XAPP708 (v1.0) February 14, 2006

133 MHz PCI-X to 128 MB DDR Small-Outline DIMM Memory Bridge

Author: Kraig Lund

Summary

This application note describes the implementation details of a bridge between a 133-MHz, 64-bit PCI-X interface and a 128 MB Double Data Rate (DDR), Small-Outline Dual Inline Memory Module (SODIMM) interface for Virtex™-4 devices. In particular, it ties together the Xilinx LogiCORE™ PCI-X core to the DDR interface described in XAPP709, *DDR SDRAM Controller Using Virtex-4 Devices*, to create a fully integrated, synthesizable, and hardware-verified reference design. The reference design is capable of reading and writing up to four KB bursts of 64-bit data at 133 MHz and has been fully tested on the ML455 PCI/PCI-X Development Kit [Ref 3].

Introduction

The Virtex-4 features, such as the programmable IDELAY and built-in FIFO support, simplify the bridging of a high-speed, PCI-X core to large amounts of DDR-SDRAM memory. One challenge is meeting the PCI-X target initial latency specification. PCI-X Protocol Addendum to the PCI Local Bus Specification Revision 2.0a ([Ref 6]) dictates that when a target signals a data transfer, "the target must do so within 16 clocks of the assertion of FRAME#." PCI termination transactions, such as Split Response/Complete, are commonly used to meet the latency specifications. This method adds complexity to the design, as well as additional system latency. Another solution is to increase the ratio of the memory frequency to the PCI-X bus frequency. However, this solution increases the required power and clock resource usage.

This application note avoids these methods and implements a simple solution for attaching a large piece of memory to a 64-bit PCI-X bus operating at up to 133 MHz. Features of the implementation include:

- Reads/writes for byte and DWORD at up to four KB bursts of data
- Simple target interface, resulting in no bus arbitration
- Conformance to the PCI-X target initial latency specification without the use of split transactions
- Optimization for low global clock resource usage
- Data mask for byte and burst writes

Implementation Details

Figure 1 shows an overall view of the reference design. The DDR interface portion of this reference design works with the Micron MT8VDDT1664HDG-265 DDR SODIMM memory located on the ML455 PCI/PCI-X development board.

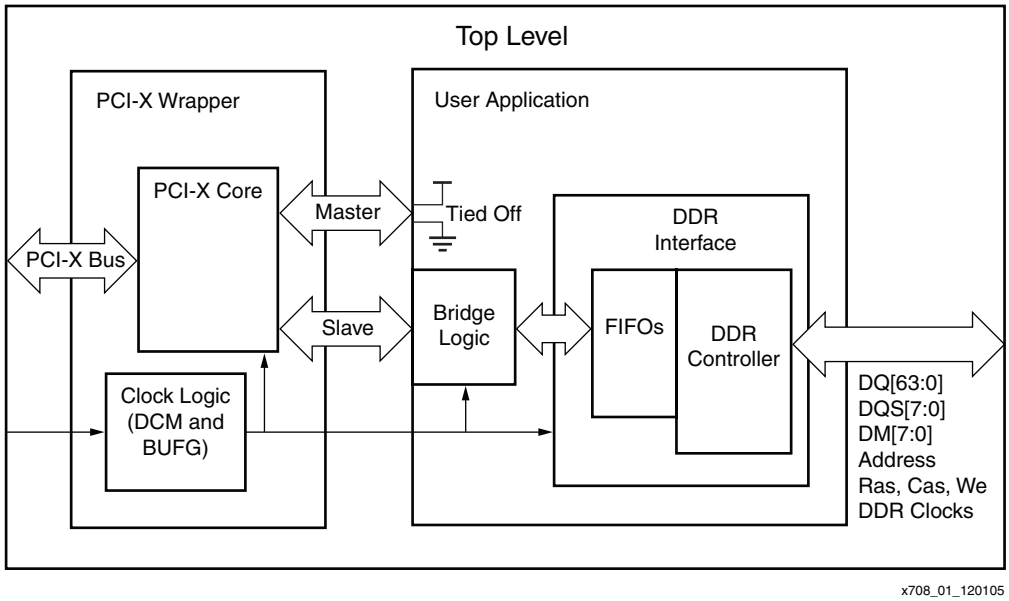

Figure 1: **Reference Design**

Specifications of this memory are:

- 200-pin, dual-rank SODIMM
- 133 MHz
- 266 MT/S
- 128 MB
- Programmable burst length of 2, 4, or 8
- CAS latency of 2.5
- 2.5V SSTL_2

For this particular design, a burst length of four was chosen as a compromise between performance and power usage. A burst length of eight is possible only by relying on Split Response/Complete and/or Retry transactions to meet the PCI-X target initial latency specification.

