XILINX®

Xilinx PCI

The CORE™ of a GREAT IDEA

Data Book

Xilinx PCI

Data Book

LogiCORE/AllianceCORE Fax:	+1 408-377-3259
Tech Support E-mail:	hotline@xilinx.com
Tech Support Phone Number:	+1 800-255-7778
Information & Feedback E-mail:	logicore@xilinx.com (all LogiCORE)
Xilinx Home Page URL:	www.xilinx.com
PCI Solutions URL:	www.xilinx.com/products/logicore/pci/pci_sol.htm

XILINX®

2100 Logic Drive
San Jose, CA 95124
United States of America
Telephone: +1 408-559-7778
Fax: +1 408-559-7114

Dear PCI customer,

On behalf of the PCI Team at Xilinx, and our CORE partners, welcome to our May 1998 PCI Data Book, and thank you for your interest in Xilinx PCI Solutions.

As the inventor and leading provider of Field Programmable Gate Array Technology, we want to pledge our continuing commitment to support your great ideas when it comes to logic design and PCI applications.

Our mission is to provide you with a high-quality PCI solution that provides better flexibility, higher performance, and lower cost than any other available solution. Xilinx PCI allows you to integrate a PCI interface with your unique logic, into one flexible programmable device. Since the first PCI product introduction in February 1996, we have developed a complete solution for PCI including super-fast FPGAs, low-cost HardWire FpgASICs, easy-to-use predictable LogiCORE modules with guaranteed timing, as well as PCI boards, drivers and design examples. We think you will find Xilinx PCI Solution interesting and we hope that you consider us for future designs.

Together we can bring the great ideas to life!

Sincerely

Per Holmberg

LogiCORE Product Manager
CORE Solutions Group

XILINX®

Section Titles

Introduction

PCI Products

PCI32 4000 Master & Slave Interfaces Version 2.0

PCI32 Spartan Master & Slave Interface

Synthesizable PCI Bridge Design Examples

Synchronous FIFO

HotPCI Spartan Prototyping Board

Driver::Works Windows Device Driver Development Kit Version 2.0

VtoolsD Windows Device Driver Development Kit Version 3.0

FPGA Products

XC4000XLT Family Field Programmable Gate Arrays

Spartan and Spartan-XL Families Field Programmable Gate Arrays

Design Methodology

PCI Compliance Checklists

XC4000XLT PCI Compliance Checklist

Spartan PCI Compliance Checklist

LogiCORE PCI32 Cores PCI Compliance Checklist

Pinout and Configuration

Resources

Waveforms

Ordering Information and License Agreement

Sales Offices, Sales Representatives, and Distributors

XILINX®

Introduction

Introduction

Xilinx PCI team - Mission

We provide the most cost-effective and highest-performance PCI solution in the market by leveraging the flexibility of Xilinx Field Programmable Gate Arrays (FPGAs). We make PCI easy to design by providing a complete solution of proven cores, intuitive development tools, and comprehensive support.

Why Xilinx PCI?

By integrating a fully-compliant PCI interface with an application-specific back-end design, into one FPGA, you can achieve higher integration and higher performance than other PCI solutions.

Because Xilinx FPGAs integrate the PCI interface plus 15,000 to 124,000 user gates, no external PLD is required for glue logic. The result is a highly integrated, one-chip PCI solution at a lower cost than most standard PCI chip sets. See Figure 1 for a comparison of the Xilinx PCI and standard PCI chips.

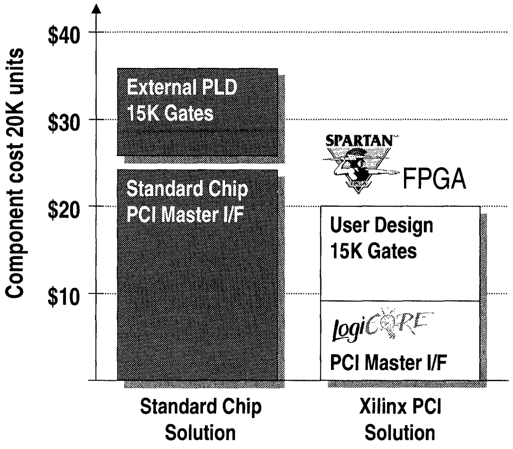


Figure 1: The cost of a Xilinx Spartan FPGA integrating a fully compliant PCI interface and 15,000 user gates, compared to a standard chip solution requiring an external PLD for glue logic.

The Xilinx PCI solution can be customized for a specific application and, as a result, the highest possible performance is achieved. Xilinx high-speed FPGAs support zero wait-state burst operations and by integrating scalable, dual-port FIFOs on the chip, a sustained bandwidth of up to 132 MBytes/sec can be achieve, system constraints being the only limitation.

The flexibility of Xilinx FPGAs makes it possible to update the PCI board, through software alone, in development or in the field. This significantly reduces your design risk and cuts development time.

Xilinx and its partners can also provide reference design examples, prototyping boards, PCI drivers, driver development tools, and design services.

All LogiCORE PCI products incorporates Xilinx Smart-IP technology, which leverages the Xilinx architectural advantages, such as look-up tables (LUTs), distributed RAM, and segmented routing, and floorplanning information, such as logic mapping, placement and routing constraints. The result is a highly predictable core, which is fixed to meet the critical PCI timing. This combined with a modular architecture, illustrated in Figure 2, allows the designer to customize the back-end design including FIFOs and DMA functions without jeopardizing the critical timing on the PCI side.

The predictable LogiCORE PCI interface is customized and downloaded using an intuitive graphical user interface available on Xilinx web-site, see the Design Methodologies chapter. The core is delivered with VHDL and Verilog simulation models, testbenches, and design examples.

Xilinx PCI32 Design Kit

To minimize the learning curve for PCI and to enable rapid development and prototyping, Xilinx provides a complete PCI Design Kit including cores, a prototyping board and driver development tools. By forming a partnership with Virtual Computer Corporation, leading provider of rapid prototyping boards and reconfigurable computing systems, and Vireo Software, leading provider of device driver tools, our customers will always have access to the leading industry expertise.

Xilinx PCI32 Design Kit contains:

- LogiCORE PCI32 interfaces (Xilinx)
- PCI System Architecture book (MindShare, Inc.)
- Bridge Design Examples (Xilinx)
- HotPCI Prototyping Board (VCC)
- Demo software (VCC)
- Canned reference drivers (Vireo)
- Driver development tools (Vireo)

Xilinx PCI32 Interfaces

- **PCI32 4000 – the high-density PCI solution.** This solution integrates a PCI interface plus up to 124,000 system gates. The core supports zero wait-state burst operations and a sustained bandwidth of up to 132 Mbytes per second. See the PCI32 4000 data sheet for more details.
- **PCI32 Spartan – the low-cost PCI solution.** This solution integrates a PCI interface plus up to 30,000 system gates at a price below standard chip solutions. The core supports zero wait-state burst operations and a sustained bandwidth of up to 132 Mbytes per second. See the PCI32 Spartan data sheet for more details.

See the Ordering Information chapter for more details.

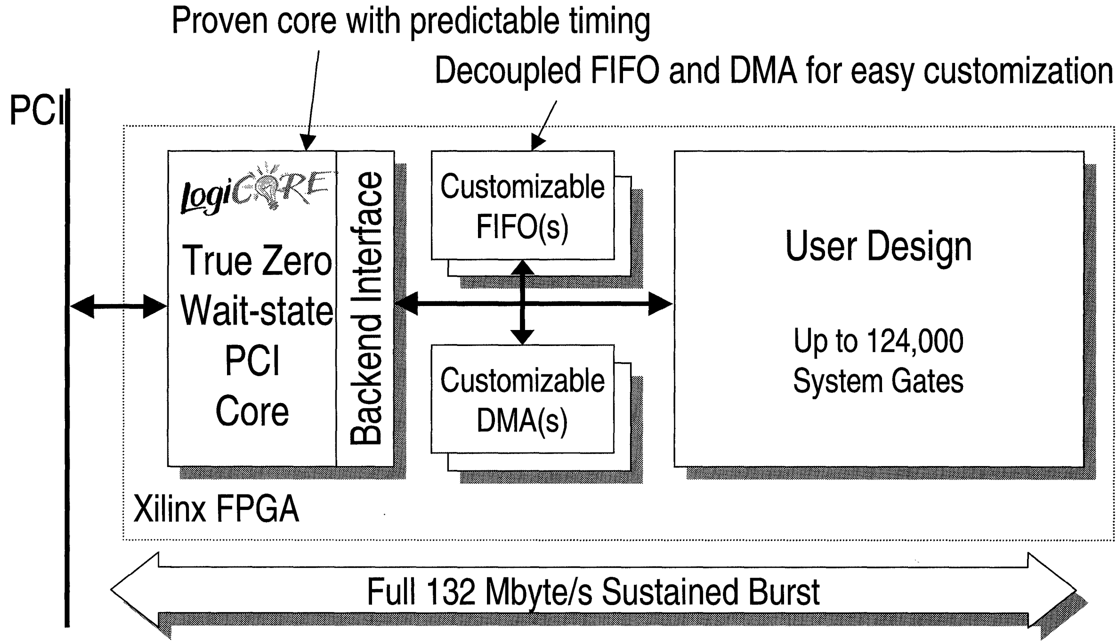

Proven core with predictable timing
Decoupled FIFO and DMA for easy customization
PCI
logiCORE
True Zero Wait-state PCI Core
Backend Interface
Customizable FIFO(s)
Customizable DMA(s)
User Design
Up to 124,000 System Gates
Xilinx FPGA
Full 132 Mbyte/s Sustained Burst

Figure 2: Block Diagram of PCI32 4000 Interface and Backend Interface

About this Catalog

The information in this catalog is also available on the Xilinx web-site, WebLINX at

`www.xilinx.com/products/logicore/pci/pci_sol.htm`

Xilinx will use the web as primary means of delivering and updating this information since it is so dynamic by nature. We strongly recommend that customers consult the web for the latest information on new product availability and datasheet revisions.

XILINX®

PCI Products

PCI32 4000 Master & Slave Interfaces Version 2.0

May, 1998

Data Sheet

XILINX®

Xilinx Inc.
2100 Logic Drive
San Jose, CA 95124
Phone: +1 408-559-7778
Fax: +1 408-377-3259
E-mail: Techsupport: hotline@xilinx.com
Feedback: logicore@xilinx.com
URL: http://www.xilinx.com

Introduction

With Xilinx LogiCORE PCI32 4000 Master and Slave interfaces Version 2.0, a designer can build a customized, 32 bit, 33 MHz fully PCI compliant system with the highest possible sustained performance, 132 Mbytes/s, and up to 124,000 system gates in a XC4000XLT FPGA.

Features

- Fully 2.1 PCI compliant 32 bit, 33 MHz PCI Interface
  - Master (Initiator/Target)
  - Slave (Target-only)
- Programmable single-chip solution with customizable back-end functionality
- Pre-defined implementation for predictable timing in Xilinx XC4000XLT FPGAs or HardWire™ FpgASICs (see *LogiCORE Facts* for listing of supported devices)
- Incorporates Xilinx Smart-IP Technology
- 3.3 V Operation with XC4000XLT devices
- Zero wait-state burst operation
- Fully verified design
  - Tested with the Xilinx internal testbench
  - Tested in hardware (proven in FPGAs and HardWire devices)
- Configurable on-chip dual-port FIFOs can be added for maximum burst speed (see *Xilinx Documents* section)
- Design Once™ - automatic conversion to a HardWire FpgASIC for cost reduction
- Supported Initiator functions (PCI Master only)
  - Initiate Memory Read, Memory Write, Memory Read Multiple (MRM), Memory Read Line (MRL) commands
  - Initiate I/O Read, I/O Write commands
  - Initiate Configuration Read, Configuration Write commands
  - Bus Parking
  - Basic Host Bridging

LogiCORE™ Facts

Core Specifics	
Device Family	XC4000XLT
CLBs Used	178 - 308
IOBs Used[1]	53/51
System Clock f_{max}	0 - 33MHz
Device Features Used	Bi-directional data buses SelectRAM™ (optional user FIFO) Boundary scan (optional)

Supported Devices/Resources Remaining		
	I/O[1,2]	CLB[3]
XC4013XLT PQ208	99/101	268 - 398
XC4013XLT PQ240	133/135	268 - 398
XC4028XLT HQ240	133/135	716 - 846
XC4062XLT HQ240	133/135	1996 - 2126
XC4062XLT BG432	293/295	1996 - 2126

Provided with Core	
Documentation	*PCI32 User's Guide* *PCI Data Book*
Design File Formats	VIEW*logic* schematics VHDL, Verilog Simulation Model NGO Netlist[4]
Constraint Files	M1 User Constraint File (UCF) M1 Guide files
Verification Tools	VIEW*logic* command files VHDL Testbench Verilog Testbench
Core Symbols	VIEW*logic*, VHDL, Verilog
Reference designs & application notes	Example design: Ping Reference Design[5] Synthesizable PCI Bridge
Additional Items	Reference book: *PCI System Architecture*

Design Tool Requirements	
Xilinx Core Tools	M1.4
Entry/Verification Tools[6]	For CORE instantiation: VHDL, Verilog, Schematic For changing source files: Workview Office V7.1.2 or V7.2

Notes: See next page.

LogiCORE™ Facts (cont.)
Support
Xilinx provides technical support for this LogiCORE™ product when used as described in the User's Guide or supporting Application Notes. Xilinx cannot guarantee timing, functionality, or support of the product if implemented in devices not listed above, or customized beyond that referenced in the product documentation, or if any changes are done in sections of the design marked as "DO NOT MODIFY".

Notes:
1. Master/Slave.
2. The XLT devices use 8 I/O locations for Vtt pins; see XC4000XLT Data Sheet.
3. The exact number of CLBs depends on user configuration of the core and level of resource sharing with adjacent logic. Factors that can affect the size of the design are number and size of the BARs, zero vs. one wait-state, and medium vs. slow decode. These numbers include a 16 x 32 FIFO.
4. Available on Xilinx Home Page, in the LogiCORE PCI Lounge: www.xilinx.com/products/logicore/pci/pci_sol.htm
5. Slave only.
6. See Xilinx Home Page for supported EDA tools.

Features (cont.)

- Supported Target functions (PCI Master and Slave)
  - Type 0 Configuration Space Header
  - Up to 3 Base Address Registers (memory or I/O with adjustable block size from 16 bytes to 2 GBytes, slow or medium decode speed)
  - Parity Generation (PAR), Parity Error Detection (PERR# and SERR#)
  - Memory Read, Memory Write, Memory Read Multiple (MRM), Memory Real Line (MRL), Memory Write, Invalidate (MWI) commands
  - I/O Read, I/O Write commands
  - Configuration Read, Configuration Write commands
  - 32-bit data transfers, burst transfers with linear address ordering
  - Target Abort, Target Retry, Target Disconnect
  - Full Command/Status Register
- Available for configuration and download on the Web
  - Web-based configuration with intuitive GUI
  - Generation of proven design files

Applications

- PCI add-in boards such as graphic cards, video adapters, LAN adapters and data acquisition boards
- Embedded applications within telecommunication and industrial systems
- CompactPCI boards
- Other applications that need PCI

General Description

The LogiCORE™ PCI32 4000 Master and Slave Interfaces V2.0 are pre-implemented and fully tested modules for Xilinx XC4000XLT FPGAs (see *LogiCORE Facts* for listing of supported devices). The pin-out and the relative placement of the internal Configurable Logic Blocks (CLBs) are predefined. Critical paths are controlled by TimeSpecs and guide files to ensure that timing is always met. This significantly reduces engineering time required to implement the PCI portion of your design. Resources can instead be

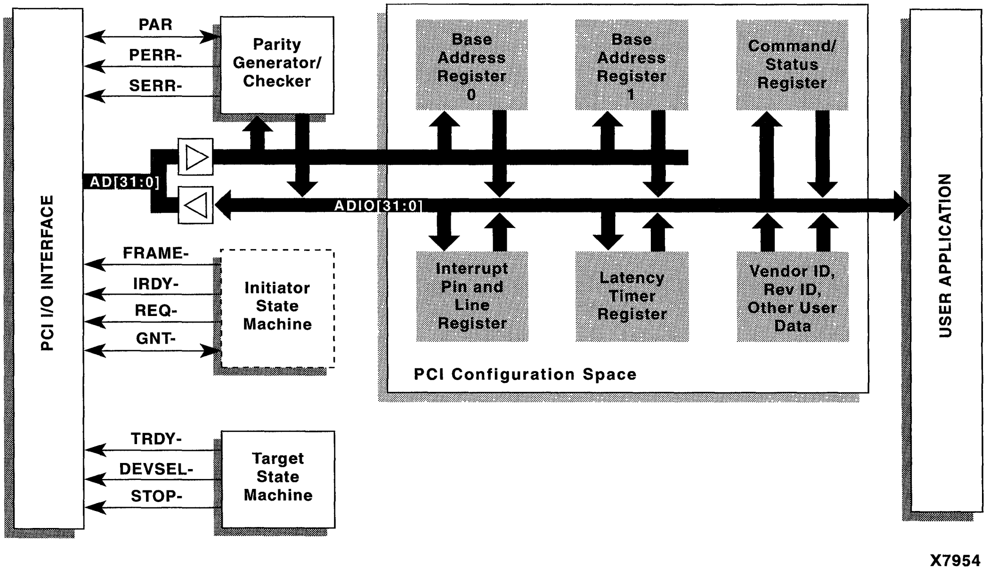


Figure 1: LogiCORE™ PCI32 4000 Interface Block Diagram (BAR 2 not shown)

focused on the unique back-end logic in the FPGA and the system level design. As a result, the LogiCORE™ PCI products can cut your development time by several months.

Xilinx XC4000XLT Series FPGAs enable designs of fully PCI-compliant systems. The devices meet all required electrical and timing parameters including AC output drive characteristics, input capacitance specifications (10pF), 7 ns setup and 0 ns hold to system clock, and 11 ns system clock to output. These devices meet all specifications for 3.3 V PCI. Although the XLT devices have a 3V driver they can be used in a 5V PCI system and meet timing for up to 8 loads.

The XC4000XLT devices differ from regular XL devices by the addition of clamp diodes, required by the PCI 3.3 V electrical specification. For more details about this see the *XC4000XLT FPGA Data Sheet*.

The PCI Compliance Checklist (later in this databook) has additional details about electrical compliance. Other features that enable efficient implementation of a complete PCI system in the XC4000XLT includes:

- Select-RAM™ memory: on-chip ultra-fast RAM with synchronous write option and dual-port RAM option. Used in the PCI Interfaces to implement the FIFO.
- Individual output enable for each I/O
- Internal 3-state bus capability
- 8 global low-skew clock or signal distribution networks
- IEEE 1149.1-compatible boundary scan logic support

The module is carefully optimized for best possible performance and utilization in the XC4000XLT FPGA architecture. When implemented in the XC4013, more than 50% of the FPGA's resources remain for integrating a unique back-end interface and other system functions into a fully programmable one-chip solution. When implemented in a XC4062, 90% of the FPGA's resources remain.

Xilinx DesignOnce™ service allows an automatic conversion to a low cost HardWire™ device for high-volume production.

Smart-IP Technology

Drawing on the architectural advantages of Xilinx FPGAs, new Xilinx Smart-IP technology ensures highest performance, predictability, repeatability, and flexibility in PCI designs. The Smart-IP technology is incorporated in every LogiCORE PCI Core.

Xilinx Smart-IP technology leverages the Xilinx architectural advantages, such as look-up tables (LUTs), distributed RAM, and segmented routing, and floorplanning information, such as logic mapping and relative location constraints. This technology provides the best physical layout, predictability, and performance. Additionally, these predetermined features allow for significantly reduced compile times over competing architectures.

The PCI32 Spartan Interface can parameterized, allowing for design flexibility in which users can create the exact PCI interface needed. PCI Cores made with Smart-IP technology are unique by maintaining their performance and predictability regardless of the device size.

Functional Description

The LogiCORE PCI32 4000 Interfaces are partitioned into five major blocks, plus the user application, shown in Figure 1. Each block is described below.

PCI I/O Interface Block

The I/O interface block handles the physical connection to the PCI bus including all signaling, input and output synchronization, output three-state controls, and all request-grant handshaking for bus mastering.

Parity Generator/Checker

Generates/checks even parity across the AD bus, the CBE lines, and the PAR signal. Reports data parity errors via PERR- and address parity errors via SERR-.

Target State Machine

This block manages control over the PCI interface for Target functions. The states implemented are a subset of equations defined in "Appendix B" of the *PCI Local Bus Specification*. The controller is a high-performance state machine using state-per-bit (one-hot) encoding for maximum performance. State-per-bit encoding has narrower and shallower next-state logic functions that closely match the Xilinx FPGA architecture.

Initiator State Machine (PCI Master only)

This block manages control over the PCI interface for Initiator functions. The states implemented are a subset of equations defined in "Appendix B" of the *PCI Local Bus Specification*. The Initiator Control Logic also uses state-per-bit encoding for maximum performance.

PCI Configuration Space

This block provides the first 64 bytes of Type 0, version 2.1, Configuration Space Header (CSH) (see Table 1) to support software-driven "Plug-and Play" initialization and configuration. This includes Command, Status, and three Base Address Registers (BARs). BAR 2 is not shown in figure 1. These BARs illustrate how to implement memory- or I/O-mapped address spaces. Each BAR sets the base address for the interface and allows the system software to determine the addressable range required by the interface. Using a combination of Configurable Logic Block (CLB) flip-flops for the read/write registers and CLB look-up tables for the read-only registers results in optimized packing density and layout.

With this release, the hooks for extending configuration space has been built into the backend interface. These hooks, including the ability to implement a CapPtr in configuration space, allows the user to implement functions such as Advanced Configuration and Power Interface (ACPI) in the backend design.

Table 1: PCI Configuration Space Header

31	16	15	0	
Device ID		Vendor ID		00h
Status		Command		04h
Class Code			Rev ID	08h
BIST	Header Type	Latency Timer	*Cache Line Size*	0Ch
Base Address Register 0 (BAR0)				10h
Base Address Register 1 (BAR1)				14h
Base Address Register 2 (BAR2)				18h
Base Address Register 3 (BAR3)				1Ch
Base Address Register 4 (BAR5)				20h
Base Address Register 5 (BAR5)				24h
Cardbus CIS Pointer				28h
Subsystem ID		Subsystem Vendor ID		2Ch
Expansion ROM Base Address				30h
Reserved			CapPtr	34h
Reserved				38h
Max_Lat	Min_Gnt	Interrupt Pin	Interrupt Line	3Ch
Reserved				40h-FFh

Note:
Italicized address areas are not implemented in the LogiCORE PCI32 4000 Interface default configuration. These locations return zero during configuration read accesses.

User Application with Optional Burst FIFOs

The LogiCORE PCI32 4000 Interfaces provide a simple, general-purpose interface with a 32-bit data path and latched address for de-multiplexing the PCI address/data bus. The general-purpose user interface allows the rest of the device to be used in a wide range of applications.

Typically, the user application contains burst FIFOs to increase PCI system performance (An Application Note is available, please see the *Xilinx Documents* section). An on-chip read/write FIFO, built from the on-chip synchronous dual-port RAM (SelectRAM™) available in XC4000XLT devices, supports data transfers in excess of 33 MHz.

Interface Configuration

The LogiCORE PCI32 4000 Interfaces can easily be configured to fit unique system requirements using Xilinx web-based graphical configuration tool or changing the VHDL, Verilog, or VIEW*logic* configuration file. The following customization is supported by the LogiCORE product and described in accompanying documentation.

- Initiator or target functionality (PCI Master only)
- Base Address Register configuration (1 - 3 Registers, size and mode)
- Configuration Space Header ROM
- Initiator and target state machine (e.g., termination conditions, transaction types and request/transaction arbitration)
- Burst functionality
- User Application including FIFO (back-end design)

Table 2: PCI Bus Commands

CBE [3:0]	Command	PCI Master	PCI Slave
0000	Interrupt Acknowledge	No[1]	Ignore
0001	Special Cycle	No[1]	Ignore
0010	I/O Read	Yes	Yes
0011	I/O Write	Yes	Yes
0100	Reserved	Ignore	Ignore
0101	Reserved	Ignore	Ignore
0110	Memory Read	Yes	Yes
0111	Memory Write	Yes	Yes
1000	Reserved	Ignore	Ignore
1001	Reserved	Ignore	Ignore
1010	Configuration Read	Yes	Yes
1011	Configuration Write	Yes	Yes
1100	Memory Read Multiple	Yes	Yes
1101	Dual Address Cycle	No[1]	Ignore
1110	Memory Read Line	Yes	Yes
1111	Memory Write Invalidate	No[1]	Yes

Note:
[1] The Initiator can present these commands, however, they either require additional user-application logic to support them or have not been thoroughly tested.

Supported PCI Commands

Table 2 illustrates the PCI bus commands supported by the LogiCORE™ PCI32 4000 Interfaces. The PCI Compliance Checklist, found later in this data book, has more details on supported and unsupported commands.

Burst Transfer

The PCI bus derives its performance from its ability to support burst transfers. The performance of any PCI application depends largely on the size of the burst transfer. A FIFO to support PCI burst transfer can efficiently be implemented using the XC4000XLT on-chip RAM feature, SelectRAM™. Each XC4000XLT CLB supports two 16x1 RAM blocks. This corresponds to 32 bits of single-ported RAM or 16 bits of dual-ported RAM, with simultaneous read/write capability.

Bandwidth

The Xilinx PCI32 4000 Interfaces support a sustained bandwidth of up to 132 MBytes/sec. The design can be configured to take advantage of the ability of the LogiCORE PCI32 Interface to do very long bursts. Since the FIFO isn't a fixed size, burst can go on as long as the chipset arbiter will allow. Furthermore, since the FIFOs and DMA are decoupled from the proven core, a designer can modify these functions without effecting the critical PCI timing.

The flexible Xilinx backend, combined with support for many different PCI features, gives users a solution that lends itself to being used in many high-performance applications. Xilinx is able to support different depths of FIFOs as well as dual port FIFOs, synchronous or asynchronous FIFOs and multiple FIFOs. The user is not locked into one DMA engine, hence, a DMA that fits a specific application can be designed.

The theoretical maximum bandwidth of a 32 bit, 33 MHz PCI bus is 132 MB/s. How close you get to this maximum will depend on several factors, including the PCI design used, PCI chipset, the processor's ability to keep up with your data stream, the maximum capability of your PCI design and other traffic on the PCI bus. Older chipsets and processors will tend to allow less bandwidth than newer ones.

This version of the Interface, supports a selectable wait-state for burst operation. The XC4013XLT-1, XC4028XLT-09 and XC4062XLT-09 support zero wait-state burst, equal to a sustained bandwidth of up to 132 MBytes/sec. The XC4028XLT-1 and the XC4062XLT-1 support zero wait-states while sinking data, but require one wait-state while sourcing data. See Table 3 for a list of required speed grades.

Table 3: Required Speed Grade for Desired Number of wait-states.

Device	Required Speed Grade	
	One Wait-State	Zero Wait-State
XC4013XLT	-1	-1
XC4028XLT	-1	-09
XC4062XLT PQ240	-1	TBD
XC4062XLT BG432	-1	-09

In the Zero wait-state mode, no wait-states are inserted either while sourcing data or receiving data. This allows a 100% burst transfer rate in both directions with full PCI compliance. No additional wait-states are inserted in response to a wait-state from another agent on the bus. Either IRDY or TRDY is kept asserted until the current data phase ends, as required by the V2.1 PCI Specification.

In one wait-state mode, the LogiCORE PCI32 4000 Interface automatically inserts a wait-state when sourcing data (Initiator Write, Target Read) during a burst transfer. In this mode, the LogiCORE PCI32 4000 Interface can accept data at 100% burst transfer rate and supply data at 50%.

See Table 4 for a PCI bus transfer rates for various operations in either zero or one wait-state mode.

Table 4: LogiCORE PCI32 4000 Transfer Rates

Zero Wait-State Mode	
Operation	**Transfer Rate**
Initiator Write (PCI ← LogiCORE)	3-1-1-2
Initiator Read (PCI → LogiCORE)	4-1-1-2
Target Write (PCI→ LogiCORE)	5-1-1-1
Target Read (PCI ← LogiCORE)	6-1-1-1
One Wait-State Mode	
Operation	**Transfer Rate**
Initiator Write (PCI ← LogiCORE)	3-2-2-2
Initiator Read (PCI → LogiCORE)	4-1-1-2
Target Write (PCI→ LogiCORE)	5-1-1-1
Target Read (PCI ← LogiCORE)	6-2-2-2

Note: Initiator Read and Target Write operations have effectively the same bandwidth for burst transfer.

Timing Specification

The XC4000XLT family, together with the LogiCORE PCI32 products enables design of fully compliant PCI systems. Backend design can affect the maximum speed your design is capable of. Factors in your back-end designs that can affect timing include loading of hot signals coming directly from the PCI bus, and gate count. Table 5 shows the key timing parameters for the LogiCORE PCI32 Interfaces that must be met for full PCI compliance.

Table 5: Timing Parameters [ns]

Parameter	Ref.	PCI Spec.		LogiCORE PCI32 4000, XC4000XLT-1	
		Min	Max	Min	Max
CLK Cycle Time		30	∞	30[1]	∞
CLK High Time		11		11	
CLK Low Time		11		11	
CLK to Bus Signals Valid[3]	T_{ICKOF}	2	11	2[2]	8.5
CLK to REQ# and GNT# Valid[3]	T_{ICKOF}	2	12	2[2]	11
Tri-state to Active		2		2[2]	
CLK to Tri-state			28		28[1]

Parameter	Ref.	PCI Spec.		LogiCORE PCI32 4000, XC4000XLT-1	
		Min	Max	Min	Max
Bus Signal Setup to CLK (IOB)	T_{PSD}		7		7
Bus Signal Setup to CLK (CLB)			7		7[1]
GNT# Setup to CLK	T_{PSD}		10		7
GNT# Setup to CLK (CLB)	T_{PSD}		10		10
Input Hold Time After CLK (IOB)	T_{PHD}		0		0
Input Hold Time After CLK (CLB)			0		0[2]
RST# to Tri-state			40		40[2]

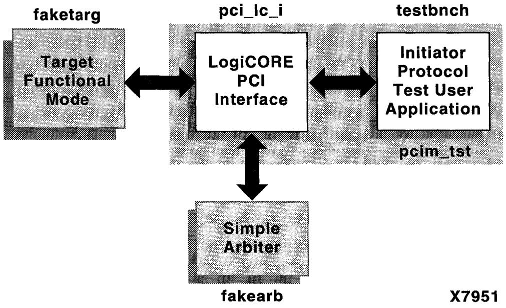
Notes:
1. Controlled by TIMESPECS, included in product
2. Verified by analysis and bench-testing
3. IOB configured for Fast slew rate

Verification Methods

Xilinx has developed a testbench with numerous vectors to test the Xilinx PCI design; this is included with the LogiCORE PCI32 4000 Master and Slave Interfaces A version of this testbench is also used internally by the Xilinx PCI team to verify the PCI32 Interfaces. Additionally, the PCI32 Interfaces have been tested in hardware for electrical, functional and timing compliance.

The testbench shipped with the interface verifies the PCI interface functions according to the test scenarios specified in the *PCI Local Bus Specification, V2.1*; see Figure 2. This testbench consists of 28 test scenarios, each designed to test a specific PCI bus operation. Refer to the checklists chapter in this databook for a complete list of scenarios.

Figure 2: PCI Protocol Testbench



Ping Reference Design

The Xilinx LogiCORE PCI "PING" Application Example, delivered in VHDL and Verilog, has been developed to provide an easy-to-understand example which demonstrates many of the principles and techniques required to successfully use a LogiCORE PCI32 4000 Interface in a System On A Chip solution.

Synthesizable PCI Bridge Design Example

Synthesizable PCI bridge design examples, delivered in Verilog and VHDL, are available to demonstrate how to interface to the LogiCORE PCI32 4000 V2.0 Interfaces and provides a modular foundation upon which to base other designs. See separate data sheet for details.

Device Utilization

The Target-Only and Target/Initiator options require a variable amount of CLB resources for the PCI32 4000 Interfaces. Choosing between one and zero wait-states device (e.g. XC4062-1 vs. -09) will change the amount of logic used. The core now includes a switch to force the entire deletion of unused Base Address Registers.

Utilization can vary widely, depending on the configuration choices made by the designer. Options that can affect the size of the core are:

- Initiator vs. Target-Only. The Initiator requires about 12 CLBs more than the target (not set in the cfg file; set at the time the core is generated).
- Number of Base Address Registers Used. Turning off any unused BARs will save on resources.
- Size of the BARs. Setting the BAR to a smaller size requires more flip-flops. A smaller address space requires more flip-flops to decode.
- Decode Speed. Medium decode requires slightly more logic than slow decode.
- Number of wait-states. Zero wait-states requires more logic than one wait-state.
- Latency timer. Disabling the latency timer will save a few resources. It must be enabled for bursting.

Recommended Design Experience

The LogiCORE PCI32 4000 Interfaces are pre-implemented allowing engineering focus at the unique back-end functions of a PCI design. Regardless, PCI is a high-performance system that is challenging to implement in any technology, ASIC or FPGA. Therefore, we recommend previous experience with building high-performance, pipelined FPGA designs using Xilinx implementation software, TIMESPECs, and guide files. The challenge to implement a complete PCI design including back-end functions varies depending on configuration and functionality of your application. Contact your local Xilinx representative for a closer review and estimation for your specific requirements.

LogiCORE™

PCI32 Spartan Master & Slave Interface

May, 1998

Data Sheet

XILINX®

Xilinx Inc.
2100 Logic Drive
San Jose, CA 95124
Phone: +1 408-559-7778
Fax: +1 408-377-3259
E-mail: Techsupport: hotline@xilinx.com
Feedback: logicore@xilinx.com
URL: http://www.xilinx.com

Introduction

With Xilinx LogiCORE PCI32 Spartan Master & Slave Interface, a designer can build a customizable, low-cost 32-bit, 33MHz fully PCI compliant system in a Spartan-family FPGA.

Features

- Fully 2.1 PCI compliant 32 bit, 33MHz PCI Interface
  - Master (Initiator/Target)
  - Slave (Target-only)
- Pre-defined implementation for predictable timing in Xilinx Spartan FPGAs (see *LogiCORE Facts* for listing of supported devices)
- Incorporates Xilinx Smart-IP Technology
- 5 V Operation with Spartan devices
- Zero wait-state burst operation
- Fully verified design
  - Tested with the Xilinx internal testbench
  - Tested in hardware (silicon proven)
- Configurable on-chip dual-port FIFOs can be added for maximum burst speed (see *Xilinx Documents* section)
- Programmable single-chip solution with customizable back-end functionality
- Supported Initiator functions
  - Initiate Memory Read, Memory Write, Memory Read Multiple (MRM), Memory Read Line (MRL) commands
  - Initiate I/O Read, I/O Write commands
  - Initiate Configuration Read, Configuration Write commands
  - Bus Parking
  - Basic Host Bridging

LogiCORE™ Facts

Core Specifics	
Device Family	XCS Spartan
CLBs Used[1]	152 - 268
IOBs Used	53
System Clock f_{max}	0 – 33MHz
Device Features Used	Bi-directional data buses SelectRAM™ (optional user FIFO) Boundary scan (optional)

Supported Devices/Resources Remaining		
	I/O	CLB[1]
XCS30 PQ208	107	308 - 424
XCS30 PQ240	141	308 - 424
XCS40 PQ208	107	516 - 632
XCS40 PQ240	141	516 - 632

Provided with Core	
Documentation	*PCI32 User's Guide* *PCI Data Book*
Design File Formats	VHDL, Verilog Simulation Models NGO Netlist[2]
Constraint Files	M1 User Constraint File (UCF) M1 Guide files
Verification Tools	VHDL and Verilog Testbench
Core Symbols	VHDL, Verilog
Reference designs	Synthesizable PCI Bridge Design

Design Tool Requirements	
Xilinx Core Tools	M1.4
Entry/Verification Tools[4]	VHDL, Verilog, Schematic

Support
Xilinx provides technical support for this LogiCORE product when used as described in the User's Guide or supporting Application Notes. Xilinx cannot guarantee timing, functionality, or support of the product if implemented in devices not listed above, or customized beyond that referenced in the product documentation.

Notes:

[1] The exact number of CLBs depends on user configuration of the core and level of resource sharing with adjacent logic. Factors that can affect the size of the design are number and size of the BARs, and medium vs. slow decode. These numbers include a 16 x 32 FIFO.

[2] Available on Xilinx Home Page, in the LogiCORE PCI Lounge: www.xilinx.com/products/logicore/pci/pci_sol.htm

[3] See Xilinx Home Page for supported EDA tools

Features (cont.)

- Supported Target functions
  - Type 0 Configuration Space Header
  - Up to 2 Base Address Registers (memory or I/O with adjustable block size from 16 bytes to 2 GBytes, slow decode speed)
  - Parity Generation (PAR), Parity Error Detection (PERR# and SERR#
  - ACPI Configuration Registers (backend module)
  - Memory Read, Memory Write, Memory Read Multiple (MRM), Memory Real Line (MRL), Memory Write, Invalidate (MWI) commands
  - I/O Read, I/O Write commands
  - Configuration Read, Configuration Write commands
  - 32-bit data transfers, burst transfers with linear address ordering
  - Target Abort, Target Retry, Target Disconnect
  - Full Command/Status Register
- Available for configuration and download on the web
  - Web-based configuration
  - Generation of proven design files

Applications

- PCI add-in boards such as graphic cards, video adapters, LAN adapters and data acquisition boards.
- Embedded applications within telecommunication and industrial systems.
- CompactPCI boards,
- Other applications that need PCI

General Description

The LogiCORE™ PCI32 Spartan Master and Slave Interfaces are pre-implemented and fully tested modules for Xilinx Spartan FPGAs (see *LogiCORE Facts* for listing of supported devices). The pin-out and the relative placement of the internal Configurable Logic Blocks (CLBs) are predefined. Critical paths are controlled by TimeSpecs and guide files to ensure that timing is always met. This significantly reduces engineering time required to implement the PCI portion of your design. Resources can instead be focused on the unique back-end logic in the FPGA and the system level design. As a result, the LogiCORE PCI products can cut your development time by several months.

Xilinx Spartan Series FPGAs enables designs of fully PCI compliant systems. The devices meet all required electrical and timing parameters including AC output drive characteristics, input capacitance specifications (10pF), 7 ns setup and 0 ns hold to system clock, and 11 ns system clock to output. These devices meet all specifications for 5 V PCI.

The PCI Compliance Checklists, found later in this data book, have additional details. Other features that enable efficient implementation of a complete PCI system in the Spartan family includes:

- Select-RAM™ memory: on-chip ultra-fast RAM with synchronous write option and dual-port RAM option. Used in the PCI32 Spartan Interface to implement the FIFO.
- Individual output enable for each I/O
- Internal 3-state bus capability
- 8 global low-skew clock or signal distribution networks
- IEEE 1149.1-compatible boundary scan logic support

See *Spartan FPGA Data Sheet* for more details.

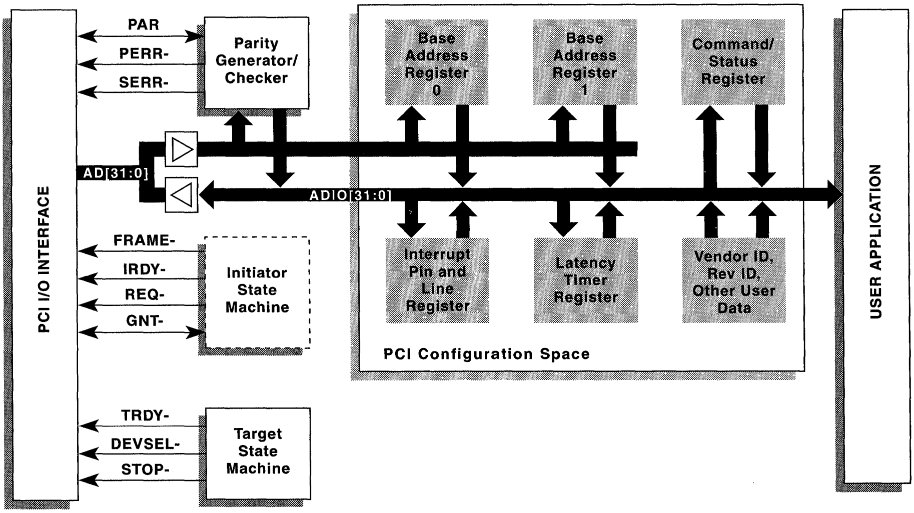


Figure 3: LogiCORE PCI32 Spartan Interface Block Diagram

The module is carefully optimized for best possible performance and utilization in the Spartan FPGA architecture. When implemented in the XCS30, more than 50% of the FPGA's resources remain for integrating a unique back-end interface and other system functions into a fully programmable one-chip solution. When implemented in the XCS40, more than 65% of the FPGA's resources remain for integrating a unique back-end interface and other system functions into a fully programmable one-chip solution.

Smart-IP Technology

Drawing on the architectural advantages of Xilinx FPGAs, new Xilinx Smart-IP technology ensures highest performance, predictability, repeatability, and flexibility in PCI designs. The Smart-IP technology is incorporated in every LogiCORE PCI Core.

Xilinx Smart-IP technology leverages the Xilinx architectural advantages, such as look-up tables (LUTs), distributed RAM, and segmented routing, and floorplanning information, such as logic mapping and relative location constraints. This technology provides the best physical layout, predictability, and performance. Additionally, these predetermined features allow for significantly reduced compile times over competing architectures.

The PCI32 Spartan Interface can parameterized, allowing for design flexibility in which users can create the exact PCI interface needed. PCI Cores made with Smart-IP technology are unique by maintaining their performance and predictability regardless of the device size.

Functional Description

The LogiCORE PCI32 Spartan Interface is partitioned into five major blocks, plus the user application, shown in Figure 1. Each block is described below.

PCI I/O Interface Block

The I/O interface block handles the physical connection to the PCI bus including all signaling, input and output synchronization, output three-state controls, and all request-grant handshaking for bus mastering.

Parity Generator/Checker

Generates/checks even parity across the AD bus, the CBE lines, and the PAR signal. Reports data parity errors via PERR- and address parity errors via SERR-.

Target State Machine

This block manages control over the PCI32 Spartan Interface for Target functions. The states implemented are a subset of equations defined in "Appendix B" of the *PCI Local Bus Specification*. The controller is a high-performance state machine using state-per-bit (one-hot) encoding for maximum performance. State-per-bit encoding has narrower and shallower next-state logic functions that closely match the Xilinx FPGA architecture.

Initiator State Machine

This block manages control over the PCI32 Spartan Interface for Initiator functions. The states implemented are a subset of equations defined in "Appendix B" of the *PCI Local Bus Specification*. The Initiator Control Logic also uses state-per-bit encoding for maximum performance.

PCI Configuration Space

This block provides the first 64 bytes of Type 0, version 2.1, Configuration Space Header (CSH) (see Table 1) to support software-driven "Plug-and Play" initialization and configuration. This includes Command, Status, and two Base Address Registers (BARs). These BARs illustrate how to implement memory- or I/O-mapped address spaces. Each BAR sets the base address for the interface and allows the system software to determine the addressable range required by the interface. Using a combination of Configurable Logic Block (CLB) flip-flops for the read/write registers and CLB look-up tables for the read-only registers results in optimized packing density and layout.

With this release, the hooks for extending configuration space has been built into the backend interface. Setting the CapPtr and bit 15 of the Status Register allows the user to implement functions such as Advanced Configuration and Power Interface (ACPI) in the backend design.

User Application with Optional Burst FIFOs

The LogiCORE PCI32 Spartan Interface provides a simple, general-purpose interface with a 32-bit data path and latched address for de-multiplexing the PCI address/data bus. The general-purpose user interface allows the rest of the device to be used in a wide range of applications.

Typically, the user application contains burst FIFOs to increase PCI system performance (An Application Note is available, please see the *Xilinx Documents* section). An on-chip read/write FIFO, built from the on-chip synchronous dual-port RAM (SelectRAM™) available in Spartan devices, supports data transfers in excess of 33 MHz.

Table 1: PCI Configuration Space Header

31	16	15	0	
Device ID		Vendor ID		00h
Status		Command		04h
Class Code			Rev ID	08h
BIST	Header Type	Latency Timer	*Cache Line Size*	0Ch
Base Address Register 0 (BAR0)				10h
Base Address Register 1 (BAR1)				14h
Base Address Register 2 (BAR2)				18h
Base Address Register 3 (BAR3)				1Ch
Base Address Register 4 (BAR5)				20h
Base Address Register 5 (BAR5)				24h
Cardbus CIS Pointer				28h
Subsystem ID		Subsystem Vendor ID		2Ch
Expansion ROM Base Address				30h
Reserved			CapPtr	34h
Reserved				38h
Max_Lat	Min_Gnt	Interrupt Pin	Interrupt Line	3Ch
Reserved				40h-FFh

Note:
Italicized address areas are not implemented in the LogiCORE PCI32 Spartan Interface default configuration. These locations return zero during configuration read accesses.

Interface Configuration

The LogiCORE PCI32 Spartan Interface can easily be configured to fit unique system requirements using Xilinx web-based PCI Configuration and Download Tool. The following customization is supported by the LogiCORE product and described in accompanying documentation.

- Initiator and target functionality
- Base Address Register configuration (1 - 2 Registers, size and mode)
- Configuration Space Header ROM
- Initiator and target state machine (e.g., termination conditions, transaction types and request/transaction arbitration)
- Burst functionality
- User Application including FIFO (back-end design)

Table 2: PCI Bus Commands

CBE [3:0]	Command	PCI Master	PCI Slave
0000	Interrupt Acknowledge	No[1]	Ignore
0001	Special Cycle	No[1]	Ignore
0010	I/O Read	Yes	Yes
0011	I/O Write	Yes	Yes
0100	Reserved	Ignore	Ignore
0101	Reserved	Ignore	Ignore
0110	Memory Read	Yes	Yes
0111	Memory Write	Yes	Yes
1000	Reserved	Ignore	Ignore
1001	Reserved	Ignore	Ignore
1010	Configuration Read	Yes	Yes
1011	Configuration Write	Yes	Yes
1100	Memory Read Multiple	Yes	Yes
1101	Dual Address Cycle	No[1]	Ignore
1110	Memory Read Line	Yes	Yes
1111	Memory Write Invalidate	No[1]	Yes

Note:
1. The Initiator can present these commands, however, they either require additional user-application logic to support them or have not been thoroughly tested.

Supported PCI Commands

Table 2 illustrates the PCI bus commands supported by the LogiCORE PCI32 Spartan Interface. The compliance checklist later in this data book have more details on supported and unsupported commands.

Burst Transfer

The PCI bus derives its performance from its ability to support burst transfers. The performance of any PCI application depends largely on the size of the burst transfer. A FIFO to support PCI burst transfer can efficiently be implemented using the Spartan on-chip RAM feature, SelectRAM™. Each Spartan CLB supports two 16x1 RAM blocks. This corresponds to 32 bits of single-ported RAM or 16 bits of dual-ported RAM, with simultaneous read/write capability.

Bandwidth

The Xilinx PCI32 Spartan Interface supports a sustained bandwidth of up to 132 MBytes/sec. See the Xilinx web for the supported device/speed grade/wait-states mode combinations.The design can be configured to take advantage of the ability of the LogiCORE PCI32 Interface to do very long bursts. Since the FIFO isn't a fixed size, burst can go on as long as the chipset arbiter will allow. Furthermore, since the FIFOs and DMA are decoupled from the proven core, a designer can modify these functions without effecting the critical PCI timing.

The flexible Xilinx backend, combined with support for many different PCI features, gives users a solution that lends itself to being used in many high-performance applications. Xilinx is able to support different depths of FIFOs as well as dual port FIFOs, synchronous or asynchronous FIFOs and multiple FIFOs. The user is not locked into one DMA engine, hence, a DMA that fits a specific application can be designed.

The theoretical maximum bandwidth of a 32 bit, 33 MHz PCI bus is 132 MB/s. How close you get to this maximum will depend on several factors, including the PCI design used, PCI chipset, the processor's ability to keep up with your data stream, the maximum capability of your PCI design and other traffic on the PCI bus. Older chipsets and processors will tend to allow less bandwidth than newer ones.

In the Zero wait-state mode, no wait-states are inserted either while sourcing data or receiving data. This allows a 100% burst transfer rate in both directions with full PCI compliance. No additional wait-states are inserted in response to a wait-state from another agent on the bus. Either IRDY or TRDY is kept asserted until the current data phase ends, as required by the V2.1 PCI Specification.

In one wait-state mode, the LogiCORE PCI32 Spartan Interface automatically inserts a wait-state when sourcing data (Initiator Write, Target Read) during a burst transfer. In this mode, the LogiCORE PCI32 Spartan Interface can accept data at 100% burst transfer rate and supply data at 50%.

See Table 3 for a PCI bus transfer rates for various operations in either zero or one wait-state mode.

Table 3: LogiCORE PCI32 Spartan Transfer Rates

Zero Wait-State Mode	
Operation	**Transfer Rate**
Initiator Write (PCI ← LogiCORE)	3-1-1-2
Initiator Read (PCI → LogiCORE)	4-1-1-2
Target Write (PCI→ LogiCORE)	5-1-1-1
Target Read (PCI ← LogiCORE)	6-1-1-1
One Wait-State Mode	
Operation	**Transfer Rate**
Initiator Write (PCI ← LogiCORE)	3-2-2-2
Initiator Read (PCI → LogiCORE)	4-1-1-2
Target Write (PCI→ LogiCORE)	5-1-1-1
Target Read (PCI ← LogiCORE)	6-2-2-2

Note: Initiator Read and Target Write operations have effectively the same bandwidth for burst transfer.

Timing Specification

The XCS family, together with the LogiCORE PCI32 Spartan Interface enables design of fully compliant PCI systems. Backend design can affect the maximum speed your design is capable of. Factors in your back-end designs that can affect timing include loading of hot signals coming directly from the PCI bus, gate count and floor planning. Table 4 shows the key timing parameters for the LogiCORE PCI32 Spartan Interface that must be met for full PCI compliance.

Verification Methods

Xilinx has developed a testbench with numerous vectors to test the Xilinx PCI design; this is included with the LogiCORE PCI32 Spartan Master and Slave Interfaces A version of this testbench is also used internally by the Xilinx PCI team to verify the PCI32 Interfaces. Additionally, the PCI32 Interfaces have been tested in hardware for electrical, functional and timing compliance.

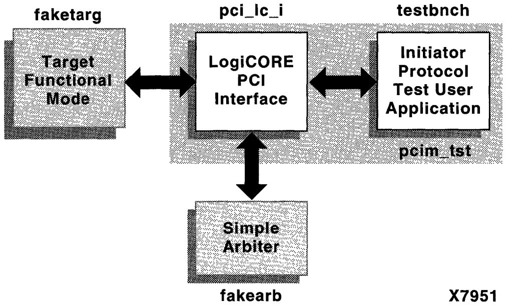
Table 4. Advanced Timing Parameters [ns]

Parameter	Ref.	PCI Spec.		LogiCORE PCI32, XCS-4	
		Min	Max	Min	Max
CLK Cycle Time		30	∞	30[1]	∞
CLK High Time		11		11	
CLK Low Time		11		11	
CLK to Bus Signals Valid[3]	T_{ICK-OF}	2	11	2[2]	9.6
CLK to REQ# and GNT# Valid[3]	T_{ICK-OF}	2	12	2[2]	9.6
Tri-state to Active		2		2[2]	
CLK to Tri-state			28		28[1]
Bus Signal Setup to CLK (IOB)	T_{PSU}		7		7
Bus Signal Setup to CLK (CLB)			7		7[1]
GNT# Setup to CLK	T_{PSU}		10		5.2
Input Hold Time After CLK (IOB)	T_{PH}		0		0
Input Hold Time After CLK (CLB)			0		0[2]
RST# to Tri-state			40		40[2]

Notes:
1. Controlled by TIMESPECs, included in product
2. Verified by analysis and bench-testing
3. IOB configured for Fast slew rate

The testbench shipped with the interface verifies the PCI interface functions according to the test scenarios specified in the *PCI Local Bus Specification, V2.1*; see Figure 2. This testbench consists of 28 test scenarios, each designed to test a specific PCI bus operation. Refer to the checklists chapter in this databook for a complete list of scenarios.

Figure 2. PCI Protocol Testbench



Ping Reference Design

The Xilinx LogiCORE PCI "PING" Application Example, delivered in VHDL and Verilog, has been developed to provide an easy-to-understand example which demonstrates many of the principles and techniques required to successfully use a LogiCORE PCI32 Spartan Interface in a System On A Chip solution.

Synthesizable PCI Bridge Design Example

Synthesizable PCI bridge design examples, delivered in Verilog and VHDL, are available to demonstrate how to interface to the LogiCORE PCI32 Spartan Interface and provides a modular foundation upon which to base other designs. See separate data sheet for details.

Device Utilization

The Target-Only and Target/Initiator options require a variable amount of CLB resources for the PCI32 Spartan Interface. The core includes a switch to force the entire deletion of unused Base Address Registers.

Utilization can vary widely, depending on the configuration choices made by the designer. Options that can affect the size of the core are:

- Initiator vs. Target-Only. The Initiator requires about 12 CLBs more than the target (not set in the cfg file; set at the time the core is generated).
- Number of Base Address Registers Used. Turning off any unused BARs will save on resources.
- Size of the BARs. Setting the BAR to a smaller size requires more flip-flops. A smaller address space requires more flip-flops to decode.
- Decode Speed. Medium decode requires slightly more logic than slow decode.
- Latency timer. Disabling the latency timer will save a few resources. It must be enabled for bursting.

Recommended Design Experience

The LogiCORE PCI32 Spartan Interface is pre-implemented allowing engineering focus at the unique back-end functions of a PCI design. Regardless, PCI is a high-performance system that is challenging to implement in any technology, ASIC or FPGA. Therefore, we recommend previous experience with building high-performance, pipelined FPGA designs using Xilinx implementation software, TIMESPECs, and guide files. The challenge to implement a complete PCI design including back-end functions varies depending on configuration and functionality of your application. Contact your local Xilinx representative for a closer review and estimation for your specific requirements.

Synthesizable PCI Bridge Design Examples

May, 1998

Data Sheet

Xilinx Inc.
2100 Logic Drive
San Jose, CA 95124
Phone: +1 408-559-7778
Fax: +1 408-377-3259
E-mail: Techsupport: hotline@xilinx.com
Feedback: logicore@xilinx.com
URL: http://www.xilinx.com

Introduction

This synthesizable PCI bridge designs are a set of example application bridges for use with the LogiCORE PCI32 Interfaces. They are delivered in Verilog and VHDL and have been fully tested with various devices. These examples demonstrate how to interface to the PCI core and provide a modular foundation upon which to base other designs. The reference designs can be easily modified to remove select portions of functionality. The facts table lists the set of features and specifics for each design.

