initiates a transaction with the Memory Write and Invalidate command ignores the Latency Timer until a cacheline boundary. When the Latency Timer has expired (and gntn is deasserted), the core asserts lm_timeoutn. The backend must terminate the transaction at next cacheline boundary by asserting lm_abort.
3. During Master Read operation the signal lm_data_xfern always reflects valid data in the local data bus. But during Master Write operation, due to data prefetch ahead of the transactions on PCI bus, lm_data_xfern along with the lm_status reflects the data validity. If lm_status is 0100, (meaning a Bus Termination) ignore the lm_data_xfern assertion because the data being prefetched is not sent out on the PCI bus due to termination.

PCI Configuration Space Setup

Determining the correct settings for the Configuration Space is an essential step in designing a PCI application, because the device may not function properly if the Configuration Space is not properly configured. The PCI IP core supports all of the required and some additional Configuration Space registers that apply to the PCI IP core (refer to *PCI Local Bus Specifications, Revision 3.0*, Chapter 6). Figure 2-2 shows the supported Configuration Space for the PCI IP core. This section describes the first 64 bytes of the Configuration Space in the PCI IP core and its customization method. For more information on the parameters used to customize the Configuration Space, refer to the Lattice PCI IP core Configuration Options section.

Figure 2-2. PCI IP Core Configuration Space

Device ID		Vendor ID		00h
Status Register		Command Register		04h
Class Code			Revision ID	08h
BIST	Header Type	Latency Timer	Cache Line Size	0Ch
Base Address 0				10h
Base Address 1				14h
Base Address 2				18h
Base Address 3				1Ch
Base Address 4				20h
Base Address 5				24h
Cardbus CIS Pointer				28h
Subsystem ID		Subsystem Vendor ID		2Ch
Expansion ROM Base Address				30h
Reserved			Cap Ptr	34h
Reserved				38h
MAX_LAT	MIN_GNT	Interrupt Pin	Interrupt Line	3Ch

Note: Shaded sections indicate reserved and unused sections in the configuration space. All unused and reserved registers return 0s.

Vendor ID: The Vendor ID is a 16-bit, read-only field used to identify the manufacturer of the product. The Vendor ID is set using the `VENDOR_ID` parameter. The Vendor ID is assigned by the PCI SIG to ensure uniqueness. Contact PCI SIG (www.pcisig.org) to obtain a unique Vendor ID.

Device ID: The Device ID is a 16-bit, read-only field that is defined by the manufacturer used to uniquely identify a particular product or model. The Device ID is set using the `DEVICE_ID` parameter. Its default value is 0000h.

Revision ID: The Revision ID is an 8-bit, read-only device-specific field that is set using the `REVISION_ID` parameter. This field is used by the manufacturer and should be viewed as an extension of the Device ID to distinguish between different functional versions of a PCI product.

Class Code: The Class Code is a 24-bit, read-only register and is used to identify the generic functionality of a device. The value of this register is determined by the `CLASS_CODE` parameter. The Class Code is broken up into three bytes. The upper byte holds the base class code; the middle byte holds the sub-class code. In addition, the lower byte holds the programming interface. The Class Code information is located in the *PCI Local Bus Specification, Revision 3.0*. The default setting for this register is FF0000h.

Command Register: The Command Register is a 16-bit read/write register that provides coarse control over the device. It is located at the lower 16 bits of address 04h in the Configuration Space. Using this register, the memory and I/O space can be disabled to allow only configuration accesses. This register also controls the parity error response and the `serrn` signal. Figure 2-3 and Table 2-5 illustrate the command register that is implemented in the PCI IP core.