The reference design is designed exclusively to operate on a PCI-X bus, i.e., it is not PCI backwards compatible. Nominally, the design is set to run at 133 MHz, however a simple modification to the DDR refresh counter allows operation from 80 – 133 MHz (see the README file in the reference design for directions.)

PCI-X Core

This design has been tested in hardware with Rev 5.0.100 of the Xilinx LogiCORE PCI-X core. UG160, *LogiCORE PCI-X v5.0 User Guide*, provides detailed information about the core.

The following features of this core are used:

- 133 MHz
- 64 bits
- Target only interface; all master signals have been tied off
- S_WAIT: to generate wait states during reads
- S_RETRY: to generate retries only when memory requests occur while the memory is in Autorefresh mode

The reference design ties two 32-bit Base Address Registers (BARs), BAR2 and BAR3, together to generate a single BAR for 64-bit addressing on the PCI-X bus. The combined BAR is recognized as a 128 Mb memory space, as pictured in Figure 2. The PCI-X core asserts the S_HIT[2] signal when a PCI-X transaction occurs within the memory space decoded by the BAR.

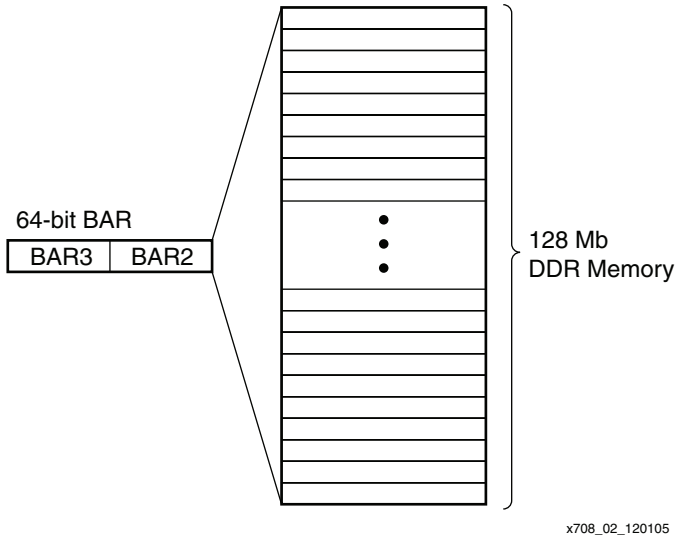

Figure 2: **PCI-X BAR Memory Map**

Command and Address Interfaces

Command Interface

The reference design responds to the subset of PCI-X target commands listed in Table 1. The table also lists the corresponding command value (C/BE#[3:0]) that is transferred over the PCI-X bus during the address phase of a valid PCI-X transaction.

Table 1: **Reference Design Commands**

PCI-X Command	C/BE#[3:0] (Hex Format)
DWORD Memory Read	0x6
DWORD Memory Write	0x7
Block Memory Read	0xE
Block Memory Write	0xF
Configuration Space Read	0xA
Configuration Space Write	0xB

Because the PCI-X and DDR clocks are identical, this reference design implements the Command FIFO as a register (Table 2) to reduce the latency to a minimum. The S_HIT[2] signal from the LogiCORE PCI-X core writes the information to the FIFO. If the DDR memory operates at a faster speed than the PCI-X bus, a Virtex-4 FIFO16 can replace the register.

Table 2: **24-Bit Command FIFO (Register)**

MSB																LSB							
23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Command				Byte Count												Byte Enable							

Bit	Description
[7:0]	Byte enable for a DWORD write (from PCI attribute phase)
[19:8]	Byte count for a burst transaction (from PCI attribute phase)
[23:20]	PCI-X command (from PCI attribute phase)

Address Interface

The Address FIFO is also implemented as a register, similar to the Command FIFO. It is written using the S_HIT[2] signal and read by the DDR controller state machine. The Address FIFO can be replaced by a FIFO16 element in designs with different clock frequencies.

Generally, DDR SDRAM devices read and write data bursts addressed on larger boundaries than byte boundaries. For example, a 64-bit DDR device programmed for burst of four is addressed in 32-byte boundaries. The ML455 board contains a 128 MB, 64-bit DDR DIMM set up for burst-of-four transfers. Table 3 shows a 27-bit PCI-X byte address divided into rank, bank, row, and column addresses for the DDR memory. Because multiple banks can be active simultaneously, the bank address has less significance than the row address, and bursting can cross banks rather than rows. Bursting across banks ensures that once a burst-read data transfer begins it will complete without the need for disconnect, which requires the use of the Master interface on the PCI-X core.