General Description

Part of or all of the design is available at no cost to all registered LogiCORE PCI32 Interface customers, who can download it from the LogiCORE PCI Lounge at

http://www.xilinx.com/products/logicore/pci/pci_sol.htm.

See the Ordering Information chapter for details.

These designs are general purpose data transfer engines to be used with the LogiCORE PCI32 Interfaces. Figure 1 presents a block diagram of the DE01 design. Typically, the user will customize the local interface to conform to a particular peripheral bus (ISA, VME, i960)or attach to a memory device. The design is modular so that unused portions may be removed. DE02 and DE03 are subsets of this design, and do not contain parts of the target functionality as indicated in the facts table.

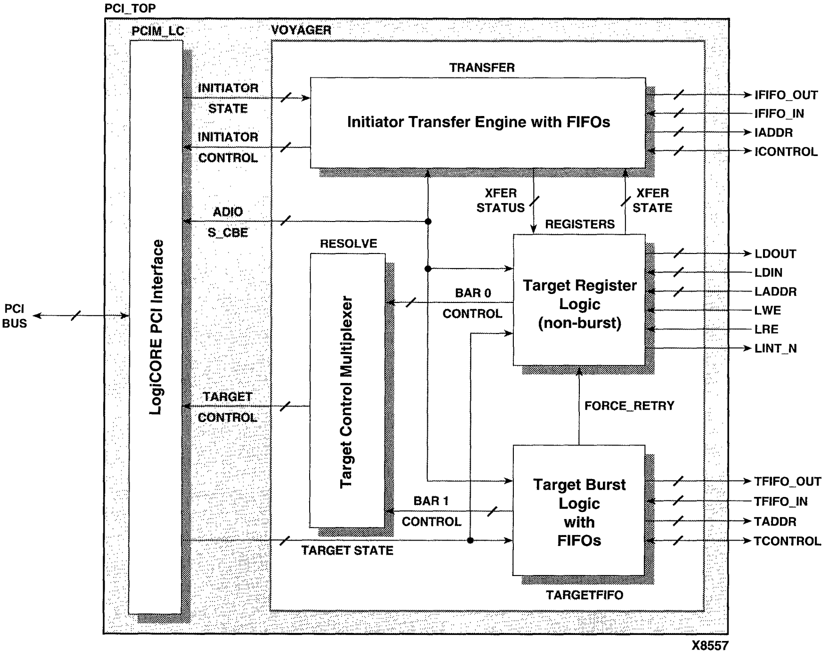


Figure 3: Synthesizable PCI Bridge Block diagram

Design Example Facts

Features	DE01	DE02	DE03
Initiator Functions			
Separate read and write FIFOs (unidirectional)	✓	✓	✓
Simple block data transfer engine (DMA)	✓	✓	✓
Programmable Burst sizes fixed by transfer counter	✓	✓	✓
Auto data delivery (handles terminations)	✓	✓	✓
Discard counter to prevent deadlock	✓	✓	✓
Initiator address counter	✓	✓	✓
Target Functions			
BAR 0 - Supports single data phase transfers	✓	✓	✓
BAR 0 - Region 1 demonstrates doorbells	✓		
BAR 0 - Region 2 demonstrates mailboxes	✓		
BAR 0 - Region 3 demonstrates long latency accesses	✓		
BAR 0 - Region 4 contains control registers for initiator	✓	✓	✓
BAR 1 - Separate read and write FiFOs (unidirectional)	✓		
BAR 1 - Delayed completion discard after time-out	✓		
BAR 1 - Generates target abort on address wrap	✓		
Target address counter	✓	✓	✓
Target functions independent of initiator	✓	✓	✓
Specifics			
Tested Device[1]	XC4062XLT-09 BG432	XC4013XLT-1HQ240 XC4028XLT-1HQ240	XCS40-4PQ240
Tested Platform[2]	FPGA Compiler, A1.4	FPGA Compiler & Express, A1.4, F1.4	Foundation Express, F1.4
CLBs Used[3]	Up to 990	Up to 530	Up to 530
IOBs Used[4]	Up to 320	Up to 150	Up to 150
LogiCORE product	PCI32 4000, V2.0.1	PCI32 4000, V2.0.1	PCI32 Spartan

Notes

1. Listed are the devices that Xilinx used to verify the design. Other devices can be used, however, without guaranteed performance.
2. Listed are the design tools that Xilinx used to verify the design. Other tools can be used. See the LogiCORE PCI data sheets for supported design tools.
3. The CLB count includes the full design and PCI interface. Actual count depends on the implemented feature set.
4. The IOB count includes the full design and PCI interface, assuming all signals are routed off chip. The actual count depends on the implemented feature set.

Functional Description

This design example supports target functionality in two memory spaces. Initiator functionality is controlled by writing into registers. The local bus interface signals are distinct for each block in the design, allowing blocks to be added or removed. Data transfer is pipelined for high clock rate. The functional description listed here describes DE01; DE02 and DE03 will contain a subset of this functionality as listed in the facts table.

BAR0 Configuration

BAR0 is configured as a 4 kilobyte MEM space which maps to a number of registers. This space does not support multiple data phase transfers. All accesses to this space terminate with target disconnect with data.

This space is logically divided into four regions based on functionality. The four regions, and the functions of the registers, are discussed below.

Region One: Doorbells (DE01 Only)

Register DBELL_P1 is a PCI-to-local doorbell. A PCI agent may create an interrupt on the local side by setting any bit of the register. A PCI agent is permitted to read back the status of this register with no side-effects.

When the local side services the interrupt, it reads this register to determine the cause of the interrupt, then clears the interrupt by writing a one to that bit. The local side may read this register without side-effects.

Similarly, DBELL_L1 is a local-to-PCI doorbell. To prevent spurious interrupts, an interrupt may not be cleared by the agent that requested it. The recipient of the interrupt must clear the interrupt. To enforce this, doorbell register bits may not be cleared from the requesting side. Before doorbell interrupts may occur, the doorbell interrupt enable bits in the CONTROL register must be set.

Region Two: Mailboxes (DE01 Only)

Register MBOX_P1 is a PCI-to-local mailbox. A PCI agent may deliver mail to an empty mailbox for a local agent to pick up. When a PCI agent writes to this register, the data is registered and a "full" flag is set. Subsequent writes to a full mailbox have no effect. The PCI agent may not read back delivered mail. Reads of the mailbox from the PCI bus side return the state of the full flag (replicated in all bits).

When the local side reads the mailbox, the "full" flag is cleared. Subsequent reads of an empty mailbox return the last valid data present in the mailbox.

Similarly, MBOX_L1 is a local-to-PCI mailbox. The "full" flag may be monitored in two ways. Mailbox "full" flags are always observable in the CONTROL register, so both PCI agents and local agents may poll the CONTROL register to watch for new messages. Optionally, full mailboxes may create interrupts. Interrupts are created on the recipient's side, and are cleared by reading the mailbox. Before mailbox interrupts may occur, the mailbox interrupt enable bits in the CONTROL register must be set.

Region Three: Bounded Latency Accesses (DE01 Only)

The two registers in this region are used for demonstrating bounded latency non-burst accesses. This type of access may be used in situations where the user application has a short latency with a known upper bound of 16 PCI clocks from the time the initiator asserts FRAME#. This is done by inserting wait states until the target is capable of completing the transaction.

Register BL_CTRL controls the initial latency of read and write operations for itself and BL_DATA. Only the least-significant four bits of the register are implemented, and the register is only accessible from the PCI bus. The local side has no access to this register, so local reads will return all zeroes and writes have no effect.

The second register, BL_DATA, is a general purpose, read/ write register that responds according to the settings in BL_CTRL. This data register is only accessible from the PCI bus. The local side has no access to this register, so local reads will return all zeroes and writes have no effect.

Region Four: Control Registers

The first three registers in this region control the initiator transfer engine.

Register XFER_LEN is used to indicate the length of the data block to be transferred. The low half of the register is not implemented. The high half is implemented as a loadable 16-bit counter.

This register is accessible from both the PCI bus and local bus, and may be read at any time. Status of transfers in progress may be obtained by reading this register. With each successful transfer, the counter decrements.

Register XFER_PADR contains the current PCI bus address for transfers performed by the transfer engine. Depending on the direction of the transfer, this address may be a source or destination.

This register is accessible from both the PCI bus and local bus, and may be read at any time. Status of transfers in progress may be obtained by reading this register. With each successful transfer, the address increments.

Register XFER_LADR contains the current local address for transfers performed by the transfer engine. Depending on the direction of the transfer, this address may be a source or destination. Only the low half of this register is implemented.

This register is accessible from both the PCI bus and local bus, and may be read at any time. Status of transfers in progress may be obtained by reading this register. With each successful transfer, the address increments.

These registers must not be written to while the initiator is active. To ensure this does not occur, writes to these register are disabled while the initiator is active.

BAR1 Configuration (DE01 Only)

BAR1 is configured as a 64 kilobyte MEM space which maps to the target FIFOs. This space supports multiple data phase transfers. Transfers beyond the end of the address space result in target abort. For all other accesses, this region will respond according to how it is accessed. Consult the PCI specification regarding delayed transactions. Data transfer between the local side and PCI bus is achieved using retries and delayed transactions as needed.

Posted Writes

The target performs posted writes. On writes to an idle target, the FIFO accepts incoming data until it is full or the write transaction has ended, whichever occurs first. In the event of a full FIFO, the target issues a disconnect. After the PCI transaction is complete, the target empties the FIFO by writing the data out to the local side until the FIFO is empty. To achieve this, the target latches the destination address for use during write out.

On writes to a busy target (the FIFO is still busy from the previous transaction) the target responds with retry, without putting the request in a retry queue.

Prefetched Reads

For reads, the target may not anticipate the length of the transaction or have the data available in time. For this reason, the target puts the transaction in a retry queue and responds with a retry termination. Then the target prefetches data to fill the FIFO. When the initiator returns to retry the transaction, the data will be available.

If the initiator returns to retry the transaction, and does not completely empty the FIFO, the FIFO is flushed after the transaction is complete. If the initiator does empty the FIFO, and attempts to read more, the target issues a disconnect.

If the initiator never retries the original transaction, deadlock may occur. For this reason, there exists a discard timer that signals a waiting delayed completion should be discarded. This timer times out after 32,768 PCI clocks. This period may be shortened to allow simulation of this event in a reasonable amount of time.

Register File Interface

The operation of this block is synchronous to the PCI clock. This block contains all the control and status registers discussed in the functional description. The local bus access port is defined in Table 1.

Table 1: Local Bus Register Interface

Name	Direction	Function
LWE	input	Write enable for registers
LRE	input	Read enable for registers
LADDR	input	Address input
LDIN	input	Data input
LDOUT	output	Data output
LINT_N	output	Active low local interrupt

Target FIFO Interface

The operation of this block is synchronous to the PCI clock. This block is interfaced to two FIFOs; one is the target read (TRF) FIFO, and the other is the target write (TWF) FIFO.

The FIFOs are identical, but data flows in opposite directions. Table 2 lists the signals used in the interface.

Table 2: Local Bus Target Fifo Interface

Name	Direction	Function
TRF_LD TWF_ST	output output	Data requested or available
TRF_ADDR TWF_ADDR	output output	Transfer starting address
TRF_AF, TWF_AE	output	Transfer almost done flag
TRF_WR, TWF_RD	input	FIFO write and read enable
TRF_DIN TWF_DOUT	input output	Data transfer ports

Initiator FIFO Interface

The operation of this block is controlled by the contents of registers in the register block. This block is interfaced to two FIFOs, similar to the memory interface block. One is the initiator read (IRF) FIFO, and the other is the initiator write (IWF) FIFO.

The FIFOs are identical, but data flows in opposite directions. A description of a FIFO follows in the FIFO section. Table 3 lists the signals used in the interface.

Table 3: Local Bus Initiator Fifo Interface

Name	Direction	Function
IWF_LD IRF_ST	output output	Data requested or available
IF_ADDR	output	Transfer starting address
IWF_AF, IRF_AE	output	Transfer almost done flag
IWF_WR, IRF_RD	input	FIFO write and read enable
IWF_DIN IRF_DOUT	input output	Data transfer ports

Pinout

The register file and FIFO interface pinouts are not fixed to specific FPGA I/O pads, allowing flexibility in customization. The PCI bus specific signals are constrained as part of the LogiCORE PCI32 implementation.

As shipped, all of the register file and FIFO interface signals are brought off-chip, but it is not necessary that any interface signals be brought off chip at all in single FPGA designs.

Core Modifications

Modifications can be done to remove the initiator functionality or selected portions of the target functionality. The full design may be expanded as needed or reduced to a very small subset of the original design. The PCI interface itself is also configurable by the designer.

Verification Methods

This design example includes a system level testbench that simulates a four-slot PCI system. This simulation testbench includes a behavioral host bridge (with programmable arbiter) capable of generating burst transactions and a programmable behavioral target.

Recommended Design Experience

The challenge to implement a complete PCI design varies depending on configuration and functionality of your application. We recommend previous experience with building high-performance, pipelined FPGA designs using the Xilinx implementation software and familiarity with either VHDL or Verilog.

Synchronous FIFO

May, 1998

Data Sheet

XILINX®

Xilinx Inc.
2100 Logic Drive
San Jose, CA 95124
Phone: +1 408-559-7778
Fax: +1 408-377-3259
E-mail: Techsupport: hotline@xilinx.com
Feedback: logicore@xilinx.com
URL: http://www.xilinx.com

Introduction

Xilinx LogiCORE FIFO is a high-performance scalable FIFO module that is provided at no cost with the Xilinx CORE Generator. It is easily parameterized and generated using the intutive graphical user interface of the CORE Generator.

Features

- Data width from 4 to 80 bits
- Depths from 16 to 256 words
- SelectRAM™ used for high density and performance
- Registered output
- Drop-in modules for the XC4000E, EX, XL, XV and Spartan families
- High performance and density guaranteed through Relational Placed Macro (RPM) mapping and placement technology
- Available in Xilinx CORE Generator

Functional Description

The synchronous First-In, First-Out Memory module has a single clock port for both data-read and data-write operations. Data presented at the module's data-input port (D) is written into the next available empty memory location on a rising clock-edge when the write-enable input (WE) is High. The memory-full status output (FULL) indicates that no more empty locations remain in the module's internal memory, and that further data-writes must be avoided until the FULL output returns Low. Data can be read-out of the FIFO via the module's data-output port (Q) in the order in which it was written by asserting the read-enable input (RE) prior to a rising clock edge. The memory-empty status output (EMPTY) indicates that no more data resides in the module's internal memory, and that further data-reads should be avoided until the EMPTY output returns to LOW.

The internal read and write counters that maintain the FIFO status can be reset by asserting the RESET input prior to a rising clock edge. This effectively empties the FIFO, discarding any data that may have been stored in the module but which had not been read-out. Note that RESET has priority over RE and WE.

The FIFO provides two additional outputs - BUFCTR_CE and BUFCTR_UPDN - that may be used to control an external up/down counter such that the counter's output indicates the number of data words that are currently stored in the FIFO. See Figure 1. This may be useful in applications where more information about the FIFO's empty/full status is required - such as half-full, for example.

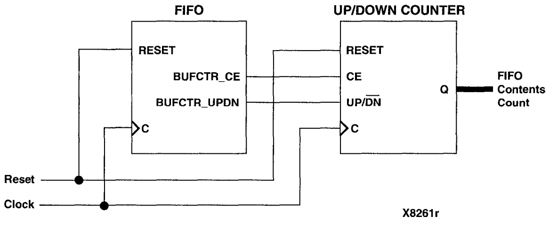


Figure 1: Connections for Up/Down Counter

The FIFO offers the designer the choice of implementing its internal memory with single- or dual-port SelectRAM™. Single-port memory is the most area efficient, but has the restriction that data may not be written and read at the same time. (I.e. WE = High and RE = High is illegal - and will be ignored - when single-port memory is used.) Dual-port memory permits simultaneous data-read and data-write operations, but incurs a higher area cost. See Figures 3 and 4.

The FIFO's internal state machine masks attempts to write data into a full FIFO (i.e. WE = High when FULL = High) and attempts to read data from an empty FIFO (i.e. RE = High when Empty = High). Either of these erroneous situations will be ignored by the FIFO, although no error flag is generated to indicate that the requested operation has not occurred.

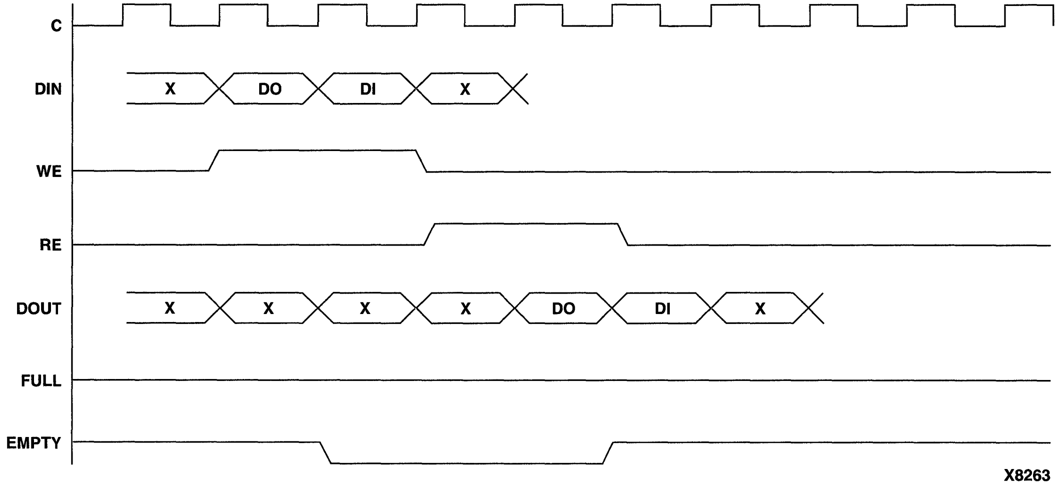


Figure 3: Timing Diagram for Single and Dual Ports

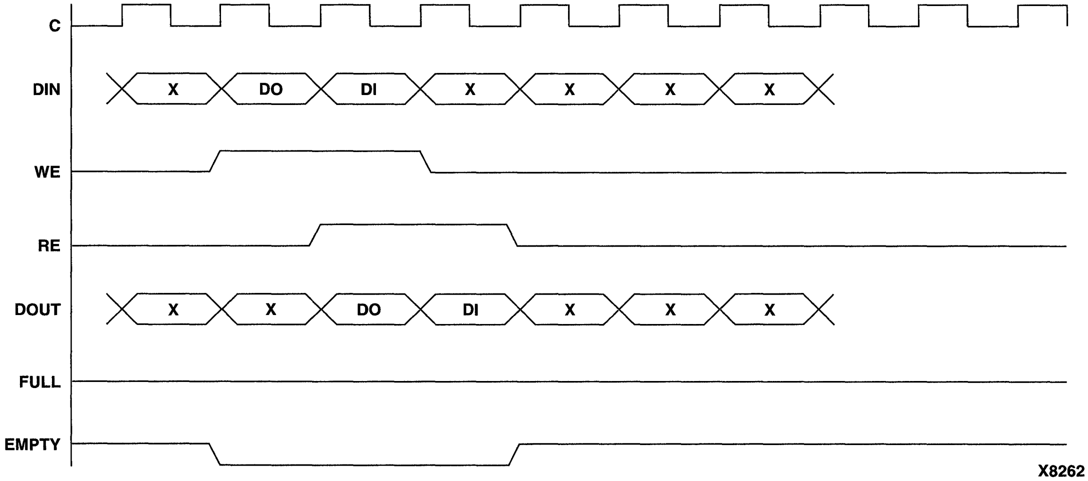


Figure 4: Timing Diagram for Dual Ports - Simultaneous Read and Write

Pinout

Port names for the schematic symbol are shown in Figure 2 and described in Table 1. Timing diagrams for the single and dual ports FIFO are shown in Figures 3 and 4.

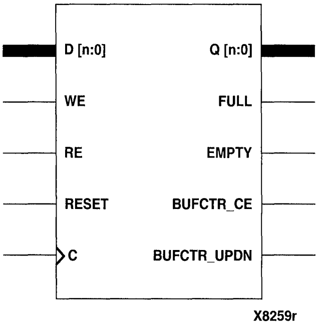


Figure 2: Core Schematic Symbol

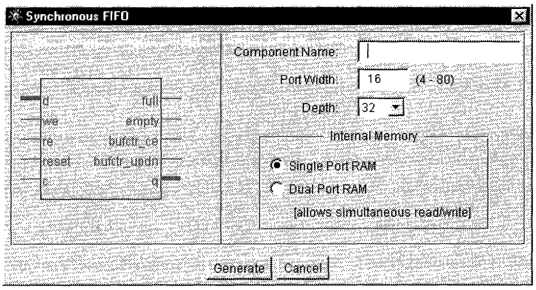
Table 1: Core Signal Pinout

Signal	Signal Direction	Description
D [n:0]	*Input*	DATA INPUT –port through which data is written into the FIFO.
WE	*Input*	WRITE ENABLE – active high signal used to allow the transfer of data from the input data port into the FIFO.
RE	*Input*	READ ENABLE – active high signal used to allow the transfer of data from the FIFO onto the output data pins.
RESET	*Input*	SYNCHRONOUS RESET – clears the FIFO
C	*Input*	CLOCK – with the exception of asynchronous control inputs (where applicable), control and data inputs are captured, and new output data formed on rising clock transitions.
Q [n:0]	*Output*	REGISTERED DATA OUT – the registered output of the FIFO.
FULL	*Output*	FULL – this signal is asserted when the FIFO is full.
EMPTY	*Output*	EMPTY – this signal is asserted when the FIFO is empty.
BUFCTR_CE	*Output*	Optional connect to the clock enable of an external up-down counter.
BUFCTR_UPDN	*Output*	Optional connect to the up-down pin of an external up-down counter.

CORE Generator Parameters

The CORE Generator dialog box for this macro is shown in Figure 5. The parameters are as follows:

- **Component Name**: Enter a name for the output files generated for this module.
- **Port Width**: Select an input bit width from the pull-down menu. The valid range is 4-80.
- **Depth**: Select the number of words in the FIFO from the pull-down menu. The valid range is 16-256 in multiples of 16.
- **Dual Port**: Select either Single Port or Dual Port RAMs.
- Internal Memory: You must choose Dual Port if simultaneous read/write operations are required.



Component Name:
Port Width: 16 (4 - 80)
Depth: 32
Internal Memory
● Single Port RAM
○ Dual Port RAM
[allows simultaneous read/write]

Figure 5: Parameterization Window

Notes:

Xilinx PCI

HotPCI Spartan Prototyping Board

May, 1998

Data Sheet

VIRTUAL COMPUTER CORPORATION

Virtual Computer Corporation

6925 Canby Ave. #103
Reseda, CA 91335 USA
Phone: +1 818-342-8294
Fax: +1 818-342-0240
E-mail: info@vcc.com
Website: http://www.vcc.com

Introduction

To facilitate rapid development of PCI Designs using the Xilinx LogiCORE PCI32 Interface, VCC has developed the HotPCI Spartan Prototyping Board. This allows the designer to quickly develop, modify, and test Xilinx LogiCORE PCI32 designs in-system. The HotPCI Board is provided with the Xilinx PCI32 Design Kit and should be used in conjunction with the LogiCORE PCI32 Spartan Interface.

Features

The Spartan HotPCI board comes standard with the following hardware and features:

- 5 V PCI 2.1 Compliance
- Xilinx Spartan XCS40-4, reconfigurable via PCI bus or Xchecker cable
- Xilinx XC95108-10 for reconfiguration management
- 8x128K bytes of fast SRAM organized as 2 Independent Banks of 32-bit RAM (four 8-bit x 128K)
- Configuration Flash 128KB for initial configuration
- Configuration RAM Cache 128KB for reconfiguration
- Programmable Clock Generator (360KHz to 100 MHz)
- Mezzanine Connectors for daughter cards
- LEDs to indicate configuration finished (DONE), 5 V, and 3.3 V
- 3 Split Power Planes & 1 Ground Plane
- 4 Signal Layers
- Xchecker/Download Cable Module
- PCI Demo bitstream
- Backend design example files for RAM interface
- PCB design files for the PCI Interface only
- Reference drivers for Windows 95 and Windows NT

Options

All options are available directly from VCC. See the Ordering Information chapter for more details.

- Programmable Voltage Control Module (3.3v to 1.8v)
- License for the Configuration Cache Manager™
- Prototyping Daughter Card
- XC40125 Extended Logic Daughter card
- XC6264 RPU Daughter Card
- HotPCI Card with a Xilinx XC4062XL FPGA

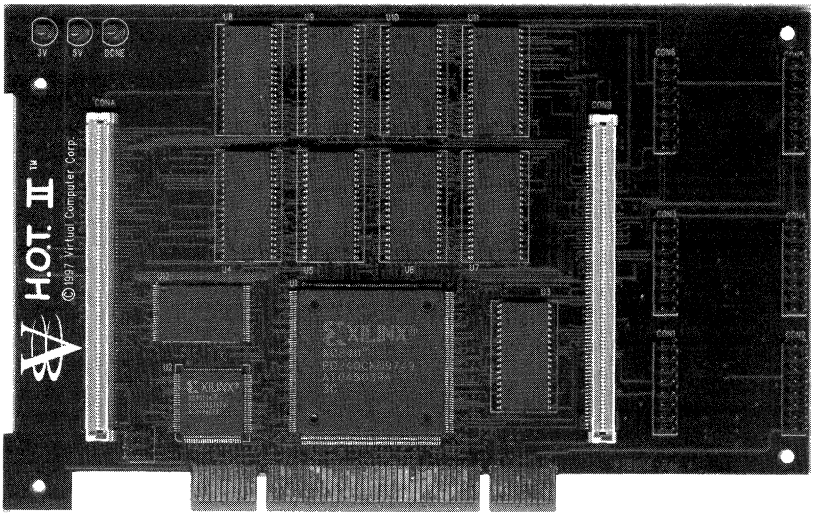


Figure 1: Spartan HotPCI Spartan Prototyping Board

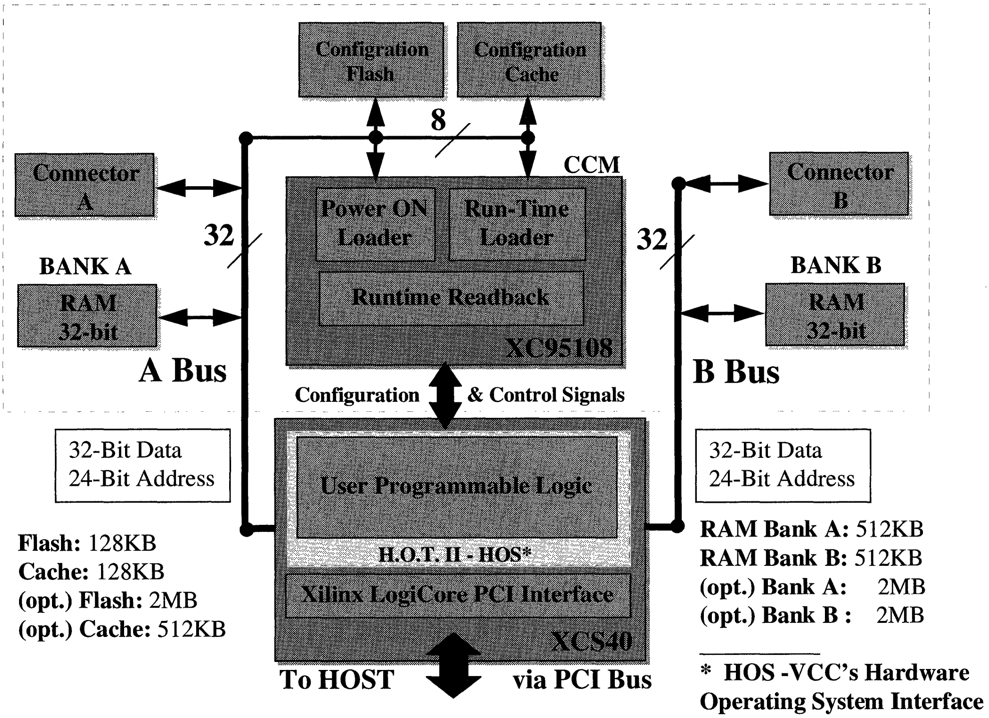


Figure 2: Spartan HotPCI Block Diagram

General Description

The Spartan HotPCI board is a PCI bus based general purpose PCI Prototyping Board, which is provided with the Xilinx PCI32 Design Kit (see the Ordering Information chapter of this data book for details). When used in conjunction with the LogiCORE PCI32 Spartan Interface, designers can quickly modify and test their PCI designs in-system. It features the XCS40 FPGA as the PCI bus interface chip, and a single bank of SRAM plus a VCC proprietary reconfiguration system, the Configuration Cache Manager™ (CCM). The Spartan HotPCI Spartan Prototyping Board is valuable for evaluating, customizing, and verifying the Xilinx LogiCORE PCI32 product line. The mezzanine daughter card connectors offer expandability for prototyping additional features or extending the programmable logic capabilities of the HotPCI board. A series of daughter cards are available from VCC, including a prototyping card for wire-wrap projects, an extended logic card with the XC40125, and a XC6200 Reconfigurable Processing Unit (RPU) card offering microsecond dynamic partial reconfiguration.

Software

The HotPCI Spartan Prototyping Board is supplied with drivers made by Vireo Software, Inc. The Xilinx PCI32 Design Kit also includes driver development tools from Vireo. Together with VCC's HotPCI Board, Vireo's development tools allow easy customization and prototyping of a complete PCI system. See the *Driver::Works Windows Device Driver Development Kit Version 2.0* and the *VtoolsD Windows Device Driver Development Kit Version 3.0* data sheets for more details on the driver development tools. Included with the board is also a bitstream and a Windows '95 application CD that allow demonstration of the card.

Functional Description

The Spartan XCS40 FPGA contains the Xilinx LogiCORE PCI32 Interface Macro and the backend design. VCC supplies a customized backend that allows users to communicate with two fully independent 32-bit banks of RAM and the Configuration Cache Manager™ (CCM). The CCM controls the Run-Time Reconfiguration (RTR) behavior of the system.

The HotPCI board has two independent buses, each with 32-bit data and 24-bit address. There is a daughter board I/O connector for each of these two buses.

Configuration with the CCM

At power-on the FPGA is set to slave serial mode; then the CCM obtains the configuration data from flash and loads the FPGA. Through the PCI bus, the user can load a new configuration into the Configuration Ram Cache. The user then writes to the CCM to start the reconfiguration of the FPGA. During this time access to the board is disabled by the driver. When the FPGA comes back on-line it signals the driver, which reloads the PCI Header information into the LogiCORE PCI Core. A 128KB Configuration Cache RAM can hold 3 XCS40 configurations. See Figure 2 for a block diagram of the board.

Configuration with an Xchecker cable

Configuration can also be performed through an Xchecker cable or download cable. The supplied Xchecker module occupies one of the daughter card I/O connectors. The PCI bus of the host computer must be reset when this occurs.

Notes:

Driver::Works Windows Device Driver Development Kit Version 2.0

May, 1998

Data Sheet

Vireo Software

Vireo Software, Inc.

30 Monument Square, Suite 135
Concord, MA 01742 USA
Phone: +1 978-369-3380
Fax: +1 978-318-6946
E-mail: info@vireo.com
support@vireo.com
Website: http://www.vireo.com

Introduction

To facilitate rapid development of PCI Designs using the Xilinx LogiCORE PCI32 Interface, Vireo is providing the Driver::Works Windows Device Driver Development Kit. This kit includes an interactive GUI Wizard that runs in conjunction with the Microsoft Visual C++ 4.2 and later. Provided at no extra cost with the Xilinx PCI Design Kit, is a full-featured version of Driver::Works licensed for prototyping fully functional drivers and testing them with the HotPCI board. An unrestricted license is available directly from Vireo.

Support

Support for Driver::Works is provided only from Vireo. See Vireo's home page for contact instructions and other details.

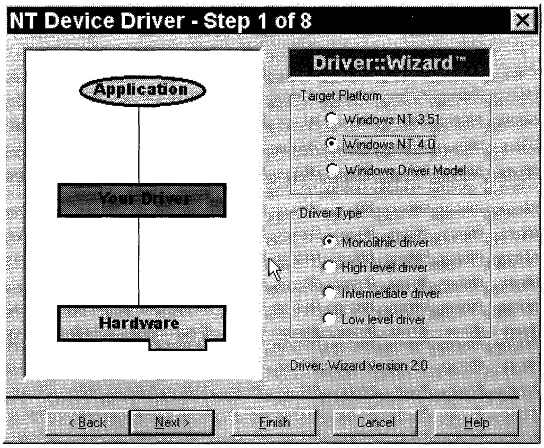
Features

- Windows NT support
- Windows 98 support
- Driver::Wizard™, Vireo's Code Generation Wizard Interface. The Wizard includes automated support for all PCI functions, including:
  - PCI Configuration
  - DMA
  - Mapped Memory
  - Interrupt handling
  - IO Ports
  - Application interface
  - Registry interface
  - Plug and Play handling
- Support for MSVC++ 4.2 and later
- C++ Class Library for NT/WDM driver development
- Full library source code is included
- Dozens of sample drivers, with full source code
- Full Plug and Play support
- Driver Access Architecture (DAA) supports portability between Windows NT, Windows 95, Windows 98, and Win32 Driver Model (WDM)
- Driver::Monitor™ - monitor driver activity without a debugger.
- Ready-to-use examples tested on real-world hardware
- Full technical support through Vireo
- RISC platform support
- More than 700 pages of printed and online documentation

Description

Driver::Works is a next-generation environment for device driver development based on a powerful and flexible C++ class library coupled with a powerful code generation wizard.

Over time, Windows application development has evolved to class libraries such as MFC and development tools such Microsoft's Application Wizard. Vireo provides a similar environment for Windows NT and WDM device driver development.



NT Device Driver - Step 1 of 8

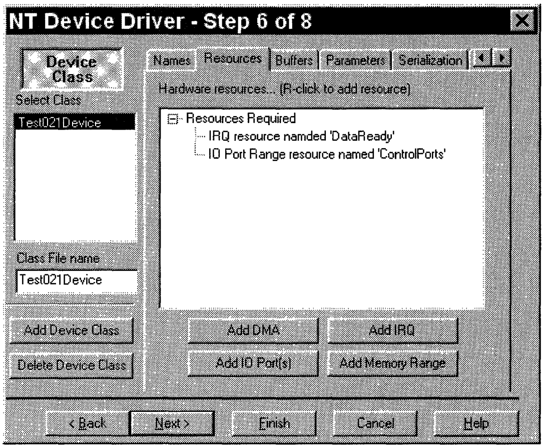
Driver::Wizard™

Target Platform
○ Windows NT 3.51
● Windows NT 4.0
○ Windows Driver Model

Driver Type
● Monolithic driver
○ High level driver
○ Intermediate driver
○ Low level driver

Driver::Wizard version 2.0

Figure 1: Driver::Wizard GUI (earlier version shown)



Select Class: Test021Device

Class File name: Test021Device

```
⊟ Resources Required
    ├── IRQ resource namded 'DataReady'
    └── IO Port Range resource named 'ControlPorts'
```

Figure 2: Driver::Wizard GUI (earlier version shown)

The Driver::Works class library offers thousands of lines of tested code that reduce many complex tasks to simple library calls. In fact, Driver::Works offers by far the most complete device driver library available.

Driver::Works also ships with complete examples that are designed to be used as a basis for further development.

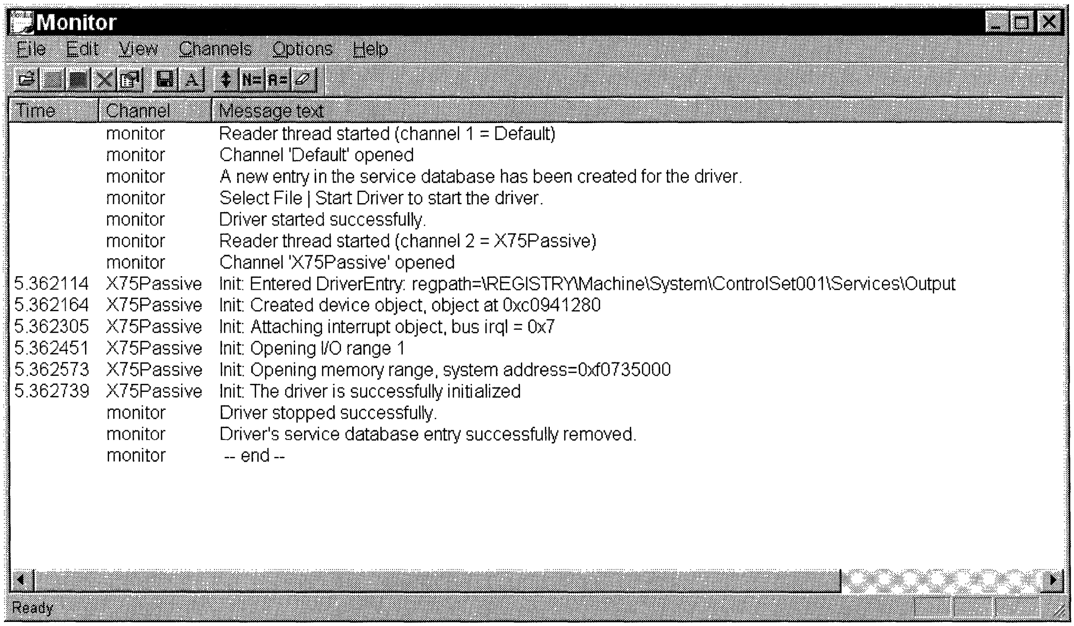
Driver::Works also includes Vireo's unique Driver::Wizard technology, shown in Figures 1 & 2. Driver::Wizard guides you through a series of steps that identify many characteristics of your device. Driver::Wizard then generates source code tailored to your driver. The Driver::Works class library, framework, and Wizard provide access to tens of thousands of lines of working, debugged code that will allow you to develop your device drivers quickly.

Driver::Works implements Vireo's Device Access Architecture (DAA) interfaces. Using DAA, device driver source code can be easily ported between Windows 95, Windows 98, and all versions of Windows NT. Drivers written with DAA provide optimal performance on each platform while at the same time offering a common set of objects and interfaces that provide source code portability with no limitations or overhead.

Driver::Works requires Microsoft Visual C++ version 4.2 or later, and the Microsoft NT DDK, or the Windows 98 DDK. Driver::Works drivers have been tested on both Alpha and Intel single and dual processor platforms.

Driver::Works incorporates years of class-library design experience into a clean, object-oriented system that accurately reflects the underlying system architecture while avoiding the use of arcane C++ language features.

The Driver::Monitor tool, shown in Figure 3, provides a unique workbench for loading, testing, tracing, and unloading your device driver.



```
Time      Channel     Message text
          monitor     Reader thread started (channel 1 = Default)
          monitor     Channel 'Default' opened
          monitor     A new entry in the service database has been created for the driver.
          monitor     Select File | Start Driver to start the driver.
          monitor     Driver started successfully.
          monitor     Reader thread started (channel 2 = X75Passive)
          monitor     Channel 'X75Passive' opened
5.362114  X75Passive  Init: Entered DriverEntry: regpath=\REGISTRY\Machine\System\ControlSet001\Services\Output
5.362164  X75Passive  Init: Created device object, object at 0xc0941280
5.362305  X75Passive  Init: Attaching interrupt object, bus irql = 0x7
5.362451  X75Passive  Init: Opening I/O range 1
5.362573  X75Passive  Init: Opening memory range, system address=0xf0735000
5.362739  X75Passive  Init: The driver is successfully initialized
          monitor     Driver stopped successfully.
          monitor     Driver's service database entry successfully removed.
          monitor      -- end --
```

Figure 3: Driver::Monitor Interface

Licensing

The version of Driver::Works included in the Xilinx PCI Design Kit is fully functional and includes all libraries and software. It is licensed for use in driver development and prototyping only. Vireo offers Xilinx PCI customers the opportunity to purchase a royalty-free distribution license. Contact Vireo for pricing and details.

Vireo provides free bug fixes available for immediate download. Timely new versions provide support for a new compiler versions, and operating system revisions. Vireo also provides new examples and bug fixes on a regular basis.

Technical support on this product is available *only* through Vireo Software Inc.

Notes:

VtoolsD Windows Device Driver Development Kit Version 3.0

May, 1998

Data Sheet

Vireo Software, Inc.

30 Monument Square, Suite 135
Concord, MA 01742 USA
Phone: +1 978-369-3380
Fax: +1 978-318-6946
E-mail: info@vireo.com
support@vireo.com
Website: http://www.vireo.com

Introduction

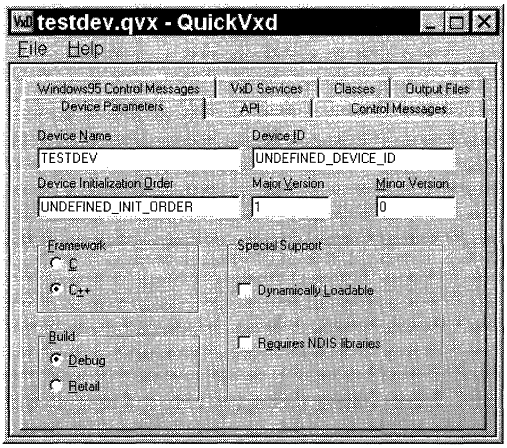
To facilitate rapid development of PCI Designs using the Xilinx LogiCORE PCI Interface, Vireo is providing the VtoolsD Windows Device Driver Development Kit. This kit includes an interactive GUI Wizard that allows the creation of a drive driver framework with a few simple selection and mouse clicks. Both the Microsoft Visual C++ 4.2 and later and the Borland C++ 4.x and later compilers are supported. Provided at no extra cost with the Xilinx PCI Design Kit, is a full-featured, fully functional version of VtoolsD licensed for prototyping drivers and testing them with the HotPCI board.

Support

Support for VtoolsD is provided only from Vireo. See Vireo's home page for contact information and other details.

Features

- Windows 95, 98, 3.X Support
- VtoolsD Interface
- Works with MS or Borland C/C++ compilers
- More than 50 sample drivers
  - Over 2 dozen example drivers written in C
  - Over 1 dozen example drivers written in C++
- Detailed on-line and printed documentation
- C and C++ system interfaces
- C Run Time Library
- C++ Class Library
- Includes complete source code for all libraries
- Thunks and wraps for every VMM/ VxD service and handler
- Microsoft DDK components bundled with VtoolsD for Windows 95
- Debug kernel executables and symbol tables
- WDEB386 system-level debugger for VxDs
- More than 1900 online help topic pages
- DDK documentation and help files
- Supports Driver Access Architecture (DAA)
- QuickVxD Wizard for quick device driver framework development.
- Driver Access Architecture (DAA) supports portability between Windows NT, Windows 95, Windows 98, and Win32 Driver Model (WDM)
- Complete access to over 900 interfaces from C/C++
- More than 60 classes designed for VxD operation
- More than 80 ANSI-compatible C Run Time Library functions
- Driver::Monitor™ - monitor driver activity without a debugger.
- QuickVxD source code generator - a VxD Wizard
- Microsoft and Borland compiler support
- Dynamic VxD Loader
- VxD Viewer



Device Name: TESTDEV

Device ID: UNDEFINED_DEVICE_ID

Device Initialization Order: UNDEFINED_INIT_ORDER

Major Version: 1

Minor Version: 0

Framework: ○ C ● C++

Build: ● Debug ○ Retail

Special Support: □ Dynamically Loadable □ Requires NDIS libraries

Figure 1: VtoolsD™ QuickVxD GUI

Description

VtoolsD is the easiest and fastest way to build Virtual Device Drivers (VxDs) for Microsoft Windows. Designed for both novice and experienced VxD developers, VtoolsD provides the comprehensive C or C++ solution for all VxD development challenges. Shipping since July 1994, VtoolsD is a mature, professional product used by thousands of developers world wide.

VtoolsD supports all of the system interfaces that the Microsoft DDK provides, plus an additional set of services provided by the VtoolsD libraries. VtoolsD can be used to write any kind of VxD, and makes that easier than it would be using the DDK.

Driver::Works implements Vireo's Device Access Architecture (DAA) interfaces. Using DAA, device driver source code can be easily ported between Windows 95, Windows 98, and all versions of Windows NT. Drivers written with DAA provide optimal performance on each platform while at the same time offering a common set of objects and interfaces that provide source code portability with no limitations or overhead.

The Driver::Monitor tool, shown in Figure 2, provides a unique workbench for loading, testing, tracing, and unloading your device driver.

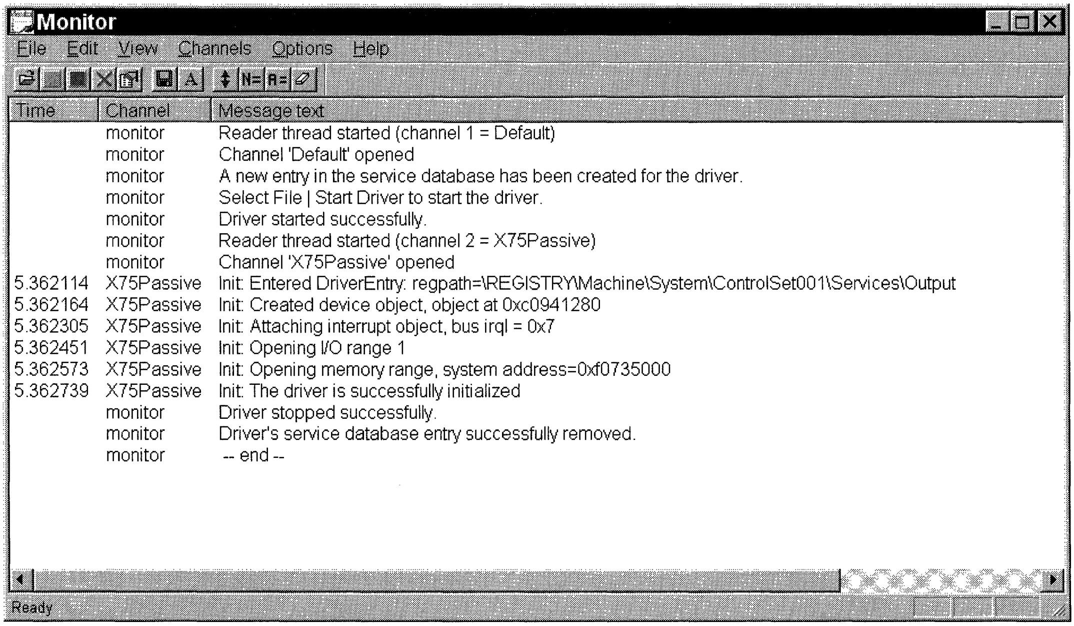
VtoolsD requires either MSVC++ 4.2 or later or the Borland C++ 4.X and later compilers. The Microsoft DDK is not required to use VtoolsD.

Licensing

The version of VtoolsD, included in the Xilinx PCI Design Kit, is fully functional and includes all libraries and software. It is licensed for use in driver development and prototyping only. Vireo offers Xilinx PCI customers the opportunity to purchase a royalty-free distribution license. Contact Vireo for pricing and details.

Vireo provides free bug fixes available for immediate download. Timely new versions provide support for a new compiler versions, and operating system revisions. Vireo also provides new examples and bug fixes on a regular basis.

Technical support on this product is available *only* through Vireo Software Inc.



```
Time       Channel      Message text
           monitor      Reader thread started (channel 1 = Default)
           monitor      Channel 'Default' opened
           monitor      A new entry in the service database has been created for the driver.
           monitor      Select File | Start Driver to start the driver.
           monitor      Driver started successfully.
           monitor      Reader thread started (channel 2 = X75Passive)
           monitor      Channel 'X75Passive' opened
5.362114   X75Passive   Init: Entered DriverEntry: regpath=\REGISTRY\Machine\System\ControlSet001\Services\Output
5.362164   X75Passive   Init: Created device object, object at 0xc0941280
5.362305   X75Passive   Init: Attaching interrupt object, bus irql = 0x7
5.362451   X75Passive   Init: Opening I/O range 1
5.362573   X75Passive   Init: Opening memory range, system address=0xf0735000
5.362739   X75Passive   Init: The driver is successfully initialized
           monitor      Driver stopped successfully.
           monitor      Driver's service database entry successfully removed.
           monitor       -- end --
```

Figure 2: Driver::Monitor Interface

XILINX®

FPGA Products

XILINX®

XC4000XLT Family
Field Programmable Gate Arrays

May, 1998

Data Sheet

XC4000XLT Features

Note: This data sheet describes the XC4000XLT Family devices. This information does not necessarily apply to the other Xilinx families: XC4000, XC4000A, XC4000D, XC4000H, XC4000L, XC4000E, XC4000EX, XC4000XL or XC4000XV. For information on these devices, or for the most current information regarding the XC4000XLT family, see the Xilinx WEBLINX at http://www.xilinx.com.

- System featured Field-Programmable Gate Arrays
  - Select-RAM™ memory: on-chip ultra-fast RAM with
    - synchronous write option
    - dual-port RAM option
  - Abundant flip-flops
  - Flexible function generators
  - Dedicated high-speed carry logic
  - Hierarchy of interconnect lines
  - Internal 3-state bus capability
  - 8 global low-skew clock or signal distribution networks
- System performance beyond 80 MHz
- Low power segmented routing architecture
- Systems-oriented features
  - IEEE 1149.1-compatible boundary scan logic support
  - Individually programmable output slew rate
  - Programmable input pull-up or pull-down resistors
- Configured by loading binary file
  - Unlimited reprogrammability
- Readback capability
  - Program verification
  - Internal node observability

XC4000XLT Family FPGAs

- **3.3V PCI Compliant**
  XC4000XLT devices provide PCI compliant I/O. They differ from XC4000XL devices only in that they enable the positive input signal clamping function required by PCI specifications.
- **New Packages enable Positive Signal Clamping**
  The XC4000XLT family of FPGAs is a new packaging option for the XC4000XL FPGAs. For XLT devices, Vtt, the positive clamping supply is made available to device pins. These Vtt pins replace 8 normal I/O pins. By connecting the Vtt pins to a positive power supply, the positive clamping diodes present in the IOBs are enabled.

XC4000XLT Electrical Features

- Low-Voltage Device Functions at Vcc=3.0 - 3.6 Volts
- Vtt supply allows positive signal clamping to Vtt +0.6 V
- Fully 3.3 V PCI compliant I/O (Vtt connected to 3.3V)
- 5.0 V PCI compatible I/O for embedded systems with 8 loads or less (Vtt connected to 5.0 V)
- 5.0 V TTL compatible I/O (Vtt connected to 5.0V)
- 3.3 V LVTTL, LVCMOS compatible I/O

Additional XC4000XLT Family Features

- **Highest Performance** — XC4000XL architecture
- **Highest Capacity** — Over 130,000 system gates
- **Low Power** — 3.3 V technology
- **Software Compatibility** — Bitstream compatible with XC4000XL devices
- **Package Compatibility** — Footprint compatible with XC4000XL devices (except for Vtt power pins)
- **Advanced Technology** — 0.35 micron CMOS process
- Buffered interconnect for maximum speed
- New latch capability in configurable logic blocks
- Improved VersaRing™ I/O interconnect for better fixed pinout flexibility
- Flexible high-speed clock network
  - 8 additional Early Buffers for shorter clock delays
  - Virtually unlimited number of clock signals
- Optional Multiplexer or 2-input function generator on device outputs
- 26 Address bits in master parallel configuration mode

PCI Compatible Features

- LogiCORE PCI32 4000 Interface available
  - 33 MHz 32-Bit PCI interface
  - Master or Target mode
  - Implemented entirely in programmable logic
  - Up to 124,000 gates available for user logic

- Fully compliant 3.3 V PCI I/O
  - < 7 nsec input setup time
  - 0 nsec input hold time
  - < 11 nsec clock to output
  - Positive and negative input signal clamping
  - Meets 5.0 V PCI timing for up to 8 loads
    - 80 mA sink current at minimum AC drive point (2.2V)

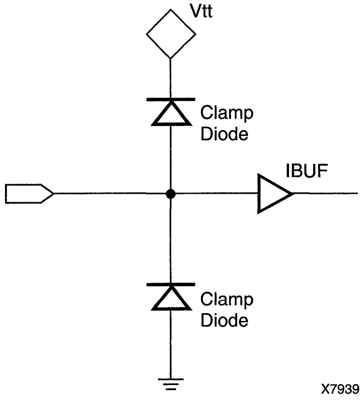


Figure 1: Clamp Diodes Present in the XL, XLT IOBs

PCI Requirements for Clamp Diodes

Clamp diodes are electrical protection devices placed in the I/O buffer of a chip. Both 5 V PCI and 3.3 V PCI signalling environments require clamp diodes to ground, which all Xilinx 4K family devices have. The 3.3 V PCI specification also requires clamp diodes to 3.3 V. The clamp diode serves two purposes. It offers device protection and it controls the bus waveforms as signals are transitioning on the bus. The latter function is vital to the signal integrity of the bus and is why clamp diodes are mandatory in a 3.3 V PCI system.

5.0 V PCI Requirement for Maximum AC Ratings and Device Protection

The upper clamp diode is optional in 5 V systems. For 5 V signaling, the PCI specification simply requires the devices be able to withstand a maximum overshoot voltage of 11 V for a minimum of 11 nsec through a 55 ohm resistor. See the PCI Specification v2.1, p126 for more details on this particular test. XC4000XL/XLT devices have a maximum input voltage requirement of 7.0 V (for < 10 nsec). In order to meet the PCI test requirements and provide device protection, it is necessary to connect the Vtt pins to the 5.0 V power supply.

Difference between the XC4000XLT and XC4000XL FPGAs

The only difference between XLT and XL devices is that in XLT devices, the Vtt supply is connected to package pins. By connecting the Vtt supply pins to a positive voltage, positive input signal clamping is enabled. The Vtt pins assigned to the Vtt supply are named in the pinout guide for the XC4013XLT, XC4028XLT, and XC4062XLT FPGAs. There are 8 Vtt pins in all package options.

I/O signaling compliance

The I/O signaling compliance is a function of how the Vtt pins are connected. Connecting Vtt to a power supply programs the compliance for all the IOBs on the device. All 8 of the Vtt pins must be connected to the same voltage source.

- **Vtt floating**
  When Vtt is left floating, the I/O characteristics of the XLT devices will be identical to XL devices. I/O will be LVTTL and LVCMOS compatible.