Figure 2-3. Command Register

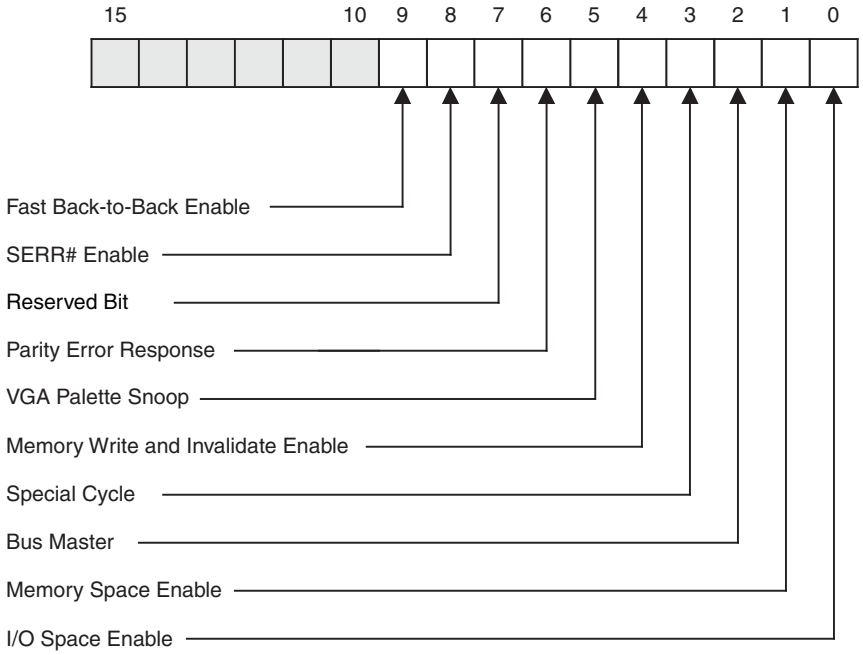


Table 2-5. Command Register Description

Bit Location	Description
0	**I/O Space Enable** controls a device's response to I/O space accesses. I/O space accesses are enabled if the bit is set to a 1. After reset the I/O space enable bit is set to a 0.
1	**Memory Space Enable** controls a device's response to memory space accesses. Memory space accesses are enabled if the bit is set to a 1. After reset the memory space enable bit is set to a 0.
2	**Bus Master** enables the PCI IP core to act as a master on the PCI bus when this bit is set to 1. After reset the Bus Master enable bit is set to a 0.
3	**Special Cycle** controls a device's action on special cycle operations. Special cycle accesses are enabled if the bit is set to 1. After reset the bit is set to 0.
4	**Memory Write and Invalidate Enable** controls the PCI IP core's ability to execute the Memory Write and Invalidate cycle on the PCI bus. The Core, when required, will issue the Memory Write and Invalidate command if this bit is set to a 1. After reset this bit is set to a 0.
5	**VGA Palette Snoop**
6	**Parity Error Response** is used to control a device's response to parity errors. If the bit is 0, a parity error causes the Detected Parity Error status bit to be set in the status register but does not drive the perrn signal. After reset the bit is set to 0. This is the enable for parity error checking. However, even with the `perrn` signal disabled, the device is still required to generate parity.
7	**Reserved Bit**
8	**SERR Enable** is used to enable the `serrn` driver. To enable, this bit is set to a 1. After reset this bit is set to 0.
9	**Fast Back-to-Back Enable** allows the PCI IP core to execute fast back-to-back transactions to different devices. If the fast back-to-back enable is set to a 1, the Core executes fast back-to-back transactions. After reset this bit is set to 0.
10-15	**Reserved Bits** The returned value for these bits is 0 when this register is read.

Status Register

The Status Register is a 16-bit read/write register that provides information on the capabilities of the PCI IP core. It also reports the error status of the PCI IP core. The Status Register is located at the upper 16 bits of register location 04h. Writes to the Status Register from the PCI bus are slightly different, given that bits can be reset but not set. Writing a 1 to a bit in the status register resets it, but only if the current value of the bit is a 1. Writing a 0 to a bit has no effect. Figure 2-4 and Table 2-6 describe the Status Register that is implemented in the PCI IP core.