Table 3: **27-Bit Address FIFO (Register)**

MSB																							LSB
26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Rank	Row Address												Bank Address		Column Address							Data Mask			Unused	

Bit	Description
[1:0]	Unused bits
[4:2]	Bits for generating the data mask to the DIMM
[11:5]	Column address bits of the PCI-X byte address
[13:12]	Bank address bits of the PCI-X byte address
[25:14]	Row address bits of the PCI-X byte address
[26]	Rank bit of the PCI-X byte address

PCI and PCI-X addressing always occurs in byte boundaries, however, because bits 1 and 0 are not used in this design, PCI-X burst transfers must start on DWORD-aligned addresses. Although this address mapping violates the PCI-X protocol, which states that burst transfers can start on any byte address, it is generally not a problem because many PCI-X systems begin and end bursts on larger boundaries than bytes.

Write Interface

Datapath

Figure 3 shows the write interface datapath logic. The PCI-X interface presents the write data on a 64-bit bus, S_DATA_OUT[63:0], and steers the data to a 128-bit bus. The steering depends on PCI-X address bit 3. DWORD and burst writes beginning on odd, 16-byte address boundaries are steered to the correct 64-bit rise/fall slot. The output of the steering logic is presented to the user side of the Write FIFO. The Write FIFO uses four FIFO modules, each 32 bits wide. The output of the FIFO feeds the I/O DDR registers as rising and falling data.

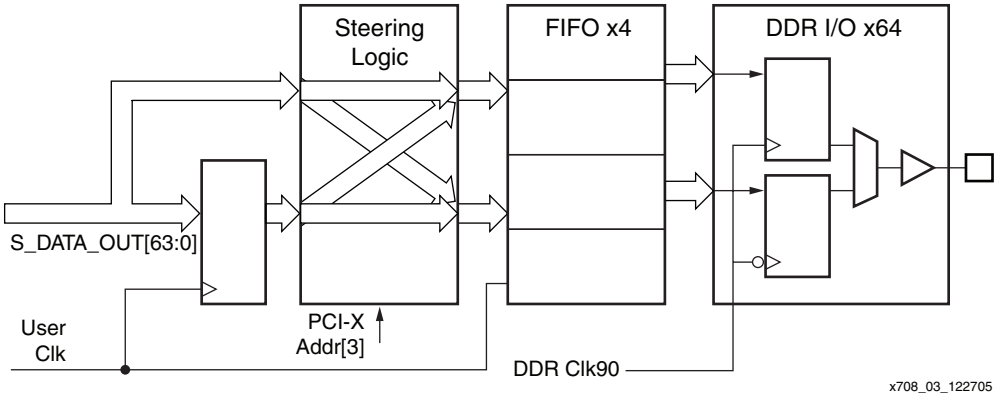

Figure 3: **Write Datapath Overview**

Data Mask Module

Data masking is used to allow individual byte writes to the DDR memory. Typically, one data mask (DM) bit corresponds to a single byte of data (DQ). The reference design contains a module for driving the data mask bits DM[7:0] on the DDR SDRAM SODIMM. It is configured for burst-of-four transactions and is capable of both individual byte writes and bursting on DWORD boundaries. Figure 4 shows an example of a single byte-write while Figure 5 shows a burst write. DM Low allows writes and DM High blocks writes.