- **Vtt connected to 5.0V power**.
  If Vtt is connected to the 5.0 V power supply, the XLT device will be TTL, LVTTL, LVCMOS and 5V-PCI compatible for up to 8 PCI loads.

- **Vtt connected to 3.3V power**
  If Vtt is connected to 3.3V power, the I/O will be LVTTL, LVCMOS, and 3.3V-PCI compliant. Note that 5V TTL and 5V CMOS is not allowed.

Table 1: XC4000XLT Family Field Programmable Gate Arrays

Device	Logic Cells	Max Logic Gates (No RAM)	Max. RAM Bits (No Logic)	Typical Gate Range (Logic and RAM)*	CLB Matrix	Total CLBs	Number of Flip-Flops	Max. User I/O
XC4013XLT	1368	13,000	18,432	10,000 - 30,000	24 x 24	576	1,536	184
XC4028XLT	2432	28,000	32,768	18,000 - 50,000	32 x 32	1,024	2,560	185
XC4062XLT	5472	62,000	73,728	40,000 - 130,000	48 x 48	2,304	5,376	352

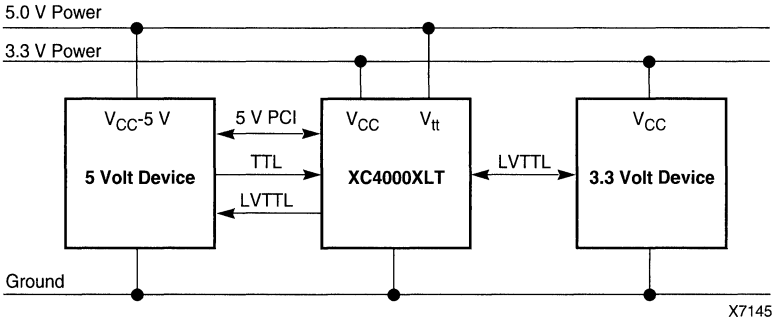


Figure 2: XLT Power supply and signaling environment with Vtt connected to 5.0 V power supply

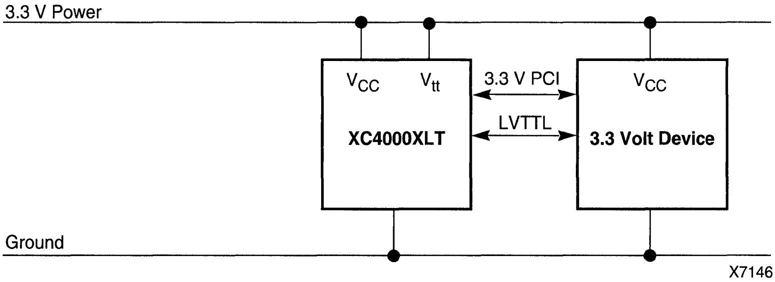


Figure 3: XLT Power supply and signaling environment with Vtt connected to 3.3V power supply

Pin Locations for XC4013XLT Devices

XC4013XLT Pad Name	PQ208	PQ240
VCC	P183	P212
I/O (A8)	P184	P213
I/O (A9)	P185	P214
I/O (A19)	P186	P215
I/O (A18)	P187	P216
I/O	P188	P217
I/O	P189	P218
I/O (A10)	P190	P220
I/O (A11)	P191	P221
VCC	-	P222
VTT	192	P223
I/O	-	P224
I/O	-	P225
I/O	P193	P226
GND	P194	P227
I/O	P195	P228
I/O	P196	P229
I/O	P197	P230
I/O	P198	P231
I/O (A12)	P199	P232
I/O (A13)	P200	P233
I/O	-	P234
I/O	-	P235
I/O	P201	P236
I/O	P202	P237
I/O (A14)	P203	P238
I/O, GCK8 (A15)	P204	P239
VCC	P205	P240
GND	P2	P1
I/O, GCK1 (A16)	P4	P2
I/O (A17)	P5	P3
I/O	P6	P4
I/O	P7	P5
I/O, TDI	P8	P6
I/O, TCK	P9	P7
I/O	P10	P8
I/O	P11	P9
I/O	P12	P10
I/O	P13	P11
I/O	-	P12
I/O	-	P13
GND	P14	P14
I/O	P15	P15
I/O	P16	P16
I/O, TMS	P17	P17
VTT	P18	P18
VCC	-	P19
I/O	-	P20
I/O	-	P21
I/O	P19	P23
I/O	P20	P24
I/O	P21	P25
I/O	P22	P26
I/O	P23	P27
I/O	P24	P28
GND	P25	P29
VCC	P26	P30
I/O	P27	P31
I/O	P28	P32
I/O	P29	P33
I/O	P30	P34
I/O	P31	P35
I/O	P32	P36
I/O	-	P38
I/O	-	P39
VCC	-	P40
VTT	P33	P41
I/O	P34	P42
I/O	P35	P43
I/O	P36	P44
GND	P37	P45
I/O	-	P46
I/O	-	P47
I/O	P38	P48
I/O	P39	P49
I/O	P40	P50
I/O	P41	P51
I/O	P42	P52
I/O	P43	P53
I/O	P44	P54
I/O	P45	P55
I/O	P46	P56
I/O, GCK2	P47	P57
O (M1)	P48	P58
GND	P49	P59
I (M0)	P50	P60
VCC	P55	P61
I (M2)	P56	P62
I/O, GCK3	P57	P63
I/O (HDC)	P58	P64
I/O	P59	P65
I/O	P60	P66
I/O	P61	P67
I/O (LDC)	P62	P68
I/O	P63	P69
I/O	P64	P70
I/O	P65	P71
I/O	P66	P72
I/O	-	P73
I/O	-	P74
GND	P67	P75
I/O	P68	P76
I/O	P69	P77
I/O	P70	P78
VTT	P71	P79
VCC	-	P80
I/O	P72	P81
I/O	P73	P82
I/O	-	P84
I/O	-	P85
I/O	P74	P86
I/O	P75	P87
I/O	P76	P88
I/O (INIT)	P77	P89
VCC	P78	P90
GND	P79	P91
I/O	P80	P92
I/O	P81	P93
I/O	P82	P94
I/O	P83	P95
I/O	P84	P96
I/O	P85	P97
I/O	-	P99
I/O	-	P100
VCC	-	P101
VTT	P86	P102
I/O	P87	P103
I/O	P88	P104
I/O	P89	P105
GND	P90	P106
I/O	-	P107
I/O	-	P108
I/O	P91	P109
I/O	P92	P110
I/O	P93	P111
I/O	P94	P112
I/O	P95	P113
I/O	P96	P114
I/O	P97	P115
I/O	P98	P116
I/O	P99	P117
I/O, GCK4	P100	P118
GND	P101	P119
DONE	P103	P120
VCC	P106	P121
PROGRAM	P108	P122

XC4013XLT Pad Name	PQ208	PQ240
I/O (D7)	P109	P123
I/O, GCK5	P110	P124
I/O	P111	P125
I/O	P112	P126
I/O	-	P127
I/O	-	P128
I/O (D6)	P113	P129
I/O	P114	P130
I/O	P115	P131
I/O	P116	P132
I/O	P117	P133
I/O	P118	P134
GND	P119	P135
I/O	-	P136
I/O	-	P137
I/O	P120	P138
VTT	P121	P139
VCC	-	P140
I/O (D5)	P122	P141
I/O (CS0)	P123	P142
I/O	P124	P144
I/O	P125	P145
I/O	P126	P146
I/O	P127	P147
I/O (D4)	P128	P148
I/O	P129	P149
VCC	P130	P150
GND	P131	P151
I/O (D3)	P132	P152
I/O (RS)	P133	P153
I/O	P134	P154
I/O	P135	P155
I/O	P136	P156
I/O	P137	P157
I/O (D2)	P138	P159
I/O	P139	P160
VCC	-	P161
VTT	P140	P162
I/O	P141	P163
I/O	-	P164
I/O	-	P165
GND	P142	P166
I/O	-	P167
I/O	-	P168
I/O	P143	P169
I/O	P144	P170
I/O	P145	P171
I/O	P146	P172
I/O (D1)	P147	P173
I/O (RCLK, RDY/BUSY)	P148	P174
I/O	P149	P175
I/O	P150	P176
I/O (D0, DIN)	P151	P177
I/O, SGCK4 †, GCK6 †† (DOUT)	P152	P178
CCLK	P153	P179
VCC	P154	P180
O, TDO	P159	P181
GND	P160	P182
I/O (A0, WS)	P161	P183
I/O, GCK7 (A1)	P162	P184
I/O	P163	P185
I/O	P164	P186
I/O (CS1, A2)	P165	P187

XC4013XLT Pad Name	PQ208	PQ240
I/O (A3)	P166	P188
I/O	-	P189
I/O	-	P190
I/O	P167	P191
I/O	P168	P192
I/O	P169	P193
I/O	P170	P194
GND	P171	P196
I/O	P172	P197
I/O	-	P198
I/O	-	P199
VTT	173	P200
VCC	-	P201
I/O (A4)	P174	P202
I/O (A5)	P175	P203
I/O	P176	P205
I/O	P177	P206
I/O (A21)	P178	P207
I/O (A20)	P179	P208
I/O (A6)	P180	P209
I/O (A7)	P181	P210
GND	P182	P211

8/21/1997

* Pads labelled GND* or VCC* are internally bonded to Ground or VCC planes within the package. They have no direct connection to any specific package pin.

Additional XC4013XLT Package Pins

PQ208

N.C. Pins					
P1	P3	P51	P52	P53	P54
P102	P104	P105	P107	P155	P156
P157	P158	P206	P207	P208	-

5/5/97

PQ240

GND Pins					
P22‡	P37‡	P83‡	P98‡	P143‡	P158‡
P204‡	P219‡	-	-	-	-
N.C. Pins					
P195	-	-	-	-	-

6/9/97

‡ Pins marked with this symbol are used for Ground connections on some revisions of the device. These pins may not physically connect to anything on the current device revision. However, they should be externally connected to Ground, if possible.

Pin Locations for XC4028XLT Devices

XC4028XLT Pad Name	HQ240
VCC	P212
I/O (A8)	P213
I/O (A9)	P214
I/O (A19)	P215
I/O (A18)	P216
I/O	P217
I/O	P218
I/O (A10)	P220
I/O (A11)	P221
GND	-
I/O	-
I/O	-
I/O	-
I/O	-
VCC	P222
VTT	P223
I/O	P224
I/O	P225
I/O	P226
GND	P227
I/O	-
I/O	-
I/O	P228
I/O	P229
I/O	P230
I/O	P231
I/O (A12)	P232
I/O (A13)	P233
GND	-
VCC	-
I/O	-
I/O	-
I/O	P234
I/O	P235
I/O	P236
I/O	P237
I/O (A14)	P238
I/O, GCK8, (A15)	P239
VCC	P240
GND	P1
I/O, GCK1, (A16)	P2
I/O (A17)	P3
I/O	P4
I/O	P5
I/O, TDI	P6
I/O, TCK	P7
I/O	-
I/O	-
VCC	-
GND	-
I/O	P8
I/O	P9
I/O	P10
I/O	P11
I/O	P12
I/O	P13
I/O	-
I/O	-
GND	P14
I/O	P15
I/O	P16
I/O, TMS	P17
VTT	P18
VCC	P19
I/O	P20
I/O	P21
I/O	-
I/O	-
GND	P22
I/O	-
I/O	-
I/O	P23
I/O	P24
I/O	P25
I/O	P26
I/O	P27
I/O	P28
GND	P29
VCC	P30
I/O	P31
I/O	P32
I/O	P33
I/O	P34
I/O	P35
I/O	P36
I/O	-
I/O	-
GND	P37
I/O	-
I/O	-
I/O	P38
I/O	P39
VCC	P40
VTT	P41
I/O	P42
I/O	P43
I/O	P44
GND	P45
I/O	-
I/O	-
I/O	P46
I/O	P47
I/O	P48
I/O	P49
I/O	P50
I/O	P51
GND	-
VCC	-
I/O	-
I/O	-
I/O	P52
I/O	P53
I/O	P54
I/O	P55
I/O	P56
I/O, GCK2	P57
O (M1)	P58
GND	P59
I (M0)	P60
VCC	P61
I (M2)	P62
I/O, GCK3	P63
I/O (HDC)	P64
I/O	P65
I/O	P66
I/O	P67
I/O ($\overline{\text{LDC}}$)	P68
I/O	-
I/O	-
VCC	-
GND	-
I/O	P69
I/O	P70
I/O	P71
I/O	P72
I/O	P73
I/O	P74

XC4028XLT Pad Name	HQ240
I/O	-
I/O	-
GND	P75
I/O	P76
I/O	P77
I/O	P78
VTT	P79
VCC	P80
I/O	P81
I/O	P82
I/O	-
I/O	-
GND	P83
I/O	-
I/O	-
I/O	P84
I/O	P85
I/O	P86
I/O	P87
I/O	P88
I/O (INIT)	P89
VCC	P90
GND	P91
I/O	P92
I/O	P93
I/O	P94
I/O	P95
I/O	P96
I/O	P97
I/O	-
I/O	-
GND	P98
I/O	-
I/O	-
I/O	P99
I/O	P100
VCC	P101
VTT	P102
I/O	P103
I/O	P104
I/O	P105
GND	P106
I/O	-
I/O	-
I/O	P107
I/O	P108
I/O	P109
I/O	P110
I/O	P111
I/O	P112
GND	-
VCC	-
I/O	-
I/O	-
I/O	P113
I/O	P114
I/O	P115
I/O	P116
I/O	P117
I/O, GCK4	P118
GND	P119
DONE	P120
VCC	P121
PROGRAM	P122
I/O (D7)	P123
I/O, GCK5	P124
I/O	P125
I/O	P126
I/O	P127
I/O	P128
I/O	-
I/O	-
VCC	-
GND	-
I/O (D6)	P129
I/O	P130
I/O	P131
I/O	P132
I/O	P133
I/O	P134
I/O	-
I/O	-
GND	P135
I/O	P136
I/O	P137
I/O	P138
VTT	P139
VCC	P140
I/O (D5)	P141
I/O (CS0)	P142
I/O	-
I/O	-
GND	P143
I/O	-
I/O	-
I/O	P144
I/O	P145
I/O	P146
I/O	P147
I/O (D4)	P148
I/O	P149
VCC	P150
GND	P151
I/O (D3)	P152
I/O (RS)	P153
I/O	P154
I/O	P155
I/O	P156
I/O	P157
I/O	-
I/O	-
GND	P158
I/O	-
I/O	-
I/O (D2)	P159
I/O	P160
VCC	P161
VTT	P162
I/O	P163
I/O	P164
I/O	P165
GND	P166
I/O	-
I/O	-
I/O	P167
I/O	P168
I/O	P169
I/O	P170
I/O	P171
I/O	P172
GND	-
VCC	-
I/O (D1)	P173
I/O (RCLK,RDY/BUSY)	P174
I/O	-
I/O	-
I/O	P175
I/O	P176
I/O (D0, DIN)	P177
I/O, GCK6 (DOUT)	P178

XC4028XLT Pad Name	HQ240
CCLK	P179
VCC	P180
O, TDO	P181
GND	P182
I/O (A0, $\overline{\text{WS}}$)	P183
I/O, GCK7 (A1)	P184
I/O	P185
I/O	P186
I/O (CS1, A2)	P187
I/O (A3)	P188
I/O	-
I/O	-
VCC	-
GND	-
I/O	P189
I/O	P190
I/O	P191
I/O	P192
I/O	P193
I/O	P194
I/O	P195
I/O	-
GND	P196
I/O	P197
I/O	P198
I/O	P199
VTT	P200
VCC	P201
I/O	-
I/O	-
I/O	-
I/O	-
GND	-
I/O (A4)	P202
I/O (A5)	P203
I/O	P205
I/O	P206
I/O (A21)	P207
I/O (A20)	P208
I/O (A6)	P209
I/O (A7)	P210
GND	P211

8/21/1997

Additional XC4028XLT Package Pins

HQ240

GND Pins					
P204	P219	-	-	-	-

10/7/97

Note: These pins may be N.C. for this device revision, however for compatibility with other devices in this package, these pins should be tied to GND.

Pin Locations for XC4062XLT Devices

XC4062XLT Pad Name	HQ240	BG432
VCC	P212	VCC*
I/O (A8)	P213	D17
I/O (A9)	P214	A17
I/O	-	C17
I/O	-	B17
I/O	-	-
I/O	-	-
GND	-	GND*
I/O (A19)	P215	C18
I/O (A18)	P216	D18
I/O	P217	B18
I/O	P218	A19
I/O (A10)	P220	B19
I/O (A11)	P221	C19
VCC	-	VCC*
GND	-	GND*
I/O	-	D19
I/O	-	A20
I/O	-	B20
I/O	-	C20
I/O	-	B21
I/O	-	D20
GND	-	GND*
I/O	-	C21
I/O	-	A22
VCC	P222	VCC*
VTT	P223	B22
I/O	P224	C22
I/O	P225	B23
I/O	P226	A24
GND	P227	GND*
I/O	-	D22
I/O	-	C23
I/O	P228	B24
I/O	P229	C24
I/O	-	-
I/O	-	-
GND	-	GND*
I/O	-	D23
I/O	-	B25
I/O	P230	A26
I/O	P231	C25
I/O (A12)	P232	D24
I/O (A13)	P233	B26
GND	-	GND*
VCC	-	VCC*
I/O	-	A27
I/O	-	D25
I/O	-	C26
I/O	-	B27
I/O	-	A28
I/O	-	D26
GND	-	GND*
I/O	P234	C27
I/O	P235	B28
I/O	P236	D27
I/O	P237	B29
I/O (A14)	P238	C28
I/O, GCK8 (A15)	P239	D28
VCC	P240	VCC*
GND	P1	GND*
I/O, GCK1 (A16)	P2	D29
I/O (A17)	P3	C30
I/O	P4	E28
I/O	P5	E29
I/O, TDI	P6	D30
I/O, TCK	P7	D31
GND	-	GND*
I/O	-	F28
I/O	-	F29
I/O	-	E30
I/O	-	E31
I/O	-	G28
I/O	-	G29
VCC	-	VCC*
GND	-	GND*
I/O	-	F30
I/O	-	F31
I/O	P8	H28
I/O	P9	H29
I/O	P10	G30
I/O	P11	H30
GND	-	GND*
I/O	-	-
I/O	-	-
I/O	P12	J28
I/O	P13	J29
I/O	-	H31
I/O	-	J30
GND	P14	GND*
I/O	P15	K28
I/O	P16	K29
I/O, TMS	P17	K30
VTT	P18	K31
VCC	P19	VCC*
I/O	P20	L29
I/O	P21	L30
GND	-	GND*
I/O	-	M30
I/O	-	M28
I/O	-	M29
I/O	-	M31
I/O	-	N31
I/O	-	N28
GND	P22	GND*
VCC	-	VCC*
I/O	-	N29
I/O	-	N30
I/O	-	P30
I/O	-	P28
I/O	P23	P29
I/O	P24	R31
GND	-	GND*
I/O	P25	R30
I/O	P26	R28
I/O	-	-
I/O	-	-
I/O	P27	R29
I/O	P28	T31
GND	P29	GND*
VCC	P30	VCC*
I/O	P31	T30
I/O	P32	T29
I/O	-	-
I/O	-	-
I/O	P33	U31
I/O	P34	U30
GND	-	-
I/O	P35	U28
I/O	P36	U29
I/O	-	V30
I/O	-	V29
I/O	-	V28
I/O	-	W31
VCC	-	VCC*
GND	P37	GND*
I/O	-	W30

XC4062XLT Pad Name	HQ240	BG432
I/O	-	W29
I/O	-	W28
I/O	-	Y31
I/O	-	Y30
I/O	-	Y29
GND	-	GND*
I/O	P38	Y28
I/O	P39	AA30
VCC	P40	VCC*
VTT	P41	AA29
I/O	P42	AB31
I/O	P43	AB30
I/O	P44	AB29
GND	P45	GND*
I/O	-	AB28
I/O	-	AC30
I/O	P46	AC29
I/O	P47	AC28
I/O	-	-
I/O	-	-
GND	-	GND*
I/O	-	AD31
I/O	-	AD30
I/O	P48	AD29
I/O	P49	AD28
I/O	P50	AE30
I/O	P51	AE29
GND	-	GND*
VCC	-	VCC*
I/O	-	AF31
I/O	-	AE28
I/O	-	AF30
I/O	-	AF29
I/O	P52	AG31
I/O	P53	AF28
GND	-	GND*
I/O	-	AG30
I/O	-	AG29
I/O	P54	AH31
I/O	P55	AG28
I/O	P56	AH30
I/O, GCK2	P57	AJ30
O (M1)	P58	AH29
GND	P59	GND*
I (M0)	P60	AH28
VCC	P61	VCC*
I (M2)	P62	AJ28
I/O, GCK3	P63	AK29
I/O (HDC)	P64	AH27
I/O	P65	AK28
I/O	P66	AJ27
I/O	P67	AL28
I/O ($\overline{\text{LDC}}$)	P68	AH26
GND	-	GND*
I/O	-	AK27
I/O	-	AJ26
I/O	-	AL27
I/O	-	AH25
I/O	-	AK26
I/O	-	AL26
VCC	-	VCC*
GND	-	GND*
I/O	P69	AH24
I/O	P70	AJ25
I/O	P71	AK25
I/O	P72	AJ24
I/O	-	AH23
I/O	-	AK24
GND	-	GND*
I/O	-	-
I/O	-	-
I/O	P73	AL24
I/O	P74	AH22
I/O	-	AJ23
I/O	-	AK23
GND	P75	GND*
I/O	P76	AJ22
I/O	P77	AK22
I/O	P78	AL22
VTT	P79	AJ21
VCC	P80	VCC*
I/O	P81	AH20
I/O	P82	AK21
GND	-	GND*
I/O	-	AJ20
I/O	-	AH19
I/O	-	AK20
I/O	-	AJ19
I/O	-	AL20
I/O	-	AH18
GND	P83	GND*
VCC	-	VCC*
I/O	-	AK19
I//O	-	AJ18
I/O	P84	AL19
I/O	P85	AK18
I/O	P86	AH17
I/O	P87	AJ17
GND	-	GND*
I/O	-	-
I/O	-	-
I/O	-	AK17
I/O	-	AL17
I/O	P88	AJ16
I/O ($\overline{\text{INIT}}$)	P89	AK16
VCC	P90	VCC*
GND	P91	GND*
I/O	P92	AL16
I/O	P93	AH15
I/O	-	AL15
I/O	-	AJ15
I/O	-	-
I/O	-	-
GND	-	GND*
I/O	P94	AK15
I/O	P95	AJ14
I/O	P96	AH14
I/O	P97	AK14
I/O	-	AL13
I/O	-	AK13
VCC	-	VCC*
GND	P98	GND*
I/O	-	AJ13
I/O	-	AH13
I/O	-	AL12
I/O	-	AK12
I/O	-	AJ12
I/O	-	AK11
GND	-	GND*
I/O	P99	AH12
I/O	P100	AJ11
VCC	P101	VCC*
VTT	P102	AL10
I/O	P103	AK10
I/O	P104	AJ10
I/O	P105	AK9
GND	P106	GND*
I/O	-	AL8
I/O	-	AH10
I/O	P107	AJ9

XC4062XLT Pad Name	HQ240	BG432
I/O	P108	AK8
I/O	-	-
I/O	-	-
GND	-	GND*
I/O	-	AJ8
I/O	-	AH9
I/O	P109	AK7
I/O	P110	AL6
I/O	P111	AJ7
I/O	P112	AH8
GND	-	GND*
VCC	-	VCC*
I/O	-	AK6
I/O	-	AL5
I/O	P113	AH7
I/O	P114	AJ6
I/O	-	AK5
I/O	-	AL4
GND	-	GND*
I/O	-	AH6
I/O	-	AJ5
I/O	P115	AK4
I/O	P116	AH5
I/O	P117	AK3
I/O, GCK4	P118	AJ4
GND	P119	GND*
DONE	P120	AH4
VCC	P121	VCC*
PROGRAM	P122	AH3
I/O (D7)	P123	AJ2
I/O, GCK5	P124	AG4
I/O	P125	AG3
I/O	P126	AH2
I/O	-	AH1
I/O	-	AF4
GND	-	GND*
I/O	P127	AF3
I/O	P128	AG2
I/O	-	AG1
I/O	-	AE4
I/O	-	AE3
I/O	-	AF2
VCC	-	VCC*
GND	-	GND*
I/O (D6)	P129	AF1
I/O	P130	AD4
I/O	P131	AD3
I/O	P132	AE2
I/O	-	AD2
I/O	-	AC4
GND	-	GND*
I/O	-	-
I/O	-	-
I/O	P133	AC3
I/O	P134	AD1
I/O	-	AC2
I/O	-	AB4
GND	P135	GND*
I/O	P136	AB3
I/O	P137	AB2
I/O	P138	AB1
VTT	P139	AA3
VCC	P140	VCC*
I/O (D5)	P141	AA2
I/O (CS0)	P142	Y2
GND	P143	GND*
I/O	-	Y4
I/O	-	Y3
I/O	-	Y1
I/O	-	W1
I/O	-	W4
I/O	-	W3
GND	-	GND*
VCC	-	VCC*
I/O	-	W2
I/O	-	V2
I/O	-	V4
I/O	-	V3
I/O	P144	U1
I/O	P145	U2
GND	-	GND*
I/O	P146	U4
I/O	P147	U3
I/O	-	-
I/O	-	-
I/O (D4)	P148	T1
I/O	P149	T2
VCC	P150	VCC*
GND	P151	GND*
I/O (D3)	P152	T3
I/O (RS)	P153	R1
I/O	-	-
I/O	-	-
I/O	P154	R2
I/O	P155	R4
GND	-	GND*
I/O	P156	R3
I/O	P157	P2
I/O	-	P3
I/O	-	P4
I/O	-	N1
I/O	-	N2
VCC	-	VCC*
GND	P158	GND*
I/O	-	N3
I/O	-	N4
I/O	-	M1
I/O	-	M2
I/O	-	M3
I/O	-	M4
GND	-	GND*
I/O (D2)	P159	L2
I/O	P160	L3
VCC	P161	VCC*
VTT	P162	K1
I/O	P163	K2
I/O	P164	K3
I/O	P165	K4
GND	P166	GND*
I/O	-	J2
I/O	-	J3
I/O	P167	J4
I/O	P168	H1
I/O	-	-
I/O	-	-
GND	-	GND*
I/O	P169	H2
I/O	P170	H3
I/O	P171	H4
I/O	P172	G2
I/O	-	G3
I/O	-	F1
GND	-	GND*
VCC	-	VCC*
I/O (D1)	P173	G4
I/O (RCLK, RDY/BUSY)	P174	F2
I/O	-	F3
I/O	-	E1
I/O	-	F4
I/O	-	E2

XC4062XLT Pad Name	HQ240	BG432
GND	-	GND*
I/O	-	E3
I/O	-	D1
I/O	P175	E4
I/O	P176	D2
I/O (D0, DIN)	P177	C2
I/O, GCK6 (DOUT)	P178	D3
CCLK	P179	D4
VCC	P180	VCC*
O, TDO	P181	C4
GND	P182	GND*
I/O (A0, $\overline{\text{WS}}$)	P183	B3
I/O, GCK7 (A1)	P184	D5
I/O	P185	B4
I/O	P186	C5
I/O	-	A4
I/O	-	D6
GND	-	GND*
I/O	-	B5
I/O	-	C6
I/O (CS1, A2)	P187	A5
I/O (A3)	P188	D7
I/O	-	B6
I/O	-	A6
VCC	-	VCC*
GND	-	GND*
I/O	P189	D8
I/O	P190	C7
I/O	P191	B7
I/O	P192	D9
I/O	-	B8
I/O	-	A8
GND	-	GND*
I/O	-	-
I/O	-	-
I/O	P193	D10
I/O	P194	C9
I/O	P195	B9
I/O	-	C10
GND	P196	GND*
I/O	P197	B10
I/O	P198	A10
I/O	P199	C11
VTT	P200	D12
VCC	P201	VCC*
I/O	-	B11
I/O	-	C12
GND	-	GND*
I/O	-	D13
I/O	-	B12
I/O	-	C13
I/O	-	A12
I/O	-	D14
I/O	-	B13
GND	-	GND*
VCC	-	VCC*
I/O (A4)	P202	C14
I/O (A5)	P203	A13
I/O	P205	B14
I/O	P206	D15
I/O (A21)	P207	C15
I/O (A20)	P208	B15
GND	-	GND*
I/O	-	-
I/O	-	-
I/O	-	A15
I/O	-	C16
I/O (A6)	P209	B16
I/O (A7)	P210	A16
GND	P211	GND*

8/21/1997

* Pads labelled GND* or VCC* are internally bonded to Ground or VCC planes within the package. They have no direct connection to any specific package pin.

Additional XC4062XLT Package Pins

HQ240

GND Pins					
P204	P219	-	-	-	-

5/5/97

Note: These pins may be N.C. for this device revision, however for compatibility with other devices in this package, these pins should be tied to GND.

BG432

VCC Pins						
A1	A11	A21	A31	C3	C29	D11
D21	L1	L4	L28	L31	AA1	AA4
AA28	AA31	AH11	AH21	AJ3	AJ29	AL1
AL11	AL21	AL31	-	-	-	-
GND Pins						
A2	A3	A7	A9	A14	A18	A23
A25	A29	A30	B1	B2	B30	B31
C1	C31	D16	G1	G31	J1	J31
P1	P31	T4	T28	V1	V31	AC1
AC31	AE1	AE31	AH16	AJ1	AJ31	AK1
AK2	AK30	AK31	AL2	AL3	AL7	AL9
AL14	AL18	AL23	AL25	AL29	AL30	-
N.C. Pins						
C8	-	-	-	-	-	-

5/5/97

XILINX®

Spartan and Spartan-XL Families Field Programmable Gate Arrays

May, 1998

Advance Product Specification

Introduction

The Spartan™ Series is the first high-volume production FPGA solution to deliver all the key requirements for ASIC replacement up to 40,000 gates. These requirements include high performance, on-chip RAM, Core Solutions and prices that, in high volume, approach and in many cases are equivalent to mask programmed ASIC devices.

The Spartan series is the result of more than thirteen years of FPGA design experience and feedback from thousands of customers. By streamlining the Spartan feature set, leveraging advanced hybrid process technologies and focusing on total cost management, the Spartan series delivers the key features required by ASIC and other high volume logic users while avoiding the initial cost, long development cycles and inherent risk of conventional ASICs.

The Xilinx LogiCORE PCI32 Spartan Interface will support the XCS30 and XCS40 devices, as shown in Table 1.

Spartan Series Features

Note: The Spartan series devices described in this data sheet include the Spartan™ family of devices and the Spartan-XL™ family of devices.

- Next generation ASIC replacement technology
  - First ASIC replacement FPGA for high-volume production with on-chip RAM
  - Advanced Ultradense™ 0.35µm/0.50µm process
  - Density up to 1862 logic cells or 40,000 system gates
  - Streamlined feature set based on XC4000 architecture
  - System performance beyond 80 MHz
  - Broad set of AllianceCORE™ and LogiCORE™ solutions available
  - Unlimited reprogrammability
- System level features
  - Available in both 5.0 Volt and 3.3 Volt versions
  - On-chip Select-RAM™ memory
  - Fully PCI compliant
  - Low power segmented routing architecture
  - Full readback capability for program verification and internal node observability
  - Dedicated high-speed carry logic
  - Internal 3-state bus capability
  - 8 global low-skew clock or signal distribution networks
  - IEEE 1149.1-compatible boundary scan logic support
- Versatile I/O and packaging
  - Low cost plastic packages available in all densities
  - Footprint compatibility in common packages across all Spartan and Spartan-XL devices
  - Individually programmable output slew-rate control maximizes performance and reduces noise
  - Hold time of 0.0 ns for input registers simplifies system timing
  - 12-mA sink current per output
- Fully supported by powerful Xilinx development system
  - Foundation series: Fully integrated, shrink-wrap software
  - Alliance series: Over 100 PC and engineering workstation 3rd party development systems supported
  - Fully automatic mapping, placement and routing
  - Interactive design editor for design optimization

Table 1: LogiCORE PCI32 Spartan Supported Spartan and Spartan-XL[1] Series Field Programmable Gate Arrays

Device	Logic Cells	Max System Gates	Typical Gate Range (Logic and RAM)*	CLB Matrix	Total CLBs	Number of Flip-Flops	Max. User I/O
XCS30 & XCS30XL	1368	30,000	10,000 - 30,000	24 x 24	576	1,536	192
XCS40 & XCS40XL	1862	40,000	13,000 - 40,000	28 x 28	784	2,016	224

* Max values of Typical Gate Range include 20-30% of CLBs used as RAM.

[1] Spartan-XL devices and PCI32 Spartan support for Spartan-XL available later 1998

General Overview

Spartan Series FPGAs are implemented with a regular, flexible, programmable architecture of Configurable Logic Blocks (CLBs), interconnected by a powerful hierarchy of versatile routing resources (routing channels), and surrounded by a perimeter of programmable Input/Output Blocks (IOBs), as seen in Figure 1. They have generous routing resources to accommodate the most complex interconnect patterns.

The devices are customized by loading configuration data into internal static memory cells. Re-programming is possible an unlimited number of times. The values stored in these memory cells determine the logic functions and interconnections implemented in the FPGA. The FPGA can either actively read its configuration data from an external serial PROM (Master Serial mode), or the configuration data can be written into the FPGA from an external device (Slave Serial mode).

Spartan FPGAs can be used where hardware must be adapted to different user applications. FPGAs are ideal for shortening design and development cycles, and also offer a cost-effective solution for production rates well beyond 50,000 systems per month.

Spartan Series devices achieve high-performance, low-cost operation through the use of an advanced architecture and semiconductor technology. Spartan and Spartan-XL provide system clock rates exceeding 80 MHz and internal performance in excess of 150 MHz. In contrast to other FPGA devices, Spartan offers the most cost-effective solution while maintaining leading-edge performance. In addition to the conventional benefit of high volume programmable logic solutions Spartan also offers on-chip edge-triggered single-port and dual-port RAM, clock enables on all flip-flops, fast carry logic, and many other features.

The Spartan Series leverages the highly successful XC4000 architecture with many of that family's features and benefits. Technology advancements have been derived from the XC4000XL and XC4000XV process developments.

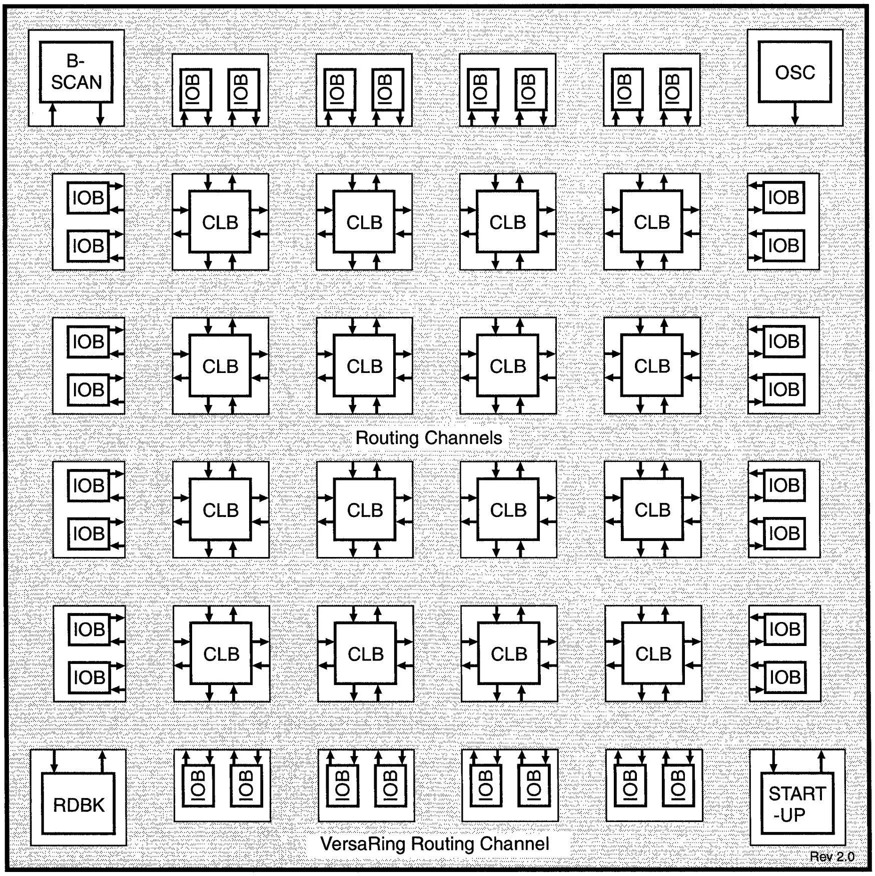


Figure 1: Basic FPGA Block Diagram

Device-Specific Pinout Tables

Device-specific tables include all packages for each PCI Supported Spartan and Spartan-XL device. They follow the pad locations around the die, and include boundary scan register locations.

Pin Locations for XCS30 & XCS30XL Devices

XCS30 & XCS30XL Pad Name	PQ208	PQ240
VCC	P183	P212
I/O	P184	P213
I/O	P185	P214
I/O	P186	P215
I/O	P187	P216
I/O	P188	P217
I/O	P189	P218
I/O	P190	P220
I/O	P191	P221
VCC	P192	P222
I/O	-	P223
I/O	-	P224
I/O	P193	P225
I/O	P194	P226
GND	P195	P227
I/O	P196	P228
I/O	P197	P229
I/O	P198	P230
I/O	P199	P231
I/O	P200	P232
I/O	P201	P233
I/O	P202	P234
I/O	P203	P235
I/O	P204	P236
I/O	P205	P237
I/O	P206	P238
I/O, SGCK1	P207	P239
VCC	P208	P240
GND	P1	P1
I/O, PGCK1	P2	P2
I/O	P3	P3
I/O	P4	P4
I/O	P5	P5
I/O, TDI	P6	P6
I/O, TCK	P7	P7
I/O	P8	P8
I/O	P9	P9
I/O	P10	P10
I/O	P11	P11
I/O	P12	P12
I/O	-	P13
GND	P13	P14
I/O	P14	P15
I/O	P15	P16
I/O, TMS	P16	P17
I/O	P17	P18
VCC	P18	P19
I/O	-	P20
I/O	-	P21
I/O	P19	P23
I/O	P20	P24
I/O	P21	P25
I/O	P22	P26
I/O	P23	P27
I/O	P24	P28
GND	P25	P29
VCC	P26	P30
I/O	P27	P31
I/O	P28	P32
I/O	P29	P33
I/O	P30	P34
I/O	P31	P35
I/O	P32	P36
I/O	-	P38
I/O	-	P39
VCC	P33	P40
I/O	P34	P41
I/O	P35	P42
I/O	P36	P43
I/O	P37	P44
GND	P38	P45
I/O	-	P46
I/O	P39	P47
I/O	P40	P48
I/O	P41	P49
I/O	P42	P50
I/O	P43	P51
I/O	P44	P52
I/O	P45	P53
I/O	P46	P54
I/O	P47	P55
I/O	P48	P56
I/O, SGCK2	P49	P57
Don't Connect	P50	P58
GND	P51	P59
MODE	P52	P60
VCC	P53	P61
Don't Connect	P54	P62
I/O, PGCK2	P55	P63
I/O (HDC)	P56	P64
I/O	P57	P65
I/O	P58	P66
I/O	P59	P67
I/O ($\overline{\text{LDC}}$)	P60	P68
I/O	P61	P69
I/O	P62	P70
I/O	P63	P71
I/O	P64	P72
I/O	P65	P73
I/O	-	P74
GND	P66	P75
I/O	P67	P76
I/O	P68	P77
I/O	P69	P78
I/O	P70	P79
VCC	P71	P80
I/O	P72	P81
I/O	P73	P82
I/O	-	P84
I/O	-	P85
I/O	P74	P86
I/O	P75	P87
I/O	P76	P88
I/O ($\overline{\text{INIT}}$)	P77	P89
VCC	P78	P90
GND	P79	P91
I/O	P80	P92
I/O	P81	P93
I/O	P82	P94
I/O	P83	P95
I/O	P84	P96
I/O	P85	P97
I/O	-	P99
I/O	-	P100

XCS30 & XCS30XL Pad Name	PQ208	PQ240
VCC	P86	P101
I/O	P87	P102
I/O	P88	P103
I/O	P89	P104
I/O	P90	P105
GND	P91	P106
I/O	-	P107
I/O	P92	P108
I/O	P93	P109
I/O	P94	P110
I/O	P95	P111
I/O	P96	P112
I/O	P97	P113
I/O	P98	P114
I/O	P99	P115
I/O	P100	P116
I/O	P101	P117
I/O, SGCK3	P102	P118
GND	P103	P119
DONE	P104	P120
VCC	P105	P121
PROGRAM	P106	P122
I/O	P107	P123
I/O, PGCK3	P108	P124
I/O	P109	P125
I/O	P110	P126
I/O	-	P127
I/O	P111	P128
I/O	P112	P129
I/O	P113	P130
I/O	P114	P131
I/O	P115	P132
I/O	P116	P133
I/O	P117	P134
GND	P118	P135
I/O	-	P136
I/O	-	P137
I/O	P119	P138
I/O	P120	P139
VCC	P121	P140
I/O	P122	P141
I/O	P123	P142
I/O	P124	P144
I/O	P125	P145
I/O	P126	P146
I/O	P127	P147
I/O	P128	P148
I/O	P129	P149
VCC	P130	P150
GND	P131	P151
I/O	P132	P152
I/O	P133	P153
I/O	P134	P154
I/O	P135	P155
I/O	P136	P156
I/O	P137	P157
I/O	P138	P159
I/O	P139	P160
VCC	P140	P161
I/O	P141	P162
I/O	P142	P163
I/O	-	P164
I/O	-	P165
GND	P143	P166
I/O	-	P167
I/O	P144	P168
I/O	P145	P169
I/O	P146	P170
I/O	P147	P171
I/O	P148	P172
I/O	P149	P173
I/O	P150	P174
I/O	P151	P175
I/O	P152	P176
I/O (DIN)	P153	P177
I/O, SGCK4 (DOUT)	P154	P178
CCLK	P155	P179
VCC	P156	P180
O, TDO	P157	P181
GND	P158	P182
I/O	P159	P183
I/O, PGCK4	P160	P184
I/O	P161	P185
I/O	P162	P186
I/O	P163	P187
I/O	P164	P188
I/O	P165	P189
I/O	-	P190
I/O	P166	P191
I/O	P167	P192
I/O	P168	P193
I/O	P169	P194
GND	P170	P196
I/O	P171	P197
I/O	P172	P198
I/O	-	P199
I/O	-	P200
VCC	P173	P201
I/O	P174	P202
I/O	P175	P203
I/O	P176	P205
I/O	P177	P206
I/O	P178	P207
I/O	P179	P208
I/O	P180	P209
I/O	P181	P210
GND	P182	P211

10/22/97

* Pads labelled GND* or VCC* are internally bonded to Ground or VCC planes within the package. They have no direct connection to any specific package pin.

Additional XCS30/XL Package Pins

PQ240

GND Pins					
P22‡	P37‡	P83‡	P98‡	P143‡	P158‡
P204‡	P219‡	-	-	-	-
Not Connected Pins					
P195	-	-	-	-	-

9/24/97

‡ Pins marked with this symbol are used for Ground connections on some revisions of the device. These pins may not physically connect to anything on the current device revision. However, they should be externally connected to Ground, if possible.

Pin Locations for XCS40 & XCS40XL Devices

XCS40 & XCS40XL Pad Name	PQ208	PQ240
VCC	P183	P212
I/O	P184	P213
I/O	P185	P214
I/O	P186	P215
I/O	P187	P216
I/O	P188	P217
I/O	P189	P218
I/O	P190	P220
I/O	P191	P221
I/O	-	-
I/O	-	-
VCC	P192	P222
I/O	-	P223
I/O	-	P224
I/O	P193	P225
I/O	P194	P226
GND	P195	P227
I/O	P196	P228
I/O	P197	P229
I/O	P198	P230
I/O	P199	P231
I/O	P200	P232
I/O	P201	P233
I/O	P202	P234
I/O	P203	P235
I/O	P204	P236
I/O	P205	P237
I/O	P206	P238
I/O, SGCK1	P207	P239
VCC	P208	P240
GND	P1	P1
I/O, PGCK1	P2	P2
I/O	P3	P3
I/O	P4	P4
I/O	P5	P5
I/O, TDI	P6	P6
I/O, TCK	P7	P7
I/O	P8	P8
I/O	P9	P9
I/O	P10	P10
I/O	P11	P11
I/O	P12	P12
I/O	-	P13
GND	P13	P14
I/O	P14	P15
I/O	P15	P16
I/O, TMS	P16	P17
I/O	P17	P18
VCC	P18	P19
I/O	-	P20
I/O	-	P21
I/O	-	-
I/O	-	-
I/O	P19	P23
I/O	P20	P24
I/O	P21	P25
I/O	P22	P26
I/O	P23	P27
I/O	P24	P28
GND	P25	P29
VCC	P26	P30
I/O	P27	P31
I/O	P28	P32
I/O	P29	P33
I/O	P30	P34
I/O	P31	P35
I/O	P32	P36
I/O	-	-
I/O	-	P38
I/O	-	P39
VCC	P33	P40
I/O	P34	P41
I/O	P35	P42
I/O	P36	P43
I/O	P37	P44
GND	P38	P45
I/O	-	P46
I/O	P39	P47
I/O	P40	P48
I/O	P41	P49
I/O	P42	P50
I/O	P43	P51
I/O	P44	P52
I/O	P45	P53
I/O	P46	P54
I/O	P47	P55
I/O	P48	P56
I/O, SGCK2	P49	P57
Don't Connect	P50	P58
GND	P51	P59
MODE	P52	P60
VCC	P53	P61
Don't Connect	P54	P62
I/O PGCK2	P55	P63
I/O (HDC)	P56	P64
I/O	P57	P65
I/O	P58	P66
I/O	P59	P67
I/O (LDC)	P60	P68
I/O	P61	P69
I/O	P62	P70
I/O	P63	P71
I/O	P64	P72
I/O	P65	P73
I/O	-	P74
GND	P66	P75
I/O	P67	P76
I/O	P68	P77
I/O	P69	P78
I/O	P70	P79
VCC	P71	P80
I/O	P72	P81
I/O	P73	P82
I/O	-	-
I/O	-	-
I/O	-	P84
I/O	-	P85
I/O	P74	P86
I/O	P75	P87
I/O	P76	P88
I/O (INIT)	P77	P89
VCC	P78	P90
GND	P79	P91
I/O	P80	P92
I/O	P81	P93
I/O	P82	P94
I/O	P83	P95
I/O	P84	P96
I/O	P85	P97
I/O	-	-
I/O	-	-
I/O	-	P99
I/O	-	P100
VCC	P86	P101
I/O	P87	P102
I/O	P88	P103
I/O	P89	P104

XCS40 & XCS40XL Pad Name	PQ208	PQ240
I/O	P90	P105
GND	P91	P106
I/O	-	P107
I/O	P92	P108
I/O	P93	P109
I/O	P94	P110
I/O	P95	P111
I/O	P96	P112
I/O	P97	P113
I/O	P98	P114
I/O	P99	P115
I/O	P100	P116
I/O	P101	P117
I/O, SGCK3	P102	P118
GND	P103	P119
DONE	P104	P120
VCC	P105	P121
PROGRAM	P106	P122
I/O	P107	P123
I/O, PGCK3	P108	P124
I/O	P109	P125
I/O	P110	P126
I/O	-	P127
I/O	P111	P128
I/O	P112	P129
I/O	P113	P130
I/O	P114	P131
I/O	P115	P132
I/O	P116	P133
I/O	P117	P134
GND	P118	P135
I/O	-	P136
I/O	-	P137
I/O	P119	P138
I/O	P120	P139
VCC	P121	P140
I/O	P122	P141
I/O	P123	P142
I/O	-	-
I/O	P124	P144
I/O	P125	P145
I/O	P126	P146
I/O	P127	P147
I/O	P128	P148
I/O	P129	P149
VCC	P130	P150
GND	P131	P151
I/O	P132	P152
I/O	P133	P153
I/O	P134	P154
I/O	P135	P155
I/O	P136	P156
I/O	P137	P157
I/O	-	-
I/O	P138	P159
I/O	P139	P160
VCC	P140	P161
I/O	P141	P162
I/O	P142	P163
I/O	-	P164
I/O	-	P165
GND	P143	P166
I/O	-	P167
I/O	P144	P168

XCS40 & XCS40XL Pad Name	PQ208	PQ240
I/O	P145	P169
I/O	P146	P170
I/O	P147	P171
I/O	P148	P172
I/O	P149	P173
I/O	P150	P174
I/O	P151	P175
I/O	P152	P176
I/O (DIN)	P153	P177
I/O, SGCK4 (DOUT)	P154	P178
CCLK	P155	P179
VCC	P156	P180
O, TDO	P157	P181
GND	P158	P182
I/O	P159	P183
I/O, PGCK4	P160	P184
I/O	P161	P185
I/O	P162	P186
I/O	P163	P187
I/O	P164	P188
I/O	P165	P189
I/O	-	P190
I/O	P166	P191
I/O	P167	P192
I/O	P168	P193
I/O	P169	P194
GND	P170	P196
I/O	P171	P197
I/O	P172	P198
I/O	-	P199
I/O	-	P200
VCC	P173	P201
I/O	-	-
I/O	-	-
I/O	P174	P202
I/O	P175	P203
I/O	P176	P205
I/O	P177	P206
I/O	P178	P207
I/O	P179	P208
I/O	P180	P209
I/O	P181	P210
GND	P182	P211

10/23/97

Additional XCS40/XL Package Pins

PQ240

GND Pins					
P22‡	P37‡	P83‡	P98‡	P143‡	P158‡
P204‡	P219‡	-	-	-	-
Not Connected Pins					
P195	-	-	-	-	-

6/9/97

‡ Pins marked with this symbol are used for Ground connections on some revisions of the device. These pins may not physically connect to anything on the current device revision. However, they should be externally connected to Ground, if possible.

XILINX®

Design Methodology

Design Methodology

The LogiCORE PCI32 Interfaces are highly optimized for the XC4000XLT and Spartan FPGAs. UCF and guide files provide easy implementation of the PCI interface. Pre-placed and pre-routed guide files guarantee timing performance on critical control signals like IRDY-, TRDY-, and FRAME-.

The user connects the LogiCORE PCI Interface to other modules to complete the design. For example, to complete a PCI adapter card interface using the XC4013XLT, a designer first configures the PCI core using the intuitive graphical user interface on the Xilinx web, and downloads proven PCI design. See Figure 3 for the GUI. A designer would then create a single page schematic, VHDL, or Verilog description, using the LogiCORE PCI32 Interface component together with the required user application. The user can then simulate the core with the customer design using provided VHDL and Verilog simulation models. Next, the user compiles the design using the M1 software and the PCI place and route constraints file for the targeted FPGA. At this point the design can be resimulated or downloaded to the target device. See Figure 1 for a complete design flow.

LogiCORE PCI Configuration

To support multiple design environments, the LogiCORE PCI interface can be configured and downloaded from the web complete with a netlist, constraints, simulation model, and testbench in VHDL or Verilog. This methodology is described in our on-line documentation. For more details see:

`www.xilinx.com/products/logicore/logicore.htm`

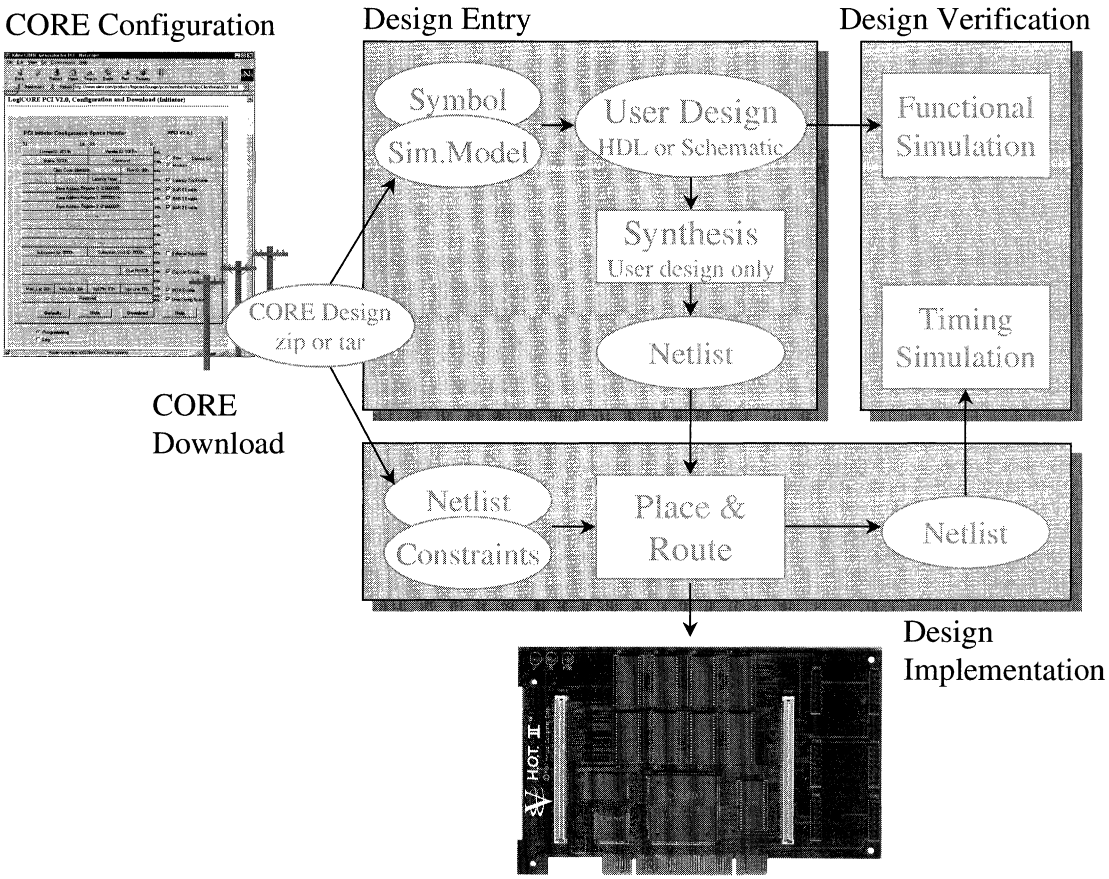


Figure 1: PCI32 Interface Design and Simulation Methodology

This LogiCORE PCI configuration methodology allows HDL users and schematic to modify the PCI macro. After configuring the macro, the user can download the design files in a PC or UNIX format. Using the macro in a HDL flow is described in the *PCI32 User's Guide*.