Figure 2-4. Status Register

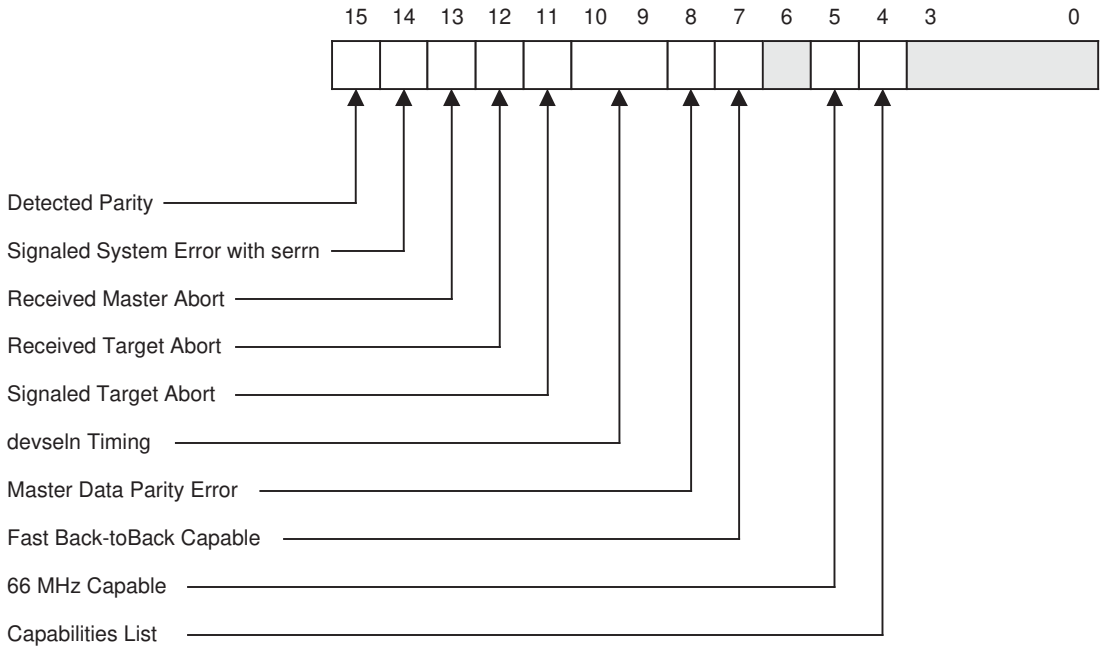


Table 2-6. Status Register Descriptions

Bit Location	Description
4	**Capabilities List** is a read-only bit that indicates whether or not the device contains an address pointer to the start of the Capabilities list. The bit is set to a 1 to indicate that the Capabilities Pointer at location 34h is valid. After reset the value is set to a 0. The CAP_PTR_ENA parameter initializes this bit.
5	**66MHz Capable** is a read-only bit that is used to indicate that the device is capable of running at 66MHz. The bit is set to a 1 if the device is 66MHz capable. The PCI_66MHZ_CAP parameter initializes this bit.
7	**Fast Back-to-Back Capable** is a read-only bit that indicates if the device is capable of handling fast back-to-back transactions. The bit is set to a 1 if the device can accept these transactions. The FAST_B2B_CAP parameter initializes this bit.
8	**Master Data Parity Error** indicates that the bus master has detected a parity error during a transaction. A value of 1 means a parity error has occurred. After rest the bit is set to 0.
9-10	**DEVSEL Timing** bits indicate the slowest time for a device to assert the devseln signal for all accesses except the configuration accesses. The PCI IP core only supports the slow decode setting. The DEVSEL_TIMING parameter (bits 2 and 1) determines the DEVSEL timing. 00 - Fast (not supported) 01- Medium (not supported) 10 - Slow 11 - Reserved
11	**Signaled Target Abort** is set when the target terminates the cycle with a Target-Abort. Writing a 1 clears the Signaled Target Abort.
12	**Received Target Abort** is set to a 1 by the Core after it terminates a cycle with a target abort.

Table 2-6. Status Register Descriptions (Continued)

Bit Location	Description
13	**Received Master Abort** is set to a 1 by the Core after it terminates a cycle with a master abort with the exception of special cycles.
14	**Signaled System Error** with `serrn` is set when the device asserts serrn. Writing a one clears the bit.
15	**Detected Parity Error** is used to indicate a parity error even if the parity error handling is disabled.
0,1,2,3,6	**Reserved Bits** The returned value for each of these bits is 0 when this register is read.

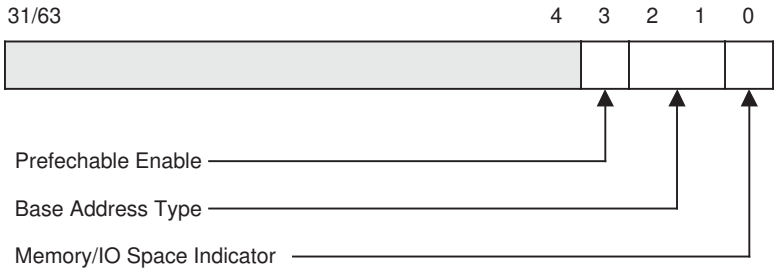
Base Address Registers

The PCI IP core supports up to six Base Address Registers (BARs) for Master/Target and Target configurations. The BAR holds the base address for the PCI IP core, and it is used to point to the starting address of the PCI IP core in the system memory map. They are configured differently based on whether they are mapped in memory or I/O space. A memory location is addressed using 32 bits or 64 bits while I/O locations are limited to 32-bit addresses. The six BARs consist of 192 bits in the Configuration Space and are located in address locations 0x10 to 0x27.