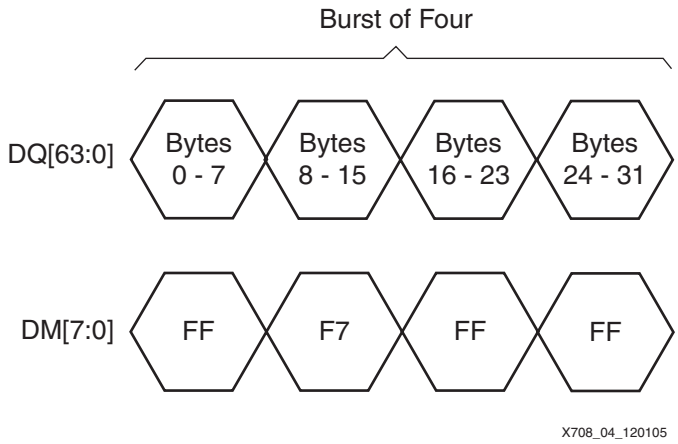

Figure 4: **Single Byte-Write to Byte #12**

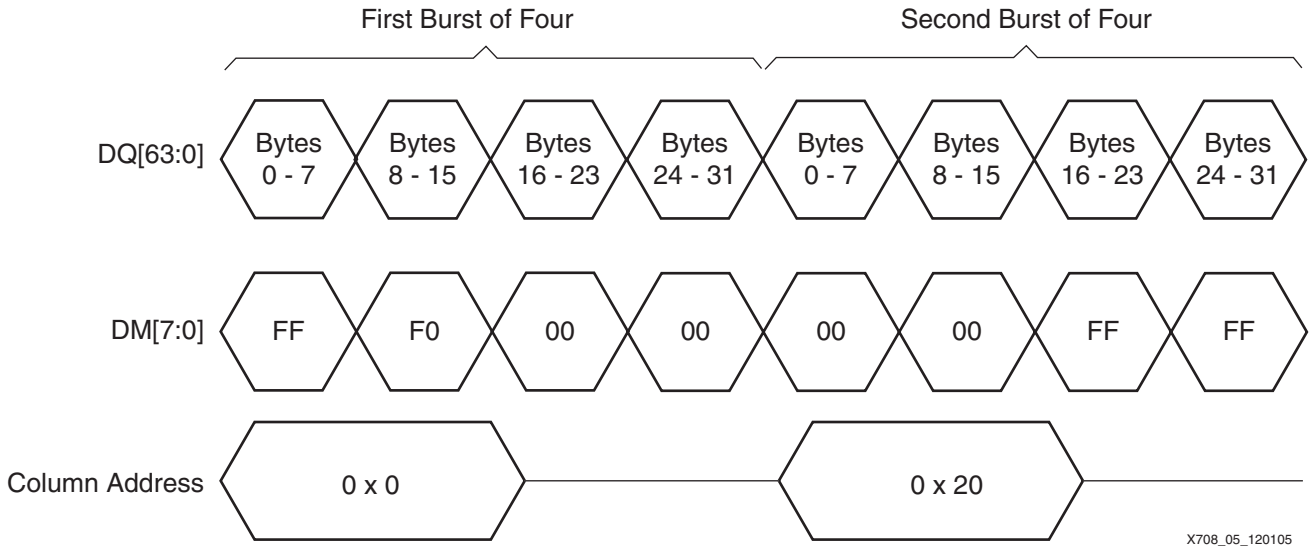

Figure 5: **36 Byte Burst Write From Address 0xC to 0X2C**

The data mask module (Figure 6) uses the start address and byte count provided by the PCI-X core, as well as the burst count and start signal from the DDR controller state machine, to output 16 bits of mask data (8 bits rising and 8 bits falling) to the DDR I/O registers.

A block RAM stores masking patterns for the first and last beats of any burst transaction. The masking patterns are addressed using an index based on the start address and byte count information. The outputs of the block RAM are multiplexed. This multiplexer is controlled by a state machine that uses the burst length data to determine when to steer the first, last, and middle (`0x0000`) data to the output. For single DWORD transactions, byte enables provided by the PCI-X core are ORed with the output.