For users that cannot access the configuration tool on the web, the core can be configured locally by editing a single schematic, VHDL or Verilog module.

Configuration Module

This is a self contained module, that allows migration between various HDL and schematic EDA tools. In Figure 2 the Viewlogic version of the configuration module is shown.

Schematic

For Viewlogic users, all modifiable elements are in the cfg.1 schematic. To make changes, simply edit the cfg.1 file as describe in the *PCI32 User's Guide*. This will configure the schematics to desired PCI configuration.

VHDL and Verilog

If using an VHDL or Verilog synthesis tool, two configuration options are available. The LogiCORE PCI interface can be configured and downloaded from the web or the user can simply edit the configuration file. To modify the design for either implementation or simulation in Verilog, edit the cfg.v file. If using VHDL, edit the cfg.vhd file included with the design files. This file supplies the configuration information to both the implementation black box, which is instantiated in your design, and the functional simulation model. More information about HDL flows can be found in the *PCI32 User's Guide.*

Selectable Options

Wait States

The LogiCORE PCI32 Interfaces are switch selectable between one and zero wait states in the configuration module. See LogiCORE PCI Data Sheets for details on bandwidth options.

Both zero and one wait state modes are fully PCI compliant; they do not deassert IRDY# or TRDY# when the other PCI bus agent inserts a wait state. See the LogiCORE PCI32 User's Guide for additional details on using this mode.

Decode Speed

The LogiCORE PCI32 4000 Interfaces supports both medium and slow decode. This doesn't affect the timing of the backend designs, only the assertion of DEVSEL.

CapPtr and User Config Space

The LogiCORE PCI32 Interfaces supports the new PCI Capability Pointer and the Additional Config Space it needs. For more details on using this option, refer to the *LogiCORE PCI32 User's Guide*.

The Base Address Registers are fully modifiable. Unused BARs can be turned off for better utilization.

The Interrupt Enable sets Bit 0 of the Interrupt Pin Register.

The Latency Timer must be enabled in any design that needs to do bursting. If the design doesn't require bursting, disabling it will save some resources. Refer to the *LogiCORE PCI32 User's Guide* for more details on these and any other options.

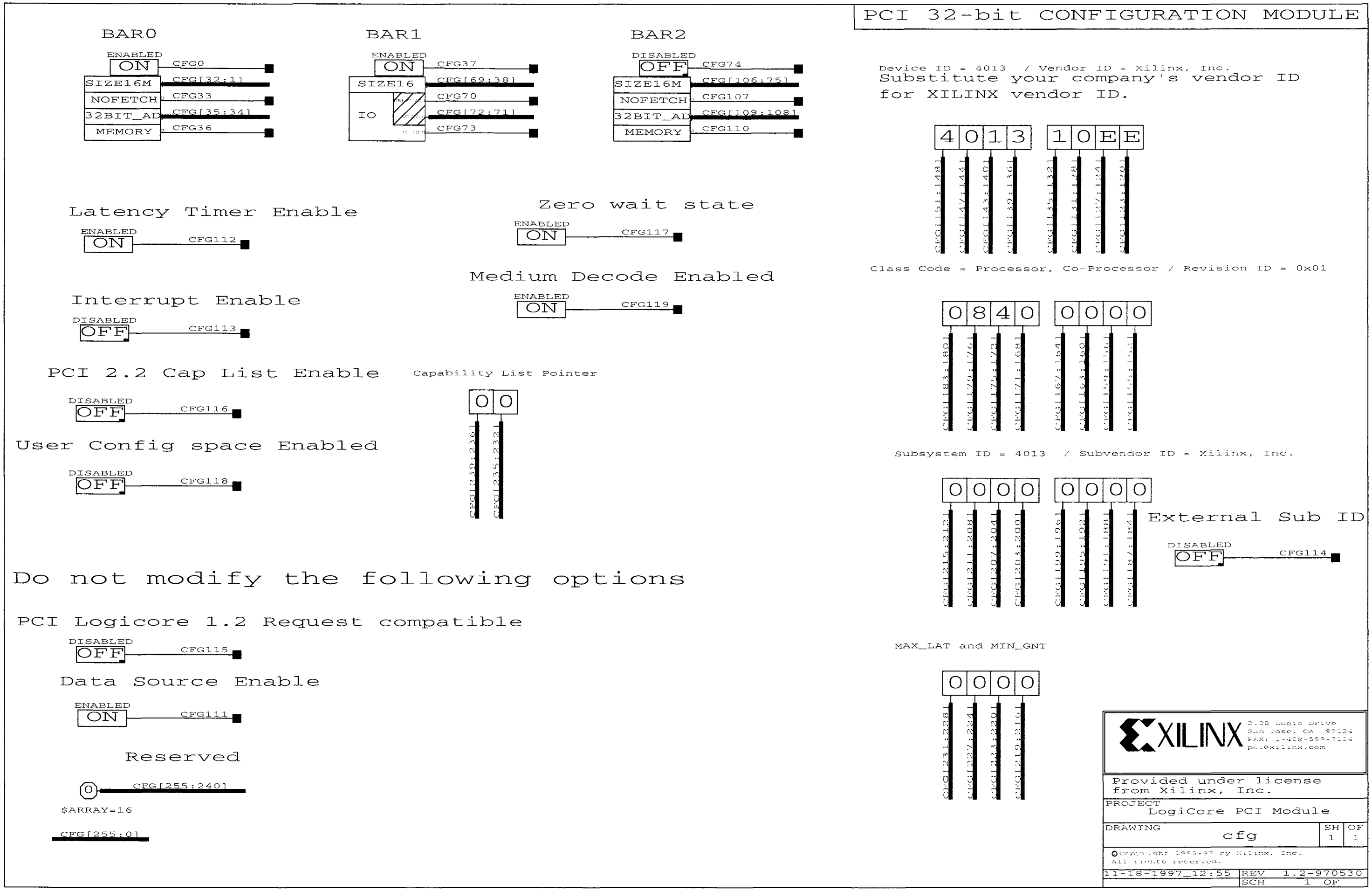
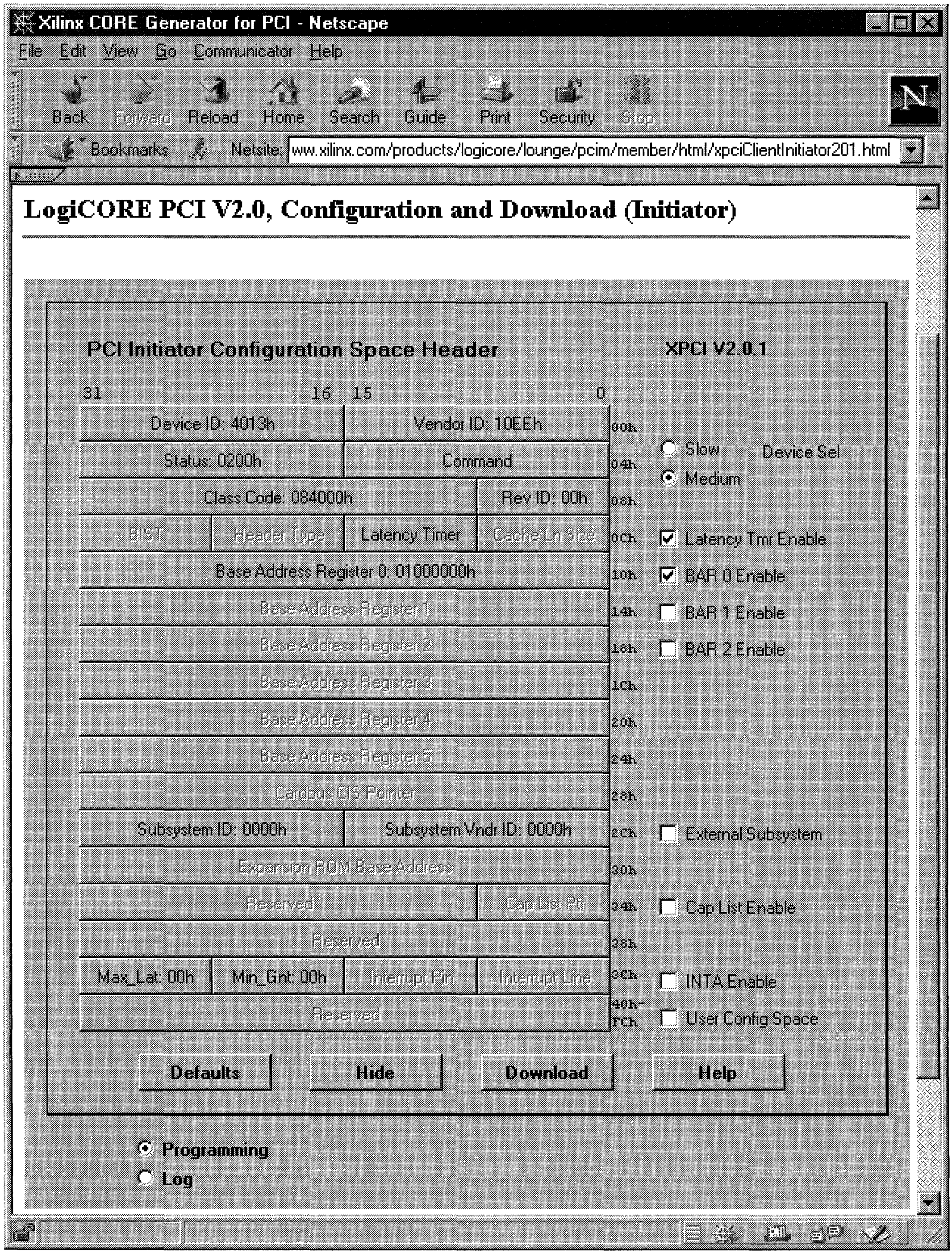


Figure 2: Schematic of cfg.1

Figure 3: PCI Configuration and Download GUI



ww.xilinx.com/products/logicore/lounge/pcim/member/html/xpciClientInitiator201.html

LogiCORE PCI V2.0, Configuration and Download (Initiator)

PCI Initiator Configuration Space Header

XPCI V2.0.1

31	16	15	0	
Device ID: 4013h		Vendor ID: 10EEh		00h
Status: 0200h		Command		04h
Class Code: 084000h			Rev ID: 00h	08h
BIST	Header Type	Latency Timer	Cache Ln Size	0Ch
Base Address Register 0: 01000000h				10h
Base Address Register 1				14h
Base Address Register 2				18h
Base Address Register 3				1Ch
Base Address Register 4				20h
Base Address Register 5				24h
Cardbus CIS Pointer				28h
Subsystem ID: 0000h		Subsystem Vndr ID: 0000h		2Ch
Expansion ROM Base Address				30h
Reserved			Cap List Ptr	34h
Reserved				38h
Max_Lat: 00h	Min_Gnt: 00h	Interrupt Pin	Interrupt Line	3Ch
Reserved				40h-FCh

○ Slow Device Sel
● Medium

☑ Latency Tmr Enable

☑ BAR 0 Enable

□ BAR 1 Enable

□ BAR 2 Enable

□ External Subsystem

□ Cap List Enable

□ INTA Enable

□ User Config Space

● Programming
○ Log

PCI Compliance Checklists

Xilinx PCI

XC4000XLT
PCI Compliance Checklist

May, 1998

Data Sheet

Component Product Information

Date	May, 1998
Vendor Name	Xilinx, Inc.
Vendor Street Address	2100 Logic Drive
Note Vendor City, State, Zip	San Jose, CA 95124 U.S.A
Vendor Phone Number	+1 408-559-7778
Vendor Contact, Title	Per Holmberg LogiCORE Marketing Manager
Product Name	XC4000XLT FPGAs
Product Model Number	XC4013XLT XC4028XLT XC4062XLT
Product Revision Level	

Component Electrical Checklist

This checklist applies to the following Component/Manufacturer: XC4000XLT / Xilinx, Inc.

All items were verified over the following range of junction temperatures: ____min ____max

OR

All items were verified over the following range of CASE temperatures: 25 ° C min 85 ° C max

5 V Signaling

Type	Description	Yes or N/A
CE1.	Component supports 5V signaling environment?	yes ____ na✓
	if "na", skip to section "3.3V Signaling" below.	
CE2.	Component operates over voltage range 5V +/- 5%?	na___ yes ___ no___
	"na" allowed for components that support 5V signaling, but draw power from a supply other than Vcc 5V.	
CE3.	Voltages between 2.0V and Vcc+0.5V are recognized as logic high?	yes ___ no___
CE4.	Voltages between -0.5V and 0.8V are recognized as logic low?	yes ___ no___
CE5.	All inputs sink less than 70uA when pulled to 2.7V DC?	yes ___ no___
CE6.	All inputs source less than 70uA when pulled to 0.5V DC?	yes ___ no___
CE7.	All outputs drive to 2.4V (min) in the high state while sourcing 2mA?	yes ___ no___
CE8.	All outputs drive to 0.55V (max) in the low state, sinking 3 or 6mA?	yes ___ no___
CE9.	Outputs source at least 44mA at 1.4V in the high state? *proven at: ___ Vcc= min, ___ process=worst/slow, ___ junction temp=_____ (max) by: ___ SPICE simulation, __ device characterization, ___ other:_______________*	yes ___ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE10.	Outputs source no more than 142mA at 3.1V in the high state? *proven at: __ Vcc=max, ___ process=best/fast, ___ junction temp=______ (min)by: ___ SPICE simulation, ___ device characterization, ___ other:_______________*	yes ___ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE11.	Outputs sink at least 95mA at 2.2V in the low state? *proven at: ___ Vcc=max, ___ process=worst/slow, ___ junction temp=______ (max) by: ___ SPICE simulation, ___ device characterization, ___ other:* _______________	yes ___ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE12.	Outputs sink no more than 206mA at 0.71V in the low state? *proven at: ___ Vcc=max, ___ process=best/fast, ___ junction temp=______ (min) by: ___ SPICE simulation, ___ device characterization, ___ other:* _______________	yes ___ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	

Type	Description	Yes or N/A
CE13.	REQ#, GNT# outputs source at least 22mA at 1.4V in the high state? *proven at: __ Vcc=max, ___ process=worst/slow, ___ junction temp=_____ (max) by: ___ SPICE simulation, __ device characterization, ___ other:_______________*	yes ___ no___
CE14.	REQ#, GNT# outputs sink at least 47mA at 2.2V in the low state? *proven at: __ Vcc=min, ___ process=worst/slow, ___ junction temp=______ (max) by: ___ SPICE simulation, __ device characterization, ___ other:________________*	na ___ yes ___ no___
CE15.	Clamps on all signals source at least 25mA at -1V, and 91mA at -2V? *proven by: __ SPICE simulation, __ device characterization, other:_____________*	na ___ yes ___ no___
CE16.	Unloaded rise times are no lower than 1 V/nS between 0.4 and 2.4V? The unloaded maximum rise time is: ___V/nS (measured at pin)	yes ___ no___
CE17.	Unloaded fall times are no lower than 1 V/nS between 2.4 and 0.4V? The unloaded maximum fall time is: ____V/nS (measured at pin)	yes ___ no___

3.3 V Signaling

Type	Description	Pass/NA
CE18.	Component supports 3.3V signaling environment?	yes ✓ na___
	if "na", skip to section "Loading and Device Protection" below.	
CE19.	Component operates over voltage range 3.3V +/- 0.3V?	yes ✓ no___
CE20.	Voltages between 0. 5Vcc and Vcc+0.5V are recognized as logic high?	yes ✓ no___
CE21.	Voltages between -0.5V and 0.3Vcc are recognized as logic low?	yes ✓ no___
CE22.	All inputs sink/source less than 10 uA at any voltage from 0V to Vcc?	yes ✓ no___
CE23.	All outputs drive to 0.9Vcc (min) in the high state while sourcing 500uA?	yes ✓ no___
CE24.	All outputs drive to 0.1Vcc (max) in the low state, sinking 1500uA?	yes ✓ no___
CE25.	Outputs source at least 36mA at 0.9V in the high state? *proven at: ✓ Vcc=3.0V, ___ process=worst/slow, ___ junction temp= 85 ° C (max) by: ___ SPICE simulation, ✓ device characterization, other:* ________________	yes ✓ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, RST#, and SERR#	
CE26.	Outputs source no more than 115mA at 2.5V in the high state? *proven at: ✓ Vcc=3.6V, ___ process=best/fast, ___ junction temp= 25 ° C (min) by: ___ SPICE simulation, ✓ device characterization, other:* ________________	yes ✓ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, RST#, and SERR#	
CE27.	Outputs sink at least 48mA at 1.8V in the low state? *proven at: ✓ Vcc=3.0V, ___ process=worst/slow, ___ junction temp= 85 ° C (max) by: ___ SPICE simulation, ✓ device characterization, other:* ____________	yes ✓ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE28.	Outputs sink no more than 137mA at 0.65V in the low state? *proven at: ✓ Vcc=3.6V, ___ process=best/fast, ___ junction temp= 25 ° C (min) by: ___ SPICE simulation,✓ device characterization, other:* ________________	yes ✓ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE29.	REQ#, GNT# outputs source at least 18mA at 0.9V in the high state? *proven at: ✓ Vcc=3.0V, ___ process=worst/slow, ___ junction temp= 85 ° C (max) by: ___ SPICE simulation, ✓ device characterization, other:* ____________	yes ✓ no___
CE30.	REQ#, GNT# outputs sink at least 24mA at 1.8V in the low state? *proven at: ✓ Vcc=3.0V, ___ process=worst/slow, ___ junction temp= 85 ° C (max) by: ___ SPICE simulation, ✓ device characterization, other:* ____________	yes ✓ no __
CE31.	Clamps on all signals source at least 25mA at -1V, and 91mA at -2V? *proven by: ___ SPICE simulation, ✓ device characterization, other:*____________	yes ✓ no___
CE32.	Clamps on all signals sink at least 25mA at Vcc+1V, and 91mA at Vcc+2V? *proven by: ___ SPICE simulation, ✓ device characterization, other:*____________	na ___ yes ✓ no___

Type	Description	Pass/NA
CE33.	Unloaded rise times are no lower than 1 V/nS between 0.2Vcc and 0.6Vcc? The unloaded maximum rise time is: 1.14 V/nS (measured at pin)	na ___ yes ✓ no___
CE34.	Unloaded fall times are no lower than 1 V/nS between 0.6Vcc and 0.2Vcc? The unloaded maximum fall time is: 1.38 V/nS (measured at pin)	yes ✓ no___

Loading and Device Protection

Type	Description	Yes/No
CE35.	Capacitance on all PCI signals (except CLK, IDSEL) is less than or equal to 10 pF?	yes ✓ no___
CE36.	Capacitance on CLK signal is between 5 and 12 pF?	yes ✓ no___
CE37.	Capacitance on IDSEL signal is less than 8 pF? *capacitance guaranteed by: device characterization* ✓ *other* _______________. The maximum inductance on any PCI pin is: 15.9 nH.	yes ✓ no ___
CE38.	Read, understand section "Maximum AC Ratings and Device Protection"? ✓ *believe to be non-issue given technology used proven robustness when exposed to prescribed test condition.*	yes ✓ no___

Timing Specification

Type	Description	Pass / N/A
CE39.	Component is operational at any frequency between DC and 33 MHz?	yes ✓ na___
	Notes: "na" implies component intended for motherboard use only. To satisfy this requirement, designs are allowed to require software to place the component in the proper state before stopping the clock and return it to an operational state after restarting the clock.	
CE40.	Component is operational with a CLK High Time of 11 nS for 33 Mhz PCI, 6 ns for 66 Mhz PCI?	na ___ yes ✓ no___
CE41.	Component is operational with a CLK Low Time of 11 nS for 33 Mhz PCI, 6 ns for 66 Mhz PCI?	na ___ yes ✓ no___
CE42.	All bussed signals are driven valid between 2 and 11 nS after CLK for 33 Mhz PCI, between 2 and 6 ns for 66 Mhz PCI?	yes ✓ no___
CE43.	REQ# and GNT# signals are driven valid between 2 and 12 nS after CLK for 33 Mhz PCI, between 2 and 6 ns for 66 Mhz PCI?	yes ✓ no___
CE44.	All Tri-state signals become active no earlier than 2 nS after CLK?	yes ✓ no___
CE45.	All Tri-state signals float no later than 28 nS after CLK for 33 Mhz PCI, no later than 14 nS for 66 Mhz PCI?	yes ✓ no___
CE46.	All bussed inputs require no more than 7 nS setup to CLK for 33 Mhz PCI, no more than 3 nS for 66 Mhz PCI?	yes ✓ no___
CE47.	REQ# requires no more than 12 nS setup to CLK for 33 Mhz PCI, no more than 5 nS for 66 Mhz PCI?	na ___ yes ✓ no___
CE48.	GNT# requires no more than 10 nS setup to CLK for 33 Mhz PCI, no more than 5 nS for 66 Mhz PCI?	na ___ yes ✓ no___
CE49.	All inputs require no more than 0 nS of hold time after CLK?	yes ✓ no___
CE50.	All outputs are Tri-stated within 40 nS after RST# goes low?	yes ✓ no___

all timings (CE39 through CE50 verified by (check all that apply)

____ *static timing design tools (MOTIVE, QTV, QuickPath, Veritime...)*

____ *dynamic timing design tools (Verilog, Qsim, Quicksim, ViewSim, VHDL, ...)*

✓ *silicon AC testing*

____*other*______________________________

NOTE: Maximum and minimum timings assume different output loadings for both 5.0V and 3.3V parts. See PCI Spec Rev 2.1 page 134 note #2.

64-bit Components

Type	Description	Pass or N/A
	Component is 32-bit only, this section is	NA ✓
CE51.	Component senses, during RST# active, its connection to 64-bit wires?	yes ___ no___
CE52.	64-bit input signals will be stable when not connected?	yes ___ no__

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

Xilinx PCI

Spartan
PCI Compliance Checklist

May, 1998

Data Sheet

Component Product Information

Date	May, 1998
Vendor Name	Xilinx, Inc.
Vendor Street Address	2100 Logic Drive
Note Vendor City, State, Zip	San Jose, CA 95124 U.S.A
Vendor Phone Number	+1 408-559-7778
Vendor Contact, Title	Per Holmberg LogiCORE Marketing Manager
Product Name	Spartan Series FPGAs
Product Model Number	XCS30, XCS40
Product Revision Level	

Component Electrical Checklist

This checklist applies to the following Component/Manufacturer: Spartan XCS30 / Xilinx

All items were verified over the following range of junction temperatures: ____min ____max

OR

All items were verified over the following range of CASE temperatures: 25 ° C min 85 °C max

5 V Signaling

Type	Description	Yes or N/A
CE1.	Component supports 5V signaling environment?	yes ✓ na___
	if “na”, skip to section “3.3V Signaling” below.	
CE2.	Component operates over voltage range 5V +/- 5%?	na___ yes ✓ no___
	“na” allowed for components that support 5V signaling, but draw power from a supply other than Vcc 5V.	
CE3.	Voltages between 2.0V and Vcc+0.5V are recognized as logic high?	yes ✓ no___
CE4.	Voltages between -0.5V and 0.8V are recognized as logic low?	yes ✓ no___
CE5.	All inputs sink less than 70uA when pulled to 2.7V DC?	yes ✓ no___
CE6.	All inputs source less than 70uA when pulled to 0.5V DC?	yes ✓ no___
CE7.	All outputs drive to 2.4V (min) in the high state while sourcing 2mA?	yes ✓ no___
CE8.	All outputs drive to 0.55V (max) in the low state, sinking 3 or 6*mA?	yes ✓ no___
CE9.	Outputs source at least 44mA at 1.4V in the high state? *proven at: ✓ Vcc= min, ___ process=worst/slow, ✓ junction temp=85 ° (max) by: ___ SPICE simulation, ✓ device characterization, ___ other:________________*	yes ✓ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE10.	Outputs source no more than 142mA at 3.1V in the high state? *proven at: ✓ Vcc=max, ___ process=best/fast, ___ junction temp=25 ° (min)by: ___ SPICE simulation, ✓ device characterization, ___ other: ________________*	yes ✓ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE11.	Outputs sink at least 95mA at 2.2V in the low state? *proven at: ✓ Vcc=max, ___ process=worst/slow, ✓ junction temp=85 ° (max) by: ___ SPICE simulation, ✓ device characterization, ___ other: ________________*	yes ✓ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE12.	Outputs sink no more than 206mA at 0.71V in the low state? *proven at: ✓ Vcc=max, ___ process=best/fast, ✓ junction temp=25 ° (min) by: ___ SPICE simulation, ✓ device characterization, ___ other: ________________*	yes ✓ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE13.	REQ#, GNT# outputs source at least 22mA at 1.4V in the high state? *proven at: ✓ Vcc=max, ___ process=worst/slow, ✓ junction temp=85 ° (max) by: ___ SPICE simulation, ✓ device characterization, ___ other:______________*	yes ✓ no___
CE14.	REQ#, GNT# outputs sink at least 47mA at 2.2V in the low state? *proven at: ✓ Vcc=min, ___ process=worst/slow, ___ junction temp=85 ° (max) by: ___ SPICE simulation, ✓ device characterization, ___ other:________*	na ___ yes ✓ no___

Type	Description	Yes or N/A
CE15.	Clamps on all signals source at least 25mA at -1V, and 91mA at -2V? *proven by: __ SPICE simulation, ✓ device characterization, other: ___________*	na ___ yes ✓ no___
CE16.	Unloaded rise times are no lower than 1 V/nS between 0.4 and 2.4V? The unloaded maximum rise time is: 1.53 V/nS (measured at pin)	yes ✓ no___
CE17.	Unloaded fall times are no lower than 1 V/nS between 2.4 and 0.4V? The unloaded maximum fall time is: 3.49 V/nS (measured at pin)	yes ✓ no___

3.3 V Signaling

Type	Description	Pass/NA
CE18.	Component supports 3.3V signaling environment?	yes ___ na ✓
	if "na", skip to section "Loading and Device Protection" below.	
CE19.	Component operates over voltage range 3.3V +/- 0.3V?	yes ___ no___
CE20.	Voltages between 0. 5Vcc and Vcc+0.5V are recognized as logic high?	yes ___ no___
CE21.	Voltages between -0.5V and 0.3Vcc are recognized as logic low?	yes ___ no___
CE22.	All inputs sink/source less than 10 uA at any voltage from 0V to Vcc?	yes ___ no___
CE23.	All outputs drive to 0.9Vcc (min) in the high state while sourcing 500uA?	yes ___ no___
CE24.	All outputs drive to 0.1Vcc (max) in the low state, sinking 1500uA?	yes___ no___
CE25.	Outputs source at least 36mA at 0.9V in the high state? *proven at: ___ Vcc=3.0V, ___ process=worst/slow, ___ junction temp= 75° C (max) by: ___ SPICE simulation,___ device characterization, other: __________*	yes ___ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, RST#, and SERR#	
CE26.	Outputs source no more than 115mA at 2.5V in the high state? *proven at: ___ Vcc=3.6V, ___ process=best/fast, ___ junction temp= 25 ° C (min) by: ___ SPICE simulation, ___ device characterization, other: _______________*	yes___ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, RST#, and SERR#	
CE27.	Outputs sink at least 48mA at 1.8V in the low state? *proven at: ___ Vcc=3.0V, ___ process=worst/slow, ___ junction temp= 75 ° C (max) by: ___ SPICE simulation, ___ device characterization, other: __________*	yes ___ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE28.	Outputs sink no more than 137mA at 0.65V in the low state? *proven at: ___ Vcc=3.6V, ___ process=best/fast, ___ junction temp= 25 ° C (min) by: ___ SPICE simulation,___ device characterization, other: _______________*	yes ___ no___
	NOTE: applies to all outputs except REQ#, GNT#, CLK, and RST#	
CE29.	REQ#, GNT# outputs source at least 18mA at 0.9V in the high state? *proven at: ___ Vcc=3.0V, ___ process=worst/slow, ___ junction temp= 75 ° C (max) by: ___ SPICE simulation, ___ device characterization, other:* ________________	yes ___ no___
CE30.	REQ#, GNT# outputs sink at least 24mA at 1.8V in the low state? *proven at: ___ Vcc=3.0V, ___ process=worst/slow, ___ junction temp= 75 ° C (max) by: ___ SPICE simulation, ___ device characterization, other: __________*	yes ___ no __
CE31.	Clamps on all signals source at least 25mA at -1V, and 91mA at -2V? *proven by: ___ SPICE simulation, ___ device characterization, other:_________*	yes ___ no___
CE32.	Clamps on all signals sink at least 25mA at Vcc+1V, and 91mA at Vcc+2V? *proven by: ___ SPICE simulation, ___ device characterization, other:___________*	na ___ yes ___ no___

Type	Description	Pass/NA
CE33.	Unloaded rise times are no lower than 1 V/nS between 0.2Vcc and 0.6Vcc? The unloaded maximum rise time is: ____ V/nS (measured at pin)	na ___ yes ___ no___
CE34.	Unloaded fall times are no lower than 1 V/nS between 0.6Vcc and 0.2Vcc? The unloaded maximum fall time is: ____ V/nS (measured at pin)	yes ___ no___

Loading and Device Protection

Type	Description	Yes/No
CE35.	Capacitance on all PCI signals (except CLK, IDSEL) is less than or equal to 10 pF?	yes ✓ no___
CE36.	Capacitance on CLK signal is between 5 and 12 pF?	yes ✓ no___
CE37.	Capacitance on IDSEL signal is less than 8 pF? *capacitance guaranteed by: device characterization* ✓ *other* _______________. The maximum inductance on any PCI pin is:_*15.9*_nH.	yes ✓ no ___
CE38.	Read, understand section "Maximum AC Ratings and Device Protection"? ✓ *believe to be non-issue given technology used proven robustness when exposed to prescribed test condition.*	yes ✓ no___

Timing Specification

Type	Description	Pass / N/A
CE39.	Component is operational at any frequency between DC and 33 MHz?	yes ✓ na___
	Notes: "na" implies component intended for motherboard use only. To satisfy this requirement, designs are allowed to require software to place the component in the proper state before stopping the clock and return it to an operational state after restarting the clock.	
CE40.	Component is operational with a CLK High Time of 11 nS for 33 Mhz PCI, 6 ns for 66 Mhz PCI?	na ___ yes ✓ no___
CE41.	Component is operational with a CLK Low Time of 11 nS for 33 Mhz PCI, 6 ns for 66 Mhz PCI?	na ___ yes ✓ no___
CE42.	All bussed signals are driven valid between 2 and 11 nS after CLK for 33 Mhz PCI, between 2 and 6 ns for 66 Mhz PCI?	yes ✓ no___
CE43.	REQ# and GNT# signals are driven valid between 2 and 12 nS after CLK for 33 Mhz PCI, between 2 and 6 ns for 66 Mhz PCI?	yes ✓ no___
CE44.	All Tri-state signals become active no earlier than 2 nS after CLK?	yes ✓ no___
CE45.	All Tri-state signals float no later than 28 nS after CLK for 33 Mhz PCI, no later than 14 nS for 66 Mhz PCI?	yes ✓ no___
CE46.	All bussed inputs require no more than 7 nS setup to CLK for 33 Mhz PCI, no more than 3 nS for 66 Mhz PCI?	yes ✓ no___
CE47.	REQ# requires no more than 12 nS setup to CLK for 33 Mhz PCI, no more than 5 nS for 66 Mhz PCI?	na ___ yes ✓ no___
CE48.	GNT# requires no more than 10 nS setup to CLK for 33 Mhz PCI, no more than 5 nS for 66 Mhz PCI?	na ___ yes ✓ no___
CE49.	All inputs require no more than 0 nS of hold time after CLK?	yes ✓ no___
CE50.	All outputs are Tri-stated within 40 nS after RST# goes low? *all timings (CE39 through CE50 verified by (check all that apply)* ____ *static timing design tools (MOTIVE, QTV, QuickPath, Veritime...)* ____ *dynamic timing design tools (Verilog, Qsim, Quicksim, ViewSim, VHDL, ...)* ✓ *silicon AC testing* ____*other*________________________________	yes ✓ no___
	NOTE: Maximum and minimum timings assume different output loadings for both 5.0V and 3.3V parts. See PCI Spec Rev 2.1 page 134 note #2.	

64-bit Components

Type	Description	Pass or N/A
	Component is 32-bit only, this section is	NA ✓
CE51.	Component senses, during RST# active, its connection to 64-bit wires?	yes ___ no___
CE52.	64-bit input signals will be stable when not connected?	yes ___ no__

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

LogiCORE PCI32 Cores
PCI Compliance Checklist

May, 1998

Data Sheet

Component Product Information

Date	May, 1998
Vendor Name	Xilinx, Inc.
Vendor Street Address	2100 Logic Drive
Note Vendor City, State, Zip	San Jose, CA 95124 U.S.A
Vendor Phone Number	+1 408-559-7778
Vendor Contact, Title	Per Holmberg LogiCORE Marketing Manager
Product Name	LogiCORE PCI Interface
Product Model Number	DO-DI-PCI32 XC4000XLT FPGA Spartan FPGA
Product Revision Level	PCI32 4000/PCI32 Spartan

Component Configuration Checklist

Organization

Type	Description	Pass or N/A
CO1.	Does each PCI resource have a configuration space based on the 256 byte template defined in section 6.1, with a predefined 64 byte header and a 192 byte device specific region?	yes ✓ no___
CO2.	Do all functions in the device support the Vendor ID, Device ID, Command, Status, Header Type and Class Code fields in the header? See figure 6-1.	yes ✓ no___
CO3.	Is the configuration space available for access at all times? **The configuration space is available for access at all times through the internal ADIO bus when other operations are not in progress. The contents of the Command/Status Register is available on the CSR[39:0] bus output from the macro. Support status bits can be set at any time.**	yes ✓ no___
CO4.	Are writes to reserved registers or read only bits completed normally and the data discarded?	yes ✓ no___
CO5.	Are reads to reserved or unimplemented registers, or bits, completed normally and a data value of 0 returned?	yes ✓ no___
CO6.	Is the vendor ID a number allocated by the PCI SIG? **The Xilinx vendor ID is the default value. However, the user should use his or her company's own vendor ID.**	yes ✓ no___
CO7.	Does the Header Type field have a valid encoding?	yes ✓ no___
CO8.	Do multi-byte transactions access the appropriate registers and are the registers in "little endian" order?	yes ✓ no___
CO9.	Are all READ ONLY register values within legal ranges? For example, the Interrupt Pin register must only contain values 0-4.	yes ✓ no___
CO10.	Is the class code in compliance with the definition in Appendix D?	yes ✓ no___
CO11.	Is the predefined header portion of configuration space accessible as bytes, words, and dwords?	yes ✓ no___
CO12.	Is the device a multifunction device?	yes __ no ✓
CO13.	If the device is multifunction, are config space accesses to unimplemented functions ignored?	yes ___ no___ N/A ✓

Indicate either N/A (Not Applicable) or Implemented by placing a check in the appropriate box. Grayed areas indicate invalid selections. This table should be completed for each function in a multifunction device.

Location	Name	Required/Optional	N/A	Implemented
00h-01h	Vendor ID	Required	▨	✓
02h-03h	Device ID	Required	▨	✓
04h-05h	Command	Required	▨	✓
06h-07h	Status	Required	▨	✓
08h	Revision ID	Required	▨	✓
09h-0Bh	Class Code	Required	▨	✓
0Ch	Cache Line Size	Required by master devices/functions that can generate Memory Write and Invalidate	✓	
0Dh	Latency Timer	Required by master devices/functions that can burst more than two data phases		✓
0Eh	Header Type	If the device is multi-functional, then bit 7 must be set to a 1. The remaining bits are required to have a defined value.		✓
0Fh	BIST	Optional	✓	
10h-27h	Base Address Registers	1 or more required for any address allocation.		✓
28h-2Bh	Cardbus CIS Pointer	Optional		✓ returns 0
2CH=2Dh	Subsystem Vendor ID	Optional		✓
2Eh-2Fh	Subsystem ID	Optional		✓
30h-2Fh	Expansion ROM Base Address	Required for devices/functions that have expansion ROM.		✓ returns 0
34h-3Bh	Reserved		▨	✓
3Ch	Interrupt Line	Required by devices/functions that use an interrupt pin.		✓
3Dh	Interrupt Pin	Required by devices/functions that use an interrupt pin.		✓
3Eh	Min_Gnt	Optional		✓
3Fh	Max_Lat	Optional		✓

Device Control

This section should be completed individually for all functions in a multifunction device.

Type	Description	Yes/No
DC1.	When the command register is loaded with a 0000h is the device/function logically disconnected from the PCI, with the exception of configuration accesses? (Devices in BOOT code path are exempt).	yes ✓ no___
DC2.	Is the device/function disabled after the assertion of PCI RST#? (Devices in BOOT code path are exempt).	yes ✓ no___

In the following tables for Command and Status Registers, an "x" in the "Target" or "Master" columns, indicates that applying the bit is appropriate. "N/A" indicates that applying the bit is not applicable, but must return a 0 when read.

Bit	Name	Required/Optional	N/A	Target	Master
0	I/O Space	Required if device/function has registers mapped into I/O space.		✓	N/A
1	Memory Space	Required if device/function responds to memory space accesses.		✓	N/A
2	Bus Master	Required		N/A	✓
3	Special Cycles	Required for devices/ functions that can respond to Special Cycles.	✓		N/A
4	Memory Write and Invalidate Enable	Required for devices/ functions that generate Memory Write and Invalidate cycles.	✓	N/A	
5	VGA Palette snoop	Required for VGA or graphical devices/functions that snoop VGA color palette.	✓		N/A
6	Parity Error Response	Required unless exempted per section 3.7.2.		✓	✓
7	Wait cycle control	Optional	✓		
8	SERR# enable	Required if device/function has SERR# pin.		✓	
9	Fast Back-to-Back Enable	Required if Master device/function can support fast back-to-back cycles among different targets.	✓	N/A	
10-15	Reserved			✓	✓

Device Status

This section should be completed individually for all functions in a multifunction device.

Device Status Questions

Type	Description	Yes/No
DS1.	Do all implemented read/write bits in the Status reset to 0?	yes ✓ no___
DS2.	Are read/write bits set to a 1 exclusively by the device/function?	yes ✓ no___
DS3.	Are read/write bits reset to a 0 when PCIRST# is asserted?	yes ✓ no___
DS4.	Are read/write bits reset to a 0 by writing a 1 to the bit?	yes ✓ no___

Bit	Name	Required/Optional	N/A	Target	Master
0-4	Reserved	Required		✓	✓
5	66 Mhz Capable	Required for 66Mhz capable devices	✓		
6	UDF Supported	Optional	✓		
7	Fast Back-to-Back Capable	Optional	✓		N/A
8	Data Parity Detected	Required		N/A	✓
9-10	DEVSEL Timing	Required		✓	N/A
11	Signaled Target Abort	Required for devices/functions that are capable of signaling target abort		✓	N/A
12	Received Target Abort	Required		N/A	✓
13	Received Master Abort	Required		N/A	✓
14	Signaled System Error	Required for devices/functions that are capable of asserting SERR#		✓	✓
15	Detected Parity Error	Required unless exempted per section 3.7.2		✓	✓

Base Addresses

This section should be completed individually for all functions in a multifunction device

Type	Description	Yes/No
BA1.	If the device/function uses expansion ROM, does it implement the Expansion ROM Base Address Register? **The expansion ROM base address is not supported.**	yes ___ no ✓
BA2.	Do all Base Address registers asking for IO space request 256 bytes or less?	yes ✓ no ___
BA3.	If the device/function has an Expansion ROM Base Address register, does the memory enable bit in the Command register have precedence over the enable bit in the Expansion ROM base Address register? **The expansion ROM base address is not supported.**	yes ___ no ✓
BA4.	Does the device/function use any address space (memory or IO) other than that assigned using Base Address registers? (i.e.; Does the device/function hard-decode any addresses?) Note: If the answer is yes, you must list decoded addresses as explanations at the end of this section.	yes ___ no ✓
BA5.	Does the device/function decode all 32-bits of IO space? **The upper 24 bits of address are decoded by the base register. The lower 8 bits would be decoded by the user application.**	yes ✓ no ___
BA6.	If the device/function has an Expansion ROM Base Address register, is the size of the memory space requested 16MB or smaller?	yes ___ no___

VGA Devices

VGA Devices (fill in this section only if component is VGA device)

Type	Description	Yes/No
VG1.	Is palette snoop implemented, including bit in Command register?	yes ___ no___
VG2.	Is Expansion ROM Base Address register implemented and provide full relocatability of the expansion ROM? (The device must NOT do a hard decode of 0C0000h).	yes ___ no ___
VG3.	Does the device come up disabled? (Bottom three bits of Command register must be initialized to zero on power-up and PCIRST#).	yes ___ no ___
VG4.	Does Class Code field indicate VGA device? (value of 030000h).	yes ___ no___
VG5.	Does the device hard-decode only standard ISA VGA addresses and their aliases? (IO addresses 3B0h through 3BBh, 3C0h through 3DFh, Memory addresses 0A0000h through 0BFFFFh)	yes ___ no___
VG6.	Does the device use Base Address Registers to allocate needed space other than standard ISA VGA Addresses? (e.g. for a linear FRAME buffer)	yes ___ no___

General Component Protocol Checklist (Master)

The following checklist is to filled out as a general verification of the IUT's protocol compliance. This checklist applies to all master operations.

Test #	Description	Pass or N/A
MP1.	All Sustained Tri-State signals are driven high for one clock before being Tri-Stated. (2.1)	yes ✓ no___
MP2.	IUT always asserts all byte enables during each data phase of a Memory Write Invalidate cycle. (3.1.1) **Memory Write and Invalidate command not supported**	yes ___ no ✓
MP3.	IUT always uses Linear Burst Ordering for Memory Write Invalidate cycles. (3.1.1) **Memory Write and Invalidate command not supported.**	yes ___ no ✓
MP4.	IUT always drives IRDY# when data is valid during a write transaction. (3.2.1)	yes ✓ no ___
MP5.	IUT only transfers data when both IRDY# and TRDY# are asserted on the same rising clock edge. (3.2.1)	yes ✓ no ___
MP6.	Once the IUT asserts IRDY# it never changes FRAME# until the current data phase completes. (3.2.1)	yes ✓ no___
MP7.	Once the IUT asserts IRDY# it never changes IRDY# until the current data phase completes. (3.2.1)	yes ✓ no ___
MP8.	IUT never uses reserved burst ordering (AD[1::0] = "01". (3.2.2) **Value driven onto AD bus is controlled by the user application.**	yes ✓ no __
MP9.	IUT never uses reserved burst ordering (AD[1::0] = "11". (3.2.2) **Value driven onto AD bust is controlled by the user application.**	yes ✓ no __
MP10.	IUT always ignores configuration command unless IDSEL is asserted and AD[1::0] are "00". (3.2.2)	yes ✓ no ____
MP11.	The IUT's AD lines are driven to stable values during every address and data phase. (3.2.4)	yes ✓ no ___
MP12.	The IUT's C/BE# output buffers remain enabled from the first clock of the data phase through the end of the transaction. (3.3.1)	yes ✓ no ___
MP13.	The IUT's C/BE# lines contain valid Byte Enable information during the entire data phase. (3.3.1) **The values on the C/BE# pins are driven by the user application.**	yes ✓ no ___
MP14.	IUT never deasserts FRAME# unless IRDY# is asserted or will be asserted (3.3.3.1)	yes ✓ no ___
MP15.	IUT never deasserts IRDY# until at least one clock after FRAME# is deasserted. (3.3.3.1)	yes ✓ no ___
MP16.	Once the IUT deasserts FRAME# it never reasserts FRAME# during the same transaction. (3.3.3.1)	yes ✓ no ___
MP17.	IUT never terminates with master abort once target has asserted DEVSEL#. (3.3.3.1)	yes ✓ no ___
MP18.	IUT never signals master abort earlier than 5 clocks after FRAME# was first sampled asserted. (3.3.3.1)	yes ✓ no ___
MP19.	IUT always repeats an access exactly as the original when terminated by retry. (3.3.3.2.2) **The retry process is controlled by logic in the user's application.**	yes ✓ no ___

Test #	Description	Pass or N/A
MP20.	IUT never starts cycle unless GNT# is asserted. (3.4.1) **The IUT never starts a transaction cycle unless the following conditions are true:** - **GNT- is asserted.** - **The bus is idle.** - **The Master Enable bit is set in the Command Register.** **The IUT can transition from Initiator IDLE state to DR_BUS when GNT# is asserted and there is not active REQUEST pending.**	yes ✓ no ___
MP21.	IUT always Tri-States C/BE# and AD within one clock after GNT# negation when bus is idle and FRAME# is negated. (3.4.3)	yes ✓ no___
MP22.	IUT always drives C/BE# and AD within eight clocks of GNT# assertion when bus is idle. (3.4.3)	yes ✓ no___
MP23.	IUT always asserts IRDY# within eight clocks on all data phases. (3.5.2) **The IRDY# signal is controlled by the M_READY input from the user application.**	yes ✓ no ___
MP24.	IUT always begins lock operation with a read transaction. (3.6) **LOCK# function not supported.**	yes ___ no ✓
MP25.	IUT always releases #LOCK when access is terminated by target-abort or master-abort. (3.6) **LOCK# function not supported.**	yes ___ no ✓
MP26.	IUT always deasserts LOCK# for minimum of one idle cycle between consecutive lock operations. (3.6) **LOCK# function not supported.**	yes ✓ no ___
MP27.	IUT always uses Linear Burst Ordering for configuration cycles. (3.7.4)	yes ✓ no ___
MP28.	IUT always drives PAR within one clock of C/BE# and AD being driven. (3.8.1)	yes ✓ no ___
MP29.	IUT always drives PAR such that the number of "1"s on AD[31::0],C/BE[3:0], and PAR equals an even number. (3.8.1)	yes ✓ no ___
MP30.	IUT always drives PERR# (when enabled) active two clocks after data when data parity error is detected. (3.8.2.1)	yes ✓ no ___
MP31.	IUT always drives PERR (when enabled) for a minimum of 1 clock for each data phase that a parity error is detected. (3.8.2.1) **Dual Address command not supported.**	yes ✓ no ___
MP32.	IUT always holds FRAME# asserted for cycle following DUAL command. (3.10.1) **Dual Address command not supported.**	yes ___ no ✓
MP33.	IUT never generates DUAL cycle when upper 32-bits of address are zero. (3.10.1)	yes ___ no ✓

General Component Protocol Checklist (Target)

The following checklist is to filled out as a general verification of the IUT's protocol compliance. This checklist applies to all target operations.

General Component Protocol Checklist (Target)

Test #	Description	Pass or N/A
TP1.	All Sustained Tri-State signals are driven high for one clock before being Tri-Stated.	yes ✓ no ___
TP2.	IUT never reports PERR# until it has claimed the cycle and completed a data phase. (2.2.5)	yes ✓ no ___
TP3.	IUT never aliases reserved commands with other commands. (3.1.1)	yes ✓ no ___
TP4.	32-bit addressable IUT treats DUAL command as reserved. (3.1.1)	yes ✓ no ___
TP5.	Once IUT has asserted TRDY# it never changes TRDY# until the data phase completes. (3.2.1)	yes ✓ no ___
TP6.	Once IUT has asserted TRDY# it never changes DEVSEL# until the data phase completes. (3.2.1)	yes ✓ no ___
TP7.	Once IUT has asserted TRDY# it never changes STOP# until the data phase completes. (3.2.1)	yes ✓ no___
TP8.	Once IUT has asserted STOP# it never changes STOP# until the data phase completes. (3.2.1)	yes ✓ no ___
TP9.	Once IUT has asserted STOP# it never changes TRDY# until the data phase completes. (3.2.1)	yes ✓ no ___
TP10.	Once IUT has asserted STOP# it never changes DEVSEL# until the data phase completes. (3.2.1)	yes ✓ no ___
TP11.	IUT only transfers data when both IRDY# and TRDY# are asserted on the same rising clock edge. (3.2.1)	yes ✓ no ___
TP12.	IUT always asserts TRDY# when data is valid on a read cycle. (3.2.1)	yes ✓ no ___
TP13.	IUT always signals target-abort when unable to complete the entire IO access as defined by the byte enables. (3.2.2) **This function is implemented in the user application. Only the user application could determine if the byte enables were valid for the selected I/O devices.**	yes ✓ no ___
TP14.	IUT never responds to reserved encodings. (3.2.2)	yes ✓ no ___
TP15.	IUT always ignores configuration command unless IDSEL is asserted and AD[1::0] are "00". (3.2.2)	yes ✓ no ___
TP16.	IUT always disconnects after the first data phase when reserved burst mode is detected. (3.2.2)	yes ✓ no ___
TP17.	The IUT's AD lines are driven to stable values during every address and data phase. (3.2.4)	yes ✓ no ___
TP18.	The IUT's C/BE# output buffers remain enabled from the first clock of the data phase through the end of the transaction. (3.3.1)	yes ✓ no ___
TP19.	IUT never asserts TRDY# during turnaround cycle on a read. (3.3.1)	yes ✓ no ___

General Component Protocol Checklist (Target) (Continued)

Test #	Description	Pass or N/A
TP20.	IUT always deasserts TRDY#,STOP#, and DEVSEL# the clock following the completion of the last data phase. (3.3.3.2)	yes ✓ no ___
TP21.	IUT always signals disconnect when burst crosses resource boundary. (3.3.3.2) **This function would be implemented in the user application.**	yes ✓ no __
TP22.	IUT always deasserts STOP# the cycle immediately following FRAME# being deasserted. (3.3.3.2.1)	yes ✓ no ___
TP23.	Once the IUT has asserted STOP# it never deasserts STOP# until FRAME# is negated. (3.3.3.2.1)	yes ✓ no ___
TP24.	IUT always deasserts TRDY# before signaling target-abort. (3.3.3.2.1)	yes ✓ no ___
TP25.	IUT never deasserts STOP# and continues the transaction. (3.3.3.2.1)	yes ✓ no ___
TP26.	IUT always completes initial data phase within 16 clocks. (3.5.1.1)	yes ✓ no ___
TP27.	IUT always locks minimum of 16 bytes. (3.6) **LOCK# function not supported.**	yes ___ no ✓
TP28.	IUT always issues DEVSEL# before any other response. (3.7.1)	yes ✓ no ___
TP29.	Once IUT has asserted DEVSEL# it never deasserts DEVSEL# until the last data phase has competed except to signal target-abort. (3.7.1)	yes ✓ no ___
TP30.	IUT never responds to special cycles. (3.7.2)	yes ✓ no ___
TP31.	IUT always drives PAR within one clock of C/BE# and AD being driven. (3.8.1)	yes ✓ no ___
TP32.	IUT always drives PAR such that the number of "1"s on AD[31::0],C/BE[3:0], and PAR equals an even number. (3.8.1)	yes ✓ no ___

Component Protocol Checklist for a Master Device

For details on tests to run in order to fill in this checklist, refer to the Master Protocol Test Scenarios in Addendum A.

Definition: IUT is an acronym for “Implementation Under Test”.

Test Scenario: 1.1. PCI Device Speed (as indicated by DEVSEL) Tests

Note: The Initiator detects and reports a Master Abort condition. However, the LogiCORE Initiator de-asserts FRAME- and IRDY- one cycle later than the diagram shown as Figure 3-4 on Page 40 of the *PCI Local Bus Specification*, revision 2.1. Otherwise the LogiCORE Initiator treats Master Abort normally.

If IUT does not implement memory transactions mark 1 through 10 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 10 N/A.

Test #	Description	Pass	N/A
1	Data transfer after write to fast memory slave.	✓	
2	Data transfer after read from fast memory slave.	✓	
3	Data transfer after write to medium memory slave.	✓	
4	Data transfer after read from medium memory slave.	✓	
5	Data transfer after write to slow memory slave.	✓	
6	Data transfer after read from slow memory slave.	✓	
7	Data transfer after write to subtractive memory slave.	✓	
8	Data transfer after read from subtractive memory slave.	✓	
9	Master abort bit set after write to slower than subtractive memory slave.	✓	
10	Master abort bit set after read from slower than subtractive memory slave.	✓	

If IUT does not implement I/O transactions mark 11 through 20 N/A. Else if IUT supports both read and write transactions DO NOT mark 11 through 20 N/A.

Test #	Description	Pass	N/A
11	Data transfer after write to fast I/O slave.	✓	
12	Data transfer after read from fast I/O slave.	✓	
13	Data transfer after write to medium I/O slave.	✓	
14	Data transfer after read from medium I/O slave.	✓	
15	Data transfer after write to slow I/O slave.	✓	
16	Data transfer after read from slow I/O slave.	✓	
17	Data transfer after write to subtractive I/O slave.	✓	
18	Data transfer after read from subtractive I/O slave.	✓	
19	Master abort bit set after write to slower than subtractive I/O slave.	✓	
20	Master abort bit set after read from slower than subtractive I/O slave.	✓	

If IUT does not implement Configuration transactions mark 21 through 30 N/A. Else if IUT supports both read and write transactions DO NOT mark 21 through 30 N/A.

Note: Configuration transactions tested using VirtualChips environment.

Test #	Description	Pass	N/A
21	Data transfer after write to fast config slave.	✓	
22	Data transfer after read from fast config slave.	✓	
23	Data transfer after write to medium config slave.	✓	
24	Data transfer after read from medium config slave.	✓	
25	Data transfer after write to slow config slave.	✓	
26	Data transfer after read from slow config slave.	✓	
27	Data transfer after write to subtractive config slave.	✓	
28	Data transfer after read from subtractive config slave.	✓	
29	Master abort bit set after write to slower than subtractive config slave.	✓	
30	Master abort bit set after read from slower than subtractive config slave.	✓	

If IUT does not implement Interrupt transactions mark 31 through 35 N/A.

Test #	Description	Pass	N/A
31	Data transfer after interrupt from fast memory slave.		✓
32	Data transfer after interrupt from medium memory slave.		✓
33	Data transfer after interrupt from slow memory slave.		✓
34	Data transfer after interrupt from subtractive memory slave.		✓
35	Master abort bit set for interrupt from slower than subtractive memory slave.		✓

If IUT does not implement Special transactions mark 36 and 37 N/A.

Test #	Description	Pass	N/A
36	Data transfer after Special transaction to slave.		✓
37	Master abort bit is not set after Special transaction.		✓

Explanations

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.1.

Test Scenario: 1.2. PCI Bus Target Abort Cycles

Definition: IUT is an acronym for “Implementation Under Test”.

Note: The initiator does not repeat a transaction because it is controlled by the user application. The user application must monitor the Received Target Abort bit (CSR28) in the Command Register to prevent a retry after a Target Abort.

If IUT does not implement memory transactions mark 1 through 16 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 16 N/A.