BAR Mapped to Memory Space

When selecting the amount of required memory for a BAR, the amount of memory is saved to the `BAR0-BAR5` parameters in its 2's complement form. Bits 0 through 3 of a memory BAR describes the attributes of the BAR and do not change. The minimum recommended amount of memory a BAR should request is 4Kbytes. Figure 2-5 and Table 2-7 describe the configuration of a BAR for memory space.

Figure 2-5. Memory Base Address Register



SERR Enable is used to enable the `serrn` driver. To enable, this bit is set to a 1. After reset this bit is set to 0.

Table 2-7. Memory Base Address Register

Bit Location	Description
0	**Memory/I/O Space Indicator** indicates whether the base address is mapped to I/O or memory space. A 0 indicates mapping to the memory space. The value of this bit is set by bit 0 of the `BAR0-BAR5` parameters.
1-2	**Base Address Type** is used to determine whether the BAR is mapped into a 32-bit or 64-bit address space. These bits have the following meaning: 00 - located in 32-bit address space 01 - reserved 10 - located in 64-bit address space 11 - reserved
3	**Prefetchable Enable** is determined by bit 3. It is a read-only bit that indicates if the memory space is prefetchable. A value of 1 means the memory space is prefetchable. Bit 3 of the `BAR0-BAR5` parameters sets the value of this bit.
4-31/63	**Bits 4-31/63** are read/write to hold memory address and are initialized by the `BAR0-BAR5` parameters.

Bar Mapped to I/O Space

When selecting the amount of required I/O space for a BAR, the amount is saved to the BAR0-BAR5 parameters in its 2's complement form. Bits 0 and 1 of an I/O BAR describe the attributes of the BAR and do not change. Figure 2-6 and Table 2-8 describe the configuration of a BAR for I/O space.

Figure 2-6. I/O Base Address Register

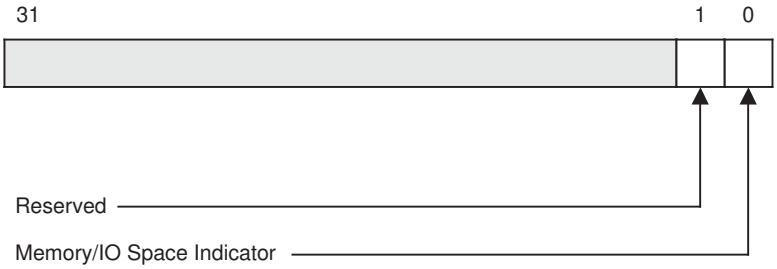


Table 2-8. I/O Base Address Register

Bit Location	Description
0	**Memory/I/O space Indicator** indicates whether the base address is mapped to I/O or memory space. A 0 indicates mapping to the memory space. The value of this bit is set by bit 0 of the BAR0-BAR5 parameters.
1	**Bit 1** is reserved and hardwired to 0. This bit is read only.
2-31	**Bits 2-31** are read/write to hold the memory address and are initialized by the BAR0-BAR5 parameters.

Cache Line Size

The Cache Line Size register is an 8-bit read/write register, located at 0Ch. It specifies the Cache Line Size in Double Words (DWORDs). During a reset the register is set to 00h. This register is output to local interface as cache[7:0].

Latency Timer

The Latency Timer register is an eight-bit read/write or read only register, located at byte address 0Dh. It specifies the Master Latency Timer value for a PCI Master on the PCI bus. During reset the register is set to 00h.

CardBus CIS Pointer

The CardBus CIS Pointer is a read-only, 32-bit register at location 28h in the Configuration Space. The CIS_POINTER parameter determines the value of the register. For more information on the CardBus CIS Pointer, refer to the CardBus specification.

Subsystem Vendor ID

The Subsystem Vendor ID is a 16-bit, read-only field and is used to further identify the manufacturer of the expansion board or subsystem. The SUBSYSTEM_VENDOR_ID parameter determines the value of the Subsystem Vendor ID register. The PCI SIG assigns the Vendor ID to ensure uniqueness. Contact PCI SIG (www.pcisig.org) to attain a unique Subsystem Vendor ID.

Subsystem ID

The Subsystem ID is a 16-bit, read-only field and is used to further identify the particular device. This field is defined by the manufacturer and is used to uniquely identify products or models. The SUBSYSTEM_ID parameter determines the value of this register.