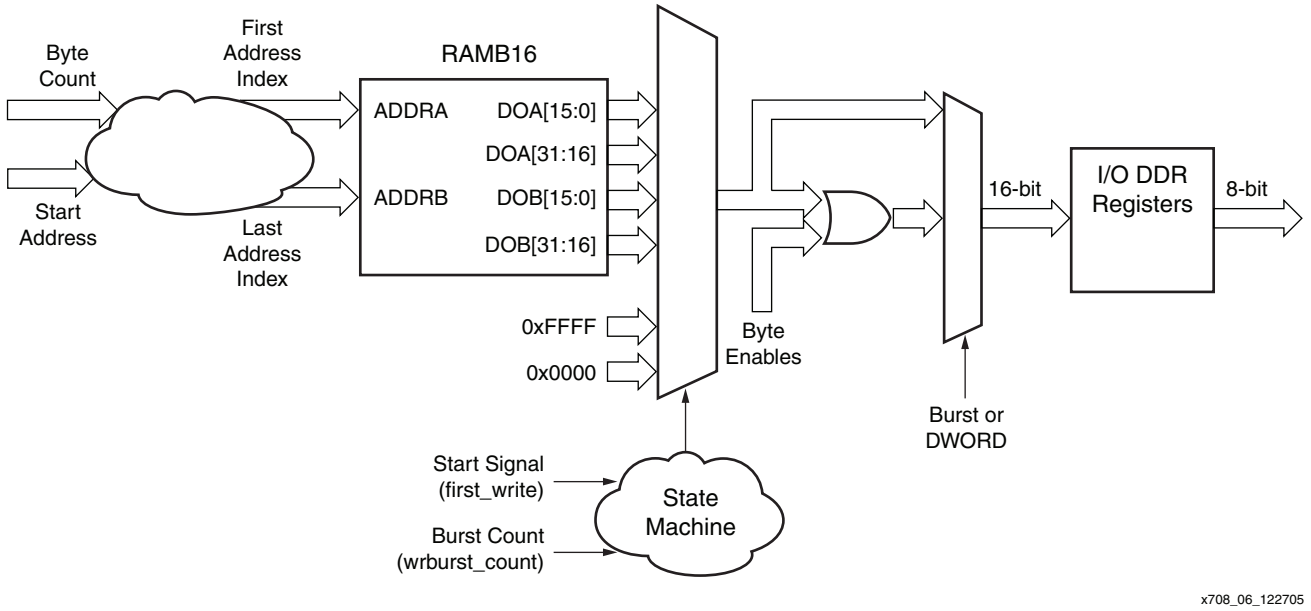

Figure 6: **Data Mask Module**

Read Interface

Datapath

Figure 7 shows the read datapath from the DDR memory to the PCI-X bus. Data is captured in the I/O DDR flip-flops using the direct clocking technique described in XAPP701, *Memory Interfaces Data Capture Using Direct Clocking Technique*. Input DDR I/Os are set to the same-edge mode for simplicity and the least amount of latency. Rising and falling data is pipelined and directed into a 64-bit commutator. The output of the commutator connects directly to the PCI-X core. The S_WAIT signal, which is driven by the DDR controller state machine, signals to the core when the read data is valid.

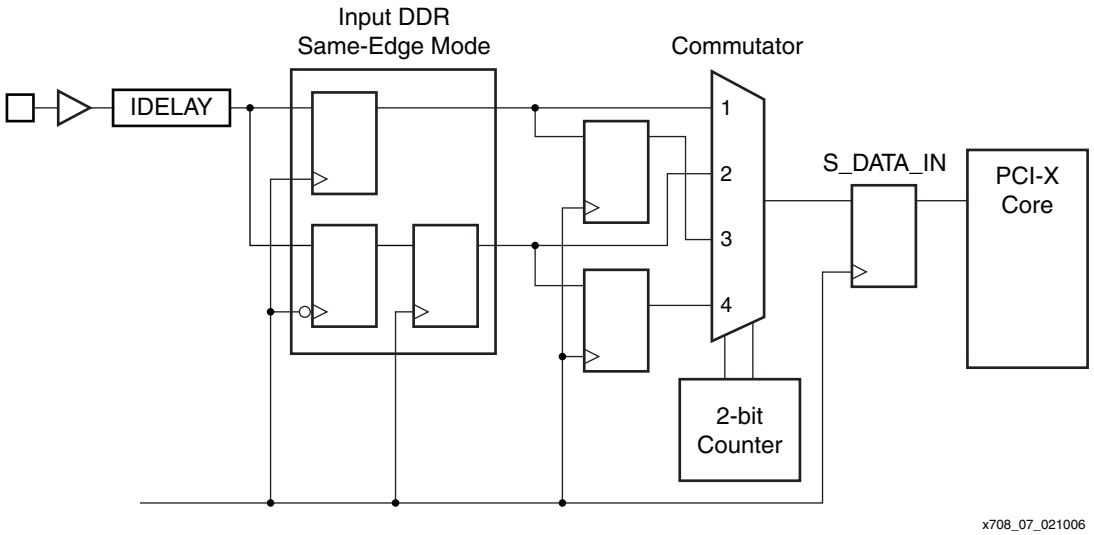

Figure 7: **Single-Bit Read Datapath**

Latency

The overall read data latency is variable (but deterministic) due to the bursting nature of the DDR memory. The memory can return data on any of the four data beats depending on the starting address of the PCI-X read request. A simple counter determines when the data is valid and drives the S_WAIT signal to the PCI-X core for wait states.

For read transactions, the PCI-X specification requires that data be present on the bus within 16 clocks of the FRAME# signal being asserted. Figure 8 shows the latency of the read datapath.

To achieve the best possible latency, the reference design employs an address decoding scheme that does not rely on the PCI-X core. The PCI_HIT_EARLY signal mimics the function of the S_HIT[2] output from the PCI-X core, except it uses combinatorial logic decoding to occur two clock cycles earlier than S_HIT[2].

Split transactions are not employed in this design. If this reference design used split transactions then pre-decoding of the address bus would not be required, nor would 4-word bursts be required for the DDR memory. In addition, the design is more parameterizable because the DDR and PCI-X clock domains can be kept separate from each other and allowed to operate at arbitrary frequency ratios. However, the disadvantages of implementing split transactions include adding the logic and complexity of using the Master interface on the PCI-X core.