Test #	Description	Pass	N/A
1	Target Abort bit set after write to fast memory slave.	✓	
2	IUT does not repeat the write transaction.	✓	
3	IUT's Target Abort bit set after read from fast memory slave.	✓	
4	IUT does not repeat the read transaction.	✓	
5	Target Abort bit set after write to medium memory slave.	✓	
6	IUT does not repeat the write transaction.	✓	
7	IUT's Target Abort bit set after read from medium memory slave.	✓	
8	IUT does not repeat the read transaction.	✓	
9	Target Abort bit set after write to slow memory slave.	✓	
10	IUT does not repeat the write transaction.	✓	
11	IUT's Target Abort bit set after read from slow memory slave.	✓	
12	IUT does not repeat the read transaction.	✓	
13	Target Abort bit set after write to subtractive memory slave.	✓	
14	IUT does not repeat the write transaction.	✓	
15	IUT's Target Abort bit set after read from subtractive memory slave.	✓	
16	IUT does not repeat the read transaction.	✓	

If IUT does not implement I/O transactions mark 17 through 32 N/A. Else if IUT supports both read and write transactions DO NOT mark 17 through 32 N/A.

Test #	Description	Pass	N/A
17	Target Abort bit set after write to fast I/O slave.	✓	
18	IUT does not repeat the write transaction.	✓	
19	IUT's Target Abort bit set after read from fast I/O slave.	✓	
20	IUT does not repeat the read transaction.	✓	
21	Target Abort bit set after write to medium I/O slave.	✓	
22	IUT does not repeat the write transaction.	✓	
23	IUT's Target Abort bit set after read from medium I/O slave.	✓	
24	IUT does not repeat the read transaction.	✓	
25	Target Abort bit set after write to slow I/O slave.	✓	
26	IUT does not repeat the write transaction.	✓	
27	IUT's Target Abort bit set after read from slow I/O slave.	✓	
28	IUT does not repeat the read transaction.	✓	
29	Target Abort bit set after write to subtractive I/O slave.	✓	
30	IUT does not repeat the write transaction.	✓	

Test #	Description	Pass	N/A
31	IUT's Target Abort bit set after read from subtractive I/O slave.	✓	
32	IUT does not repeat the read transaction.	✓	

If IUT does not implement configuration transactions mark 33 through 48 N/A. Else if IUT supports both read and write transactions DO NOT mark 33 through 48 N/A.

Note: Configuration transactions tested using VirtualChips environment.

Test #	Description	Pass	N/A
33	Target Abort bit set after write to fast config slave	✓	
34	IUT does not repeat the write transaction.	✓	
35	IUT's Target Abort bit set after read from fast config slave.	✓	
36	IUT does not repeat the read transaction.	✓	
37	Target Abort bit set after write to medium config slave.	✓	
38	IUT does not repeat the write transaction.	✓	
39	IUT's Target Abort bit set after read from medium config slave.	✓	
40	IUT does not repeat the read transaction.	✓	
41	Target Abort bit set after write to slow config slave.	✓	
42	IUT does not repeat the write transaction.	✓	
43	IUT's Target Abort bit set after read from slow config slave.	✓	
44	IUT does not repeat the read transaction.	✓	
45	Target Abort bit set after write to subtractive config slave.	✓	
46	IUT does not repeat the write transaction.	✓	
47	IUT's Target Abort bit set after read from subtractive config slave.	✓	
48	IUT does not repeat the read transaction.	✓	

If IUT does not implement interrupt transactions mark 49 through 56 N/A.

Test #	Description	Pass	N/A
49	IUT's Target Abort bit set after interrupt acknowledge from fast slave.		✓
50	IUT does not repeat the interrupt acknowledge transaction.		✓
51	IUT's Target Abort bit set after interrupt acknowledge from medium slave.		✓
52	IUT does not repeat the interrupt acknowledge transaction.		✓
53	IUT's Target Abort bit set after interrupt acknowledge from slow slave.		✓
54	IUT does not repeat the interrupt acknowledge transaction.		✓
55	IUT's Target Abort bit set after interrupt acknowledge from subtractive slave.		✓
56	IUT does not repeat the interrupt acknowledge transaction.		✓

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.2.

Test Scenario: 1.3. PCI Bus Target Retry Cycles

If IUT does not implement memory transactions mark 1 through 8 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 8 N/A.

Test #	Description	Pass	N/A
1	Data transfer after write to fast memory slave.	✓	
2	Data transfer after read from fast memory slave.	✓	
3	Data transfer after write to medium memory slave.	✓	
4	Data transfer after read from medium memory slave.	✓	
5	Data transfer after write to slow memory slave.	✓	
6	Data transfer after read from slow memory slave.	✓	
7	Data transfer after write to subtractive memory slave.	✓	
8	Data transfer after read from subtractive memory slave.	✓	

If IUT does not implement I/O transactions mark 9 through 16 N/A. Else if IUT supports both read and write transactions DO NOT mark 9 through 16 N/A.

Test #	Description	Pass	N/A
9	Data transfer after write to fast I/O slave.	✓	
10	Data transfer after read from fast I/O slave.	✓	
11	Data transfer after write to medium I/O slave.	✓	
12	Data transfer after read from medium I/O slave.	✓	
13	Data transfer after write to slow I/O slave.	✓	
14	Data transfer after read from slow I/O slave.	✓	
15	Data transfer after write to subtractive I/O slave.	✓	
16	Data transfer after read from subtractive I/O slave.	✓	

If IUT does not implement configuration transactions mark 17 through 24 N/A. Else if IUT supports both read and write transactions DO NOT mark 17 through 24 N/A.

Note: Configuration transactions tested using VirtualChips environment.

Test #	Description	Pass	N/A
17	Data transfer after write to fast config slave.	✓	
18	Data transfer after read from fast config slave.	✓	
19	Data transfer after write to medium config slave.	✓	
20	Data transfer after read from medium config slave.	✓	
21	Data transfer after write to slow config slave.	✓	
22	Data transfer after read from slow config slave.	✓	
23	Data transfer after write to subtractive config slave.	✓	
24	Data transfer after read from subtractive config slave.	✓	

If IUT does not implement interrupt transactions mark 25 through 28 N/A else do not mark 25 through 28 N/A.

Test #	Description	Pass	N/A
25	Data transfer after interrupt acknowledge from fast slave.		✓
26	Data transfer after interrupt acknowledge from medium slave.		✓
27	Data transfer after interrupt acknowledge from slow slave.		✓
28	Data transfer after interrupt acknowledge from subtractive slave.		✓

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.3.

Test Scenario: 1.4. PCI Bus Single Data Phase Disconnect Cycles

If IUT does not implement memory transactions mark 1 through 8 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 8 N/A.

Test #	Description	Pass	N/A
1	Data transfer after write to fast memory slave.	✓	
2	Data transfer after read from fast memory slave.	✓	
3	Data transfer after write to medium memory slave.	✓	
4	Data transfer after read from medium memory slave.	✓	
5	Data transfer after write to slow memory slave.	✓	
6	Data transfer after read from slow memory slave.	✓	
7	Data transfer after write to subtractive memory slave.	✓	
8	Data transfer after read from subtractive memory slave.	✓	

If IUT does not implement I/O transactions mark 9 through 16 N/A. Else if IUT supports both read and write transactions DO NOT mark 9 through 16 N/A.

Test	Description	Pass	N/A
9	Data transfer after write to fast I/O slave.	✓	
10	Data transfer after read from fast I/O slave.	✓	
11	Data transfer after write to medium I/O slave.	✓	
12	Data transfer after read from medium I/O slave.	✓	
13	Data transfer after write to slow I/O slave.	✓	
14	Data transfer after read from slow I/O slave.	✓	
15	Data transfer after write to subtractive I/O slave.	✓	
16	Data transfer after read from subtractive I/O slave.	✓	

If IUT does not implement configuration transactions mark 17 through 24 N/A. Else if IUT supports both read and write transactions DO NOT mark 17 through 24 N/A.

Note: Configuration transactions tested using VirtualChips environment.

Test #	Description	Pass	N/A
17	Data transfer after write to fast config slave.	✓	
18	Data transfer after read from fast config slave.	✓	
19	Data transfer after write to medium config slave.	✓	
20	Data transfer after read from medium config slave.	✓	
21	Data transfer after write to slow config slave.	✓	
22	Data transfer after read from slow config slave.	✓	
23	Data transfer after write to subtractive config slave.	✓	
24	Data transfer after read from subtractive config slave.	✓	

If IUT does not implement interrupt transactions mark 25 through 28 N/A else do not mark 25 through 28 N/A.

Test #	Description	Pass	N/A
25	Data transfer after interrupt acknowledge from fast slave.		✓
26	Data transfer after interrupt acknowledge from medium slave.		✓
27	Data transfer after interrupt acknowledge from slow slave.		✓
28	Data transfer after interrupt acknowledge from subtractive slave.		✓

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.4.

Test Scenario: 1.5. PCI Bus Multi-Data Phase Target Abort Cycles

Note: The Initiator does not repeat a transaction because it is controlled by the user application. The user application must monitor the Received Target Abort bit (CSR28) in the Command Register to prevent a retry after a Target Abort.

If IUT does not implement memory transactions mark 1 through 16 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 16 N/A.

Test #	Description	Pass	N/A
1	Target Abort bit set after write to fast memory slave.	✓	
2	IUT does not repeat the write transaction.	✓	
3	IUT's Target Abort bit set after read from fast memory slave.	✓	
4	IUT does not repeat the read transaction.	✓	
5	Target Abort bit set after write to medium memory slave.	✓	
6	IUT does not repeat the write transaction.	✓	
7	IUT's Target Abort bit set after read from medium memory slave.	✓	
8	IUT does not repeat the read transaction.	✓	
9	Target Abort bit set after write to slow memory slave.	✓	
10	IUT does not repeat the write transaction.	✓	
11	IUT's Target Abort bit set after read from slow memory slave.	✓	
12	IUT does not repeat the read transaction.	✓	
13	Target Abort bit set after write to subtractive memory slave.	✓	
14	IUT does not repeat the write transaction.	✓	
15	IUT's Target Abort bit set after read from subtractive memory slave.	✓	
16	IUT does not repeat the read transaction.	✓	

If IUT does not implement dual address transactions mark 17 through 32 N/A. Else if IUT supports both read and write dual address transactions DO NOT mark 17 through 32 N/A.

Test #	Description	Pass	N/A
17	Target Abort bit set after write to fast memory slave.		✓
18	IUT does not repeat the write transaction.		✓
19	IUT's Target Abort bit set after read from fast memory slave.		✓
20	IUT does not repeat the read transaction.		✓
21	Target Abort bit set after write to medium memory slave.		✓
22	IUT does not repeat the write transaction.		✓
23	IUT's Target Abort bit set after read from medium memory slave.		✓
24	IUT does not repeat the read transaction.		✓
25	Target Abort bit set after write to slow memory slave.		✓
26	IUT does not repeat the write transaction.		✓
27	IUT's Target Abort bit set after read from slow memory slave.		✓
28	IUT does not repeat the read transaction.		✓
29	Target Abort bit set after write to subtractive memory slave.		✓
30	IUT does not repeat the write transaction.		✓
31	IUT's Target Abort bit set after read from subtractive memory slave.		✓

Test #	Description	Pass	N/A
32	IUT does not repeat the read transaction.		✓

If IUT does not implement configuration transactions mark 33 through 48 N/A. Else if IUT supports both read and write transactions DO NOT mark 33 through 48 N/A.

Note: Configuration transactions tested using VirtualChips environment.

Test #	Description	Pass	N/A
33	Target Abort bit set after write to fast config. slave.	✓	
34	IUT does not repeat the write transaction.	✓	
35	IUT's Target Abort bit set after read from fast config. slave.	✓	
36	IUT does not repeat the read transaction.	✓	
37	Target Abort bit set after write to medium config. slave.	✓	
38	IUT does not repeat the write transaction.	✓	
39	IUT's Target Abort bit set after read from medium config. slave.	✓	
40	IUT does not repeat the read transaction.	✓	
41	Target Abort bit set after write to slow config. slave.	✓	
42	IUT does not repeat the write transaction.	✓	
43	IUT's Target Abort bit set after read from slow config. slave.	✓	
44	IUT does not repeat the read transaction.	✓	
45	Target Abort bit set after write to subtractive config. slave.	✓	
46	IUT does not repeat the write transaction.	✓	
47	IUT's Target Abort bit set after read from subtractive config. slave.	✓	
48	IUT does not repeat the read transaction.	✓	

If IUT does not implement MEMORY READ MULTIPLE transactions mark 49 through 56 N/A. Else DO NOT mark 49 through 56 N/A.

Test #	Description	Pass	N/A
49	IUT's Target Abort bit set after read from fast memory slave.	✓	
50	IUT does not repeat the read transaction.	✓	
51	IUT's Target Abort bit set after read from medium memory slave.	✓	
52	IUT does not repeat the read transaction.	✓	
53	IUT's Target Abort bit set after read from slow memory slave.	✓	
54	IUT does not repeat the read transaction.	✓	
55	IUT's Target Abort bit set after read from subtractive memory slave.	✓	
56	IUT does not repeat the read transaction.	✓	

If IUT does not implement MEMORY READ LINE transactions mark 57 through 64 N/A. Else DO NOT mark 57 through 64 N/A.

Test #	Description	Pass	N/A
57	IUT's Target Abort bit set after read from fast memory slave.	✓	
58	IUT does not repeat the read transaction.	✓	
59	IUT's Target Abort bit set after read from medium memory slave.	✓	

Test #	Description	Pass	N/A
60	IUT does not repeat the read transaction.	✓	
61	IUT's Target Abort bit set after read from slow memory slave.	✓	
62	IUT does not repeat the read transaction.	✓	
63	IUT's Target Abort bit set after read from subtractive memory slave.	✓	
64	IUT does not repeat the read transaction.	✓	

If IUT does not implement MEMORY WRITE AND INVALIDATE transactions mark 65 through 72 N/A. Else DO NOT mark 65 through 72 N/A.

Test #	Description	Pass	N/A
65	Target Abort bit set after write to fast memory slave.		✓
66	IUT does not repeat the write transaction.		✓
67	Target Abort bit set after write to medium memory slave.		✓
68	IUT does not repeat the write transaction.		✓
69	Target Abort bit set after write to slow memory slave.		✓
70	IUT does not repeat the write transaction.		✓
71	IUT's Target Abort bit set after read from slow memory slave.		✓
72	IUT does not repeat the write transaction.		✓

Verify that the PCI core sets the Target Abort bit after trying to perform burst I/O transactions (Xilinx-only test).

Test #	Description	Pass	N/A
73	Target Abort bit set after write to fast I/O slave.	✓	
74	IUT does not repeat the write transaction.	✓	
75	IUT's Target Abort bit set after read from fast I/O slave.	✓	
76	IUT does not repeat the read transaction.	✓	
77	Target Abort bit set after write to medium I/O slave.	✓	
78	IUT does not repeat the write transaction.	✓	
79	IUT's Target Abort bit set after read from medium I/O slave.	✓	
80	IUT does not repeat the read transaction.	✓	
81	Target Abort bit set after write to slow I/O slave.	✓	
82	IUT does not repeat the write transaction.	✓	
83	IUT's Target Abort bit set after read from slow I/O slave.	✓	
84	IUT does not repeat the read transaction.	✓	
85	Target Abort bit set after write to subtractive I/O slave.	✓	
86	IUT does not repeat the write transaction.	✓	
87	IUT's Target Abort bit set after read from subtractive I/O slave.	✓	
88	IUT does not repeat the read transaction.	✓	

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.5.

Test Scenario: 1.6. PCI Bus Multi-Data Phase Retry Cycles

If IUT does not implement memory transactions mark 1 through 8 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 8 N/A.

Test #	Description	Pass	N/A
1	Data transfer after write to fast memory slave.	✓	
2	Data transfer after read from fast memory slave.	✓	
3	Data transfer after write to medium memory slave.	✓	
4	Data transfer after read from medium memory slave.	✓	
5	Data transfer after write to slow memory slave.	✓	
6	Data transfer after read from slow memory slave.	✓	
7	Data transfer after write to subtractive memory slave.	✓	
8	Data transfer after read from subtractive memory slave.	✓	

If IUT does not implement I/O transactions mark 9 through 16 N/A. Else if IUT supports both read and write transactions DO NOT mark 9 through 16 N/A.

Test #	Description	Pass	N/A
9	Data transfer after write to fast I/O slave.	✓	
10	Data transfer after read from fast I/O slave.	✓	
11	Data transfer after write to medium I/O slave.	✓	
12	Data transfer after read from medium I/O slave.	✓	
13	Data transfer after write to slow I/O slave.	✓	
14	Data transfer after read from slow I/O slave.	✓	
15	Data transfer after write to subtractive I/O slave.	✓	
16	Data transfer after read from subtractive I/O slave.	✓	

If IUT does not implement configuration transactions mark 17 through 24 N/A. Else if IUT supports both read and write transactions DO NOT mark 17 through 24 N/A.

Note: Configuration transactions tested using VirtualChips environment.

Test #	Description	Pass	N/A
17	Data transfer after write to fast config. slave.	✓	
18	Data transfer after read from fast config. slave.	✓	
19	Data transfer after write to medium config. slave.	✓	
20	Data transfer after read from medium config. slave.	✓	
21	Data transfer after write to slow config. slave.	✓	
22	Data transfer after read from slow config. slave.	✓	
23	Data transfer after write to subtractive config. slave.	✓	
24	Data transfer after read from subtractive config. slave.	✓	

If IUT does not implement MEMORY READ MULTIPLE transactions mark 25 through 28 N/A else do not mark 25 through 28 N/A.

Test #	Description	Pass	N/A
25	Data transfer after memory read multiple from fast slave.	✓	

Test #	Description	Pass	N/A
26	Data transfer after memory read multiple from medium slave.	✓	
27	Data transfer after memory read multiple from slow slave.	✓	
28	Data transfer after memory read multiple from subtractive slave.	✓	

If IUT does not implement MEMORY READ LINE transactions mark 29 through 32 N/A. Else do not mark 29 through 32 N/A.

Test #	Description	Pass	N/A
29	Data transfer after memory read line from fast slave.	✓	
30	Data transfer after memory read line from medium slave.	✓	
31	Data transfer after memory read line from slow slave.	✓	
32	Data transfer after memory read line from subtractive slave.	✓	

If IUT does not implement MEMORY WRITE AND INVALIDATE transactions mark 33 through 36 N/A. Else do not mark 33 through 36 N/A.

Test #	Description	Pass	N/A
33	Data transfer after memory write and invalidate to fast slave.		✓
34	Data transfer after memory write and invalidate to medium slave.		✓
35	Data transfer after memory write and invalidate to slow slave.		✓
36	Data transfer after memory write and invalidate to subtractive slave.		✓

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.6.

Test Scenario: 1.7. PCI Bus Multi-Data Phase Disconnect Cycles

If IUT does not implement multi-data phase memory transactions mark 1 through 8 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 8 N/A.

Test #	Description	Pass	N/A
1	Data transfer after write to fast memory slave.	✓	
2	Data transfer after read from fast memory slave.	✓	
3	Data transfer after write to medium memory slave.	✓	
4	Data transfer after read from medium memory slave.	✓	
5	Data transfer after write to slow memory slave.	✓	
6	Data transfer after read from slow memory slave.	✓	
7	Data transfer after write to subtractive memory slave.	✓	
8	Data transfer after read from subtractive memory slave.	✓	

If IUT does not implement I/O transactions mark 9 through 16 N/A. Else if IUT supports both read and write transactions DO NOT mark 9 through 16 N/A.

Test #	Description	Pass	N/A
9	Data transfer after write to fast I/O slave.	✓	
10	Data transfer after read from fast I/O slave.	✓	
11	Data transfer after write to medium I/O slave.	✓	
12	Data transfer after read from medium I/O slave.	✓	
13	Data transfer after write to slow I/O slave.	✓	
14	Data transfer after read from slow I/O slave.	✓	
15	Data transfer after write to subtractive I/O slave.	✓	
16	Data transfer after read from subtractive I/O slave.	✓	

If IUT does not implement configuration transactions mark 17 through 24 N/A. Else if IUT supports both read and write transactions DO NOT mark 17 through 24 N/A.

Note: Configuration transactions tested using VirtualChips environment.

Test #	Description	Pass	N/A
17	Data transfer after write to fast config. slave.	✓	
18	Data transfer after read from fast config. slave.	✓	
19	Data transfer after write to medium config. slave.	✓	
20	Data transfer after read from medium config. slave.	✓	
21	Data transfer after write to slow config. slave.	✓	
22	Data transfer after read from slow config. slave.	✓	
23	Data transfer after write to subtractive config. slave.	✓	
24	Data transfer after read from subtractive config. slave.	✓	

If IUT does not implement MEMORY READ MULTIPLE transactions mark 25 through 28 N/A else do not mark 25 through 28 N/A.

Test #	Description	Pass	N/A
25	Data transfer after memory read multiple from fast slave.	✓	

Test #	Description	Pass	N/A
26	Data transfer after memory read multiple from medium slave.	✓	
27	Data transfer after memory read multiple from slow slave.	✓	
28	Data transfer after memory read multiple from subtractive slave.	✓	

If IUT does not implement MEMORY READ LINE transactions mark 29 through 32 N/A else do not mark 29 through 32 N/A.

Test #	Description	Pass	N/A
29	Data transfer after memory read line from fast slave.	✓	
30	Data transfer after memory read line from medium slave.	✓	
31	Data transfer after memory read line from slow slave.	✓	
32	Data transfer after memory read line from subtractive slave.	✓	

If IUT does not implement MEMORY WRITE AND INVALIDATE transactions mark 33 through 36 N/A else do not mark 33 through 36 N/A.

Test #	Description	Pass	N/A
33	Data transfer after memory write and invalidate to fast slave.		✓
34	Data transfer after memory write and invalidate to medium slave.		✓
35	Data transfer after memory write and invalidate to slow slave.		✓
36	Data transfer after memory write and invalidate to subtractive slave.		✓

Note: Tests 1 through 36 are performed with a 4 double-word transfer size. Test 37 and 38 test for any effect with smaller transfer sizes. Single double-word transfers are tested in Scenario 1.4.

Test #	Description	Pass	N/A
37	Data transfer after memory write and memory read operations with a 3 double-word burst transfer size to fast memory slave.	✓	
38	Data transfer after memory write and memory read operations with a 2 double-word burst transfer size to fast memory slave.	✓	

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.7.

Test Scenario: 1.8. Multi-Data Phase & TRDY# Cycles

If IUT does not implement multi-data phase memory transactions mark 1 through 12 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 12 N/A.

All the Scenario 1.8 tests are four word transfers to a fast memory slave.

Test #	Description	Pass	N/A
1	Verify that data is written to primary target when TRDY# is released after 2nd rising clock edge and asserted on 3rd rising clock edge after FRAME#	✓	
2	Verify that data is read from primary target when TRDY# is released after 2nd rising clock edge and asserted on 3rd rising clock edge after FRAME#	✓	
3	Verify that data is written to primary target when TRDY# is released after 3rd rising clock edge and asserted on 4th rising clock edge after FRAME#	✓	
4	Verify that data is read from primary target when TRDY# is released after 3rd rising clock edge and asserted on 4th rising clock edge after FRAME#	✓	
5	Verify that data is written to primary target when TRDY# is released after 3rd rising clock edge and asserted on 5th rising clock edge after FRAME#	✓	
6	Verify that data is read from primary target when TRDY# is released after 3rd rising clock edge and asserted on 5th rising clock edge after FRAME#	✓	
7	Verify that data is written to primary target when TRDY# is released after 4th rising clock edge and asserted on 6th rising clock edge after FRAME#	✓	
8	Verify that data is read from primary target when TRDY# is released after 4th rising clock edge and asserted on 6th rising clock edge after FRAME#	✓	
9	Verify that data is written to primary target when TRDY# alternately released for one clock cycle and asserted for one clock cycle after FRAME#	✓	
10	Verify that data is read from primary target when TRDY# alternately released for one clock cycle and asserted for one clock cycle after FRAME#	✓	
11	Verify that data is written to primary target when TRDY# alternately released for two clock cycles and asserted for two clock cycles after FRAME#	✓	
12	Verify that data is read from primary target when TRDY# alternately released for two clock cycles and asserted for two clock cycles after FRAME#	✓	

If IUT does not implement DUAL ADDRESS transactions mark 13 through 24 N/A. Else if IUT supports both read and write transactions DO NOT mark 13 through 24 N/A.

Test #	Description	Pass	N/A
13	Verify that data is written to primary target when TRDY# released after 3rd rising clock edge and asserted on 4th rising clock edge after FRAME#		✓
14	Verify that data is read from primary target when TRDY# released after 3rd rising clock edge and asserted on 4th rising clock edge after FRAME#		✓
15	Verify that data is written to primary target when TRDY# released after 4th rising clock edge and asserted on 5th rising clock edge after FRAME#		✓
16	Verify that data is read from primary target when TRDY# released after 4th rising clock edge and asserted on 5th rising clock edge after FRAME#		✓
17	Verify that data is written to primary target when TRDY# released after 4th rising clock edge and asserted on 6th rising clock edge after FRAME#		✓

Test #	Description	Pass	N/A
18	Verify that data is read from primary target when TRDY# released after 4th rising clock edge and asserted on 6th rising clock edge after FRAME#		✓
19	Verify that data is written to primary target when TRDY# released after 5th rising clock edge and asserted on 7th rising clock edge after FRAME#		✓
20	Verify that data is read from primary target when TRDY# released after 5th rising clock edge and asserted on 7th rising clock edge after FRAME#		✓
21	Verify that data is written to primary target when TRDY# alternately released for one clock cycle and asserted for one clock cycle after FRAME#		✓
22	Verify that data is read from primary target when TRDY# alternately released for one clock cycle and asserted for one clock cycle after FRAME#		✓
23	Verify that data is written to primary target when TRDY# alternately released for two clock cycles and asserted for two clock cycles after FRAME#		✓
24	Verify that data is read from primary target when TRDY# alternately released for two clock cycles and asserted for two clock cycles after FRAME#		✓

If IUT does not implement MEMORY READ MULTIPLE transactions mark 25 through 30 N/A else do not mark 25 through 30 N/A.

Test #	Description	Pass	N/A
25	Verify that data is read from primary target when TRDY# released after 2nd rising clock edge and asserted on 3rd rising clock edge after FRAME#	✓	
26	Verify that data is read from primary target when TRDY# released after 3rd rising clock edge and asserted on 4th rising clock edge after FRAME#	✓	
27	Verify that data is read from primary target when TRDY# released after 3rd rising clock edge and asserted on 5th rising clock edge after FRAME#	✓	
28	Verify that data is read from primary target when TRDY# released after 4th rising clock edge and asserted on 6th rising clock edge after FRAME#	✓	
29	Verify that data is read from primary target when TRDY# alternately released for one clock cycle and asserted for one clock cycle after FRAME#	✓	
30	Verify that data is read from primary target when TRDY# alternately released for two clock cycles and asserted for two clock cycles after FRAME#	✓	

If IUT does not implement MEMORY READ LINE transactions mark 31 through 36 N/A else do not mark 31 through 36 N/A.

Test #	Description	Pass	N/A
31	Verify that data is read from primary target when TRDY# released after 2nd rising clock edge and asserted on 3rd rising clock edge after FRAME#	✓	
32	Verify that data is read from primary target when TRDY# released after 3rd rising clock edge and asserted on 4th rising clock edge after FRAME#	✓	
33	Verify that data is read from primary target when TRDY# released after 3rd rising clock edge and asserted on 5th rising clock edge after FRAME#	✓	
34	Verify that data is read from primary target when TRDY# released after 4th rising clock edge and asserted on 6th rising clock edge after FRAME#	✓	

Test #	Description	Pass	N/A
35	Verify that data is read from primary target when TRDY# alternately released for one clock cycle and asserted for one clock cycle after FRAME#	✓	
36	Verify that data is read from primary target when TRDY# alternately released for two clock cycles and asserted for two clock cycles after FRAME#	✓	

If IUT does not implement MEMORY WRITE and INVALIDATE transactions mark 37 through 42 N/A else do not mark 37 through 42 N/A.

Test #	Description	Pass	N/A
37	Verify that data is written to primary target when TRDY# released after 2nd rising clock edge and asserted on 3rd rising clock edge after FRAME#		✓
38	Verify that data is written to primary target when TRDY# released after 3rd rising clock edge and asserted on 4th rising clock edge after FRAME#		✓
39	Verify that data is written to primary target when TRDY# released after 3rd rising clock edge and asserted on 5th rising clock edge after FRAME#		✓
40	Verify that data is written to primary target when TRDY# released after 4th rising clock edge and asserted on 6th rising clock edge after FRAME#		✓
41	Verify that data is written to primary target when TRDY# alternately released for one clock cycle and asserted for one clock cycle after FRAME#		✓
42	Verify that data is written to primary target when TRDY# alternately released for two clock cycles and asserted for two clock cycles after FRAME#		✓

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.8.

Test Scenario: 1.9. Bus Data Parity Error Single Cycles

If IUT is exempted from reporting parity errors per the exemptions listed in subsection 3.7.2 of the specification then mark the following N/A.

If IUT does not implement memory transactions mark 1 through 3 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 3 N/A

Test #	Description	Pass	N/A
1	Verify the IUT sets Data Parity Error Detected bit when Primary Target asserts PERR# on IUT Memory Write	✓	
2	Verify that PERR# is active two clocks after the first data phase (which had odd parity) on IUT Memory Read	✓	
3	Verify the IUT sets Parity Error Detected bit when odd parity is detected on IUT Memory read	✓	

If IUT does not implement I/O transactions mark 4 through 6 N/A. Else if IUT supports both read and write transactions DO NOT mark 4 through 6 N/A.

Test #	Description	Pass	N/A
4	Verify the IUT sets Parity Error Detected bit when Primary Target asserts PERR# on IUT I/O Write	✓	
5	Verify that PERR# is active two clocks after the first data phase (which had odd parity) on IUT I/O Read	✓	
6	Verify the IUT sets Parity Error Detected bit when odd parity is detected on IUT I/O read	✓	

If IUT does not implement configuration transactions mark 7 through 9 N/A. Else if IUT supports both read and write transactions DO NOT mark 7 through 9 N/A.

Note: Configuration transactions tested using VirtualChips environment.

Test #	Description	Pass	N/A
7	Verify the IUT sets Parity Error Detected bit when Primary Target asserts PERR# on IUT Config Write	✓	
8	Verify that PERR# is active two clocks after the first data phase (which had odd parity) on IUT Config Read	✓	
9	Verify the IUT sets Parity Error Detected bit when odd parity is detected on IUT Config read	✓	

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.9.

Test Scenario: 1.10. Bus Data Parity Error Multi-Data Phase Cycles

If IUT is exempted from reporting parity errors per the exemptions listed in subsection 3.7.2 of the specification then mark the following N/A.

If IUT does not implement memory transactions mark 1 through 3 N/A. Else if IUT supports both read and write transactions DO NOT mark 1 through 3 N/A.

Test #	Description	Pass	N/A
1	Verify the IUT sets Parity Error Detected bit when Primary Target asserts PERR# on IUT multi data phase Memory Write	✓	
2	Verify that PERR# is active two clocks after the first data phase (which had odd parity) on IUT multi data phase Memory Read	✓	
3	Verify the IUT sets Parity Error Detected bit when odd parity is detected on IUT Memory multi data phase read	✓	

If IUT does not implement DUAL ADDRESS transactions mark 4 through 6 N/A. Else if IUT supports both read and write transactions DO NOT mark 4 through 6 N/A.

Test #	Description	Pass	N/A
4	Verify the IUT sets Parity Error Detected bit when Primary Target asserts PERR# on IUT dual address multi data phase Write		✓
5	Verify that PERR# is active two clocks after the first data phase (which had odd parity) on IUT dual address multi data phase Read		✓
6	Verify the IUT sets Parity Error Detected bit when odd parity is detected on IUT dual address multi data phase read		✓

If IUT does not implement configuration transactions mark 7 through 9 N/A. Else if IUT supports both read and write transactions DO NOT mark 7 through 9 N/A.

Note: Configuration transactions tested using VirtualChips environment.

Test #	Description	Pass	N/A
7	Verify the IUT sets Parity Error Detected bit when Primary Target asserts PERR# on IUT Config multi-data phase Write	✓	
8	Verify that PERR# is active two clocks after the first data phase (which had odd parity) on IUT Config multi-data phase Read	✓	
9	Verify the IUT sets Parity Error Detected bit when odd parity is detected on IUT Config multi-data phase read	✓	

If IUT does not implement MEMORY READ MULTIPLE transactions mark 10 through 11 N/A. Else DO NOT mark 10 through 11 N/A.

Test #	Description	Pass	N/A
10	Verify that PERR# is active two clocks after the first data phase (which had odd parity) on IUT mem. rd. multiple data phase.	✓	
11	Verify the IUT sets Parity Error Detected bit when odd parity is detected on IUT mem. rd. multiple data phase.	✓	

If IUT does not implement MEMORY READ LINE transactions mark 12 through 13 N/A. Else DO NOT mark 12 through 13 N/A.

Test #	Description	Pass	N/A
12	Verify that PERR# is active two clocks after the first data phase (which had odd parity) on IUT mem. rd. line data phase.	✓	
13	Verify the IUT sets Parity Error Detected bit when odd parity is detected on IUT mem. rd. line data phase.	✓	

If IUT does not implement MEMORY WRITE and INVALIDATE transactions mark 14 N/A. Else DO NOT mark 14 N/A.

Test #	Description	Pass	N/A
14	Verify the IUT sets Parity Error Detected bit when Primary Target asserts PERR# on IUT Memory Write and Invalidate data phase.		✓

Explanations:
Test 7-9: Core cannot perform multi-data phase configuration transaction. Will always issue a disconnect with
data each configuration transaction.

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.10.

Test Scenario: 1.11. Bus Master Timeout.

Note: Configuration transactions not tested.

Test #	Description	Pass	N/A
1	Memory write transaction terminates before 4 data phases completed	✓	
2	Memory read transaction terminates before 4 data phases completed	✓	
3	Config write transaction terminates before 4 data phases completed		✓
4	Config read transaction terminates before 4 data phases completed		✓
5	Memory read multiple transaction terminates before 4 data phases	✓	
6	Memory read line transaction terminates before 4 data phases	✓	
7	Dual Address write transaction terminates before 4 data phases completed		✓
8	Dual Address read transaction terminates before 4 data phases completed		✓

If IUT does not support cache coherent transactions mark 9 N/A Else if IUT supports cache coherent transactions and has implemented the configuration register that specifies cache line length DO NOT mark 9 N/A.

Test #	Description	Pass	N/A
9	Memory write invalidate terminates on line boundary		✓

Latency timer after 2 cycles, disconnect 2 clocks after DEVSEL# asserted (Xilinx-only test).

Test #	Description	Pass	N/A
10	Memory write transaction terminates correctly	✓	
11	Memory read transaction terminates correctly	✓	

Explanations:

The LogiCore Initiator does detect and respond to a Master Abort condition (no DEVSEL# asserted by the addressed Target). However, the LogiCore Initiator deasserts FRAME# and IRDY# one cycle later than the diagram shown as Figure 3-4 on page 40 in the PCI Local Bus Specification, revision 2.1. Otherwise, the LogiCore Initiator treats Master Abort normally. Master Abort is intended to keep an Initiator from “hanging” the bus when addressing a non-existent or malfunctioning Target. A one cycle latency should not adversely affect most designs. The extra clock cycle eliminates a critical path in the LogiCore Initiator control logic.

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.11.

Test Scenario: 1.12. Target Lock

If an IUT does not support lock transactions mark 1 through 5 N/A. Else if IUT supports lock transactions DO NOT mark 1 through 5 N/A.

Test #	Description	Pass	N/A
1	IUT does not perform bus transaction (read lock) on locked resource		✓
2	IUT does establish lock after lock is released		✓
3	IUT does release lock after write to primary target		✓
4	IUT does not establish lock when it detects retry		✓
5	IUT does not establish lock when it detects target abort		✓

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.12.

Test Scenario: 1.13. PCI Bus Master Parking

Verify that the IUT is able to drive PCI bus to stable conditions if it is idle and GNT# is asserted.

Test #	Description	Pass	N/A
1	IUT drives AD[31::00] to stable values within eight PCI Clocks of GNT#.	✓	
2	IUT drives C/BE[3::0]# to stable values within eight PCI Clocks of GNT#.	✓	
3	IUT drives PAR one clock cycle after IUT drives AD[31::0]	✓	

Verify that the IUT will Tri-state the bus when GNT# is not asserted.

Test #	Description	Pass	N/A
4	IUT Tri-states AD[31::00] and C/BE[3::0] and PAR when GNT# is released.	✓	

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.13

Test Scenario: 1.14. PCI Bus Master Arbitration

Verify that the IUT is able to complete bus transaction when GNT# is deasserted coincident with FRAME# asserted.

Complete bus transaction when GNT# is de-asserted coincident with FRAME# asserted (2-cycle GNT#). This is tested with both Memory Read and Memory Write.

Test #	Description	Pass	N/A
1	IUT completes transaction when de-asserting GNT# is coincident with asserting FRAME#.	✓	

Verify that IUT doesn't assert FRAME when GNT# is asserted for only one cycle (Xilinx-only test). This is tested with both Memory Read and Memory Write.

Test #	Description	Pass	N/A
2	IUT doesn't assert FRAME# when GNT# is asserted for only one cycle.	✓	

Wait for idle bus (FRAME# and IRDY# deasserted) after receiving GNT# before starting Initiator transaction (Xilinx-only test).

Test #	Description	Pass	N/A
3	IUT waits for idle bus after receiving GNT# before completing write and read transaction.	✓	

LogiCORE PCI Initiator does not attempt bus transaction if Bus Master Enable bit is not set in the Command Register (Xilinx-only test).

Test #	Description	Pass	N/A
4	Reset Bus Master Enable bit in the IUT command register. Attempt a write transaction. Verify that the IUT does not attempt a transaction.	✓	

LogiCORE PCI Initiator should attempt bus transaction if Bus Master Enable bit is set in the Command Register and a REQUEST is pending (Xilinx-only test).

Test #	Description	Pass	N/A
5	Set Bus Master Enable bit in the IUT command register. Attempt a write transaction. Verify that the IUT successfully completes transaction.	✓	

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

End of checklist for Scenario 1.14.

Component Protocol Checklist for a Target Device

For detail on tests to run in order to fill in this checklist, refer to the Target Protocol Test Scenarios in Addendum B.

Definition: IUT is an acronym for "Implementation Under Test".

Test Scenario: 2.1. Target Reception of an Interrupt Cycle.

If IUT does not respond to Interrupt Acknowledge bus transactions mark 1 through 2 N/A

Test #	Description	Pass	N/A
1.	IUT generates Interrupts when programmed		✓
2.	IUT clears Interrupts when serviced (may include driver specific actions)		✓

Test Scenario: 2.2. Target Reception of Special Cycle.

IF IUT does not implement Special Cycles mark 1 through 5 N/A

Test #	Description	Pass	N/A
1	No DEVSEL Assertion by IUT after Special Cycle	✓	
2	IUT receives encoded special cycle **Message received but not processed in macro. Would be processed by user application.**	✓	

Test Scenario: 2.3. Target Detection of Address and Data parity Error for Special Cycle.

Test #	Description	Pass	N/A
1	IUT reports address parity error via SERR	✓	
2	IUT reports data parity error via SERR		✓
3	IUT keeps SERR active for at least one clock		✓

Test Scenario: 2.4. Target Reception of I/O Cycles With Legal and Illegal Byte Enables.

IF IUT does not support I/O cycles mark 1 through 4 N/A or if IUT claims all 32 bits during an I/O cycle mark 1 and 2 N/A

Note: The LogiCORE Target supports 32-bit I/O transfers. The macro does not automatically generate Target Abort or Disconnect during illegal transfers. However, this function can be added by the user application.

Test #	Description	Pass	N/A
1	IUT asserts TRDY following 2nd rising edge from FRAME on all Legal BE"		✓
2	IUT terminates with TARGET Abort for each illegal BE"		✓
IF IUT supports Target disconnect Check the following			
3	IUT asserts STOP		✓
4	IUT de-asserts STOP after FRAME de-assertion		✓

Test Scenario: 2.5. Target Ignores Reserved Commands.

Test #	Description	Pass	N/A
1	IUT does not respond to RESERVED COMMANDS	✓	
2	Initiator detects master abort for each transfer	✓	
FOR 32bit TARGETS			
3	IUT does not respond to 64bit cycle (dual address)	✓	

Test Scenario: 2.6. Target Receives Configuration Cycles.

Note: The LogiCORE Target does not support burst transfers in or out of its configuration space.

Test #	Description	Pass	N/A
1	IUT responds to all configuration cycles type 0 read/write cycles appropriately	✓	
2	IUT does not respond to configuration cycles type 0 with IDSEL inactive	✓	
IF IUT does not support configuration type 1 mark 3 through 5 N/A			
Test #	Description	Pass	N/A
3	IUT responds to all configuration cycles type 1 read/write cycles appropriately		✓
4	IUT responds to all configuration cycles type 0 read/write cycles appropriately		✓
5	IUT does not respond (master abort) on illegal configuration cycle types		✓

Test Scenario: 2.7. Target Receives I/O Cycles With Address and Data Parity Errors.

Test #	Description	Pass	N/A
If IUT does not support I/O cycles mark N/A			
1	IUT reports address parity error via SERR during I/O read/write cycles	✓	
2	IUT reports data parity error via PERR during I/O write cycles	✓	

Test Scenario: 2.8. Target Gets Config. Cycles With Address and Data Parity Errors.

Test #	Description	Pass	N/A
1	IUT reports address parity error via SERR during configuration read/write cycles	✓	
2	IUT reports data parity error via PERR during configuration write cycles	✓	

Test Scenario: 2.9. Target Receives Memory Cycles.

Note: The LogiCORE Target does not automatically generate Target Abort when a burst transaction crosses an address boundary. However this function can be added by the user application.

Test #	Description	Pass	N/A
IF IUT does not interface to a memory subsystem mark all N/A			
1	IUT completes single memory read and write cycles appropriately	✓	
IF IUT does not interface to main system memory or memory is NOT CACHEABLE mark 2 to 4 N/A			
2	IUT completes memory read line cycles appropriately	✓	
3	IUT completes memory read multiple cycles appropriately	✓	
4	IUT completes memory write and invalidate cycles appropriately	✓	
5	IUT completes one cycle and disconnects on RESERVED memory operations	✓	
6	IUT disconnects on burst transactions that cross its address boundary		✓

Test Scenario: 2.10. Target Gets Memory Cycles With Address and Data Parity Errors.

Test #	Description	Pass	N/A
IF IUT does not interface to a memory subsystem mark 1 to 2 N/A			
1	IUT reports address parity error via SERR during all memory read and write cycles	✓	
2	IUT reports data parity error via PERR during all memory write cycles	✓	

Test Scenario: 2.11. Target Gets Fast Back to Back Cycles.

Test #	Description	Pass	N/A
1	IUT responds to back to back memory writes appropriately		✓
2	IUT responds to memory write followed by memory read appropriately		✓
IF IUT does not implement the "Fast Back-to-Back Bit" then mark 3 and 4 N/A			
3	IUT responds to back to back memory writes with 2nd write selecting IUT		✓
4	IUT responds to memory write followed by memory read with read selecting IUT		✓

Test Scenario: 2.12. Target Performs Exclusive Access Cycles.

If the IUT does not implement LOCK mark 1-4 N/A

Test #	Description	Pass	N/A
2	IUT responds with RETRY when second initiator attempts an access		✓
3	IUT responds to access releasing LOCK by initiator		✓
4	IUT responds to access by second initiator		✓

Test Scenario: 2.13. Target Gets Cycles With IRDY Used for Data Stepping.

Test #	Description	Pass	N/A
1	IUT responds appropriately with a wait state inserted on phase 1 of 3 data phases	✓	
2	IUT responds appropriately with a wait state inserted on phase 2 of 3 data phases	✓	
3	IUT responds appropriately with a wait state inserted on phase 3 of 3 data phases	✓	
4	IUT responds appropriately with a wait state inserted on all of 3 data phases	✓	
5	IUT responds appropriately when Initiator has maximum initial latency, eight wait states before IRDY# asserted. Perform Memory Write and Read. Xilinx-only test.	✓	

Test Scenario: 2.14. Target Signals and Responds to Various Target Termination Conditions (Xilinx-only tests)

Normal Termination Conditions (S_READY asserted, S_TERM deasserted, T_ABORT deasserted, KEEPOUT deasserted).

Test #	Description	Pass	N/A
1	IUT responds appropriately to a Memory Write Operation, S_READY asserted immediately.	✓	
2	IUT responds appropriately to a Memory Read Operation, S_READY asserted immediately.	✓	
3	IUT responds appropriately to a Memory Write Operation, S_READY asserted after 7 clocks.	✓	
4	IUT responds appropriately to a Memory Read Operation, S_READY asserted after 7 clocks.	✓	

Target Disconnect (S_READY asserted, S_TERM asserted, T_ABORT deasserted, KEEPOUT deasserted).

Test #	Description	Pass	N/A
5	IUT responds appropriately to a Memory Write Operation, disconnect with data on the first cycle.	✓	
6	IUT responds appropriately to a Memory Read Operation, disconnect with data on the first cycle.	✓	
7	IUT responds appropriately to a Memory Write Operation, disconnect with data after 7 cycles, S_READY and S_TERM asserted after 7 clocks.	✓	
8	IUT responds appropriately to a Memory Read Operation, disconnect with data after 7 cycles, S_READY and S_TERM asserted after 7 clocks.	✓	
9	IUT responds appropriately to a Memory Write Operation with 2 doublewords, disconnect on the second word.	✓	
10	IUT responds appropriately to a Memory Read Operation with 2 doublewords, disconnect on the second word.	✓	

Target Retry (S_READY deasserted, S_TERM asserted, T_ABORT deasserted, KEEPOUT deasserted on first transfer).

Test #	Description	Pass	N/A
11	IUT responds appropriately to a Memory Write Operation.	✓	
12	IUT responds appropriately to a Memory Read Operation.	✓	
13	IUT responds appropriately to a Memory Write Operation. Initiator asserts FRAME# and deasserts IRDY# for seven cycles. Verify IUT asserts STOP# until FRAME# deasserted.	✓	
14	IUT responds appropriately to a Memory Read Operation. Initiator asserts FRAME# and deasserts IRDY# for seven cycles. Verify IUT asserts STOP# until FRAME# deasserted.	✓	

Keepout (S_READY deasserted, S_TERM asserted, T_ABORT deasserted, KEEPOUT asserted).

Test #	Description	Pass	N/A
15	IUT responds to a Memory Write operation with a Target Retry. Verify that ADIO internal bus is Hi-Z.	✓	
16	IUT responds to a Memory Read operation with a Target Retry. Verify that ADIO internal bus is Hi-Z.	✓	
17	IUT responds to a Configuration Write operation with a Target Retry. Verify that ADIO internal bus is Hi-Z.	✓	
18	IUT responds to a Configuration Read operation with a Target Retry. Verify that ADIO internal bus is Hi-Z.	✓	

Target Abort (T_ABORT asserted, KEEPOUT deasserted).

Test #	Description	Pass	N/A
19	IUT responds appropriately to a two double-word Memory Write operation with a Target Abort.	✓	
20	Verify that Signaled Target Abort bit was set in the Status Register.	✓	
21	IUT responds appropriately to a two double-word Memory Read operation with a Target Abort.	✓	
22	Verify that Signaled Target Abort bit was set in the Status Register.	✓	

Test Scenario 2.x. Explanations.

Explanations:

This section should be used to clarify any answers on checklist items above. Please key explanation to item number.

XILINX®

Pinout and Configuration

Pinout and Configuration

Layout Considerations

The pinouts Xilinx supplies follow the PCI-SIG suggested pinout and aligns the PCI data path (ADIO[31:0]) along the horizontal long lines in the FPGA. The horizontal Longlines support internal 3-state busses. Various registers, such as the Base Address Registers, are aligned vertically, in columns.

Since the BG432 package used by the XC4062XLT is cavity down, the LogiCORE PCI Interface is placed on the other side of the die.

Pinout Tables

The following pinout tables list the PCI pin assignment for these supported packages:

- Table 1 - XC4013XLTPQ208
- Table 2 - XC4013XLTPQ240
- Table 3 - XC4028XLTPQ240
- Table 4 - XC4062XLTPQ240
- Table 5 - XC4062XLTBG432
- Table 6 - XCS30PQ208
- Table 7 - XCS30PQ240
- Table 8 - XCS40PQ208
- Table 9 - XCS40PQ240

For each pin, both the PCI function and the fundamental device pin function are listed. Those shown in bold italics are dedicated pins for configuring the FPGA device using one of the serial configuration modes. Pins without a PCI function listed are available as additional user I/O.

Note: If there are conflicts between these tables and the constraints file, the constraints file has precedence.

Configuration Mode

The LogiCORE PCI Interface is designed to use Serial Master Mode or Slave Mode for configuring the device. An external serial configuration device, such as the Xchecker cable or an embedded processor is required for Serial Slave Mode.

Use of the XC4000XLT or Spartan fast configuration mode is recommended to minimize the FPGA power-up configuration time. The fast mode is set as part of the MakeBits options in the XDM profile read in during the design compilation phase.

Please refer to the Xilinx Programmable Logic Data Book for additional information.

The Xchecker is useful during debugging. However, the PCI system needs to be held reset while the FPGA's bitstream is loaded. If using an embedded processor, the user needs to insure that configuration will not be delayed by interrupts to the processor, and that it is capable of configuring it prior to the assertion of IDSEL. Currently the v2.1 PCI Specification does not specify a time requirement for the initial asserting of IDSEL after power-on. Currently, there is a proposal for the v2.2 PCI Specification to require 2^25 clocks before configuring the PCI bus after power-on. Until this is required in the PCI Specification, Xilinx recommends use of the Master Serial fast mode to assure the FPGA is configured at the time of IDSEL assertion.