Capabilities Pointer

The Capabilities Pointer indicates the starting location of the Capabilities List. It resides at address location 34h. The Capabilities Pointer consists of an 8-bit read-only register location. The capabilities pointer must be enabled by the CAP_PTR_ENA parameter. The CAP_POINTER parameter determines the value of this register.

Min_Gnt

The Min_Gnt read-only register is an 8-bit field that is used to specify the length of time in microseconds for the Master to control the PCI bus. It resides in the upper 8 bits of address location 3Ch. The MIN_GRANT parameter determines the value of this register.

Max_Lat

The Max_Lat read-only register is an 8-bit field that is used to specify the how often the PCI IP core the bus. It resides in the third byte of address location 3Ch. The MAX_LATENCY parameter determines the value of this register.

Interrupt Line

The Interrupt Line register is set by the interrupt handling mechanism to define the interrupt routing. This is a read/write register is handled outside the operation of the PCI IP core. This register holds system interrupt routing information.

Interrupt Pin

The Interrupt Pin register is used to indicate which of the four interrupts that the PCI IP core uses. Because the PCI IP core is a single function device, the only Interrupt Pin that can be selected is Interrupt A. If the interrupt is selected, the INTERRUPT_PIN parameter sets the register with a value of 01h. This eight-bit register is located at address location 3Dh.

Reserved

All reserved registers are read-only. Write operations to reserved registers are completed normally, and the data is discarded. A 0 is returned after the read operations to reserved registers are completed normally.

Lattice PCI IP core Configuration Options

Lattice PCI IP core allows an extensive definition of the PCI Configuration Space for optimum performance.

IPexpress User-Controlled Configurations

The IPexpress user-configurable flow provides evaluation capability for any valid combination of parameters. Configurations can have a maximum of three BARs. To create a configuration with more than three BARs, contact Lattice.

The evaluation configurations of PCI IP core have a maximum of three BARs. To order a configuration with more than three BARs, contact Lattice.

Table 2-9. IPexpress Parameters for PCI IP Core

Parameter Name	Range	Default(s)
Number of BARs	1-6	3
Bus Definition Parameters		
PCI Data Bus Size	32- or 64-bit	Note 1
Local Master Data Bus Size	32- or 64-bit	Note 2
Local Target Data Bus Size	32- or 64-bit	Note 2

Table 2-9. IPexpress Parameters for PCI IP Core

Parameter Name	Range	Default(s)
Local Address Bus Width	32- or 64-bit	32-bit

1. The value for PCI Data Bus size is set in each eval configuration as described in the appendices of the PCI IP core data sheet.
2. For 32-bit PCI Data Bus, only 32-bit Local Data Bus sizes are supported. For 64-bit PCI Data Bus, only 64-bit Local Data Bus sizes are supported.

PCI Configuration Using Core Configuration Space Port

A set of signals called the Configuration Space Port is provided at the local bus side of the core to allow the user to define the PCI configuration space as required for the user's system. The names of these Core configuration input signals are all suffixed with _p.

Appropriate parameter values are to be assigned to the designated input signals of Core configuration space port to implement the desired PCI configuration space. Here are two examples to achieve this:

1. Directly assign parameter values to the input signals of Core configuration space port. The user needs to provide hard coded values to the Core's Configuration Space Port input signals in the core instantiation.

```
module pci_top();
  …
  customer_design   cus_design _inst(
              .xxxx(xxxx),
              .yyyy(yyyy),
              …….
              .zzzz(zzzz)
              ;


  pci_core     core_inst( .framen(framen),
              …
              .vdr_id_p(16'h1234),    //vendor_id = 16'h1234
              …
              .dev_tim_p(2'b10),       //devsel_timing = 2'b10  (slow)
              …
              );
endmodule
```

2. Typically, two Verilog files, para_cfg.v and PCI_params.v, can be used to load these parameters to Core's Configuration Space Port. These files are available in Lattice PCI IP release package.
   - Edit the PCI_params.v to set correct values to the parameters. Parameter names in PCI_params.v are all suffixed with _g. Alternatively use the PCI GUI provided with Lattice's software design tools to generate the PCI_params.v. Refer the note given below.
   - Instantiate para_cfg module and appropriately connect its ports to the Core configuration input signals of PCI IP core.

   para_cfg module will load the parameters, defined in PCI_params.v, into the Core's Configuration Space Port input signals.

```
module pci_top();
  …
  wire   [15:0]   vdr_id;
  wire   [ 1:0]    dev_tim;
  …
  …
```