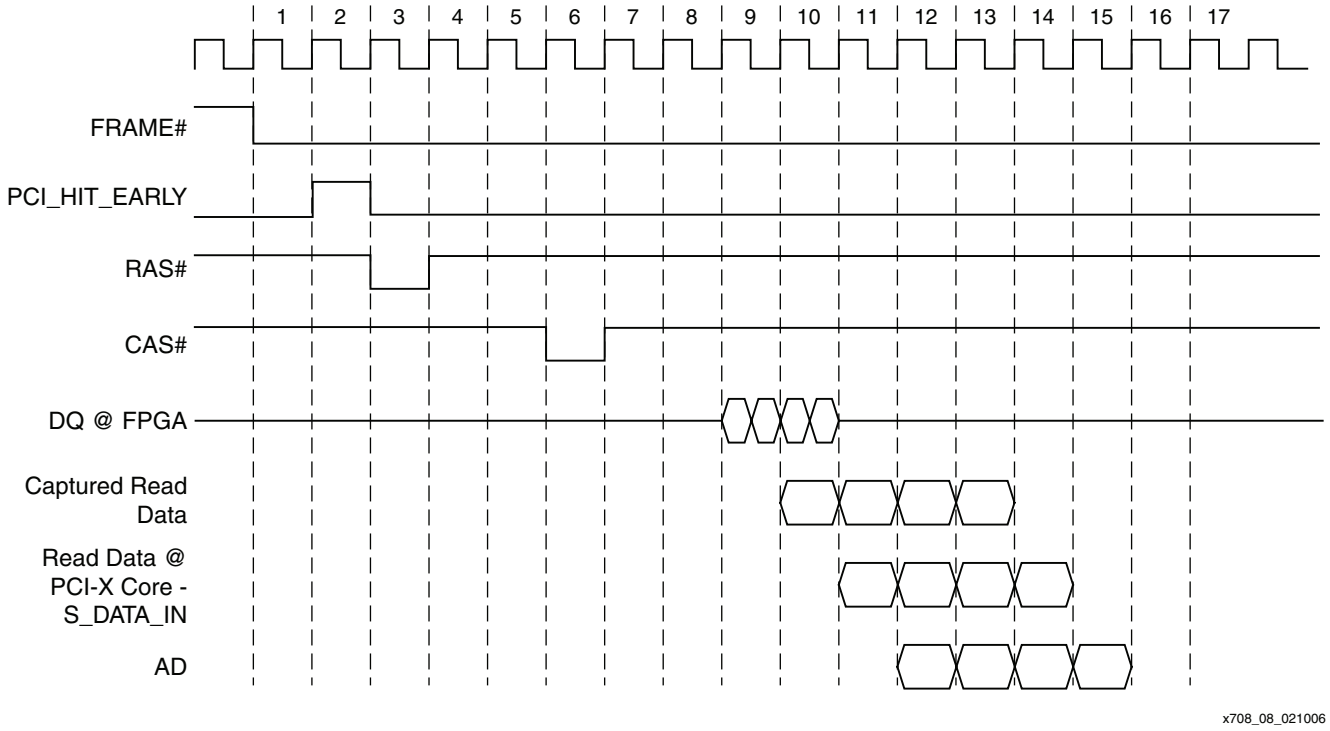

Figure 8: **Read Datapath Latency**

DDR Controller

The controller state machine issues the commands in the correct order while considering the timing requirements of the memory. Figure 9 outlines the various stages in the controller state machine.

Before the controller state machine issues commands to the memory:

- It issues a Read Enable signal to the Read/Write Address FIFO.
- It activates the necessary row(s)/bank(s) needed to support up to a 4K burst of data. At most, two rows or banks are activated. If a read burst command spans two rows/banks, the first row/bank is opened immediately. The second row/bank is opened after the first read command is issued to ensure conformance to the PCI-X latency specification, as discussed in the "Latency" section.
- It pipelines commands to synchronize them with the address signals.