Pinout for the XC4013XLT PQ208

Table 1: Pinout for the XC4013XLT PQ208

Pin Function	PCI Function	PQ208
N.C.	N.C.	P1
GND	GND	P2
N.C.	N.C.	P3
I/O, GCK1 (A16)	PCLK	P4
I/O (A17)	AD23	P5
I/O	AD22	P6
I/O	AD21	P7
I/O, TDI	TDI	P8
I/O, TCK	TCK	P9
I/O	AD20	P10
I/O	AD19	P11
I/O	AD18	P12
I/O	AD17	P13
GND	GND	P14
I/O	AD16	P15
I/O	CBE2	P16
I/O, TMS	TMS	P17
VTT	VTT	P18
I/O	GNT-	P19
I/O	FRAME-	P20
I/O	IRDY-	P21
I/O	TRDY-	P22
I/O	DEVSEL-	P23
I/O	STOP-	P24
GND	GND	P25
VCC	VCC	P26
I/O	PERR-	P27
I/O	SERR-	P28
I/O	PAR	P29
I/O	REQ-	P30
I/O		P31
I/O		P32
VTT	VTT	P33
I/O		P34
I/O	CBE1	P35
I/O	AD15	P36
GND	GND	P37
I/O	AD14	P38
I/O	AD13	P39
I/O	AD12	P40
I/O	AD11	P41
I/O	AD10	P42
I/O	AD9	P43
I/O	AD8	P44
I/O		P45

Table 1: Pinout for the XC4013XLT PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O		P46
I/O, GCK2		P47
O (M1)	**M1**	**P48**
GND	GND	P49
I (M0)	**M0**	**P50**
N.C.	N.C.	P51
N.C.	N.C.	P52
N.C.	N.C.	P53
N.C.	N.C.	P54
VCC	VCC	P55
I (M2)	**M2**	**P56**
I/O, GCK3		P57
I/O (HDC)	**HDC**	**P58**
I/O	CBE0	P59
I/O	AD7	P60
I/O	AD6	P61
I/O (LDC-)	**LDC-**	**P62**
I/O	AD5	P63
I/O	AD4	P64
I/O	AD3	P65
I/O	AD2	P66
GND	GND	P67
I/O	AD1	P68
I/O		P69
I/O	AD0	P70
VTT	VTT	P71
I/O		P72
I/O		P73
I/O		P74
I/O		P75
I/O		P76
I/O (INIT-)	**INIT-**	**P77**
VCC	VCC	P78
GND	GND	P79
I/O		P80
I/O		P81
I/O		P82
I/O		P83
I/O		P84
I/O		P85
VTT	VTT	P86
I/O		P87
I/O		P88
I/O		P89
GND	GND	P90
I/O		P91

Table 1: Pinout for the XC4013XLT PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O		P92
I/O		P93
I/O		P94
I/O		P95
I/O		P96
I/O		P97
I/O		P98
I/O		P99
I/O, GCK4		P100
GND	GND	P101
N.C.	N.C.	P102
DONE	**DONE**	**P103**
N.C.	N.C.	P104
N.C.	N.C.	P105
VCC	VCC	P106
N.C.	N.C.	P107
PROGRAM-	**PROGRAM-**	**P108**
I/O (D7)	RST-	P109
I/O, GCK5		P110
I/O		P111
I/O		P112
I/O (D6)		P113
I/O		P114
I/O		P115
I/O		P116
I/O		P117
I/O		P118
GND	GND	P119
I/O		P120
VTT	VTT	P121
I/O (D5)		P122
I/O (CS0-)		P123
I/O		P124
I/O		P125
I/O		P126
I/O		P127
I/O (D4)		P128
I/O		P129
VCC	VCC	P130
GND	GND	P131
I/O (D3)		P132
I/O (RS-)		P133
I/O		P134
I/O		P135
I/O		P136
I/O		P137

Table 1: Pinout for the XC4013XLT PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O (D2)		P138
I/O		P139
VTT	VTT	P140
I/O		P141
GND	GND	P142
I/O		P143
I/O		P144
I/O		P145
I/O		P146
I/O (D1)		P147
I/O (RCLK, RDY/ BUSY)		P148
I/O		P149
I/O		P150
I/O (D0, DIN)	**DIN**	**P151**
I/O, GCK6 (DOUT)	**DOUT**	**P152**
CCLK	CCLK	P153
VCC	VCC	P154
N.C.	N.C.	P155
N.C.	N.C.	P156
N.C.	N.C.	P157
N.C.	N.C.	P158
O, TDO	TDO	P159
GND	GND	P160
I/O (A0, WS-)		P161
I/O, GCK7 (A1)		P162
I/O		P163
I/O		P164
I/O (CS1, A2)		P165
I/O (A3)		P166
I/O		P167
I/O		P168
I/O		P169
I/O		P170
GND	GND	P171
I/O		P172
VTT	VTT	P173
I/O (A4)		P174
I/O (A5)		P175
I/O		P176
I/O		P177
I/O(A21)		P178
I/O(A20)		P179
I/O (A6)		P180
I/O (A7)		P181
GND	GND	P182

Table 1: Pinout for the XC4013XLT PQ208 (Continued)

Pin Function	PCI Function	PQ208
VCC	VCC	P183
I/O (A8)		P184
I/O (A9)		P185
I/O (A19)		P186
I/O (A18)		P187
I/O		P188
I/O		P189
I/O (A10)		P190
I/O (A11)	AD31	P191
VTT	VTT	P192
I/O	AD30	P193
GND	GND	P194
I/O	AD29	P195
I/O	AD28	P196
I/O	AD27	P197
I/O	AD26	P198
I/O (A12)	AD25	P199
I/O (A13)	AD24	P200
I/O	CBE3	P201
I/O	IDSEL	P202
I/O (A14)		P203
I/O, GCK8 (A15)		P204
VCC	VCC	P205
N.C.	N.C.	P206
N.C.	N.C.	P207
N.C.	N.C.	P208

Pinout for the XC4013XLT PQ240

Table 2: Pinout for the XC4013XLT PQ240

Pin Function	PCI Function	PQ240
GND	GND	P1
I/O, GCK1 (A16)	PCLK	P2
I/O (A17)	AD23	P3
I/O		P4
I/O	AD22	P5
I/O, TDI	TDI	P6
I/O, TCK	TCK	P7
I/O	AD21	P8
I/O	AD20	P9
I/O	AD19	P10
I/O	AD18	P11
I/O		P12
I/O		P13
GND	GND	P14
I/O	AD17	P15
I/O		P16
I/O, TMS	TMS	P17
VTT	VTT	P18
VCC		P19
I/O	AD16	P20
I/O	CBE2	P21
GND‡	GND	P22
I/O	GNT-	P23
I/O	FRAME-	P24
I/O	IRDY-	P25
I/O	TRDY-	P26
I/O	DEVSEL-	P27
I/O	STOP-	P28
GND	GND	P29
VCC	VCC	P30
I/O	PERR-	P31
I/O	SERR-	P32
I/O	PAR	P33
I/O	REQ-	P34
I/O		P35
I/O		P36
GND‡	GND	P37
I/O		P38
I/O		P39
VCC	VCC	P40
VTT	VTT	P41
I/O		P42
I/O	CBE1	P43
I/O	AD15	P44
GND	GND	P45
I/O		P46
I/O		P47
I/O	AD14	P48
I/O	AD13	P49
I/O	AD12	P50
I/O	AD11	P51
I/O	AD10	P52
I/O	AD9	P53
I/O	AD8	P54
I/O		P55
I/O		P56
I/O, GCK2		P57
O (M1)	**M1**	**P58**
GND	GND	P59
I (M0)	**M0**	**P60**
VCC	VCC	P61
I (M2)	**M2**	**P62**
I/O, GCK3		P63
I/O (HDC)	**HDC**	**P64**
I/O	CBE0	P65
I/O	AD7	P66
I/O	AD6	P67
I/O (LDC-)	**LDC-**	**P68**
I/O	AD5	P69
I/O	AD4	P70
I/O	AD3	P71
I/O	AD2	P72
I/O		P73
I/O		P74
GND	GND	P75
I/O	AD1	P76
I/O		P77
I/O	AD0	P78
VTT	VTT	P79
VCC	VCC	P80
I/O		P81
I/O		P82
GND‡	GND	P83
I/O		P84
I/O		P85
I/O		P86
I/O		P87
I/O		P88
I/O (INIT-)	**INIT-**	**P89**
VCC	VCC	P90
GND	GND	P91

Table 2: Pinout for the XC4013XLT PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P92
I/O		P93
I/O		P94
I/O		P95
I/O		P96
I/O		P97
GND‡	GND	P98
I/O		P99
I/O		P100
VCC	VCC	P101
VTT	VTT	P102
I/O		P103
I/O		P104
I/O		P105
GND	GND	P106
I/O		P107
I/O		P108
I/O		P109
I/O		P110
I/O		P111
I/O		P112
I/O		P113
I/O		P114
I/O		P115
I/O		P116
I/O		P117
I/O, GCK4		P118
GND	GND	P119
DONE	**DONE**	**P120**
VCC	VCC	P121
PROGRAM-	**PROGRAM-**	**P122**
I/O (D7)	RST-	P123
I/O, GCK5		P124
I/O		P125
I/O		P126
I/O		P127
I/O		P128
I/O (D6)		P129
I/O		P130
I/O		P131
I/O		P132
I/O		P133
I/O		P134
GND	GND	P135
I/O		P136
I/O		P137

Table 2: Pinout for the XC4013XLT PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P138
VTT	VTT	P139
VCC	VCC	P140
I/O (D5)		P141
I/O (CS0-)		P142
GND‡	GND	P143
I/O		P144
I/O		P145
I/O		P146
I/O		P147
I/O (D4)		P148
I/O		P149
VCC	VCC	P150
GND	GND	P151
I/O (D3)		P152
I/O (RS-)		P153
I/O		P154
I/O		P155
I/O		P156
I/O		P157
GND‡	GND	P158
I/O (D2)		P159
I/O		P160
VCC	VCC	P161
VTT	VTT	P162
I/O		P163
I/O		P164
I/O		P165
GND	GND	P166
I/O		P167
I/O		P168
I/O		P169
I/O		P170
I/O		P171
I/O		P172
I/O (D1)		P173
I/O (RCLK, RDY/ BUSY)		P174
I/O		P175
I/O		P176
I/O (D0, DIN)	**DIN**	**P177**
I/O, GCK6 (DOUT)	**DOUT**	**P178**
CCLK	CCLK	P179
VCC	VCC	P180
O, TDO	TDO	P181
GND	GND	P182

Table 2: Pinout for the XC4013XLT PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O (A0, WS-)		P183
I/O, GCK7 (A1)		P184
I/O		P185
I/O		P186
I/O (CS1, A2)		P187
I/O (A3)		P188
I/O		P189
I/O		P190
I/O		P191
I/O		P192
I/O		P193
I/O		P194
N.C.	N.C.	P195
GND	GND	P196
I/O		P197
I/O		P198
I/O		P199
VTT	VTT	P200
VCC	VCC	P201
I/O (A4)		P202
I/O (A5)		P203
GND‡	GND	P204
I/O		P205
I/O		P206
I/O (A21)		P207
I/O (A20)		P208
I/O (A6)		P209
I/O (A7)		P210
GND	GND	P211
VCC	VCC	P212
I/O (A8)		P213

Table 2: Pinout for the XC4013XLT PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O (A9)		P214
I/O (A19)		P215
I/O (A18)		P216
I/O		P217
I/O		P218
GND‡	GND	P219
I/O (A10)		P220
I/O (A11)		P221
VCC	VCC	P222
VTT	VTT	P223
I/O	AD31	P224
I/O		P225
I/O	AD30	P226
GND	GND	P227
I/O	AD29	P228
I/O	AD28	P229
I/O	AD27	P230
I/O	AD26	P231
I/O (A12)	AD25	P232
I/O (A13)	AD24	P233
I/O		P234
I/O		P235
I/O	CBE3	P236
I/O	IDSEL	P237
I/O (A14)		P238
I/O, GCK8 (A15)		P239
VCC	VCC	P240

‡ Pins marked with this symbol are used for Ground connections on some revisions of the device. These pins may not physically connect to anything on the current device revision. However, they should be externally connected to Ground, if possible.

Pinout for the XC4028XLT HQ240

Table 3: Pinout for the XC4028XLT HQ240

Pin Function	PCI Function	HQ240
GND	GND	P1
I/O, GCK1 (A16)	PCLK	P2
I/O (A17)	AD23	P3
I/O		P4
I/O	AD22	P5
I/O, TDI	TDI	P6
I/O, TCK	TCK	P7
I/O	AD21	P8
I/O	AD20	P9
I/O	AD19	P10
I/O	AD18	P11
I/O		P12
I/O		P13
GND	GND	P14
I/O	AD17	P15
I/O		P16
I/O, TMS	TMS	P17
VTT	VTT	P18
VCC		P19
I/O	AD16	P20
I/O	CBE2	P21
GND	GND	P22
I/O	GNT-	P23
I/O	FRAME-	P24
I/O	IRDY-	P25
I/O	TRDY-	P26
I/O	DEVSEL-	P27
I/O	STOP-	P28
GND	GND	P29
VCC	VCC	P30
I/O	PERR-	P31
I/O	SERR-	P32
I/O	PAR	P33
I/O	REQ-	P34
I/O		P35
I/O		P36
GND	GND	P37
I/O		P38
I/O		P39
VCC	VCC	P40
VTT	VTT	P41
I/O		P42
I/O	CBE1	P43
I/O	AD15	P44
GND	GND	P45

Table 3: Pinout for the XC4028XLT HQ240 (Continued)

Pin Function	PCI Function	HQ240
I/O		P46
I/O		P47
I/O	AD14	P48
I/O	AD13	P49
I/O	AD12	P50
I/O	AD11	P51
I/O	AD10	P52
I/O	AD9	P53
I/O	AD8	P54
I/O		P55
I/O		P56
I/O, GCK2		P57
O (M1)	**M1**	**P58**
GND	GND	P59
I (M0)	**M0**	**P60**
VCC	VCC	P61
I (M2)	**M2**	**P62**
I/O, GCK3		P63
I/O (HDC)	**HDC**	**P64**
I/O	CBE0	P65
I/O	AD7	P66
I/O	AD6	P67
I/O (LDC-)	**LDC-**	**P68**
I/O	AD5	P69
I/O	AD4	P70
I/O	AD3	P71
I/O	AD2	P72
I/O		P73
I/O		P74
GND	GND	P75
I/O	AD1	P76
I/O		P77
I/O	AD0	P78
VTT	VTT	P79
VCC	VCC	P80
I/O		P81
I/O		P82
GND	GND	P83
I/O		P84
I/O		P85
I/O		P86
I/O		P87
I/O		P88
I/O (INIT-)	**INIT-**	**P89**
VCC	VCC	P90
GND	GND	P91

Table 3: Pinout for the XC4028XLT HQ240 (Continued)

Pin Function	PCI Function	HQ240
I/O		P92
I/O		P93
I/O		P94
I/O		P95
I/O		P96
I/O		P97
GND	GND	P98
I/O		P99
I/O		P100
VCC	VCC	P101
VTT	VTT	P102
I/O		P103
I/O		P104
I/O		P105
GND	GND	P106
I/O		P107
I/O		P108
I/O		P109
I/O		P110
I/O		P111
I/O		P112
I/O		P113
I/O		P114
I/O		P115
I/O		P116
I/O		P117
I/O, GCK4		P118
GND	GND	P119
DONE	**DONE**	**P120**
VCC	VCC	P121
PROGRAM-	**PROGRAM-**	**P122**
I/O (D7)	RST-	P123
I/O, GCK5		P124
I/O		P125
I/O		P126
I/O		P127
I/O		P128
I/O (D6)		P129
I/O		P130
I/O		P131
I/O		P132
I/O		P133
I/O		P134
GND	GND	P135
I/O		P136
I/O		P137

Table 3: Pinout for the XC4028XLT HQ240 (Continued)

Pin Function	PCI Function	HQ240
I/O		P138
VTT	VTT	P139
VCC	VCC	P140
I/O (D5)		P141
I/O (CS0-)		P142
GND	GND	P143
I/O		P144
I/O		P145
I/O		P146
I/O		P147
I/O (D4)		P148
I/O		P149
VCC	VCC	P150
GND	GND	P151
I/O (D3)		P152
I/O (RS-)		P153
I/O		P154
I/O		P155
I/O		P156
I/O		P157
GND	GND	P158
I/O (D2)		P159
I/O		P160
VCC	VCC	P161
VTT	VTT	P162
I/O		P163
I/O		P164
I/O		P165
GND	GND	P166
I/O		P167
I/O		P168
I/O		P169
I/O		P170
I/O		P171
I/O		P172
I/O (D1)		P173
I/O (RCLK, RDY/ BUSY)		P174
I/O		P175
I/O		P176
I/O (D0, DIN)	**DIN**	**P177**
I/O, GCK6 (DOUT)	**DOUT**	**P178**
CCLK	CCLK	P179
VCC	VCC	P180
O, TDO	TDO	P181
GND	GND	P182

Table 3: Pinout for the XC4028XLT HQ240 (Continued)

Pin Function	PCI Function	HQ240
I/O (A0, WS-)		P183
I/O, GCK7 (A1)		P184
I/O		P185
I/O		P186
I/O (CS1, A2)		P187
I/O (A3)		P188
I/O		P189
I/O		P190
I/O		P191
I/O		P192
I/O		P193
I/O		P194
I/O		P195
GND	GND	P196
I/O		P197
I/O		P198
I/O		P199
VTT	VTT	P200
VCC	VCC	P201
I/O (A4)		P202
I/O (A5)		P203
GND‡	GND	P204
I/O		P205
I/O		P206
I/O (A21)		P207
I/O (A20)		P208
I/O (A6)		P209
I/O (A7)		P210
GND	GND	P211
VCC	VCC	P212
I/O (A8)		P213
I/O (A9)		P214
I/O (A19)		P215
I/O (A18)		P216
I/O		P217
I/O		P218
GND‡	GND	P219
I/O (A10)		P220
I/O (A11)		P221
VCC	VCC	P222
VTT	VTT	P223
I/O	AD31	P224
I/O		P225
I/O	AD30	P226
GND	GND	P227
I/O	AD29	P228

Table 3: Pinout for the XC4028XLT HQ240 (Continued)

Pin Function	PCI Function	HQ240
I/O	AD28	P229
I/O	AD27	P230
I/O	AD26	P231
I/O (A12)	AD25	P232
I/O (A13)	AD24	P233
I/O		P234
I/O		P235
I/O	CBE3	P236
I/O	IDSEL	P237
I/O (A14)		P238
I/O, GCK8 (A15)		P239
VCC	VCC	P240

Pinout for the XC4062XLT HQ240

Table 4: Pinout for the XC4062XLT HQ240

Pin Function	PCI Function	PQ240
GND	GND	P1
I/O, GCK1 (A16)	PCLK	P2
I/O (A17)	AD31	P3
I/O		P4
I/O	AD22	P5
I/O, TDI	TDI	P6
I/O, TCK	TCK	P7
I/O	AD21	P8
I/O	AD20	P9
I/O	AD19	P10
I/O	AD18	P11
I/O		P12
I/O		P13
GND	GND	P14
I/O	AD17	P15
I/O		P16
I/O, TMS	TMS	P17
VTT	VTT	P18
VCC		P19
I/O	AD16	P20
I/O	CBE2	P21
GND	GND	P22
I/O	GNT-	P23
I/O	FRAME-	P24
I/O	IRDY-	P25
I/O	TRDY-	P26
I/O	DEVSEL-	P27
I/O	STOP-	P28
GND	GND	P29
VCC	VCC	P30
I/O	PERR-	P31
I/O	SERR-	P32
I/O	PAR	P33
I/O	REQ-	P34
I/O		P35
I/O		P36
GND	GND	P37
I/O		P38
I/O		P39
VCC	VCC	P40
VTT	VTT	P41
I/O		P42
I/O	CBE1	P43
I/O	AD15	P44
GND	GND	P45

Table 4: Pinout for the XC4062XLT HQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P46
I/O		P47
I/O	AD14	P48
I/O	AD13	P49
I/O	AD12	P50
I/O	AD11	P51
I/O	AD10	P52
I/O	AD9	P53
I/O	AD8	P54
I/O		P55
I/O		P56
I/O, GCK2		P57
O (M1)	**M1**	**P58**
GND	GND	P59
I (M0)	**M0**	**P60**
VCC	VCC	P61
I (M2)	**M2**	**P62**
I/O, GCK3		P63
I/O (HDC)	**HDC**	**P64**
I/O	CBE0	P65
I/O	AD7	P66
I/O	AD6	P67
I/O (LDC-)	**LDC-**	**P68**
I/O	AD5	P69
I/O	AD4	P70
I/O	AD3	P71
I/O	AD2	P72
I/O		P73
I/O		P74
GND	GND	P75
I/O	AD1	P76
I/O		P77
I/O	AD0	P78
VTT	VTT	P79
VCC	VCC	P80
I/O		P81
I/O		P82
GND	GND	P83
I/O		P84
I/O		P85
I/O		P86
I/O		P87
I/O		P88
I/O (INIT-)	**INIT-**	**P89**
VCC	VCC	P90
GND	GND	P91

Table 4: Pinout for the XC4062XLT HQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P92
I/O		P93
I/O		P94
I/O		P95
I/O		P96
I/O		P97
GND	GND	P98
I/O		P99
I/O		P100
VCC	VCC	P101
VTT	VTT	P102
I/O		P103
I/O		P104
I/O		P105
GND	GND	P106
I/O		P107
I/O		P108
I/O		P109
I/O		P110
I/O		P111
I/O		P112
I/O		P113
I/O		P114
I/O		P115
I/O		P116
I/O		P117
I/O, GCK4		P118
GND	GND	P119
DONE	**DONE**	**P120**
VCC	VCC	P121
PROGRAM-	**PROGRAM-**	**P122**
I/O (D7)	RST-	P123
I/O, GCK5		P124
I/O		P125
I/O		P126
I/O		P127
I/O		P128
I/O (D6)		P129
I/O		P130
I/O		P131
I/O		P132
I/O		P133
I/O		P134
GND	GND	P135
I/O		P136
I/O		P137

Table 4: Pinout for the XC4062XLT HQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P138
VTT	VTT	P139
VCC	VCC	P140
I/O (D5)		P141
I/O (CS0-)		P142
GND	GND	P143
I/O		P144
I/O		P145
I/O		P146
I/O		P147
I/O (D4)		P148
I/O		P149
VCC	VCC	P150
GND	GND	P151
I/O (D3)		P152
I/O (RS-)		P153
I/O		P154
I/O		P155
I/O		P156
I/O		P157
GND	GND	P158
I/O (D2)		P159
I/O		P160
VCC	VCC	P161
VTT	VTT	P162
I/O		P163
I/O		P164
I/O		P165
GND	GND	P166
I/O		P167
I/O		P168
I/O		P169
I/O		P170
I/O		P171
I/O		P172
I/O (D1)		P173
I/O (RCLK, RDY/ BUSY)		P174
I/O		P175
I/O		P176
I/O (D0, DIN)	**DIN**	**P177**
I/O, GCK6 (DOUT)	**DOUT**	**P178**
CCLK	CCLK	P179
VCC	VCC	P180
O, TDO	TDO	P181
GND	GND	P182

Table 4: Pinout for the XC4062XLT HQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O (A0, WS-)		P183
I/O, GCK7 (A1)		P184
I/O		P185
I/O		P186
I/O (CS1, A2)		P187
I/O (A3)		P188
I/O		P189
I/O		P190
I/O		P191
I/O		P192
I/O		P193
I/O		P194
I/O		P195
GND	GND	P196
I/O		P197
I/O		P198
I/O		P199
VTT	VTT	P200
VCC	VCC	P201
I/O (A4)		P202
I/O (A5)		P203
GND‡	GND	P204
I/O		P205
I/O		P206
I/O (A21)		P207
I/O (A20)		P208
I/O (A6)		P209
I/O (A7)		P210
GND	GND	P211
VCC	VCC	P212
I/O (A8)		P213

Table 4: Pinout for the XC4062XLT HQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O (A9)		P214
I/O (A19)		P215
I/O (A18)		P216
I/O		P217
I/O		P218
GND‡	GND	P219
I/O (A10)		P220
I/O (A11)		P221
VCC	VCC	P222
VTT	VTT	P223
I/O	AD31	P224
I/O		P225
I/O	AD30	P226
GND	GND	P227
I/O	AD29	P228
I/O	AD28	P229
I/O	AD27	P230
I/O	AD26	P231
I/O (A12)	AD25	P232
I/O (A13)	AD24	P233
I/O		P234
I/O		P235
I/O	CBE3	P236
I/O	IDSEL	P237
I/O (A14)		P238
I/O, GCK8 (A15)		P239
VCC	VCC	P240

‡ Pins marked with this symbol are used for Ground connections on some revisions of the device. These pins may not physically connect to anything on the current device revision. However, they should be externally connected to Ground, if possible.

Pinout for the XC4062XLT BG432

Table 5: Pinout for the XC4062XLT BG432

Pin Function	PCI Function	BG432
I/O, GCK1 (A16)		D29
I/O (A17)		C30
I/O		E28
I/O		E29
I/O, TDI	TDI	D30
I/O, TCK	TCK	D31
I/O		F28
I/O		F29
I/O		E30
I/O		E31
I/O		G28
I/O		G29
I/O		F30
I/O		F31
I/O		H28
I/O		H29
I/O		G30
I/O		H30
I/O		J28
I/O		J29
I/O		H31
I/O		J30
I/O		K28
I/O		K29
I/O, TMS	TMS	K30
VTT	VTT	K31
I/O		L29
I/O		L30
I/O		M30
I/O		M28
I/O		M29
I/O		M31
I/O		N31
I/O		N28
I/O		N29
I/O		N30
I/O		P30
I/O		P28
I/O		P29
I/O		R31
I/O		R30
I/O		R28
I/O		R29
I/O		T31
I/O		T30
I/O		T29
I/O		U31
I/O		U30
I/O		U28
I/O		U29
I/O		V30
I/O		V29
I/O		V28
I/O		W31
I/O		W30
I/O		W29
I/O		W28
I/O		Y31
I/O		Y30
I/O		Y29
I/O		Y28
I/O		AA30
VTT	VTT	AA29
I/O		AB31
I/O		AB30
I/O		AB29
I/O		AB28
I/O		AC30
I/O		AC29
I/O		AC28
I/O		AD31
I/O		AD30
I/O		AD29
I/O		AD28
I/O		AE30
I/O		AE29
I/O		AF31
I/O		AE28
I/O		AF30
I/O		AF29
I/O		AG31
I/O		AF28
I/O		AG30
I/O		AG29
I/O		AH31
I/O		AG28
I/O		AH30
I/O, GCK2		AJ30
O (M1)	**M1**	**AH29**
I (M0)	**M0**	**AH28**
I (M2)	**M2**	**AJ28**
I/O, GCK3		AK29

Table 5: Pinout for the XC4062XLT BG432 (Continued)

Pin Function	PCI Function	BG432
I/O (HDC)	**HDC**	**AH27**
I/O		AK28
I/O		AJ27
I/O		AL28
I/O (LDC-)	**LDC-**	**AH26**
I/O		AK27
I/O		AJ26
I/O		AL27
I/O		AH25
I/O		AK26
I/O		AL26
I/O		AH24
I/O		AJ25
I/O		AK25
I/O		AJ24
I/O		AH23
I/O		AK24
I/O		AL24
I/O		AH22
I/O		AJ23
I/O		AK23
I/O		AJ22
I/O		AK22
I/O		AL22
VTT	VTT	AJ21
I/O		AH20
I/O		AK21
I/O		AJ20
I/O		AH19
I/O		AK20
I/O		AJ19
I/O		AL20
I/O		AH18
I/O		AK19
I/O		AJ18
I/O		AL19
I/O		AK18
I/O		AH17
I/O		AJ17
I/O		AK17
I/O		AL17
I/O		AJ16
I/O (INIT-)	**INIT-**	**AK16**
I/O		AL16
I/O		AH15
I/O		AL15
I/O		AJ15

Table 5: Pinout for the XC4062XLT BG432 (Continued)

Pin Function	PCI Function	BG432
I/O		AK15
I/O		AJ14
I/O		AH14
I/O		AK14
I/O		AL13
I/O		AK13
I/O		AJ13
I/O		AH13
I/O		AL12
I/O		AK12
I/O		AJ12
I/O		AK11
I/O		AH12
I/O		AJ11
VTT	VTT	AL10
I/O		AK10
I/O		AJ10
I/O		AK9
I/O		AL8
I/O		AH10
I/O		AJ9
I/O		AK8
I/O		AJ8
I/O		AH9
I/O		AK7
I/O		AL6
I/O		AJ7
I/O		AH8
I/O		AK6
I/O		AL5
I/O		AH7
I/O		AJ6
I/O		AK5
I/O		AL4
I/O		AH6
I/O		AJ5
I/O		AK4
I/O		AH5
I/O		AK3
I/O, GCK4		AJ4
DONE	**DONE**	**AH4**
PROGRAM-	**PROGRAM-**	**AH3**
I/O (D7)		AJ2
I/O, GCK5	PCLK	AG4
I/O	AD0	AG3
I/O	AD1	AH2
I/O	AD2	AH1

Table 5: Pinout for the XC4062XLT BG432 (Continued)

Pin Function	PCI Function	BG432
I/O	AD3	AF4
I/O	AD4	AF3
I/O	AD5	AG2
I/O	AD6	AG1
I/O	AD7	AE4
I/O		AE3
I/O		AF2
I/O (D6)		AF1
I/O	CBE0	AD4
I/O	AD8	AD3
I/O	AD9	AE2
I/O	AD10	AD2
I/O		AC4
I/O	AD11	AC3
I/O	AD12	AD1
I/O	AD13	AC2
I/O	AD14	AB4
I/O	AD15	AB3
I/O	CBE1	AB2
I/O	REQ-	AB1
VTT	VTT	AA3
I/O (D5)		AA2
I/O (CS0-)		Y2
I/O		Y4
I/O		Y3
I/O		Y1
I/O	PAR	W1
I/O	SERR-	W4
I/O		W3
I/O	PERR-	W2
I/O		V2
I/O	STOP-	V4
I/O	DEVSEL-	V3
I/O	TRDY-	U1
I/O	IRDY-	U2
I/O	FRAME-	U4
I/O	GNT-	U3
I/O (D4)		T1
I/O	CBE2	T2
I/O (D3)		T3
I/O (RS-)		R1
I/O	AD16	R2
I/O	AD17	R4
I/O	AD18	R3
I/O	AD19	P2
I/O	AD20	P3
I/O	AD21	P4

Table 5: Pinout for the XC4062XLT BG432 (Continued)

Pin Function	PCI Function	BG432
I/O		N1
I/O		N2
I/O	AD22	N3
I/O	AD23	N4
I/O	IDSEL	M1
I/O	CBE3	M2
I/O		M3
I/O		M4
I/O (D2)		L2
I/O		L3
VTT	VTT	K1
I/O	AD24	K2
I/O	AD25	K3
I/O	AD26	K4
I/O	AD27	J2
I/O	AD28	J3
I/O	AD29	J4
I/O	AD30	H1
I/O	AD31	H2
I/O		H3
I/O		H4
I/O		G2
I/O		G3
I/O		F1
I/O (D1)		G4
I/O (RCLK, RDY/ BUSY)		F2
I/O		F3
I/O		E1
I/O		F4
I/O		E2
I/O		E3
I/O		D1
I/O		E4
I/O		D2
I/O (D0, DIN)	**DIN**	**C2**
I/O, GCK6 (DOUT)	**DOUT**	**D3**
CCLK	CCLK	D4
O, TDO	TDO	C4
I/O (A0, WS-)		B3
I/O, GCK7 (A1)		D5
I/O		B4
I/O		C5
I/O		A4
I/O		D6
I/O		B5

Table 5: Pinout for the XC4062XLT BG432 (Continued)

Pin Function	PCI Function	BG432
I/O		C6
I/O (CS1, A2)		A5
I/O (A3)		D7
I/O		B6
I/O		A6
I/O		D8
I/O		C7
I/O		B7
I/O		D9
I/O		B8
I/O		A8
I/O		D10
I/O		C9
I/O	I/O	B9
I/O		C10
I/O		B10
I/O		A10
I/O		C11
VTT	VTT	D12
I/O		B11
I/O		C12
I/O		D13
I/O		B12
I/O		C13
I/O		A12
I/O		D14
I/O		B13
I/O (A4)		C14
I/O (A5)		A13
I/O		B14
I/O		D15
I/O (A21)		C15
I/O (A20)		B15
I/O		A15
I/O		C16
I/O (A6)		B16
I/O (A7)		A16
I/O (A8)		D17
I/O (A9)		A17
I/O		C17
I/O		B17
I/O (A19)		C18
I/O (A18)		D18
I/O		B18
I/O		A19
I/O (A10)		B19
I/O (A11)		C19

Table 5: Pinout for the XC4062XLT BG432 (Continued)

Pin Function	PCI Function	BG432
I/O		D19
I/O		A20
I/O		B20
I/O		C20
I/O		B21
I/O		D20
I/O		C21
I/O		A22
VTT	VTT	B22
I/O		C22
I/O		B23
I/O		A24
I/O		D22
I/O		C23
I/O		B24
I/O		C24
I/O		D23
I/O		B25
I/O		A26
I/O		C25
I/O (A12)		D24
I/O (A13)		B26
I/O		A27
I/O		D25
I/O		C26
I/O		B27
I/O		A28
I/O		D26
I/O		C27
I/O		B28
I/O		D27
I/O		B29
I/O (A14)		C28
I/O, GCK8 (A15)		D28

BG432

VCC Pins						
A1	A11	A21	A31	C3	C29	D11
D21	L1	L4	L28	L31	AA1	AA4
AA28	AA31	AH11	AH21	AJ3	AJ29	AL1
AL11	AL21	AL31	-	-	-	-
GND Pins						
A2	A3	A7	A9	A14	A18	A23
A25	A29	A30	B1	B2	B30	B31
C1	C31	D16	G1	G31	J1	J31
P1	P31	T4	T28	V1	V31	AC1
AC31	AE1	AE31	AH16	AJ1	AJ31	AK1
AK2	AK30	AK31	AL2	AL3	AL7	AL9
AL14	AL18	AL23	AL25	AL29	AL30	-
N.C. Pins						
C8	-	-	-	-	-	-

Pinout for the XCS30 PQ208

Table 6: Pinout for the XCS30 PQ208

Pin Function	PCI Function	PQ208
GND	GND	P1
I/O, PGCK1	PCLK	P2
I/O		P3
I/O		P4
I/O	AD23	P5
I/O, TDI	TDI	P6
I/O, TCK	TCK	P7
I/O	AD22	P8
I/O	AD21	P9
I/O	AD20	P10
I/O	AD19	P11
I/O	AD18	P12
GND	GND	P13
I/O	AD17	P14
I/O	AD16	P15
I/O, TMS	TMS	P16
I/O		P17
VCC	VCC	P18
I/O	GNT-	P19
I/O	FRAME-	P20
I/O	IRDY-	P21
I/O	TRDY-	P22
I/O	DEVSEL-	P23
I/O	STOP-	P24
GND	GND	P25
VCC	VCC	P26
I/O	PERR-	P27
I/O	SERR-	P28
I/O	PAR	P29
I/O	REQ-	P30
I/O		P31
I/O	CBE2	P32
VCC	VCC	P33
I/O		P34
I/O	CBE1	P35
I/O	AD15	P36
I/O	AD14	P37
GND	GND	P38
I/O	AD13	P39
I/O	AD12	P40
I/O	AD11	P41
I/O	AD10	P42
I/O	AD9	P43
I/O	AD8	P44
I/O		P45
I/O		P46
I/O		P47
I/O		P48
I/O, SGCK2		P49
N.C.	N.C.	P50
GND	GND	P51
MODE	**MODE**	P52
VCC	VCC	P53
N.C.	N.C.	P54
I/O, PGCK2		P55
I/O (HDC)	**HDC**	**P56**
I/O		P57
I/O	I/O	P58
I/O	CBE0	P59
I/O (LDC-)	**LDC-**	**P60**
I/O	AD7	P61
I/O	AD6	P62
I/O	AD5	P63
I/O	AD4	P64
I/O	AD3	P65
GND	GND	P66
I/O	AD2	P67
I/O	AD1	P68
I/O	AD0	P69
I/O		P70
VCC	VCC	P71
I/O		P72
I/O		P73
I/O		P74
I/O		P75
I/O		P76
I/O (INIT-)	**INIT-**	**P77**
VCC	VCC	P78
GND	GND	P79
I/O		P80
I/O		P81
I/O		P82
I/O		P83
I/O		P84
I/O		P85
VCC	VCC	P86
I/O		P87
I/O		P88
I/O		P89
I/O	I/O	P90
GND	GND	P91

Table 6: Pinout for the XCS30 PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O		P92
I/O		P93
I/O		P94
I/O		P95
I/O		P96
I/O		P97
I/O		P98
I/O		P99
I/O		P100
I/O		P101
I/O, SGCK3		P102
GND	GND	P103
DONE	**DONE**	**P104**
VCC	VCC	P105
PROGRAM-	**PROGRAM-**	**P106**
I/O	I/O	P107
I/O, PGCK3		P108
I/O	RST-	P109
I/O		P110
I/O		P111
I/O		P112
I/O		P113
I/O		P114
I/O		P115
I/O		P116
I/O		P117
GND	GND	P118
I/O		P119
I/O		P120
VCC	VCC	P121
I/O		P122
I/O		P123
I/O		P124
I/O		P125
I/O		P126
I/O		P127
I/O		P128
I/O		P129
VCC	VCC	P130
GND	GND	P131
I/O		P132
I/O		P133
I/O		P134
I/O		P135
I/O		P136
I/O		P137

Table 6: Pinout for the XCS30 PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O		P138
I/O		P139
VCC	VCC	P140
I/O		P141
I/O		P142
GND	GND	P143
I/O		P144
I/O		P145
I/O		P146
I/O		P147
I/O		P148
I/O		P149
I/O		P150
I/O		P151
I/O		P152
I/O (DIN)	**DIN**	**P153**
I/O, SGCK4 (DOUT)	**DOUT**	**P154**
CCLK	CCLK	P155
VCC	VCC	P156
O, TDO	TDO	P157
GND	GND	P158
I/O		P159
I/O, PGCK4		P160
I/O		P161
I/O		P162
I/O		P163
I/O		P164
I/O		P165
I/O		P166
I/O		P167
I/O		P168
I/O		P169
GND	GND	P170
I/O		P171
I/O		P172
VCC	VCC	P173
I/O		P174
I/O		P175
I/O		P176
I/O		P177
I/O		P178
I/O		P179
I/O		P180
I/O		P181
GND	GND	P182
VCC	VCC	P183

Table 6: Pinout for the XCS30 PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O		P184
I/O		P185
I/O		P186
I/O		P187
I/O		P188
I/O		P189
I/O		P190
I/O	AD31	P191
VCC	VCC	P192
I/O	AD30	P193
I/O	AD29	P194
GND	GND	P195
I/O	AD28	P196
I/O	AD27	P197
I/O	AD26	P198
I/O	AD25	P199
I/O	AD24	P200
I/O	CBE3	P201
I/O	IDSEL	P202
I/O		P203
I/O		P204
I/O		P205
I/O		P206
I/O, SGCK1		P207
VCC	VCC	P208

Pinout for the XCS30 PQ240

Table 7: Pinout for the XCS30 PQ240

Pin Function	PCI Function	PQ240
GND	GND	P1
I/O, PGCK1	PCLK	P2
I/O	AD23	P3
I/O		P4
I/O	AD22	P5
I/O, TDI	TDI	P6
I/O, TCK	TCK	P7
I/O	AD21	P8
I/O	AD20	P9
I/O	AD19	P10
I/O	AD18	P11
I/O		P12
I/O		P13
GND	GND	P14
I/O	AD17	P15
I/O		P16
I/O, TMS	TMS	P17
I/O		P18
VCC		P19
I/O	AD16	P20
I/O	CBE2	P21
GND‡	GND	P22
I/O	GNT-	P23
I/O	FRAME-	P24
I/O	IRDY-	P25
I/O	TRDY-	P26
I/O	DEVSEL-	P27
I/O	STOP-	P28
GND	GND	P29
VCC	VCC	P30
I/O	PERR-	P31
I/O	SERR-	P32
I/O	PAR	P33
I/O	REQ-	P34
I/O		P35
I/O		P36
GND‡	GND	P37
I/O		P38
I/O		P39
VCC	VCC	P40
I/O		P41
I/O		P42
I/O	CBE1	P43
I/O	AD15	P44
GND	GND	P45

Table 7: Pinout for the XCS30 PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P46
I/O		P47
I/O	AD14	P48
I/O	AD13	P49
I/O	AD12	P50
I/O	AD11	P51
I/O	AD10	P52
I/O	AD9	P53
I/O	AD8	P54
I/O		P55
I/O		P56
I/O, SGCK2		P57
N.C.		P58
GND	GND	P59
MODE	**MODE**	**P60**
VCC	VCC	P61
N.C.	N.C.	P62
I/O, PGCK2		P63
I/O (HDC)	**HDC**	**P64**
I/O	CBE0	P65
I/O	AD7	P66
I/O	AD6	P67
I/O (LDC-)	**LDC-**	**P68**
I/O	AD5	P69
I/O	AD4	P70
I/O	AD3	P71
I/O	AD2	P72
I/O		P73
I/O		P74
GND	GND	P75
I/O	AD1	P76
I/O		P77
I/O	AD0	P78
I/O		P79
VCC	VCC	P80
I/O		P81
I/O		P82
GND‡	GND	P83
I/O		P84
I/O		P85
I/O		P86
I/O		P87
I/O		P88
I/O (INIT-)	**INIT-**	**P89**
VCC	VCC	P90
GND	GND	P91

Table 7: Pinout for the XCS30 PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P92
I/O		P93
I/O		P94
I/O		P95
I/O		P96
I/O		P97
GND‡	GND	P98
I/O		P99
I/O		P100
VCC	VCC	P101
I/O		P102
I/O		P103
I/O		P104
I/O		P105
GND	GND	P106
I/O		P107
I/O		P108
I/O		P109
I/O		P110
I/O		P111
I/O		P112
I/O		P113
I/O		P114
I/O		P115
I/O		P116
I/O		P117
I/O, SGCK3		P118
GND	GND	P119
DONE	**DONE**	**P120**
VCC	VCC	P121
PROGRAM-	**PROGRAM-**	**P122**
I/O	RST-	P123
I/O, PGCK3		P124
I/O		P125
I/O		P126
I/O		P127
I/O		P128
I/O		P129
I/O		P130
I/O		P131
I/O		P132
I/O		P133
I/O		P134
GND	GND	P135
I/O		P136
I/O		P137

Table 7: Pinout for the XCS30 PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P138
I/O		P139
VCC	VCC	P140
I/O		P141
I/O		P142
GND‡	GND	P143
I/O		P144
I/O		P145
I/O		P146
I/O		P147
I/O		P148
I/O		P149
VCC	VCC	P150
GND	GND	P151
I/O		P152
I/O		P153
I/O		P154
I/O		P155
I/O		P156
I/O		P157
GND‡	GND	P158
I/O		P159
I/O		P160
VCC	VCC	P161
I/O		P162
I/O		P163
I/O		P164
I/O		P165
GND	GND	P166
I/O		P167
I/O		P168
I/O		P169
I/O		P170
I/O		P171
I/O		P172
I/O		P173
I/O		P174
I/O		P175
I/O		P176
I/O (DIN)	**DIN**	**P177**
I/O, SGCK4 (DOUT)	**DOUT**	**P178**
CCLK	CCLK	P179
VCC	VCC	P180
O, TDO	TDO	P181
GND	GND	P182
I/O		P183

Table 7: Pinout for the XCS30 PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O, PGCK4		P184
I/O		P185
I/O		P186
I/O		P187
I/O		P188
I/O		P189
I/O		P190
I/O		P191
I/O		P192
I/O		P193
I/O		P194
N.C.	N.C.	P195
GND	GND	P196
I/O		P197
I/O		P198
I/O		P199
I/O	I/O	P200
VCC	VCC	P201
I/O		P202
I/O		P203
GND‡	GND	P204
I/O		P205
I/O		P206
I/O		P207
I/O		P208
I/O		P209
I/O		P210
GND	GND	P211
VCC	VCC	P212
I/O		P213
I/O		P214
I/O		P215

Table 7: Pinout for the XCS30 PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P216
I/O		P217
I/O		P218
GND‡	GND	P219
I/O		P220
I/O		P221
VCC	VCC	P222
I/O		P223
I/O	AD31	P224
I/O		P225
I/O	AD30	P226
GND	GND	P227
I/O	AD29	P228
I/O	AD28	P229
I/O	AD27	P230
I/O	AD26	P231
I/O	AD25	P232
I/O	AD24	P233
I/O		P234
I/O		P235
I/O	CBE3	P236
I/O	IDSEL	P237
I/O		P238
I/O, SGCK1		P239
VCC	VCC	P240

‡ Pins marked with this symbol are used for Ground connections on some revisions of the device. These pins may not physically connect to anything on the current device revision. However, they should be externally connected to Ground, if possible.

Pinout for the XCS40 PQ208

Table 8: Pinout for the XCS40 PQ208

Pin Function	PCI Function	PQ208
GND	GND	P1
I/O, PGCK1	PCLK	P2
I/O		P3
I/O		P4
I/O	AD23	P5
I/O, TDI	TDI	P6
I/O, TCK	TCK	P7
I/O	AD22	P8
I/O	AD21	P9
I/O	AD20	P10
I/O	AD19	P11
I/O	AD18	P12
GND	GND	P13
I/O	AD17	P14
I/O	AD16	P15
I/O, TMS	TMS	P16
I/O		P17
VCC	VCC	P18
I/O	GNT-	P19
I/O	FRAME-	P20
I/O	IRDY-	P21
I/O	TRDY-	P22
I/O	DEVSEL-	P23
I/O	STOP-	P24
GND	GND	P25
VCC	VCC	P26
I/O	PERR-	P27
I/O	SERR-	P28
I/O	PAR	P29
I/O	REQ-	P30
I/O		P31
I/O	CBE2	P32
VCC	VCC	P33
I/O		P34
I/O	CBE1	P35
I/O	AD15	P36
I/O	AD14	P37
GND	GND	P38
I/O	AD13	P39
I/O	AD12	P40
I/O	AD11	P41
I/O	AD10	P42
I/O	AD9	P43
I/O	AD8	P44
I/O		P45

Table 8: Pinout for the XCS40 PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O		P46
I/O		P47
I/O		P48
I/O, SGCK2		P49
N.C.	N.C.	P50
GND	GND	P51
MODE	**MODE**	P52
VCC	VCC	P53
N.C.	N.C.	P54
I/O, PGCK2		P55
I/O (HDC)	**HDC**	**P56**
I/O		P57
I/O	I/O	P58
I/O	CBE0	P59
I/O (LDC-)	**LDC-**	**P60**
I/O	AD7	P61
I/O	AD6	P62
I/O	AD5	P63
I/O	AD4	P64
I/O	AD3	P65
GND	GND	P66
I/O	AD2	P67
I/O	AD1	P68
I/O	AD0	P69
I/O		P70
VCC	VCC	P71
I/O		P72
I/O		P73
I/O		P74
I/O		P75
I/O		P76
I/O (INIT-)	**INIT-**	**P77**
VCC	VCC	P78
GND	GND	P79
I/O		P80
I/O		P81
I/O		P82
I/O		P83
I/O		P84
I/O		P85
VCC	VCC	P86
I/O		P87
I/O		P88
I/O		P89
I/O	I/O	P90
GND	GND	P91

Table 8: Pinout for the XCS40 PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O		P92
I/O		P93
I/O		P94
I/O		P95
I/O		P96
I/O		P97
I/O		P98
I/O		P99
I/O		P100
I/O		P101
I/O, SGCK3		P102
GND	GND	P103
DONE	**DONE**	**P104**
VCC	VCC	P105
PROGRAM-	**PROGRAM-**	**P106**
I/O	I/O	P107
I/O, PGCK3		P108
I/O	RST-	P109
I/O		P110
I/O		P111
I/O		P112
I/O		P113
I/O		P114
I/O		P115
I/O		P116
I/O		P117
GND	GND	P118
I/O		P119
I/O		P120
VCC	VCC	P121
I/O		P122
I/O		P123
I/O		P124
I/O		P125
I/O		P126
I/O		P127
I/O		P128
I/O		P129
VCC	VCC	P130
GND	GND	P131
I/O		P132
I/O		P133
I/O		P134
I/O		P135
I/O		P136
I/O		P137

Table 8: Pinout for the XCS40 PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O		P138
I/O		P139
VCC	VCC	P140
I/O		P141
I/O		P142
GND	GND	P143
I/O		P144
I/O		P145
I/O		P146
I/O		P147
I/O		P148
I/O		P149
I/O		P150
I/O		P151
I/O		P152
I/O (DIN)	**DIN**	**P153**
I/O, SGCK4 (DOUT)	**DOUT**	**P154**
CCLK	CCLK	P155
VCC	VCC	P156
O, TDO	TDO	P157
GND	GND	P158
I/O		P159
I/O, PGCK4		P160
I/O		P161
I/O		P162
I/O		P163
I/O		P164
I/O		P165
I/O		P166
I/O		P167
I/O		P168
I/O		P169
GND	GND	P170
I/O		P171
I/O		P172
VCC	VCC	P173
I/O		P174
I/O		P175
I/O		P176
I/O		P177
I/O		P178
I/O		P179
I/O		P180
I/O		P181
GND	GND	P182
VCC	VCC	P183

Table 8: Pinout for the XCS40 PQ208 (Continued)

Pin Function	PCI Function	PQ208
I/O		P184
I/O		P185
I/O		P186
I/O		P187
I/O		P188
I/O		P189
I/O		P190
I/O	AD31	P191
VCC	VCC	P192
I/O	AD30	P193
I/O	AD29	P194
GND	GND	P195
I/O	AD28	P196
I/O	AD27	P197
I/O	AD26	P198
I/O	AD25	P199
I/O	AD24	P200
I/O	CBE3	P201
I/O	IDSEL	P202
I/O		P203
I/O		P204
I/O		P205
I/O		P206
I/O, SGCK1		P207
VCC	VCC	P208

Pinout for the XCS40 PQ240

Table 9: Pinout for the XCS40 PQ240

Pin Function	PCI Function	PQ240
GND	GND	P1
I/O, PGCK1	PCLK	P2
I/O	AD23	P3
I/O		P4
I/O	AD22	P5
I/O, TDI	TDI	P6
I/O, TCK	TCK	P7
I/O	AD21	P8
I/O	AD20	P9
I/O	AD19	P10
I/O	AD18	P11
I/O		P12
I/O		P13
GND	GND	P14
I/O	AD17	P15
I/O		P16
I/O, TMS	TMS	P17
I/O		P18
VCC		P19
I/O	AD16	P20
I/O	CBE2	P21
GND‡	GND	P22
I/O	GNT-	P23
I/O	FRAME-	P24
I/O	IRDY-	P25
I/O	TRDY-	P26
I/O	DEVSEL-	P27
I/O	STOP-	P28
GND	GND	P29
VCC	VCC	P30
I/O	PERR-	P31
I/O	SERR-	P32
I/O	PAR	P33
I/O	REQ-	P34
I/O		P35
I/O		P36
GND‡	GND	P37
I/O		P38
I/O		P39
VCC	VCC	P40
I/O		P41
I/O		P42
I/O	CBE1	P43
I/O	AD15	P44
GND	GND	P45
I/O		P46
I/O		P47
I/O	AD14	P48
I/O	AD13	P49
I/O	AD12	P50
I/O	AD11	P51
I/O	AD10	P52
I/O	AD9	P53
I/O	AD8	P54
I/O		P55
I/O		P56
I/O, SGCK2		P57
N.C.		P58
GND	GND	P59
MODE	**MODE**	**P60**
VCC	VCC	P61
N.C.	N.C.	P62
I/O, PGCK2		P63
I/O (HDC)	**HDC**	**P64**
I/O	CBE0	P65
I/O	AD7	P66
I/O	AD6	P67
I/O (LDC-)	**LDC-**	**P68**
I/O	AD5	P69
I/O	AD4	P70
I/O	AD3	P71
I/O	AD2	P72
I/O		P73
I/O		P74
GND	GND	P75
I/O	AD1	P76
I/O		P77
I/O	AD0	P78
I/O		P79
VCC	VCC	P80
I/O		P81
I/O		P82
GND‡	GND	P83
I/O		P84
I/O		P85
I/O		P86
I/O		P87
I/O		P88
I/O (INIT-)	**INIT-**	**P89**
VCC	VCC	P90
GND	GND	P91

Table 9: Pinout for the XCS40 PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P92
I/O		P93
I/O		P94
I/O		P95
I/O		P96
I/O		P97
GND‡	GND	P98
I/O		P99
I/O		P100
VCC	VCC	P101
I/O		P102
I/O		P103
I/O		P104
I/O		P105
GND	GND	P106
I/O		P107
I/O		P108
I/O		P109
I/O		P110
I/O		P111
I/O		P112
I/O		P113
I/O		P114
I/O		P115
I/O		P116
I/O		P117
I/O, SGCK3		P118
GND	GND	P119
DONE	**DONE**	**P120**
VCC	VCC	P121
PROGRAM-	**PROGRAM-**	**P122**
I/O	RST-	P123
I/O, PGCK3		P124
I/O		P125
I/O		P126
I/O		P127
I/O		P128
I/O		P129
I/O		P130
I/O		P131
I/O		P132
I/O		P133
I/O		P134
GND	GND	P135
I/O		P136
I/O		P137

Table 9: Pinout for the XCS40 PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P138
I/O		P139
VCC	VCC	P140
I/O		P141
I/O		P142
GND‡	GND	P143
I/O		P144
I/O		P145
I/O		P146
I/O		P147
I/O		P148
I/O		P149
VCC	VCC	P150
GND	GND	P151
I/O		P152
I/O		P153
I/O		P154
I/O		P155
I/O		P156
I/O		P157
GND‡	GND	P158
I/O		P159
I/O		P160
VCC	VCC	P161
I/O		P162
I/O		P163
I/O		P164
I/O		P165
GND	GND	P166
I/O		P167
I/O		P168
I/O		P169
I/O		P170
I/O		P171
I/O		P172
I/O		P173
I/O		P174
I/O		P175
I/O		P176
I/O (DIN)	**DIN**	**P177**
I/O, SGCK4 (DOUT)	**DOUT**	**P178**
CCLK	CCLK	P179
VCC	VCC	P180
O, TDO	TDO	P181
GND	GND	P182
I/O		P183

Table 9: Pinout for the XCS40 PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O, PGCK4		P184
I/O		P185
I/O		P186
I/O		P187
I/O		P188
I/O		P189
I/O		P190
I/O		P191
I/O		P192
I/O		P193
I/O		P194
N.C.	N.C.	P195
GND	GND	P196
I/O		P197
I/O		P198
I/O		P199
I/O	I/O	P200
VCC	VCC	P201
I/O		P202
I/O		P203
GND‡	GND	P204
I/O		P205
I/O		P206
I/O		P207
I/O		P208
I/O		P209
I/O		P210
GND	GND	P211
VCC	VCC	P212
I/O		P213
I/O		P214

Table 9: Pinout for the XCS40 PQ240 (Continued)

Pin Function	PCI Function	PQ240
I/O		P215
I/O		P216
I/O		P217
I/O		P218
GND‡	GND	P219
I/O		P220
I/O		P221
VCC	VCC	P222
I/O		P223
I/O	AD31	P224
I/O		P225
I/O	AD30	P226
GND	GND	P227
I/O	AD29	P228
I/O	AD28	P229
I/O	AD27	P230
I/O	AD26	P231
I/O	AD25	P232
I/O	AD24	P233
I/O		P234
I/O		P235
I/O	CBE3	P236
I/O	IDSEL	P237
I/O		P238
I/O, SGCK1		P239
VCC	VCC	P240

‡ Pins marked with this symbol are used for Ground connections on some revisions of the device. These pins may not physically connect to anything on the current device revision. However, they should be externally connected to Ground, if possible.

XILINX®

Resources

Resources

PCI Special Interest Group (PCI-SIG) Publications

The PCI-SIG publishes various PCI specifications and related documents. Most publications cost US$25 plus applicable shipping charges.

- PCI Local Bus Specification, Rev. 2.1
- PCI Compliance Checklist v2.1 (available via the World-Wide Web)
- PCI System Design Guide v1.0

Contact:

PCI Special Interest Group
2575 NE Kathryn St. #17
Hillsboro, OR 97214
Phone: +1 800-433-5177 (within USA)
Phone: +1 503-693-6232 (worldwide)
Fax: +1 503-693-8344
E-Mail: info@pcisig.com
URL: http://www.pcisig.com

PCI and FPGA Design Consultants

Listed below are design centers and design consultants that have experience with the LogiCORE PCI Products.

HighGate Design, Inc.
12380 Saratoga-Sunnyvale Road, Suite 8
Saratoga, CA 95070-3090, USA
Phone: +1 408-255-7160
Fax: +1 408-255-7162
E-mail: highgate@highgatedesign.com
URL: http://www.highgatedesign.com

Memec Design Services
1819 S. Dobson Rd. Ste. 203
Mesa, AZ 85202, USA
Phone: +1 888-360-9044
Fax: +1 602-491-4907
E-mail: info@memecdesign.com
URL: http://www.memecdesign.com

Comit Systems
1250 Oakmead Parkway, Suite 210
Sunnyvale, CA 94088, USA
Phone: +1 408-9882988
Fax: +1 408-988-2133
E-mail: preeth@comit.com
URL: http://www.comit.com

Austin Franklin
Dark Room Technologies, Inc.
126 Poor Farm Road
Harvard MA, 01451, USA
Phone: +1 508-772-9928
Fax: +1 508-772-4287
E-mail: darkroom@ix.netcom.com

Supporting PCI Tools

VCC Corporation
Provides HotPCI Rapid Prototyping Board
6925 Canby Ave. #103
Reseda, CA 91335 USA
Phone: +1 818-342-8294
Fax: +1 818-342-0240
E-mail: info@vcc.com
Website: http://www.vcc.com

Vireo Software Inc.
Provides drivers and driver development tools.
30 Monument Square, Suite 135
Concord, MA 01742 USA
Phone: +1 978-369-3380
Fax: +1 978-318-6946
E-mail: info@vireo.com
support@vireo.com
Website: http://www.vireo.com

PCI Reference Books

There are many reference books available on PCI. The following are a few that the LogiCORE development team found useful.

PCI System Architecture by Tom Shanley and Don Anderson. ISBN 1-881609-08-1. An excellent general reference book on PCI. This book is included with the LogiCORE PCI product.

Contact:

Mindshare Press
2202 Buttercup Dr.
Richardson, TX 75082
Phone: +1 214-231-2216
Fax: +1 214-783-4715

Distributed by:

Computer Literacy Bookshops, Inc.
P.O. Box 641897
San Jose, CA 95164
Phone: +1 408-435-0744
Fax: +1 408-435-1823
E-mail: info@clbooks.com
URL: http://www.clbooks.com

PCI Hardware and Software Architecture & Design by Edward Solari & George Willse. ISBN 0-929392-19-1. Everything that you ever wanted to know about PCI systems design, and more.

Contact:

Annabooks
11848 Bernardo Center Drive
Suite 110
San Diego, CA 92128

Phone:	+1 619-673-0870
	+1 800-462-1042
Fax:	+1 619-673-1432

Xilinx Documents

See the Xilinx web page at

`www.xilinx.com/products/logicore/pci/pcilit.htm`

for available literature.

IMPORTANT! Be sure to visit the Xilinx WebLINX web site for the latest information and application notes using the LogiCORE PCI interface.

LogiCORE User's Lounge Web Site

The LogiCORE User's Lounge web site provides a quick and convenient way to obtain the latest updates, documentation, design tips, application notes, and utilities. The Lounge web site is open to registered LogiCORE users. To register, point your Internet browser software to:

`www.xilinx.com/products/logicore/pci/pci_sol.htm`

Waveforms

Waveforms

These waveforms correspond with the Compliance Checklist contained herein. Please see specific sections for explanations regarding these waveforms.

Configuration Read and Write

This waveform shows a configuration read and write cycle. The LogiCORE PCI interface issues an automatic "disconnect with data" on first data phase (STOP_IO and TRDY_IO are asserted together). CNFG_VLD is asserted on the clock following the address cycle. The PCI interface in the "slow" decode mode asserts DEVSEL_IO when CFG_HIT occurs. Notice both C_TERM and C_READY are asserted.

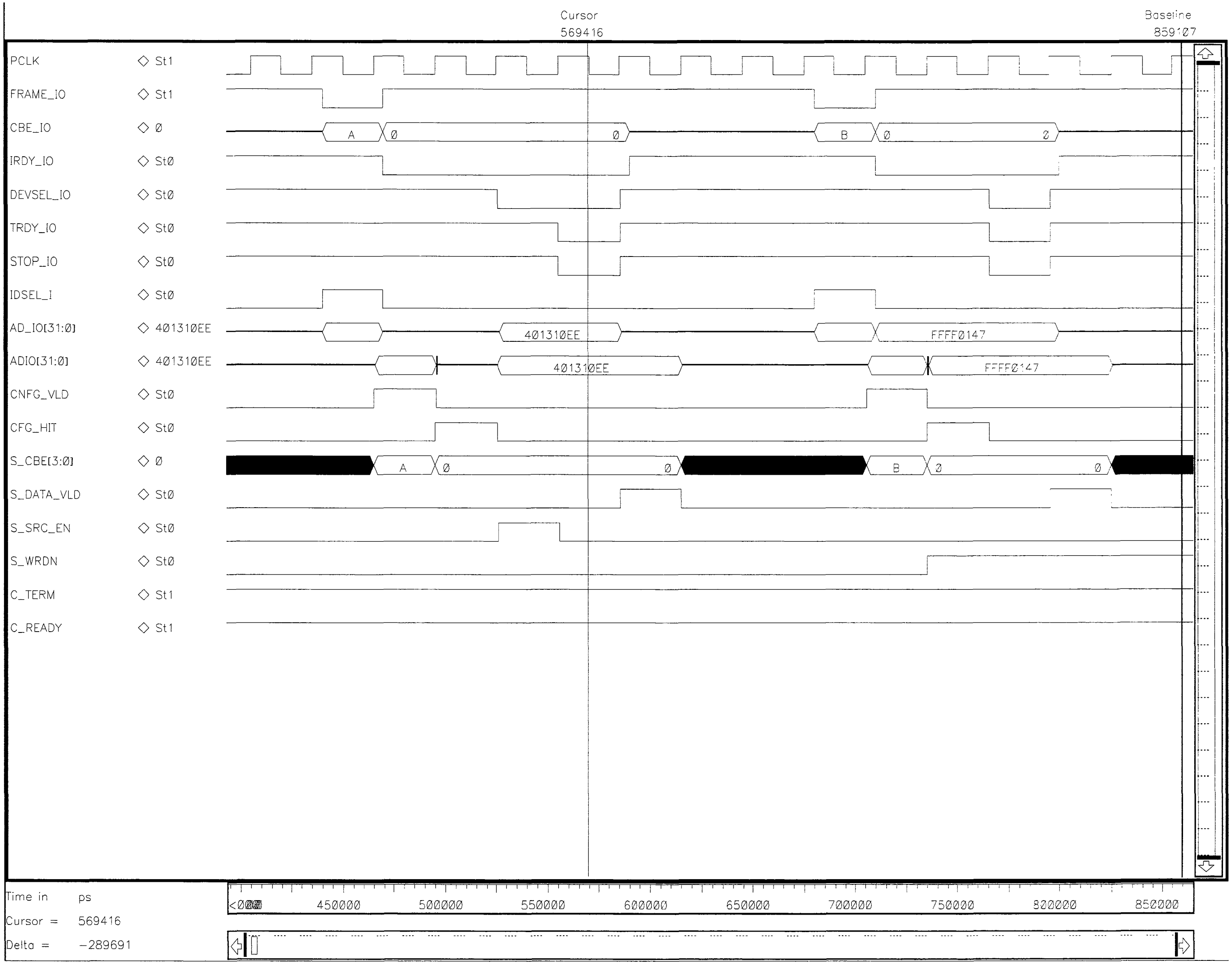


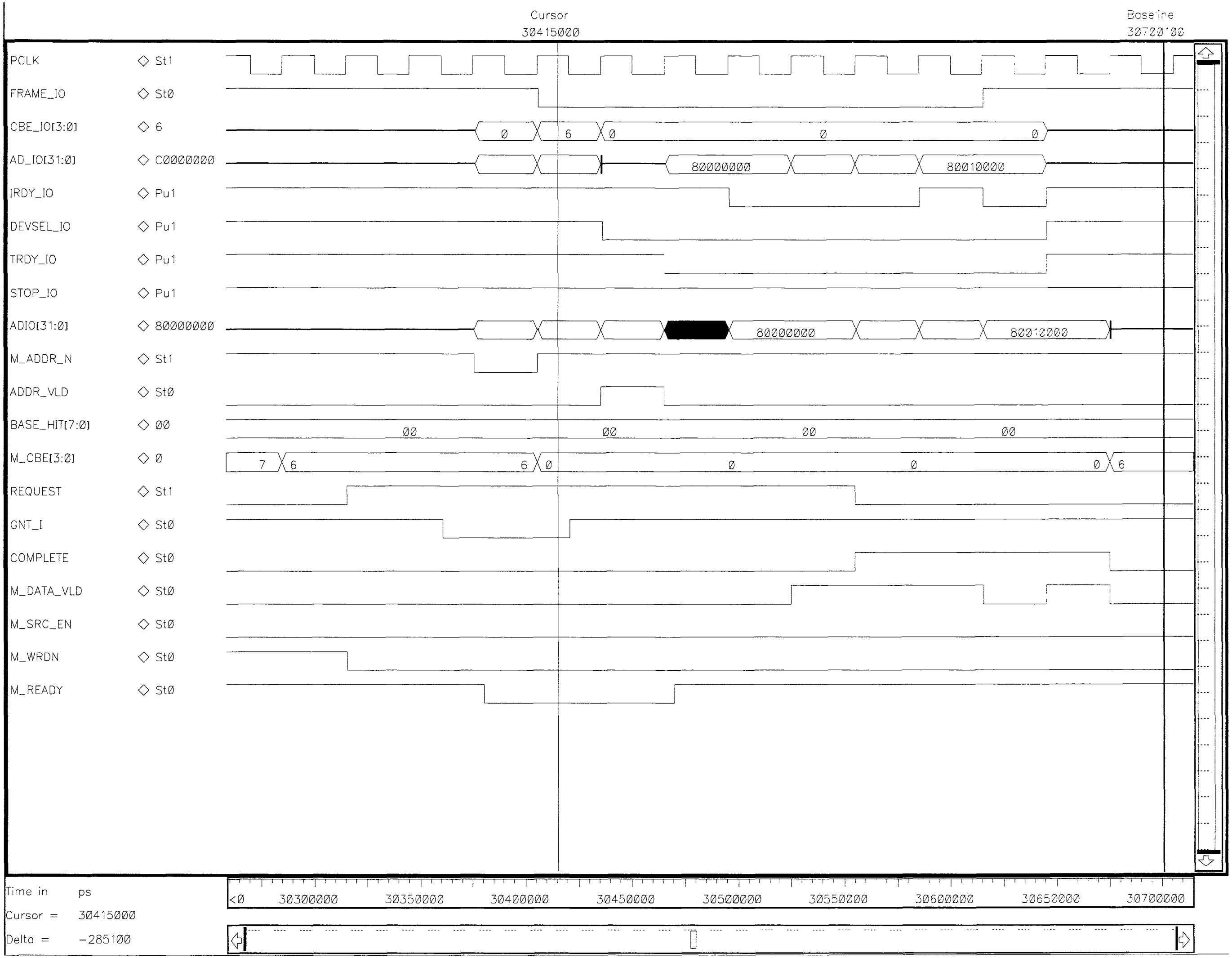
Figure 1: Configuration read and write

Initiator Burst Read

This waveform highlights a four DWord initiator burst read operation on a fast decode target. The LogiCORE PCI master inserts a single automatic waitstate during the last data phase. In this example, the user application is holding off data transfer by keeping M_READY deasserted, thus resulting in a IRDY_IO waitstate at the beginning of the transfer.

Once M_READY is asserted, it should not be deasserted until the transfer is complete. Note that COMPLETE is asserted with the next to last data.

Figure 2: Initiator burst read

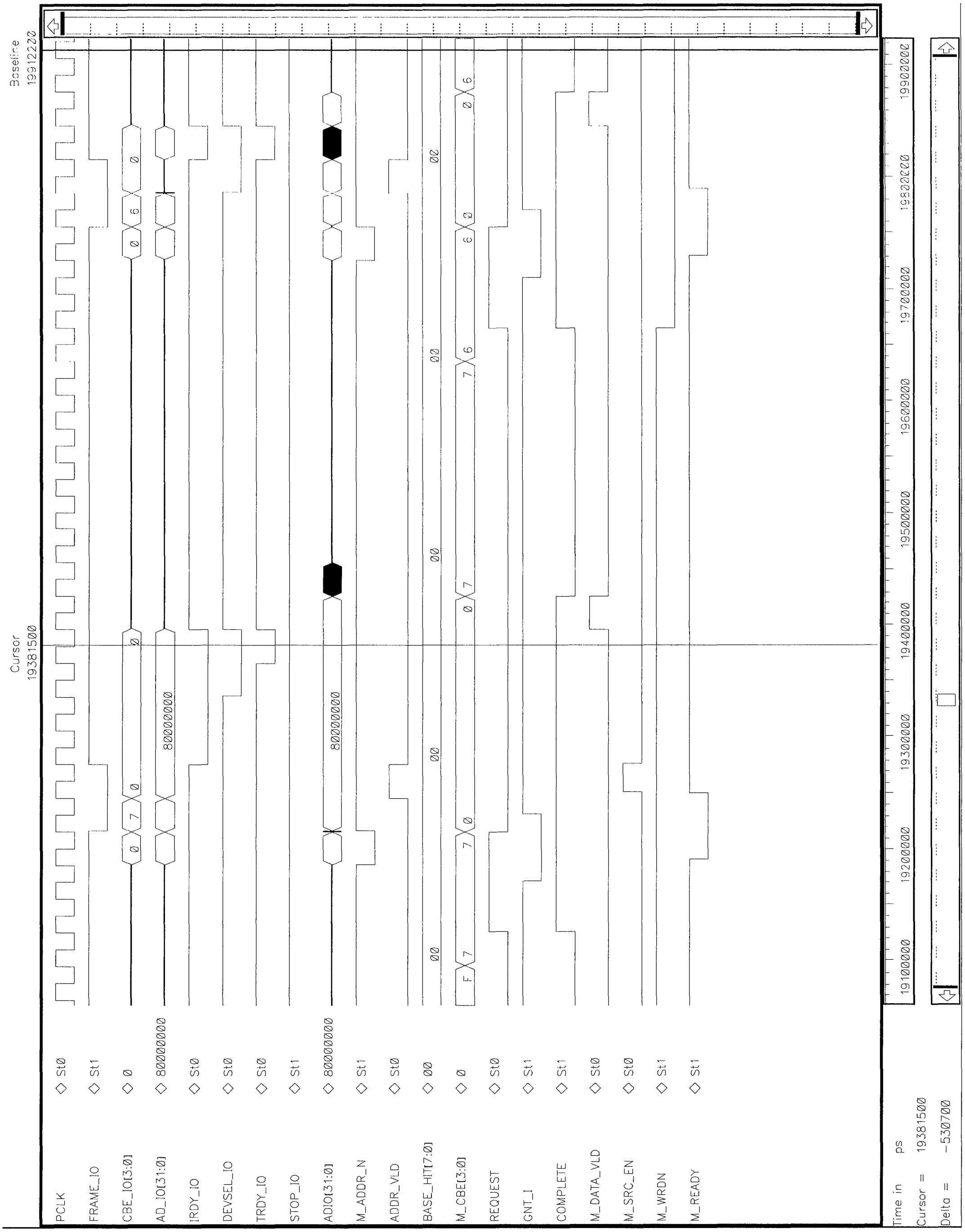


Time in ps
Cursor = 30415000
Delta = -285100

Initiator Single Read and Write

The waveforms show a single DWord initiator write and read cycle on subtractive and fast decode targets respectively. Notice that COMPLETE asserts from the very beginning of the cycles to indicate the last transfer and REQUEST is deasserted at the start of address cycle.

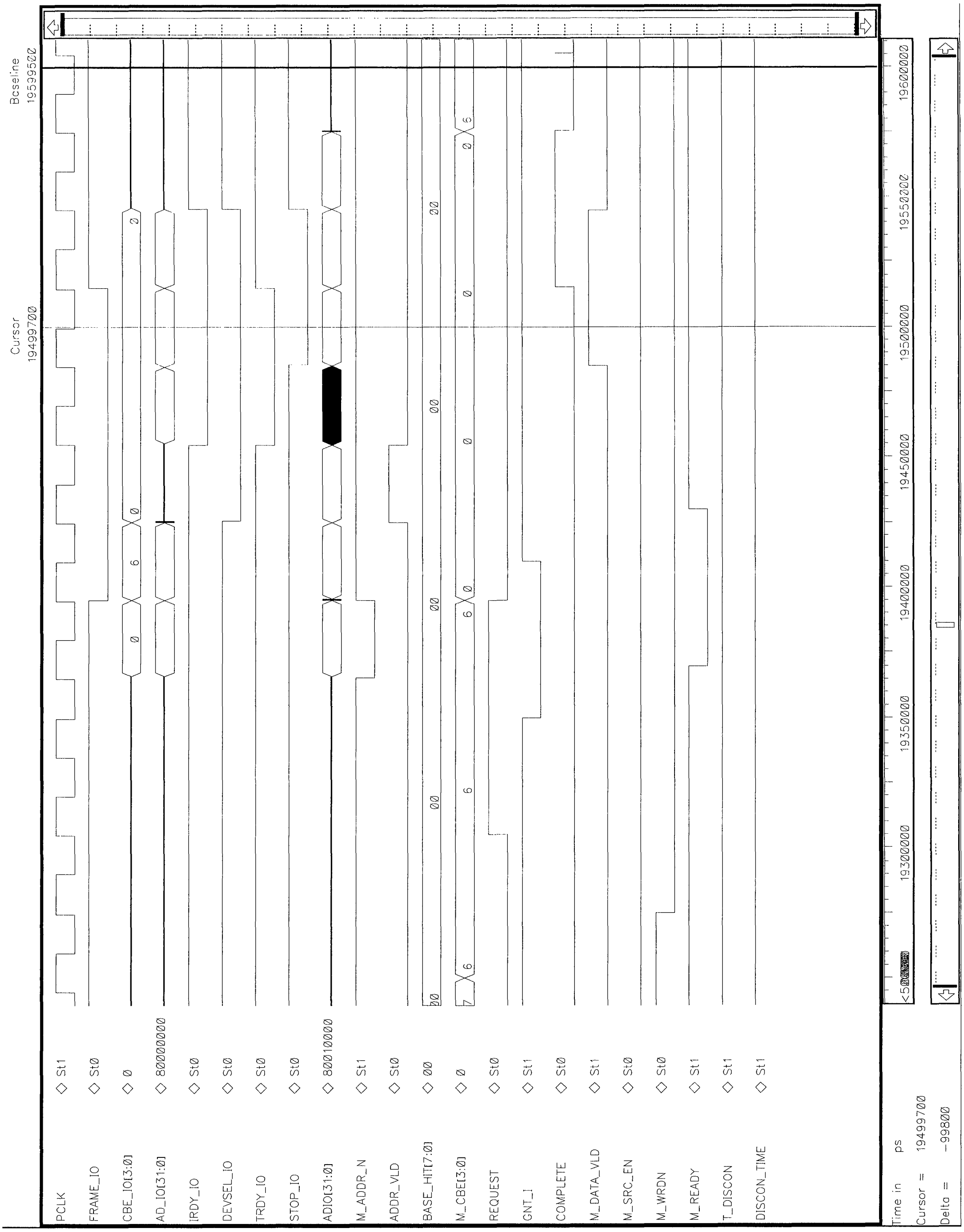
Figure 3: Initiator single read and write



Initiator Burst Write with Disconnect

The following waveform demonstrates a four DWord initiator burst read cycle on fast target performing “Disconnect with Data” on second data phase. Disconnect with data is signalled by asserting TRDY_IO and STOP_IO together. Initial IRDY_IO waitstate results from M_READY not being asserted at the start of the address phase. It is recommended that the user application always be “ready” to avoid waitstates.

Figure 4: Initiator burst write with disconnect



Initiator Burst Write (0 waitstate)

The following waveform demonstrates a four DWord initiator burst write cycle on subtractive decode target. Notice that LogiCORE master does not insert any waitstates except for the last one. Notice that COMPLETE is asserted on start of next to last data phase.

This waveform shows an example of the extra waitstate that is inserted on the last cycle of an initiator read or write. When the initiator state machine receives COMPLETE assertion, it needs to wait until the current data has transferred before deasserting FRAME# to indicate the last data phase (Recall that COMPLETE is asserted with the second to last data on the user side). Thus, the FRAME logic needs to monitor TRDY#. The current FRAME logic is too complex to add another unregistered input. The solution is to insert a wait state. Thus, the FRAME logic can look at the registered version of TRDY# instead. In terms of burst latency, if your burst size is large, the effects of this waitstate on bandwidth is relatively small.

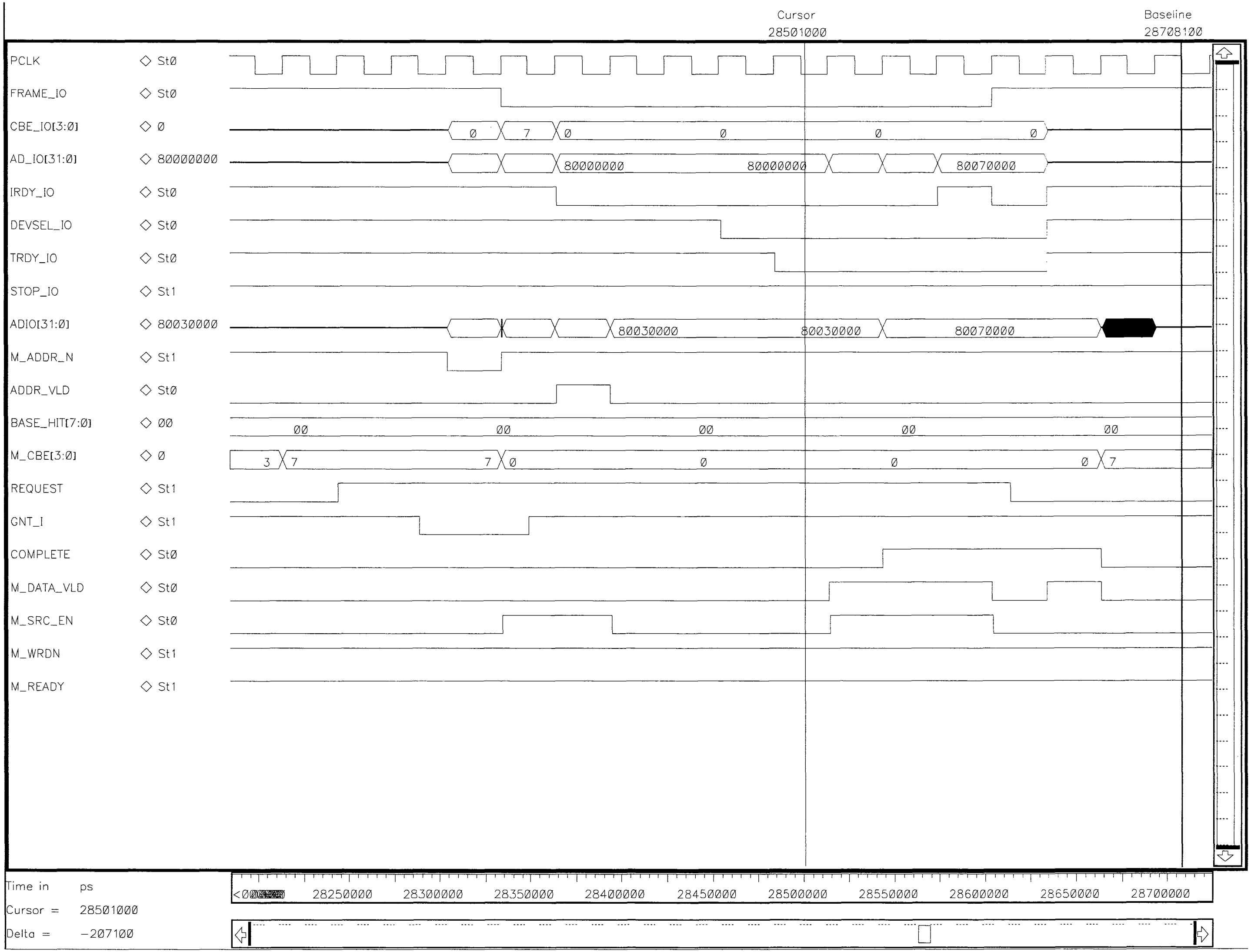
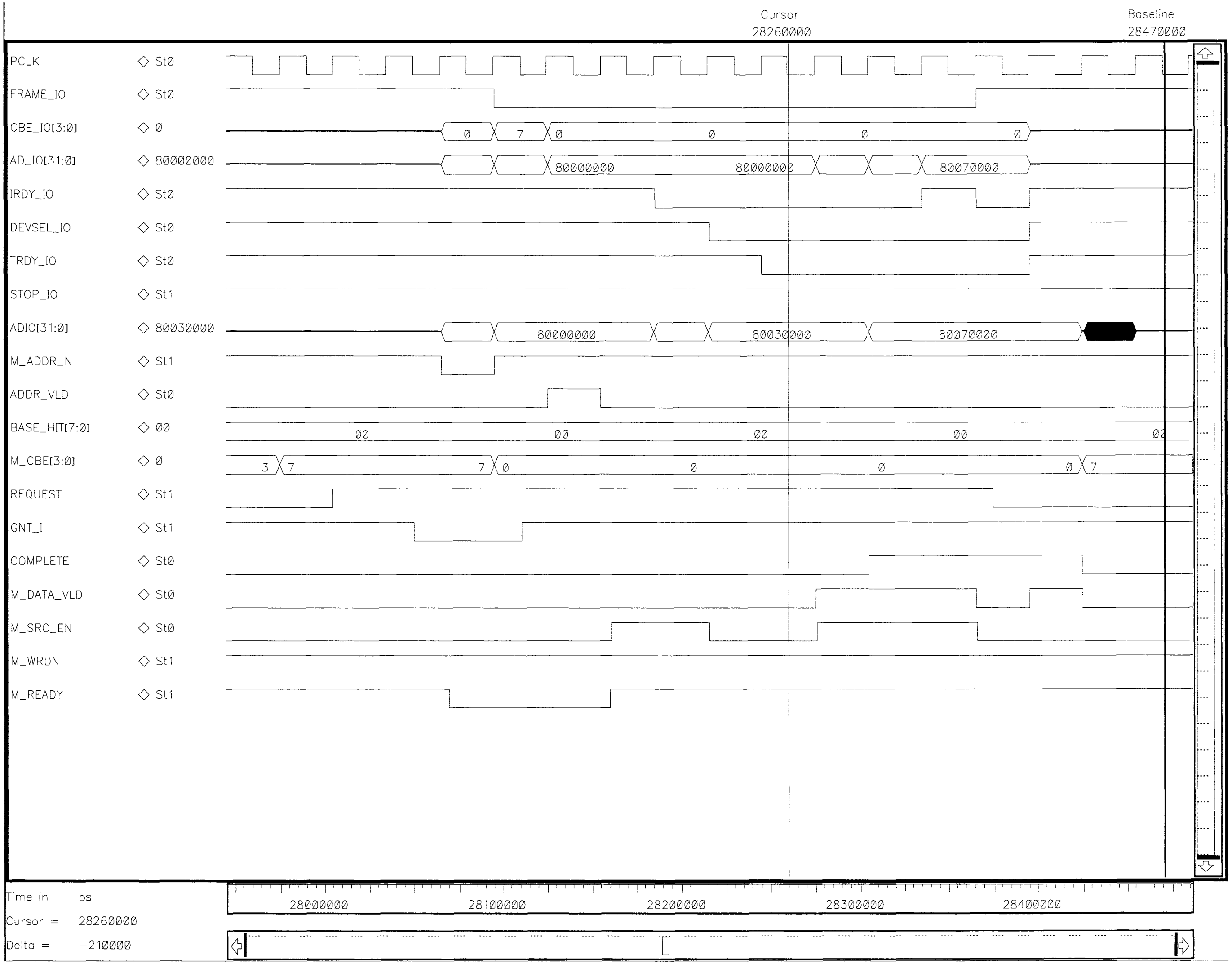


Figure 5: Initiator burst write (0 waitstate)

Initiator Burst Write (not ready, 0 waitstate)

The following waveform demonstrates a four DWord initiator burst write cycle on subtractive decode target. The cycle starts with M_READY not asserted. This shows the differences in the waveforms if the user's application is not always ready. Notice that LogiCORE master does not insert any waitstates except for the last one. Notice that COMPLETE is asserted on start of next to last data phase.

Figure 6: Initiator burst write (not ready) (0 waitstate)



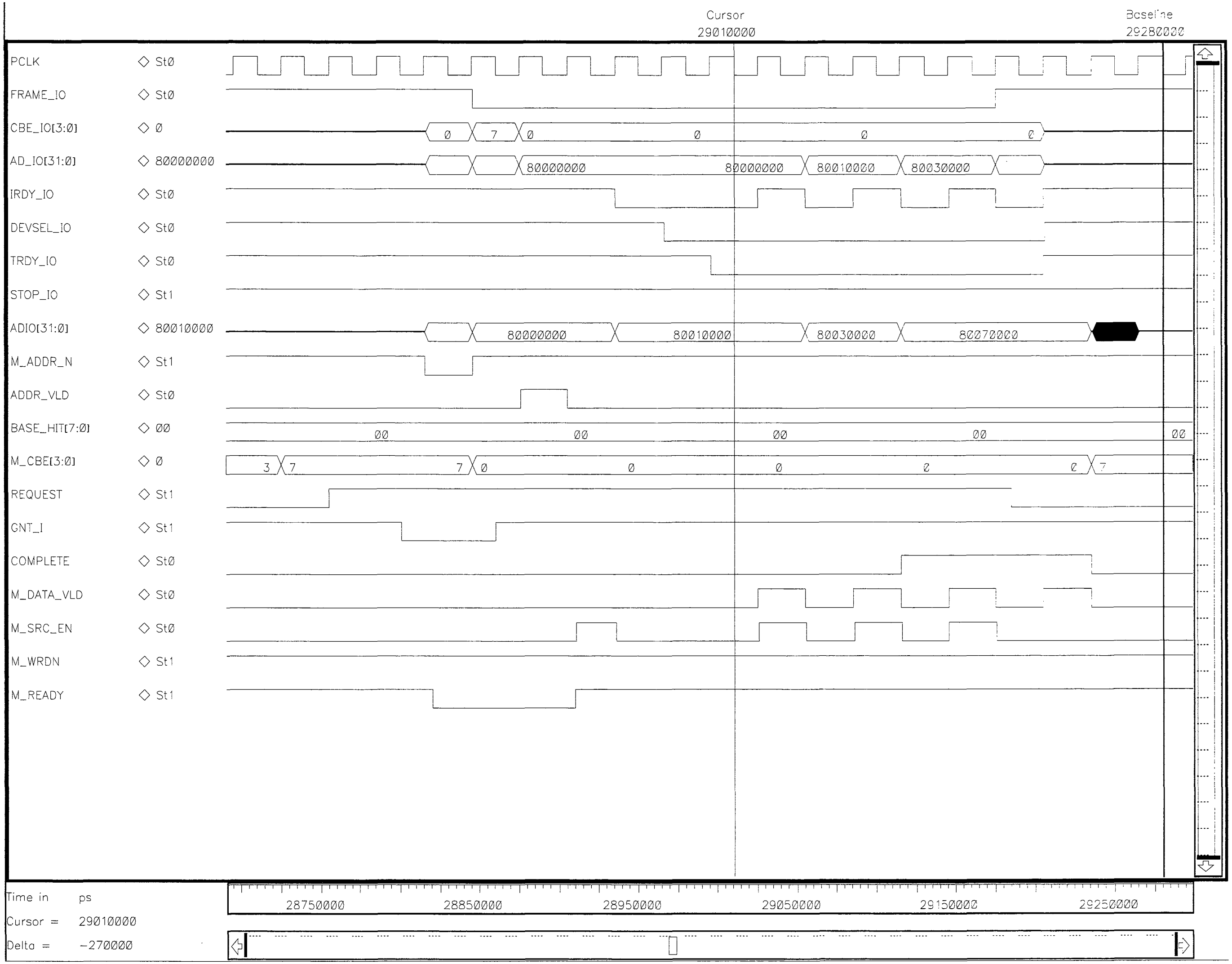
PCLK ◇ St0
FRAME_IO ◇ St0
CBE_IO[3:0] ◇ 0
AD_IO[31:0] ◇ 80000000
IRDY_IO ◇ St0
DEVSEL_IO ◇ St0
TRDY_IO ◇ St0
STOP_IO ◇ St1
ADIO[31:0] ◇ 80030000
M_ADDR_N ◇ St1
ADDR_VLD ◇ St0
BASE_HIT[7:0] ◇ 00
M_CBE[3:0] ◇ 0
REQUEST ◇ St1
GNT_I ◇ St1
COMPLETE ◇ St0
M_DATA_VLD ◇ St0
M_SRC_EN ◇ St0
M_WRDN ◇ St1
M_READY ◇ St1

Cursor 28260000
Baseline 28470000

Time in ps
Cursor = 28260000
Delta = -210000

Initiator Burst Write (1 waitstate)

The following waveform demonstrates a four DWord initiator burst write cycle on a subtractive decode target. LogiCORE master automatically inserts one waitstate on every data phase. Notice that COMPLETE asserts after next to last data phase and M_DATA_VLD asserts for each valid transfer.



Time in ps
Cursor = 29010000
Delta = -270000

Figure 7: Initiator burst write (1 waitstate)

Target Burst Write

The waveform highlights a three DWord target burst write operation in which initiator inserts one waitstate in every data phase. Notice the ADDR_VLD assertion on the clock following the address cycle. LogiCORE as "slow" decode asserts DEVSEL_IO when BASE_HIT comes and responds with TRDY_IO until last data (FRAME_IO and IRDY_IO high). S_DATA_VLD is asserted on the following clock, whenever a valid transfer occurs (both IRDY_IO and TRDY_IO asserted). Notice that S_TERM stays low while S_READY is high for a normal transaction.

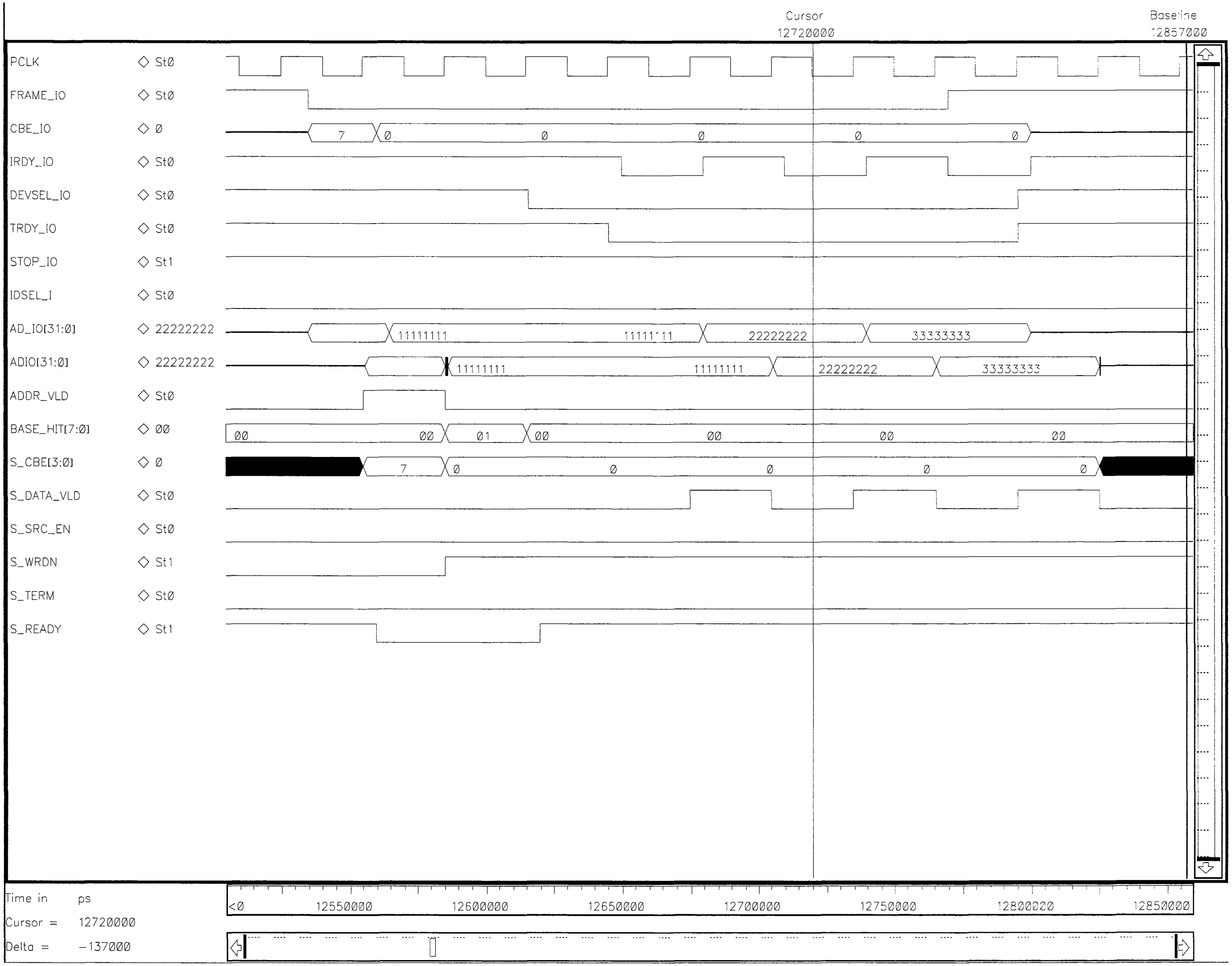
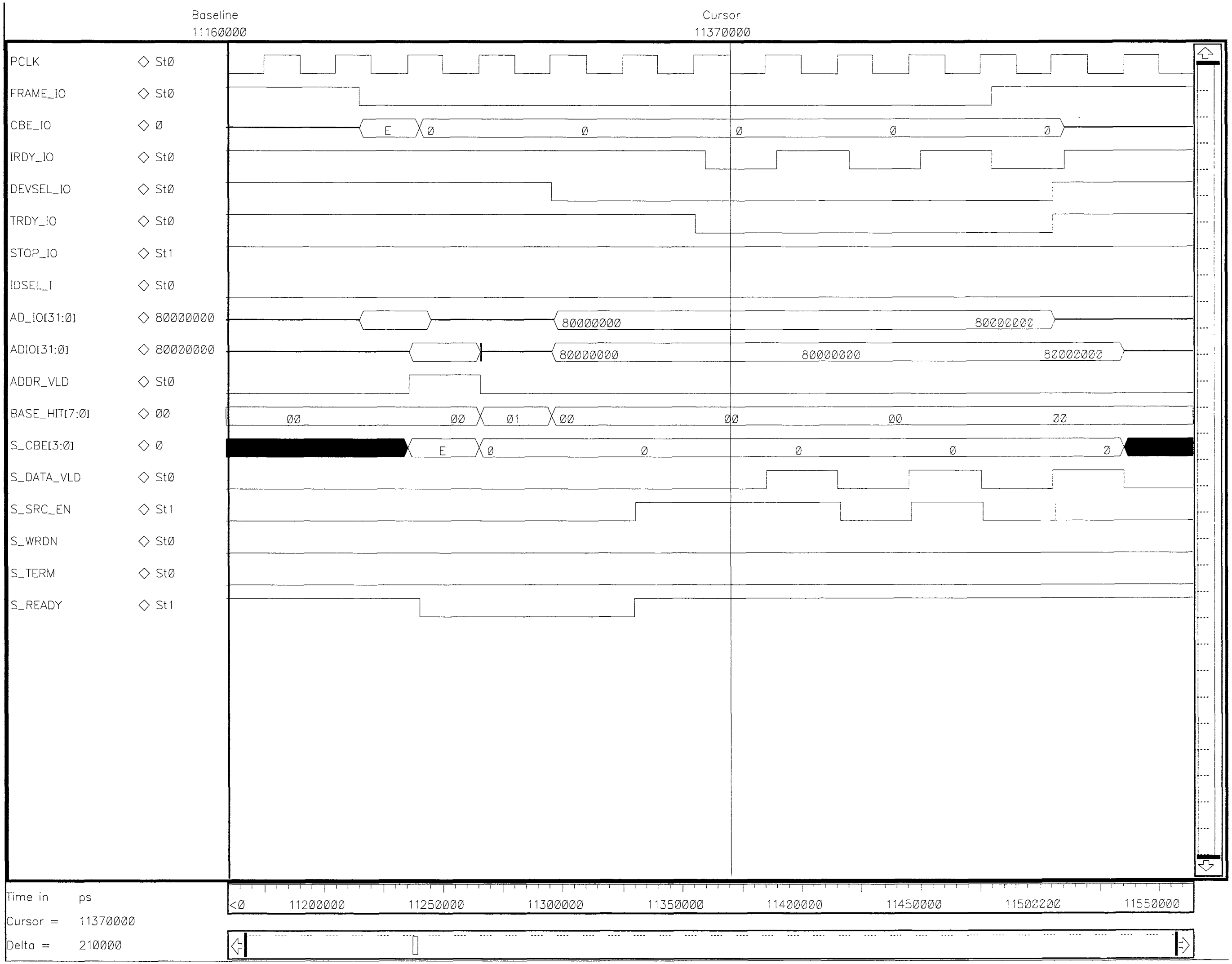


Figure 8: Target burst write

Target Burst Read (0 waitstate)

The waveform highlights a three DWord target burst read-line cycle in which initiator inserts one waitstate on data phase 2 and 3. Notice that LogiCORE target does not insert automatic waitstates. Also note how initial S_READY deassertion causes a target waitstate during initial data phase.

Figure 9: Target burst read (0 waitstate)



Baseline 11160000

Cursor 11370000

```
PCLK            ◇ St0
FRAME_IO        ◇ St0
CBE_IO          ◇ 0
IRDY_IO         ◇ St0
DEVSEL_IO       ◇ St0
TRDY_IO         ◇ St0
STOP_IO         ◇ St1
IDSEL_I         ◇ St0
AD_IO[31:0]     ◇ 80000000
ADIO[31:0]      ◇ 80000000
ADDR_VLD        ◇ St0
BASE_HIT[7:0]   ◇ 00
S_CBE[3:0]      ◇ 0
S_DATA_VLD      ◇ St0
S_SRC_EN        ◇ St1
S_WRDN          ◇ St0
S_TERM          ◇ St0
S_READY         ◇ St1
```

Time in ps

Cursor = 11370000

Delta = 210000

Target Burst Read (1 waitstate)

The waveform highlights a three DWord target burst read multiple cycle in which initiator inserts one waitstate on data phases 2 and 3. The LogiCORE target automatically inserts one waitstate during each data phase. S_SRC_EN is asserted before commencement of every data phase. Also notice how the initial S_READY deassertion causes a target waitstate during the initial data phase.

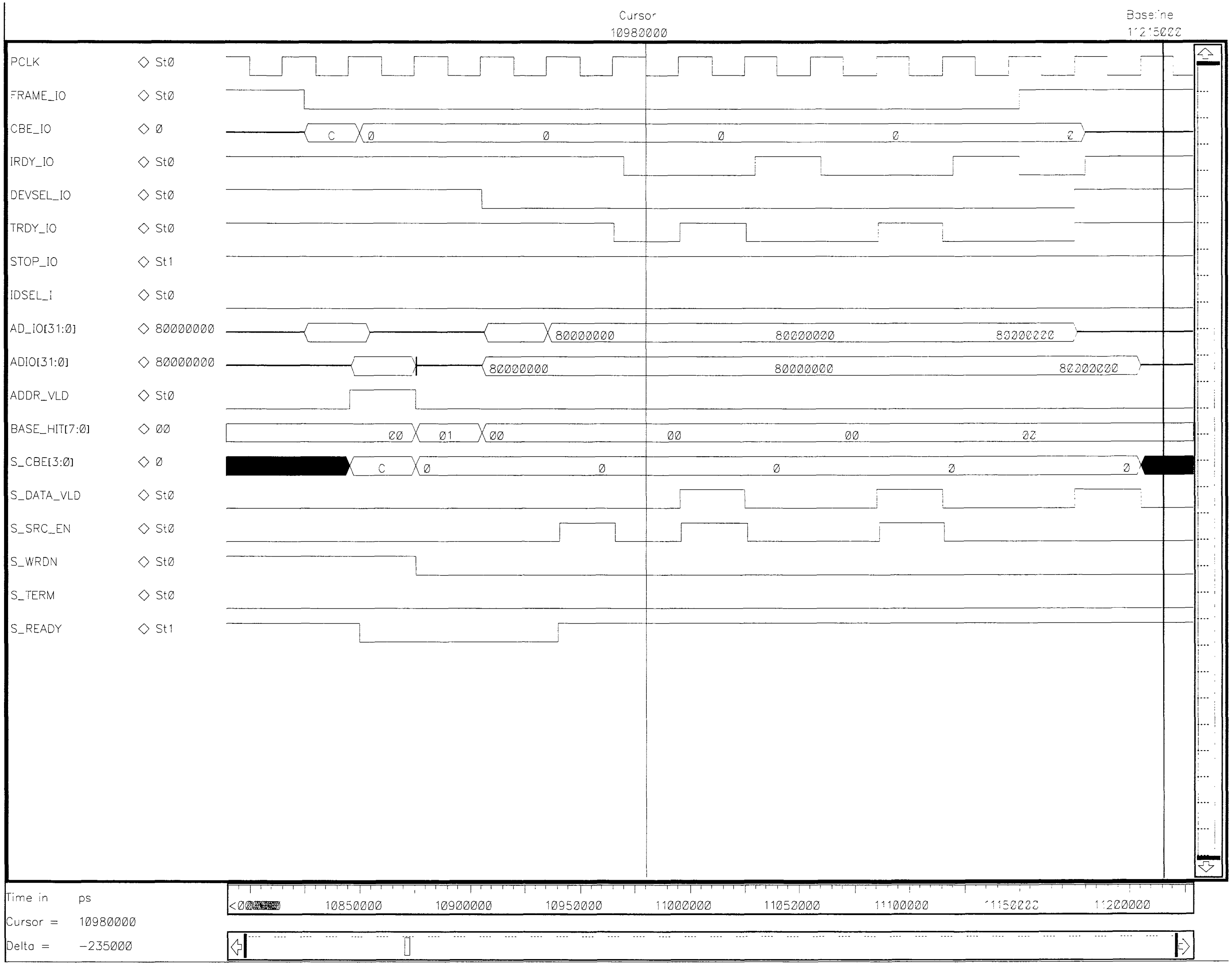


Figure 10: Target burst read (1 waitstate)

Target Single Read and Write

The waveforms show a single DWord target read and write cycle. Notice how S_READY is used by the user application to insert waitstates at the beginning of the transaction.

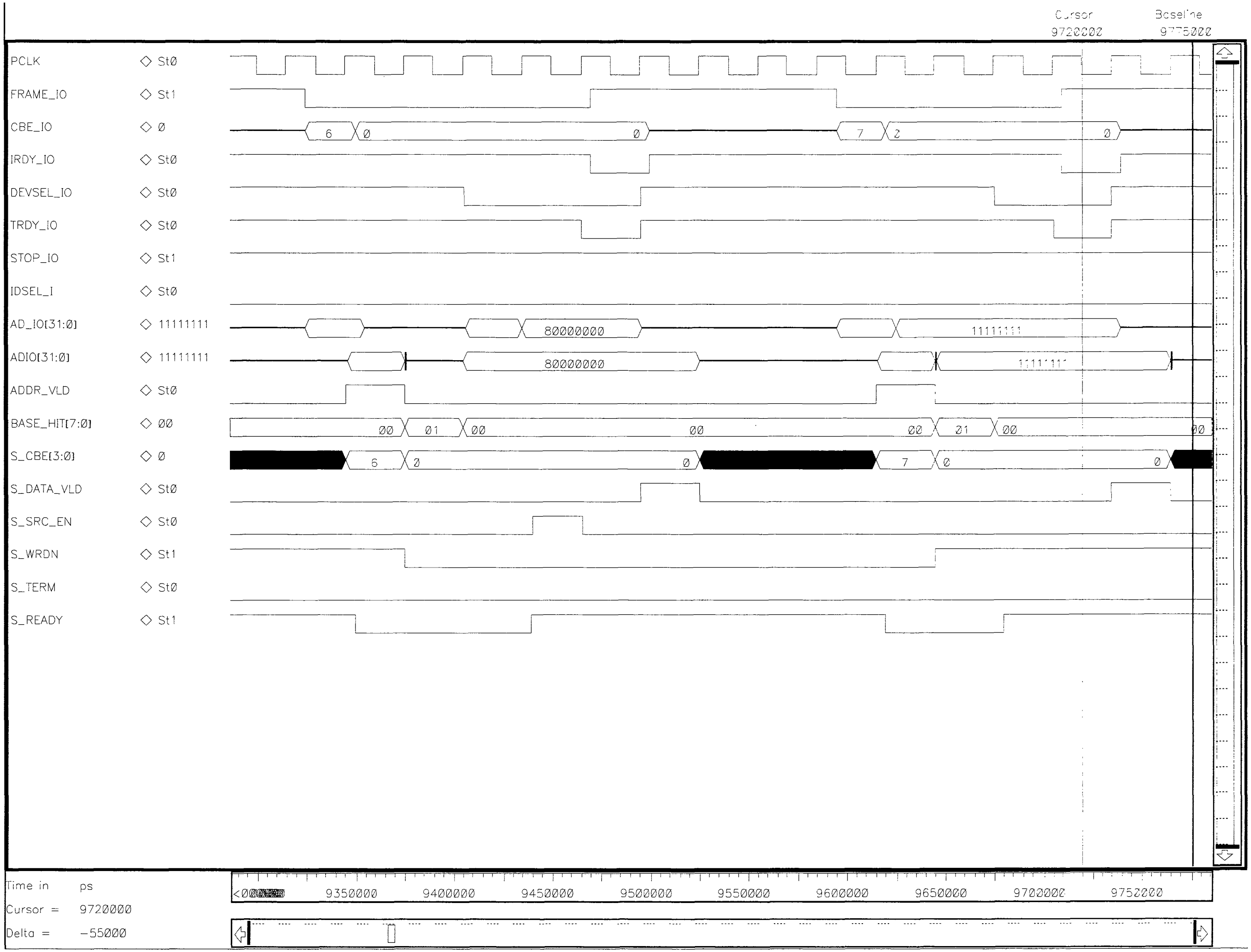


Figure 11: Target single read and write

Target Retry and Abort

The waveform shows two different target termination cycles. A) Retry during read, and B) Abort during write.

Cycle A is requested before any data is transferred and is signalled by asserting STOP_IO and not asserting TRDY_IO on the first data phase. Retry is a special case of Disconnect without data on the initial data phase. Notice that the user application needs to assert S_TERM and not assert S_READY to force a LogiCORE target retry condition.

In cycle B the LogiCORE target claims the cycle when BASE_HIT occurs by asserting DEVSEL_IO. The target signals Abort by deasserting DEVSEL_IO and asserting STOP_IO at the same time. Notice that the user backend should assert T_ABORT and not assert S_TERM and S_READY to force the LogiCORE into a target abort condition. This can be done during any data phase.

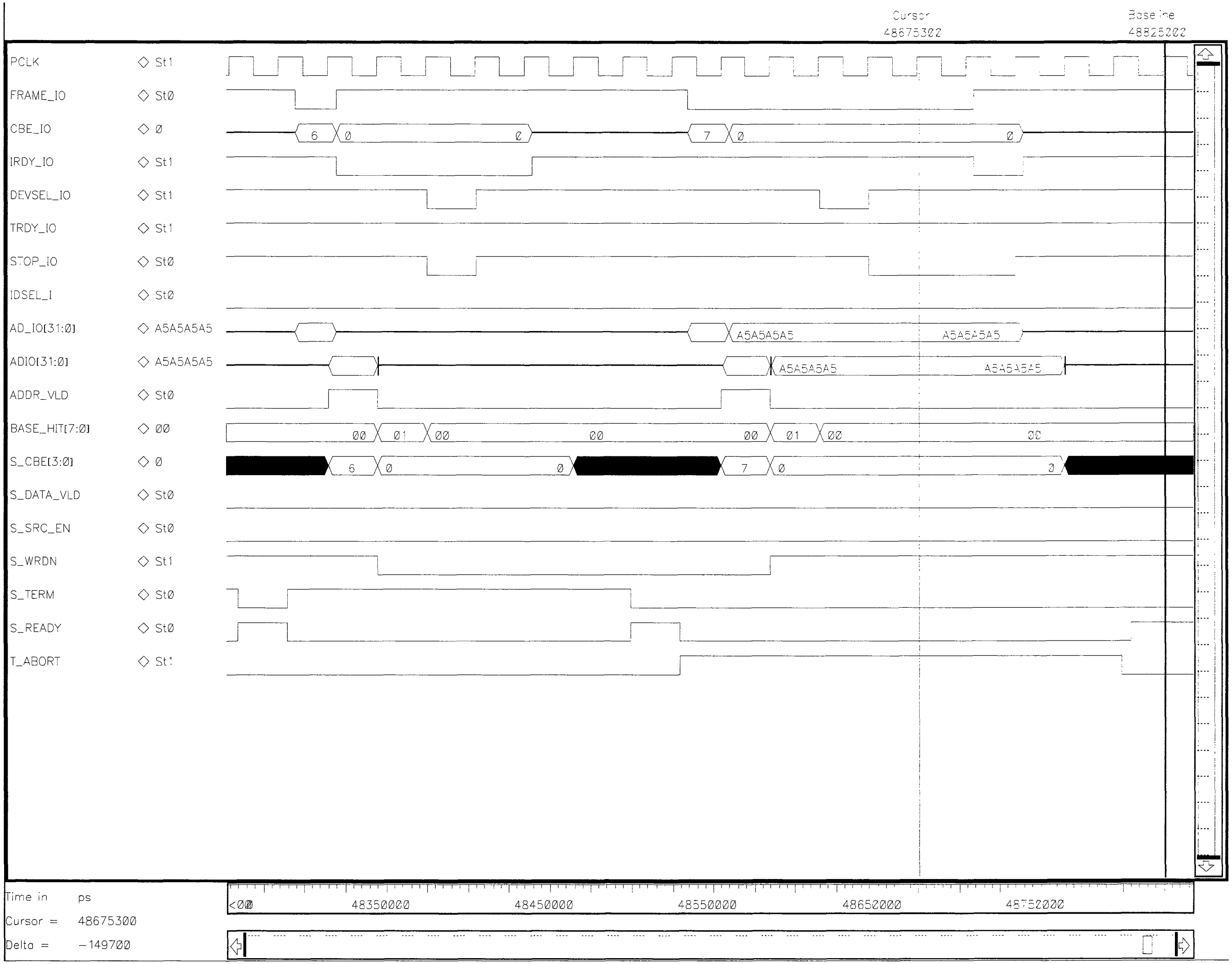
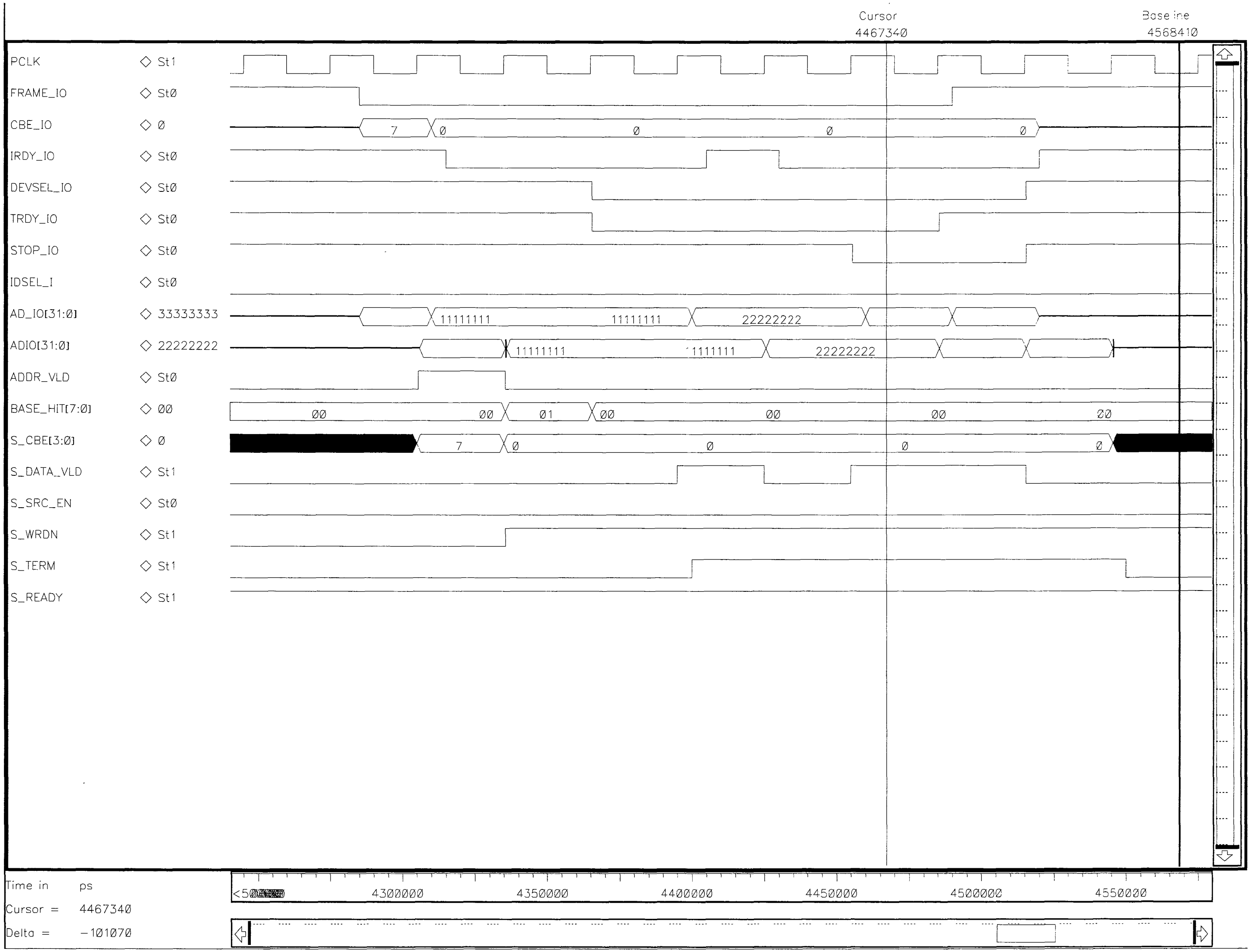


Figure 12: Target retry and abort

Target Burst Write with Disconnect

The following waveform demonstrates a four DWord target burst write cycle performing a disconnect with data on the third data phase. The initiator inserts one waitstate during the 2nd data phase. Notice that S_TERM should be asserted after the first data phase (S_DATA_VLD signal) to disconnect on phase 3. A disconnect with data is signalled by asserting TRDY_IO and STOP_IO together.

Figure 13: Target burst write with disconnect



XILINX®

Ordering Information and License Agreement

Ordering Information and License Agreement

Summary

Xilinx provides a complete solution for development of fully compliant PCI boards and CompactPCI products. This chapter describes the various product packages. See Table 1 for a quick overview. Please refer to the individual data sheets for details. Before placing an order, please read and sign the LogiCORE license agreement (last page of this chapter) and fax it to Xilinx at +1 408-377-3259. Call your local sales office for the latest availability and pricing information.

Xilinx PCI32 Design Kit

Part no: DO-DI-PCI32-DK

Overview

To minimize the learning curve for PCI and to enable rapid development and prototyping, Xilinx provides a complete PCI Design Kit including cores, a prototyping board and driver development tools. By forming a partnership with *Virtual Computer Corporation*, leading provider of rapid prototyping boards and reconfigurable computing systems, and *Vireo Software*, leading provider of device driver tools, our customers will always have access to the leading industry expertise.

Package Includes

See individual data sheets for details.

- LogiCORE PCI32 4000 Interface
  - Master and Slave (Initiator/Target) functionality
  - Configurable and downloadable from Xilinx web
  - Viewlogic source files
  - CD available upon request only
- LogiCORE PCI32 Spartan Interface
  - Master and Slave (Initiator/Target) functionality
  - Configurable and downloadable from Xilinx web
  - Viewlogic source files
  - CD available upon request only
- Synthesizable PCI Bridge Design Examples
  - VHDL and Verilog source files
  - On-line Design Examples User's Guide
- LogiCORE PCI32 User's Guide
- PCI System Architecture (Reference Book from MindShare, Inc.)
- VCC HotPCI Prototyping System
  - PCI32 Spartan prototyping board
  - Example drivers (Vireo)
  - HotPCI User's Guide
  - Demo software CD for Windows 95
- Vireo Driver::Works Development tools for Windows NT/98 drivers
  - Fully functional development tools
  - Development and prototyping license
  - Unrestricted license required for production and is available from Vireo
  - Technical support provided by Vireo only
- Vireo VtoolsD Development tools for Windows 95/98 and 3.x drivers
  - Fully functional development tools
  - Development and prototyping license
  - Unrestricted license required for production and is available from Vireo
  - Technical supported provided by Vireo only

Table 1: Xilinx PCI Product Line

Feature See individual data sheets for details	PCI32 Design Kit DO-DI-PCI32-DK	PCI32 Spartan DO-DI-PCI32-S
32-bit, 33MHz PCI Initiator & Target	✓	✓
Support for Xilinx Spartan FPGAs	✓	✓
Support for Xilinx XC4000 FPGAs	✓	
PCI Bridge Design Examples	All	RD03
Configuration and Download from web	✓	✓
LogiCORE User's Guide	✓	On-line
PCI System Architecture Book	✓	
Source Files (Viewlogic)	✓	
VCC HotPCI Prototyping System	✓	
Example Reference Drivers	✓	
Vireo Driver::Works Development Tools	✓	
Vireo VtoolsD Development Tools	✓	
12 month Maintenance Contract	✓	
Free updates	12 months	3 months

Support and Updates

A twelve-month maintenance contract including free updates is included with purchase. After expiring, the maintenance contract may be renewed annually.

- Access to Xilinx LogiCORE PCI Lounges `www.xilinx.com/products/logicore/pci/pci_sol.htm`
    - LogiCORE PCI32 design files and updates
    - Reference Designs
    - Source files for PCI PCB layout
    - Source files for example drivers
    - Extensive Application Notes
    - Known issues and design tips
- Answer Database on Xilinx web-site - www.xilinx.com
- Hotline telephone support
- Apps fax and email
- Technical email Newsletter

Technical support for Vireo Driver::Works and Vireo VtoolsD driver development tools are provided by Vireo only.

Licensing

Xilinx LogiCORE PCI32 Interfaces are licensed under the standard LogiCORE license agreement, see the last page of this chapter. Additional licenses are available for evaluation or for design with non-Xilinx technology. Contact your local Xilinx representative for more details.

The Vireo Driver::Works and Vireo VtoolsD driver development tools are licensed for development and prototyping only. An unrestricted license can be purchased from Vireo Software.

LogiCORE PCI32 Spartan

Part no: DO-DI-PCI32-S

Overview

The LogiCORE PCI32 Spartan interface is included in Xilinx PCI32 Design Kit, but may be purchased separately as a web-only, one-time license.

Package Includes

See individual data sheets for details

- LogiCORE PCI32 Spartan Interface
    - Configurable and downloadable from Xilinx web (no source files)
    - Support for Xilinx Spartan family (see data sheet for specific parts)
    - Master and Slave (Initiator/Target) functionality
    - On-line LogiCORE PCI32 User's Guide
    - One Synthesizable PCI Bridge Design Example (RD03)

Support and Updates

Core updates released within 3 months after purchase are included. Annual maintenance contract is not available for this product.

- Access to Xilinx LogiCORE PCI32 Spartan lounge `www.xilinx.com/products/logicore/pci/pci_sol.htm`
    - LogiCORE PCI32 Spartan design files and updates
    - Reference Design
    - Application Notes
    - Known issues and design tips
- Answer Database on Xilinx web-site - www.xilinx.com
- Hotline telephone support
- Apps fax and email
- Technical email Newsletter

Licensing

Xilinx LogiCORE PCI32 Interfaces are licensed under the standard LogiCORE license agreement, see the last page of this chapter.

Additional PCI Products from Virtual Computer Corporation

For pricing and availability, please contact VCC directly.

VIRTUAL COMPUTER
CORPORATION

6925 Canby Ave. #103
Reseda, CA 91335 USA
Phone: +1 818-342-8294
Fax: +1 818-342-0240
E-mail: `info@vcc.com`
Website: `http://www.vcc.com`

Products

- Programmable Voltage Control Module
    - 3.3v to 1.8v
    - Part no: HOT2-PWR
- Prototyping Daughter Card
    - Part no: HOT2-P
- XC40125 Extended Logic Daughter Card
    - Part no: HOT2-125
- XC6264 RPU Daughter Card
    - Part no: HOT2-6264
- HotPCI with Xilinx XC4062XL FPGAs
    - Part no: HOT2-XL
- Configuration Cache Manager License
    - Contact VCC

Additional PCI Products from Vireo Software

A unrestricted license for Vireo Driver::Works and VtoolsD Development tools can be purchased from Vireo. For pricing and availability, please contact Vireo directly

Vireo Software

30 Monument Square, Suite 135
Concord, MA 01742 USA
Phone: +1 978-369-3380
Fax: +1 978-318-6946
E-mail: info@vireo.com
support@vireo.com
Website: http://www.vireo.com

Obsolete products

- LogiCORE PCI Master V2.0 (Part no: DO-DI-PCIM) is no longer available for new purchases. Existing customers with valid maintenance will still receive core design file updates. Additionally, an upgrade package to the complete PCI32 Design Kit is available for customers under valid maintenance agreement. Contact Xilinx for price information.
- LogiCORE PCI Slave V2.0 (Part no: DO-DI-PCIS) is no longer available for new purchases. Existing customers with valid maintenance will still receive core design file updates.

License Agreement

XILINX LOGICORE "PCI MASTER INTERFACE" AND "PCI SLAVE INTERFACE"

PLEASE READ THIS DOCUMENT CAREFULLY BEFORE USING THE **XILINX LOGICORE PCI MASTER INTERFACE** OR **PCI SLAVE INTERFACE DESIGN**, BY USING THE XILINX LOGICORE DESIGN, YOU ARE AGREEING TO BE BOUND BY THE TERMS OF THIS LICENSE.

1. **License.** XILINX, INC. ("XILINX") hereby grants you a nonexclusive, non-transferable license to use the LOGICORE PCI MASTER/SLAVE INTERFACE design (the "Design"), solely for your use in developing designs for XILINX programmable logic devices or XILINX HardWire™ devices. Use of the Design in non-XILINX devices or technologies is prohibited unless you have entered into a separate written agreement with XILINX for such use. XILINX retains title to the Design and to any patents, copyrights, trade secrets and other intellectual property rights therein. To protect such intellectual property rights, you may not decompile, reverse engineer, disassemble, or otherwise reduce the Design to a human-perceivable form. You may not modify or prepare derivative works of the Design in whole or in part. This License allows you to make an unlimited number of copies of the Design for internal use only, provided you reproduce on each copy of the Design the copyright and any other proprietary legends that were on the original copy of the Design.
2. **Termination.** This License is effective until terminated. You may terminate this License at any time by destroying the Design and all copies thereof. This License will terminate immediately without notice from XILINX if you fail to comply with any provision of this License. Upon termination you must destroy the Design and all copies thereof.
3. **Governmental Use.** The Design is commercial computer software developed exclusively at Xilinx's expense. Accordingly, pursuant to the Federal Acquisition Regulations (FAR) Section 12.212 and Defense FAR Supplement Section 227.7202, use, duplication and disclosure of the Design by or for the Government is subject to the restrictions set forth in this License Agreement. Manufacturer is XILINX, INC., 2100 Logic Drive, San Jose, California 95124.
4. **Limited Warranty and Disclaimer.** THE DESIGN IS PROVIDED TO YOU "AS IS". XILINX AND ITS LICENSORS MAKE AND YOU RECEIVE NO WARRANTIES OR CONDITIONS, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, AND XILINX SPECIFICALLY DISCLAIMS ANY IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE. XILINX does not warrant that the functions contained in the Design will meet your requirements, or that the operation of the Design will be uninterrupted or error free, or that defects in the Design will be corrected. Furthermore, XILINX does not warrant or make any representations regarding use or the results Design in terms of correctness, accuracy, reliability or otherwise.
5. **Limitation of Liability.** IN NO EVENT WILL XILINX OR ITS LICENSORS BE LIABLE FOR ANY LOSS OF DATA, LOST PROFITS, COST OF PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES, OR FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL OR INDIRECT DAMAGES ARISING FROM THE USE OR OPERATION OF THE DESIGN OR ACCOMPANYING DOCUMENTATION, HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY. THIS LIMITATION WILL APPLY EVEN IF XILINX HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. THIS LIMITATION SHALL APPLY NOTWITHSTANDING THE FAILURE OF THE ESSENTIAL PURPOSE OF ANY LIMITED REMEDIES HEREIN.
6. **Export Restriction.** You agree that you will not export or re-export the Design, reference images or accompanying documentation in any form without the appropriate United States and foreign government licenses. Your failure to comply with this provision is a material breach of this Agreement.
7. **Third Party Beneficiary.** You understand that portions of the Design and related documentation may have been licensed to XILINX from third parties and that such third parties are intended third party beneficiaries of the provisions of this Agreement.
8. **Non-Transferable.** You may not provide design source information including, but not limited to, schematics, hardware description language source code, or netlist files, to a third party without prior written approval from XILINX. You may provide device programming files— XILINX bitstream files or PROM files or the resulting HardWire gate array to third-parties without prior approval.
9. **General.** This License shall be governed by the laws of the State of California, without reference to conflict of laws principles. If for any reason a court of competent jurisdiction finds any provision of this License, or portion thereof, to be unenforceable, that provision of this License shall be replaced to the maximum extent permissible so as to effectuate the intent of the parties, and the remainder of this License shall continue in full force and effect. This License constitutes the entire agreement between the parties with respect to the use of this Design and related documentation, and supersedes all prior or contemporaneous understandings or agreement, written or oral, regarding such subject matter.

I have read the Xilinx LogiCORE License Agreement, and I agree to comply with the terms and conditions therein.

Name: ______________________________ Date: ______________________________
(Signed)

Printed: ______________________________

Title: ______________________________ Company: ______________________________

Address: ______________________________ ______________________________

E-mail: ______________________________ PO#: ______________________________

Fax to: +1 408-377-3259 (USA)

XILINX®

Sales Offices, Sales Representatives, and Distributors

XILINX®

Sales Offices, Sales Representatives, and Distributors

Headquarters

XILINX, Inc.
2100 Logic Drive
San Jose, CA 95124
Tel: (408) 559-7778
TWX: (510) 600-8750
FAX: (408) 559-7114

Xilinx Sales Offices

NORTH AMERICA

XILINX, Inc.
1281 Oakmead Pkwy.
Suite 202
Sunnyvale, CA 94086
Tel: (408) 245-9850
FAX: (408) 245-9865

XILINX, Inc.
5690 DTC Blvd.
Suite 490W
Englewood, CO 80111
Tel: (303) 220-7541
FAX: (303) 220-8641

XILINX, Inc.
15615 Alton Parkway
Suite 280
Irvine, CA 92618
Tel: (714) 727-0780
FAX: (714) 727-3128

XILINX, Inc.
61 Spit Brook Rd.
Suite 403
Nashua, NH 03060
Tel: (603) 891-1098
FAX: (603) 891-0890

XILINX, Inc.
905 Airport Rd.
Suite 200
West Chester, PA 19380
Tel: (610) 430-3300
FAX: (610) 430-0470

XILINX, Inc.
939 North Plum Grove Road
Suite H
Schaumburg, IL 60173
Tel: (847) 605-1972
FAX: (847) 605-1976

XILINX, Inc.
6010-C Six Forks Road
Raleigh, NC 27609
Tel: (919) 846-3922
FAX: (919) 846-8316

XILINX, Inc.
4100 McEwen, Suite 237
Dallas, TX 75244
Tel: (972) 960-1043
FAX: (972) 960-0927

XILINX, Inc.
2910 South Sheridan Way,
Suite 203
Oakville, Ontario
Canada L6J7L9
Tel: (905) 829-9095
FAX: (905) 829-3045

EUROPE

XILINX, Ltd.
Benchmark House
203 Brooklands Road
Weybridge
Surrey KT13 ORH
United Kingdom
Tel: (44) 1-932-349-401
FAX: (44) 1-932-349-499

XILINX, Ltd.
(Northern European Sales)
Suite 1B Cobb House
Oyster Lane Byfleet
Surrey KT14 7DU
United Kingdom
Tel: (44) 1-932-349-403
FAX: (44) 1-932-345-519

XILINX SARL
Espace Jouy Technology
21, rue Albert Calmette, Bt. C
78353 Jouy En Josas, Cedex
France
Tel: (33) 1 34 63 01 01
FAX: (33) 1 34 63 01 09

XILINX GmbH
Süskindstr. 4
D-81929 München
Germany
Tel: (49) 89-93088-0
Tech Support Tel:
(49) 89-93088-130
FAX: (49) 89-93088-188

XILINX AB
Box 1230
Torshamnsgatan 35
S-164 28 Kista
Sweden
Tel: (46) 8-752-2470
FAX: (46) 8-750-6260
E-mail: xilinx-nordic@xilinx.com

JAPAN

XILINX K. K.
Daini-Nagaoka Bldg. 2F
2-8-5, Hatchobori, Chuo-ku
Tokyo 104
Japan
Tel: (81) 3-3297-9191
FAX: (81) 3-3297-9189

ASIA PACIFIC

XILINX Asia Pacific
Unit 4312, Tower II
Metroplaza
Hing Fong Road
Kwai Fong, N.T.
Hong Kong
Tel: (852) 2-424-5200
FAX: (852) 2-494-7159
E-mail: hongkong@xilinx.com

XILINX Korea
Room #901
Sambo-Hojung Bldg.,
14-24, Yoido-Dong
Youngdeungpo-Ku
Seoul, South Korea
Tel: (82) 2-761-4279
FAX: (82) 2-761-4278

XILINX Taiwan
Rm. 1006, 10F, No. 2, Lane 150
Sec. 5, Hsin Yin Rd.
Taipei, Taiwan, R.O.C.
Tel: (886) 2-747-1047
FAX: (886) 2-747-1046

North American Distributors

Hamilton Hallmark
Locations throughout
the U.S. and Canada.
Tel: (800) 332-8638
FAX: (800) 257-0568

Insight Electronics
(Locations throughout the U.S.)
Tel: (800) 677-7716
FAX: (619) 587-1380

Marshall Industries
(Locations throughout
the U.S. and Canada)
Tel: (800) 522-0084
FAX: (818) 307-6297

Nu Horizons
Electronics Corp.
Locations throughout the U.S.
Tel: (516) 396-5000
FAX: (516) 396-7576

U.S. Sales Representatives

ALABAMA

Electro Source, Southeast
4825 University Sq., Ste.12
Huntsville, AL 35816
Tel: (205) 830-2533
FAX: (205) 830-5567

ARIZONA

Quatra Associates
10235 S. 51st St. Suite #160
Phoenix, AZ 85044
Tel: (602) 753-5544
FAX: (602) 753-0640
E-mail: quatra@earthlink.net

ARKANSAS

Bonser-Philhower Sales
689 W. Renner Road
Suite 101
Richardson, TX 75080
Tel: (972) 234-8438
FAX: (972) 437-0897

CALIFORNIA

Norcomp
1267 Oakmead Pkwy
Sunnyvale, CA 94086
Tel: (408) 733-7707
FAX: (408) 774-1947

Norcomp
8880 Wagon Way
Granite Bay, CA 95746
Tel: (916) 791-7776
FAX: (916) 791-2223

Norcomp
30101 Agoura Ct. #234
Agoura, CA 91301
Tel: (818) 865-8330
FAX: (818) 865-2167

Norcomp
30 Corporate Park #200
Irvine, CA 92714
Tel: (714) 260-9868
FAX: (714) 260-9659

Quest-Rep Inc.
6494 Weathers Pl, Suite 200
San Diego, CA 92121
Tel: (619) 622-5040
FAX: (619) 622-9007
E-mail: questrep@ques-trep.com

COLORADO

Luscombe Engineering, Inc.
1500 Kansas Ave. Suite 1B
Longmont, CO 80501
Tel: (303) 772-3342
FAX: (303) 772-8783

CONNECTICUT

John E. Boeing, Co., Inc.
101 Harvest Park, Bldg. 1A
No. Plains Industrial Road
Wallingford, CT 06492
Tel: (203) 265-1318
FAX: 203-265-0235

DELAWARE

Delta Technical Sales, Inc.
122 N. York Rd., Suite 9
Hatboro, PA 19040
Tel: (215) 957-0600
FAX: (215) 957-0920

FLORIDA

Semtronic Assoc., Inc.
(Disti Office)
600 S. Northlake Blvd. #270
Altamonte Springs, FL 32701
Tel: (407) 831-0451
FAX: (407) 831-6055

Semtronic Assoc., Inc.
657 Maitland Avenue
Altamonte, Springs, FL 32701
Tel: (407) 831-8233
FAX: (407) 831-2844

Semtronic Assoc., Inc.
3471 NW 55th Street
Ft. Lauderdale, FL 33309
Tel: (954) 731-2384
FAX: (954) 731-1019

Semtronic Assoc., Inc.
1467 South Missouri Avenue
Clearwater, FL 33756
Tel: (813) 461-4675
FAX: (813) 442-2234

GEORGIA

Electro Source, Southeast
3280 Pointe Parkway,
Suite 1500
Norcross, GA 30092
Tel: (770) 734-9898
FAX: (770) 734-9977

IDAHO (Southwest)

Luscombe Engineering, Inc.
6901 Emerald, Suite 206
Boise, ID 83704
Tel: (208) 377-1444
FAX: (208) 377-0282

Thorson Pacific, Inc.
14575 Bel-Red Road #102
Bellevue, WA 98007
Tel: (425) 603-9393
FAX: (425) 603-9380

ILLINOIS

Advanced Technical Sales
13755 St. Charles Rock Rd.
Bridgeton, MO 63044
Tel: (314) 291-5003
FAX: (314) 291-7958

Beta Technology Sales, Inc.
1009 Hawthorn Drive
Itasca, IL 60143
Tel: (708) 250-9586
FAX: (708) 250-9592

INDIANA

Gen II Marketing, Inc.
31 E. Main St.
Carmel, IN 46032
Tel: (317) 848-3083
FAX: (317-848-1264

Gen II Marketing, Inc.
1415 Magnavox Way
Suite 130
Ft. Wayne, IN 46804
Tel: (219) 436-4485
FAX: (219) 436-1977

IOWA

Advanced Technical Sales
375 Collins Road NE
Cedar Rapids, IA 52402
Tel: (319) 393-8280
FAX: (319) 393-7258

KANSAS

Advanced Technical Sales
2012 Prairie Cir. Suite A
Olathe, KS 66062
Tel: (913) 782-8702
FAX: (913) 782-8641

KENTUCKY

Gen II Marketing, Inc.
861 Corporate Dr. #210
Lexington, KY 40503
Tel: (606) 223-9181
FAX: (606) 223-2864

LOUISIANA (Northern)

Bonser-Philhower Sales
689 W. Renner Rd., Suite 101
Richardson, TX 75080
Tel: (972) 234-8438
FAX: (972) 437-0897

LOUISIANA (Southern)

Bonser-Philhower Sales
10700 Richmond, Suite 150
Houston, TX 77042
Tel: (713) 782-4144
FAX: (713) 789-3072

MAINE

Genesis Associates
128 Wheeler Road
Burlington, MA 01803
Tel: (617) 270-9540
FAX: (617) 229-8913

MARYLAND

Micro Comp, Inc.
1421 S. Caton Avenue
Baltimore, MD 21227-1082
Tel: (410) 644-5700
FAX: (410) 644-5707

MASSACHUSETTS

Genesis Associates
128 Wheeler Road
Burlington, MA 01803
Tel: (617) 270-9540
FAX: (617) 229-8913

MICHIGAN

Miltimore Sales Inc.
22765 Heslip Drive
Novi, MI 48375
Tel: (248) 349-0260
FAX: (248) 349-0756

Miltimore Sales Inc.
3680 44th St., Suite 100-J
Kentwood, MI 49512
Tel: (616) 554-9292
FAX: (616) 554-9210

MINNESOTA

Beta Technology
18283 Minnetonka Blvd.
Suite C
Deephaven, MN 55391
Tel: (612) 473-2680
FAX: (612) 473-2690

MISSISSIPPI

Electro Source, Southeast
4835 University Sq., Ste.11
Huntsville, AL 35816
Tel: (205)-830-2533
FAX: (205) 830-5567

MISSOURI

Advanced Technical Sales
2012 Prairie Cir. Suite A
Olathe, KS 66062
Tel: (913) 782-8702
FAX: (913) 782-8641

Advanced Technical Sales
13755 St. Charles Rock Rd.
Bridgeton, MO 63044
Tel: (314) 291-5003
FAX: (314) 291-7958

MONTANA

Luscombe Engineering, Inc.
670 East 3900 South #103
Salt Lake City, UT 84107
Tel: (801) 268-3434
FAX: (801) 266-9021

NEBRASKA

Advanced Technical Sales
375 Collins Road NE
Cedar Rapids, IA 52402
Tel: (319) 393-8280
FAX: (319) 393-7258

NEVADA

Norcomp
8880 Wagon Way
Granite Bay, CA 95748
Tel: (916) 393-8280
FAX: (916) 393-7258

Quatra Associates
(Las Vegas)
4645 S. Lakeshore Dr., Suite 1
Tempe, AZ 85282
Tel: (602) 820-7050
FAX: (602) 820-7054

NEW HAMPSHIRE

Genesis Associates
128 Wheeler Road
Burlington, MA 01803
Tel: (617) 270-9540
FAX: (617) 229-8913

NEW JERSEY (Northern)

Parallax
734 Walt Whitman Road
Melville, NY 11747
Tel: (516) 351-1000
FAX: (516) 351-1606

NEW JERSEY (Southern)

Delta Technical Sales, Inc.
122 N. York Road, Suite 9
Hatboro, PA 19040
Tel: (215) 957-0600
FAX: (215) 957-0920

NEW MEXICO

Quatra Associates
600 Autumnwood Place, SE
Albuquerque, NM 87123
Tel: (505) 296-6781
FAX: (505) 292-2092

NEW YORK (Metro)

Parallax
734 Walt Whitman Road
Melville, NY 11747
Tel: (516) 351-1000
FAX: (516) 351-1606

NEW YORK

Electra Sales Corp.
333 Metro Park
Suite M103
Rochester, NY 14623
Tel: (716) 427-7860
FAX: (716) 427-0614

Electra Sales Corp.
6700 Old Collamer Rd.
E. Syracuse, NY 13057
Tel: (315) 463-1248
FAX: (315) 463-1717

NORTH CAROLINA

Electro Source, Southeast
5964-A Six Forks Rd.
Raleigh, NC 27609
Tel: (919) 846-5888
FAX: (919) 846-0408

NORTH DAKOTA

Beta Technology
18283 Minnetonka Blvd.
Suite C
Deephaven, MN 55391
Tel: (612) 473-2680
FAX: (612) 473-2690

OHIO

Bear Marketing, Inc.
3554 Brecksville Road
PO Box 427
Richfield, OH 44286-0427
Tel: (216) 659-3131
FAX: (216) 659-4823

Bear Marketing, Inc.
270 Regency Ridge Drive
Suite 115
Dayton, OH 45459
Tel: (513) 436-2061
FAX: (513) 436-9137

OKLAHOMA

Bonser-Philhower Sales
689 W. Renner Rd., Suite 101
Richardson, TX 75080
Tel: (972) 234-8438
FAX: (972) 437-0897

OREGON

Thorson Pacific, Inc.
9600 SW Oak Street,
Suite 320
Portland, OR 97223
Tel: (503) 293-9001
FAX: (503) 293-9007

PENNSYLVANIA

Bear Marketing, Inc.
4284 Rt. 8, Suite 211
Allison Park, PA 15101
Tel: (412) 492-1150
FAX: (412) 492-1155

Delta Technical Sales, Inc.
122 N. York Rd., Suite 9
Hatboro, PA 19040
Tel: (215) 957-0600
FAX: 215-957-0920

PUERTO RICO

Semtronic Assoc., Inc.
Crown Hills
125 Carite St.
Esq. Avenue Parana
Rio Piedras, P.R. 00926
Tel: (787) 766-0700/0701
FAX: (787) 763-8071

RHODE ISLAND

Genesis Associates
128 Wheeler Road
Burlington, MA 01803
Tel: (617) 270-9540
FAX: (617) 229-8913

SOUTH CAROLINA

Electro Source, Southeast
5964-A Six Forks Rd.
Raleigh NC 27609
Tel: (919) 846-5888
FAX: (919) 846-0408

SOUTH DAKOTA

Beta Technology
18283 Minnetonka Blvd.
Suite C
Deephaven, MN 55391
Tel: (612) 473-2680
FAX: (612) 473-2690

TENNESSEE

Electro Source, Southeast
4825 University Square,
Suite 12
Huntsville, AL 35816
Tel: (205) 830-2533
FAX: (205)-830-5567

TEXAS

Bonser-Philhower Sales
8240 MoPac Expwy.
Suite 295
Austin, TX 78759
Tel: (512) 346-9186
FAX: (512) 346-2393

Bonser-Philhower Sales
10700 Richmond, Suite 150
Houston, TX 77042
Tel: (713) 782-4144
FAX: (713) 789-3072

Bonser-Philhower Sales
689 W. Renner Rd., Suite 101
Richardson, TX 75080
Tel: (972) 234-8438
FAX: (972) 437-0897

TEXAS (El Paso County)

Quatra Associates
600 Autumnwood Place SE
Albuquerque, NM 87123
Tel: (505) 296-6781
FAX: (505) 292-2092

UTAH

Luscombe Engineering Co.
670 East 3900 South #103
Salt Lake City, UT 84107
Tel: (801) 268-3434
FAX: (801) 266-9021

VERMONT

Genesis Associates
128 Wheeler Road
Burlington, MA 01803
Tel: (617) 270-9540
FAX: (617) 229-8913

VIRGINIA

Microcomp, Inc.
1421 S. Caton Avenue
Baltimore, MD 21227
Tel: (410) 644-5700
FAX: (410) 644-5707

WASHINGTON

Thorson Pacific, Inc.
14575 Bel-Red Rd.
Suite 102
Bellevue, WA 98007
Tel: (206) 603-9393
FAX: (206) 603-9380

WASHINGTON (Vancouver, WA only)

Thorson Pacific, Inc.
9600 SW Oak Street
Suite 320
Portland, OR 97223
Tel: (503) 293-9001
FAX: (503) 993-9007

WASHINGTON D.C.

Micro Comp, Inc.
1421 S. Caton Avenue
Baltimore, MD 21227-1082
Tel: (410) 644-5700
FAX: (410) 644-5707

WEST VIRGINIA

Bear Marketing, Inc.
4284 Rt. 8 Suite 211
Allison Park, PA 15101
Tel: (412) 492-1150
FAX: (412) 492-1155

WISCONSIN (Western)

Beta Technology
18283 Minnetonka Blvd.
Suite C
Deephaven, MN 55391
Tel: (612) 473-2680
FAX: (612) 473-2690

WISCONSIN (Eastern)

Beta Technology Sales, Inc.
9401 N. Beloit, Suite 409
Milwaukee, WI 53227
Tel: (414) 543-6609
FAX: (414) 543-9288

WYOMING

Luscombe Engineering, Inc.
1500 Kansas Ave. Suite 1B
Longmont, CO 80501
Tel: (303) 772-3342
FAX: (303) 772-8783

International Sales Representatives

ALGERIA

Development Centre of
Advanced Technologies
128 Chemin Mohamed
GACEM
16075 El-Madania
Algiers
Algeria
Tel: (213) 2-67-73-25
FAX: (213) 2-66-26-89

ARABIA

Culato
P.O. Box 51973
Dubai
United Arab Emirates
Tel: (971)-4-284934
FAX: (971)-4-284031

AUSTRALIA

Advanced Component Dist.
Suite 5, Level 1, "Metro Centre"
124 Forest Rd.
P.O. Box 574, Hurstville 2220
Australia
Tel: (61) 2-9585-5030
FAX: (61) 2-9590-1095

Advanced Component Dist.
Unit 2, 17-19 Melrich Road
Bayswater VIC 3153
Melbourne, Australia
Tel: (61) 3-9762-4244
FAX: (61) 3-9761-1754

Advanced Component Dist.
20D William Street
Norwood SA 5067
Australia
Tel: (61) 8-364-2844
FAX: (61) 8-364-2811

Advanced Component Dist.
Ste. 1, 1048 Beaudesert Rd.
Cooper Plains
Queensland 4108
Australia

Tel: (61) 7-246-5214
FAX: (61) 7-275-3662

Advanced Component Dist.
Ste. 8, 328 Albany Highway
Victoria Park
W.A. 6100
Australia
Tel: (61) 9-9472-3232
FAX: (61) 9-9470-9632

AUSTRIA

SEI Elbatex GmbH
Eitnergasse 6
1230 Vienna
Austria
Tel: (43) 1-866-4220
FAX: (43) 1-866-42201

BELGIUM & LUXEMBURG

SEI Rodelco NV
Limburg Stirum 243
1780 Wemmel
Belgium
Tel: (32) 2-456-0757
FAX: (32) 2-460-0271

CANADA (ALBERTA)

Electro Source
2635 37th Ave NE #245
Calgary, Alberta T1Y 5Z6
Canada
Tel: (403) 735-6230
FAX: (403) 735-0599

CANADA (BRITISH COLUMBIA)

Thorson Pacific, Inc.
4170 Still Creek Dr. #200
Burnaby BC V5C 6C6
Canada
Tel: (604) 294-3999
FAX: (604) 473-7755

CANADA (OTTAWA)

Electro Source, Inc.
300 March Road, Suite 203
Kanata, Ontario K2K 2E4
Canada
Tel: (613) 592-3214
FAX: (613) 592-4256

CANADA (QUEBEC)

Electro Source
6600 TransCanada Hwy
Suite 420
Pointe Claire, Quebec H9R 4S2
Canada
Tel: (514) 630-7486
FAX: (514) 630-7421

CANADA (TORONTO)

Electro Source, Inc.
230 Galaxy Blvd.
Rexdale Ontario M9W 5R8
Canada
Tel: (416) 675-4490
FAX: (416)-675-6871

CHINA PEOPLE'S REPUBLIC

MEMEC (Beijing Rep. Office)
Rm. 410, Xuhai Bldg.
No. 85 Haidian Rd.
Beijing, 10086
P.R. China
Tel: (86) 10-6253-5475
FAX: (86) 10-6253-5475

MEMEC (Chengdu Rep. Office)
Rm. 1808, Tong Mei Mansion,
No. 76, Section 1,
Jianshe North Rd.,
Chengdu, Sichuan 610051
P.R. China
Tel: (86) 28-3399-629
FAX: (86) 28-3398-829

MEMEC (Shenzhen Rep. Office)
Rm. 715, Bao Hua Bldg.,
Hua Qiang North Rd.,
Shenzhen 518031
P.R. China
Tel: (86) 755-3241-548
FAX: (86) 755-3233-026

MEMEC Asia Pacific Ltd.
Shanghai Rep. Office
Rm. 2363, West Building
Jim Jiang Hotel
59 Mao Ming Rd. S.,
Shanghai 200020
P.R. China
Tel: (86) 21-6258-2582
FAX: (86) 21-6415-1116

CZECH REPUBLIC

SEI Eljapex/Elbatex GmbH
Prechodni 11
CZ-140 00 Praha 4
Czech Republic
Tel: (02)-6928087
FAX: (02)-4718203

DENMARK

Micronor A/S
P.O. Box 929
Torvet 1
DK-8600 Silkeborg
Denmark
Tel: (45) 8681-6522
FAX: (45) 8681-2827
E-Mail: e-mail@micronor.dk
WWW: http://www.micronor.dk

EGYPT

Guide System Integrators
27 Mokhles Al-Alfi St.
First - Zone Nasr City
Cairo
Egypt
Tel: (20) 2-401-4085
FAX: (20) 2-401-4997
E-Mail: obadr@idsc.gov.eg

FINLAND

Memec Finland Oy
Vernissakatu 6
01300
Vantaa
Finland
Tel: (358) 9-836-2600
FAX: (358) 9-8362-6027

FRANCE

REP'TRONIC
1 Bis, rue Marcel Paul
Z.I. La Bonde
91742 Massy
France
Tel: (33) 1 69 53 67 20
FAX: (33) 1 60 13 91 98
E-Mail:
100745.605@compuserve.com

AVNET/EMG.
79 Rue Pierre Sémard
92320 Châtillon
France
Tel: (33) 1 49 65 27 00
FAX: (33) 1 49 65 27 39

AVNET Composants
Sud-Ouest
Technoparc
Bât.4, Voie 5, BP 404
31314 Labège Cedex
France
Tel: (33) 5 61 39 21 12
FAX: (33) 5 61 39 21 40

AVNET Composants
Rhône-Auvergne
Parc Club du Moulin à Vent
Bât 32-33, rue du Dr. G. Levy
69693 Vénissieux Cedex
France
Tel: (33) 4 78 00 1280
FAX: (33) 4 78 75 95 97

AVNET Composants
Ouest
Technoparc-Bât. E
4 Av. des Peupliers, BP 43
35511 Cesson Sévigné Cedex
France
Tel: (33) 2 99 83 84 85
FAX: (33) 2 99 83 80 83

AVNET Composants
Rhône-Alpes
Zac des Béalières
23 Av. de Granier
38240 Meylan
France
Tel: (33) 4 76 90 11 88
FAX: (33) 4 76 41 04 90

AVNET Composants
Nantes
Le Sillon de Bretagne
23e étage-Aile C
8 Av. des Thébaudières
44802 Saint Herblain
France
Tel: (33) 2 40 16 98 50
FAX: (33) 2 40 63 22 88

AXESS Technology
49 rue de l'Estérel
Silic 600
94663 Rungis Cedex
France
Tel: (33) 1 49 78 94 94
Fax: (33) 1 49 78 03 24

AXESS Technology
Agence: Bretagne
3 Square du Chêne Germain
35510 Cesson-Sévigne
France
Tel: (33) 2 99 36 83 83
FAX: (33) 2 99 36 53 92

AXESS Technology
Agence: Sud-ouest
7, Impasse de la Marjerine
31240 L'Union
France
Tel: (33) 5 61 09 99 44
FAX: (33) 5 61 09 19 79

AXESS Technology
Agence: Rhône-Alpes
23 le Grand François
La Courtbatière
38140 Rives
France
Tel: (33) 4 76 91 45 30
FAX: (33) 4 76 91 45 34

GERMANY

Avnet EMG GmbH
Stahlgruberring 12
81829 München
Germany
Tel: (49) 89-45110-01
FAX: (49) 89-45110-129

Avnet EMG GmbH
Kurfürstenstr. 130
10785 Berlin
Germany
Tel: (49) 30-214882-0
FAX: (49) 30-2141728

Avnet EMG GmbH
Friedrich-Ebert-Damm 145
22047 Hamburg

Germany
Tel: (49) 40-696952-0
FAX: (49) 40-6962787

Avnet EMG GmbH
Benzstr. 1
70839 Gerlingen
Stuttgart, Germany
Tel: (49) 7156-4390
FAX: (49) 7156-4390-111

Avnet EMG GmbH
Heinrich-Hertz-Str. 52
40699 Erkrath
Düsseldorf, Germany
Tel: (49) 211-92003-0
FAX: (49) 211-9200399

Avnet EMG GmbH
Schmidtstr. 49
60326 Frankfurt/M.
Germany
Tel: (49) 69-973804-0
FAX: (49) 69-7380712

Avnet EMG GmbH
Fürther Str. 212
90429 Nürnberg
Germany
Tel: (49) 911-93149-0
FAX: (49) 911-320821

Intercomp
Am Hochwald 42
D-82319
Starnberg, Germany
Tel: (49) 8151-16044
FAX: (49) 8151-79270
E-Mail:
intercomp.tiefenthaler@t-online.de

Intercomp
Kniebisstr. 40/1
78628 Rottweil
Stuttgart, Germany
Tel: (49) 741-14845
FAX: (49) 741-15220
E-Mail:
intercomp.klink@t-online.de

Intercomp
Schustergasse 35
55278 Köngernheim
Frankfurt, Germany
Tel: (49) 6737-9881
FAX: (49) 6737-9882
E-Mail:
intercomp.hakam@t-online.de

Metronik GmbH
Leonhardsweg 2
82008 Unterhaching
München, Germany
Tel: (49) 89-611-080
FAX: (49) 89-611-08110

Metronik GmbH
Zum Lonnenhohl 40
D-44319 Dortmund
Germany
Tel: (49) 231-9271-10-0
FAX: (49) 231-9271-10-99
E-Mail:
10063.44@compuserve.com

Metronik GmbH
Carl-Zeiss Str.6
D-25421 Quickborn
Hamburg Germany
Tel: (49) 4106-77-30-50
FAX: (49) 4106-77-30-52

Metronik GmbH
Osmiastrasse 9
D-69221 Dossenheim
Mannheim, Germany
Tel: (49) 6221-8-70-44
FAX: (49) 6221-8-70-46

Metronik GmbH
Pilotystr. 27/29
90408 Nürnberg
Germany
Tel: (49) 89-611-08-145
FAX: (49) 89-611-08-144

Metronik GmbH
Löwenstrasse 37
70597 Stuttgart
Germany
Tel: (49) 711-764-033
FAX: (49) 711-765-5181
E-Mail:
100751.334@compuserve.com

Metronik GmbH
Franz-Schubert-Str.41
D-16548 Glienicke
Berlin, Germany
Tel: (49) 33056-845-0
FAX: (49) 33056-845-50

Metronik GmbH
Schönauer Strasse 113
D-04207 Leipzig
Germany
Tel: (49) 341-423-9413
FAX: (49) 341-423-9424

Metronik GmbH
Bahnstrasse 9
D-65205 Wiesbaden
Germany
Tel: (49) 611-973-84-0
FAX: (49) 611-973-8418

GREECE

Semicon
104 Aeolou Str.
10564 Athens
Greece
Tel: (30) 1-32-536-26
Fax: (30) 1-32-160-63
E-Mail: semicon@hellas.eu.net

HONG KONG

MEMEC Asia Pacific Ltd.
Units 3601-02 & 07-25, Tower I,
Metroplaza, Hing Fong Road,
Kwai Fong, N.T.
Hong Kong
Tel: (852) 2410-2780
FAX: (852) 2401-2518

HUNGARY

Elbatex HUN Kr
GF-NN
Szigeivan U.5
H-1083 Budapest
Hungary
Tel: (36) 1-2699093/95
FAX: (36) 1-2699096

INDIA

CG-CoreEl Logic System Ltd.
First Floor, Surya Bhavan
1181 Fergusson College Rd.
Pune 411 005 Maharashtra
India
Tel: (91) 212-323982, 328074, 328362
FAX: (91) 212-323985
Email: xsupport@cromp.ernet.in

CG-CoreEL Logic System Ltd.
961 Main
HAL 2nd Stage
Bangalore 560 008
India
Tel: (91) 80-527-9726
FAX: (91) 80-527-9726
E-Mail:
vishwa@giasbg01.vsni.net.in

IRELAND

Memec Ireland Ltd.
Gardner House
Bank Place
Limerick
Ireland
Tel: (353) 61-411842
FAX: (353) 61-411888
E-Mail: memec@iol.ie

ISRAEL

E.I.M International Ltd.
9 Hashiloach Street
P.O. Box 4000
Kiryat Arye
Petach Tikva 49130
Israel
Tel: (972) 3-926-4666
FAX: (972) 3-924-4857

ITALY

Acsis Srl
Via Alberto Mario, 26
20149 Milano, Italy
Tel: (39) 2-480-22522
FAX: (39) 2-480-12289
WWW: http://www.acsis.it

Acsis Srl
Via B. Battista Spagnoli, 61
00144 Roma
Tel: (39) 6-529-6589
FAX: (39) 6-529-6588

Avnet EMG
Centro Direzionale
Via Novara, 570
C.A.P 20153 Milan
Italy
Tel: (39) 2-381-901
FAX: (39) 2-380-02988

Avnet EMG
Ancona
Via Adriatica, 13
60022 Castelfidardo
Tel: (39) 71-781-9644
FAX: (39) 71-781-9699

Avnet EMG
Firenze
Via Panciatichi, 40
50127 Firenze
Tel: (39) 55-436-0392
FAX: (39) 55-431-035

Avnet EMG
Modena
Via Scaglia Est, 144
41100 Modena
Tel: (39) 59-351-300
FAX: (39) 59-344993

Avnet EMG
Napoli
Via Ferrante Imparato, 27
80146 Napoli
Tel: (39) 81-55-91477
FAX: (39) 81-55-91580

Avnet EMG
Roma
Via Zoe Fontana, 220
Technocitta
00131 Roma
Tel: (39) 64-13-1151
FAX: (39) 64-13-1161

Avnet EMG
Torino
Corso Orbassano, 336
10137 Torino
Tel: (39) 11-311-2347
FAX: (39) 11-308-2138

Avnet EMG
Treviso
Via Delle Querce, 7
31033 Castelfranco Veneto
Tel: (39) 423-722-675
FAX: (39) 423-722-671

Silverstar-Celdis
Viale Fulvio Testi, 280
20126 Milano, Italy
Tel: (39) 2-661-251
FAX: (39) 2-661-01-359

Silverstar-Celdis
Via Collamarini, 22
40138 Bologna, Italy
Tel: (39) 51-53-8500
FAX: (39) 51-53-8831

Silverstar-Celdis
Via G. Antonio Resti, 63
00143 Roma, Italy
Tel: (39) 6-519-57527
FAX: (39) 6-504-3330

Silverstar-Celdis
Centro Piero Della Francesca
Corso Svizzera, 185 Bis
10149 Torino, Italy
Tel: (39)11-77-10082
FAX: (39) 11-77-64921

Silverstar-Celdis
Centro Direzionale Benelli
Via Degli Abeti, 346
6100 Pesaro, Italy
Tel: (39) 721-26-560
FAX: (39) 721-400896

Silverstar-Celdis
Via A.da Noli, 6
50127 Firenze, Italy
Tel: (39) 55-43-5125
FAX: (39) 55-43-77184

Silverstar-Celdis
Via. delle Indus, 13
35010 Limena
Padova, Italy
Tel: (39) 49-88-40044
FAX: (39) 49-88-41079

Silverstar-Celdis
Via Famagosta, 1/5
17100 Savona, Italy
Tel: (39) 19-81-5090
FAX: (39) 19-81-5091

JAPAN

Kaga Electronics Co., Ltd.
1-26-1 Otowa
Bunkyo-ku
Tokyo 112, Japan
Tel: (81) 3-3942-6224
FAX: (81) 3-3942-6215

Marubun Corporation
8-1 Odenma-cho
Nihonbashi
Chuo-ku
Tokyo, 103 Japan
Tel: (81) 3-3639-9851
FAX: (81) 3-3639-9925

Okura Electronics Co., Ltd.
3-6, Ginza 2-chome,
Chuo-ku,
Tokyo, 104 Japan
Tel: (81) 3-3564-6871
FAX: (81) 3-3564-6870

Okura Electronics
Service Co., Ltd.
Yokota Bldg.
Ginza 2-11-5
Chuo-Ku
Tokyo, 104 Japan
Tel: (81) 3-3545-2360
FAX: (81) 3-3545-2351

Tokyo Electron Ltd.
TBS Broadcasting Center
5-3-6- Akasaka, Minato-Ku
Tokyo, 163 Japan
Tel: (81) 3-5561-7212
FAX: (81) 3-5561-7389

Towa Elex Co., Ltd.
Genbe Bldg.
5-4 Nihonbashi Tomisawa-cho
Chuo-Ku, Tokyo 103
Japan
Tel: (81) 3-5640-1241
FAX: (81) 3-5640-1240

THE NETHERLANDS

Rodelco BV
P.O. Box 6824
Takkebijsters 2
4802 HV Breda
The Netherlands
Tel: (31) 76-5722700
FAX: (31) 76-5710029

NEW ZEALAND

Advanced Component Dist.
Unit 7, 110 Mays Rd.
Penrose, Auckland
New Zealand
Tel: (64) 9-636-5984
FAX: (64) 9-636-5985

Advanced Component Dist.
Worchester Chambers
69 Worcester St.
Christchurch
New Zealand
Tel: (64) 3-379-3889
FAX: (64) 3-379-3072

MEMEC EBV (NZ) Ltd.
Suite 5a, Level 4,
North City Plaza,
Titahi Bay Rd., Porirua,
Wellington, New Zealand
Tel: (64) 4-237-9711
FAX: (64) 4-237-9718

NORWAY

BIT Elektronikk AS
Smedsvingen 4
P.O. Box 194
1360 Nesbru
Norway
Tel: (47) 66-77-65-00
FAX: (47) 66-77-65-01

POLAND

SEI NEljapex/Elbatex GmbH
Ul. Wilcza 50/52
PL-00679 Warszawa
Poland
Tel: (48) 2621-7122
FAX: (48) 2623-0605

PORTUGAL

ADM Electronica SA
en 107, No 743
Aguas Santas
4445 Ermesinde
Portugal
Tel: (35) 1-297-36957
FAX: (35) 1-297-36958

RUSSIA

Scan
10/32 "B" Druzhby St.
117330 Moskva
Russia
Tel: (7) 095-232-2343
FAX: (7) 095-938-2247

Scan
42 Ordjonikidze
196143 St. Petersburg
Russia
Tel: (7) 812-299-7028
FAX: (7) 812-264-6000

Scan
14 Plekhanova St.
Voronezh 394000
Russia
Tel: (7) 0732-718189
FAX: same

Scan Pulsar
9 Rogaliova St.
Dniepropetrovsk 320030
Ukrain
Tel: (7) 0562-472870
FAX: (7) 0562-451115

SINGAPORE

Frontline Technologies Ptc. Ltd
2 Science Park Drive #57
The Faraday
Singapore Science Park
Singapore 118222
Tel: (65)-779111
FAX: (65) 779-4455

MEMEC Asia Pacific Ltd.
Singapore Representative
Office
10 Anson Road #23-08
International Plaza
Singapore 079903
Tel: (65)-222-4962
FAX: (65)-222-4939

SLOVAK REPUBLIC

Elbatex Sk S.R.O.
Kasmirska 7
SK 84104 Bratislava
Tel: (42) 7-205007
FAX: (42) 7-5220600

SLOVENIA/CROATIA

Eljapex D.O.O.
Stenge 25
SLO-81000 Lalbach
Tel: (38) 661-159-7198
FAX: (38) 661-159-2398

SOUTH AFRICA

Avnet - ASD
Avnet Kopp (Pty) Ltd.
PO Box 3853
Rivonia 2128
South Africa
Tel: (27) 11 444 2333
FAX: (27) 11 444 7778

SOUTH AMERICA

DTS Ltda.
Rosas 1444
Santiago
Chile
Tel: (56) 2-6970991
FAX: (56) 2-6993316

Hitech
Rua Branco de Moraes 489
Ch. Santo Antonio
Sao Paulo
04718-010 - SP - Brazil
Tel: (55) 11-882-4000
FAX: (55) 11-882-4100

Reycom Electronica S.A.
Bdo. de Irigoyen
972 Piso 2do "B"
1304 Buenos Aires
Argentina
Tel: (54) 1-304-2018
FAX: (54) 1-304-2010

SOUTH KOREA

Hyunmyung Electronics Co.
Ltd. 3 Fl, Dukwha Bldg.,
444-17, Seokyo-Dong,
Mapo-Ku, Seoul
South Korea
Tel: (82) 2-3141-0147
FAX: (82) 2-3141-0149

MEMEC Asia Pacific Ltd.
Rm. 501, Dae Ha Bldg.
14-11 Yoido-Dong
Youngdeungpo-Ku
Seoul, 150-715
South Korea
Tel: (82) 2-786-8180
FAX: (82) 2-761-4121

Seodu Inchip
4 Fl, Myoungi Bldg.,
142-21, Samsung-Dong,
Kangnam-Ku, Seoul
South Korea
Tel: (82) 2-563-8008
FAX: (82) 2- 563-8411

SOUTHEAST ASIA

MEMEC Asia Pacific Ltd.
Units 3601-02 & 07-25, Tower I,
Metroplaza, Hing Fong Road,
Kwai Fong, N.T.
Hong Kong
Tel: (852) 2410-2780
FAX: (852) 2401-2518

SPAIN

ADM Electronics SA
Calle Tomas Breton,
No 50, 3-2
28045 Madrid
Spain
Tel: (34) 1-530-4121
FAX: (34) 1-5300164

ADM Electronics SA
Calle Mallorca 1
08014 Barcelona
Spain
Tel: (34) 3-426-6892
FAX: (34) 3-425-0544

ADM Electronica, SA
Herriko Gudarien, 8-1B
48200 Durango (Vizcaya)
Spain
Tel: (34) 4-620-1572
FAX: (34) 4- 620-2331

SWEDEN

DipCom Electronics AB
Torshamnsgatan 35
P.O. Box 1230
S-164 28 Kista
Sweden
Tel: (46) 8 752 24 80
FAX: (46) 8 751 3469

SWITZERLAND

Fenner Elektronik AG
Abteilung Bauteile
Gewerbestrasse 10
CH-4450 Sissach
Switzerland
Tel: (41) 61 975 00 00
FAX: (41) 61 971 5608

TAIWAN

MEMEC Asia Pacific Ltd.
Rm. 1005, 10F, No.2, Lane 150
Sec.5, Hsin Yin Rd.
Taipei, Taiwan R.O.C.
Tel: (886) 2-760-2028
FAX: (886) 2-765-1488

TURKEY

Eltronik Elektronik Sanayii Ve
Ticaret Ltd. Stl.
Farabi Sokak No. 44/4
06690 Cankaya
Ankara, Turkey
Tel: (90) 312-467-9952
Fax: (90) 312-467-9828

Eltronik Inc.
8427 Kennedy Boulevard
North Bergen
New Jersey 07047
USA
Tel: (201) 453-0410
FAX: (201) 453-0959

UNITED KINGDOM

Avnet EMG Ltd.
Jubilee House
Jubilee Road
Letchworth
Hertfordshire SG6 1QH
England
Tel: (44) 1- 462-488500
FAX: (44) 1- 462-488567

Cedar Technologies
Unit One Old Barns
Rycote Lane Farms
Milton Common
Oxfordshire
OX9 2NZ
England
Tel: (44) 1-844-278278
FAX: (44) 1-844-278378

Cedar Technologies
32 Enterprise House
Sprinkerse Business Park
Striling SK7 7UF
England
Tel: (44) 1-786-446221
FAX: (44) 1-786-446223

Microcall Ltd.
The Gate House
Alton House Business Park
Gatehouse Way
Aylesburg, Bucks HP 19 3DL
England
Tel: (44) 1-296-330061
FAX: (44) 1-296-330065

XILINX®

The Programmable Logic Company℠

2100 Logic Drive
San Jose, CA 95124-3400
Tel: 1-408-559-7778
Fax: 1-408-559-7114
e-mail: hotline@xilinx.com
web: www.xilinx.com

0401764

Printed in U.S.A.