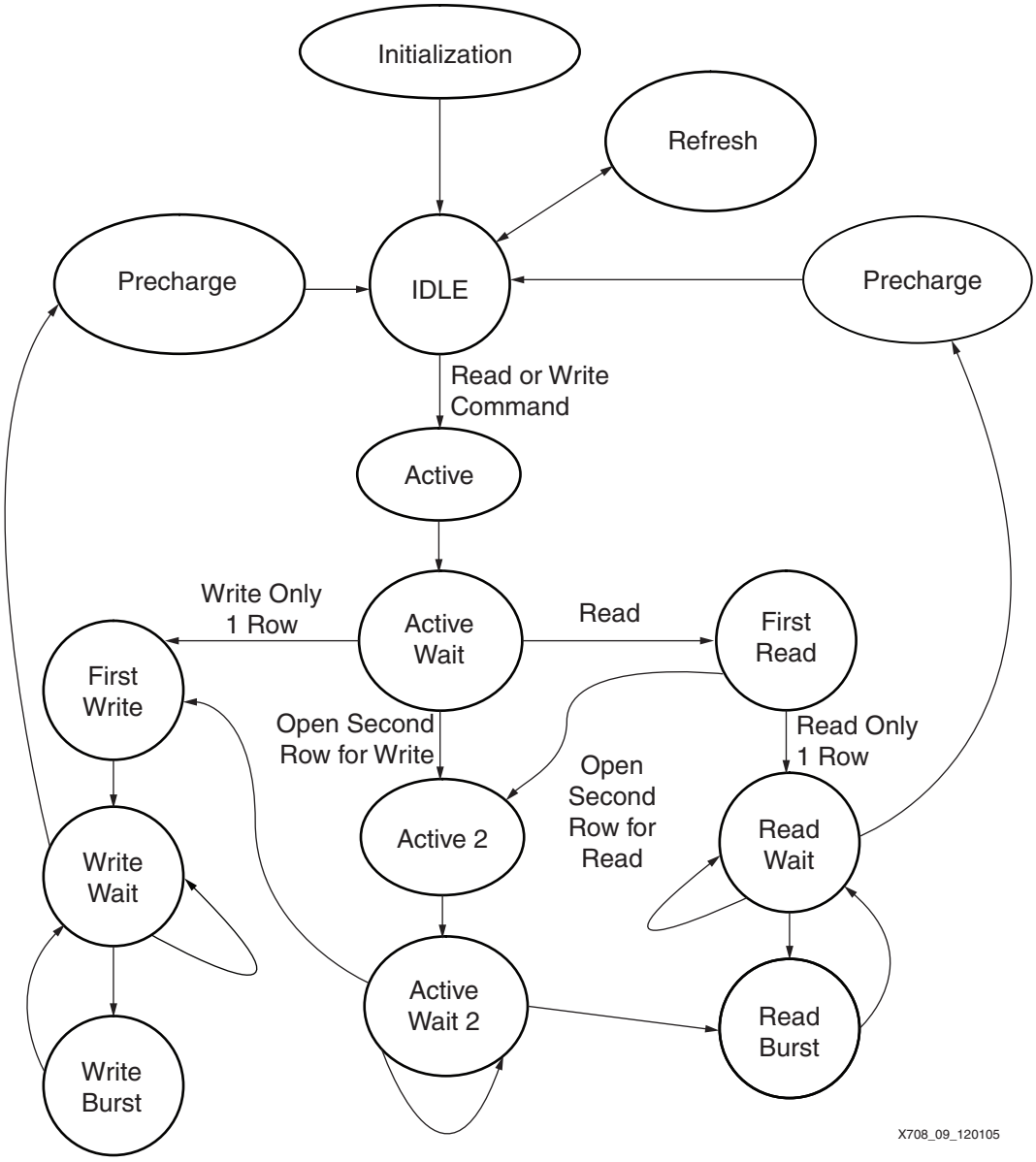

Figure 9: **DDR Controller State Machine**

Clocking

The reference design generates all necessary clocks from the PCI-X PCLK using one DCM. CLK0 is used for the PCI-X core, user, and DDR logic. CLK90 is used to center-align the DDR write data. CLKDV, which can be replaced by a simple counter, is used to generate an autorefresh timer for the DDR controller state machine. The 133 MHz clock is divided by 16 and then further divided by 96 to create an 11.52 µs timer. A 11.52 µs timer meets the 15.6 µs Autorefresh requirement even if a 4 KB burst occurs right before timer expiration. If the design operates at a frequency slower than 133 MHz, change the timer divisor to ensure the 15.6 µs requirement is met (change in ddr_controller.v/vhd). Figure 10 shows the clock scheme.

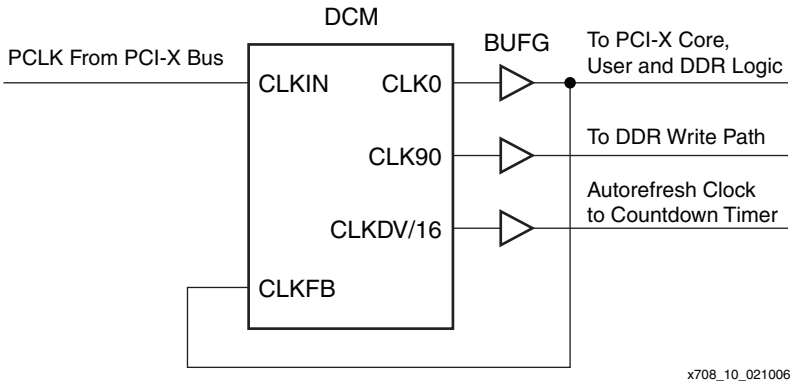

Figure 10: **Clock Scheme**

Hardware Platform

The reference design was tested on the ML455 PCI/PCI-X development board. Features of this board include:

- XC4VLX25-FF668 -11 speed grade FPGA
- 200-pin, 2.5V SODIMM socket with 128 MB (16M x 64 bit) DDR SDRAM SODIMM
- Three on-board clock sources:
  - 200 MHz Epson 2.5V EG-2121CA LVPECL
  - 133 MHz Epson 2.5V EG-2121CA LVDS
  - 33 MHz LVCMOS
- One DB9-M RS232 port (serial cable not provided)
- Support for up to four FPGA design images in a Xilinx XCF32P-FSG48C Platform Flash configuration PROM
- Static or dynamic device reconfiguration support with the XC2C32 CoolRunner™ II CPLD
- 64-bit, 3.3V system board keyed PCI or PCI-X connector
- Top-mounted, 64-bit, 3.3V keyed PCI or PCI-X expansion socket (on top of board)
- User push-button switches, LEDs, and general-purpose I/O header
- Device configuration through on-board Platform Flash, Xilinx PC-IV JTAG cable, or Xilinx Platform Cable USB
- PCI clocking support for global and regional clocking applications
- On-board power regulators (3.0V PCI, 2.5V, 1.8V, 1.2V, 1.25V V_{TT})

More details about the board are described in UG084, *Virtex-4 ML455 PCI/PCI-X Development Kit User Guide*.

Table 4 lists the jumper settings needed to run the reference design.

Table 4: **ML455 Jumper Settings**

Signal	Jumper Setting
PCIXCAP	Open
M66EN	Open

Jungo Driver

The reference design has been tested using a driver built with the WinDriver PCI/ISA/CardBus v7.00 kit by Jungo Software Technologies. This kit includes a wizard that allows a user to quickly build a driver and application test code to test the functionality of a PCI device. More details about this kit are available on the Jungo website at www.jungo.com.

Reference Design Specifications and Files

The reference design was hardware verified at 133 MHz. Table 5 provides the design specifications, and Table 6 lists the Verilog module files (the VHDL files are equivalent with a .vhd file extension) and the User Constraint File (UCF) for the design. The files are located at http://www.xilinx.com/bvdocs/appnotes/xapp708.zip.

Table 5: **Reference Design Specifications**

Description	Parameter	Specifications
Frequency of operation	DDR266 – PC2100	133 MHz
Virtex-4 device speed grade		11
Device utilization	Slices	3157 (from Verilog design)
	GCLK	4
	DCM	1
	RAMB16	5
	IOBs	203
Supported burst modes for DIMM		4
HDL language		Verilog, VHDL
Bus width	PCI-X	64 bits
	DIMM	144 bits
Device for DIMM verification		MT8VDDT3264HD-128MB

Table 6: **Design File List**

Design File	File Description
pcix_top.v	Specifies the top-level module for the PCI-X core and DDR bridge
cfg_test_x.v	Specifies the PCI-X core configuration
pcix_lc_64x.v	Specifies the PCI-X core wrapper, which includes the DCM used in this design to generate clocks for the PCI-X core and DDR memory
pcix_core.v	Specifies the PCI-X core simulation model
userapp.v	Wrapper file connects the PCI-X core to the DDR Memory Controller
ddr_top.v	Specifies the top-level module for the DDR controller and physical layer
clk_module.v	Instantiates DDR-specific clocking resources for the memory interface and the 200 MHz clock for the IDELAYCTRL module
data_path.v	Specifies the top level for the physical layer and instantiates the following modules: tap_ctrl, data_tap_inc, idelay_ctrl, idelay_rd_en, v4_dqs_iob, v4_dq_iob, and rd_data_fifo
data_tap_inc.v	Implements the tap selection controller for the data bits associated with a strobe

Table 6: **Design File List *(Continued)***

Design File	File Description
idelay_ctrl.v	Instantiates the IDELAYCTRL primitive (required when the IDELAY primitive is used in the design)
tap_ctrl.v	Detects two transitions of a DQS signal and determines the tap delay required for the associated data bits to center them with respect to the internal FPGA clock, CLK
v4_dm_iob.v	Instantiates the IOB flip-flops for the data mask bits
v4_dq_iob.v	Instantiates the IDELAY primitive and IOB flip-flops for the bidirectional data
v4_dqs_iob.v	Instantiates the IDELAY primitive and IOB flip-flops for the bidirectional strobe
ddr_controller.v	• Provides the Read Enable signals to the Write Address, Write Data, and Read Address FIFOs • Includes the controller state machine. Supplies the command signals to the DDR device. Causes the controller to generate Auto Refresh commands based on the Auto Refresh Command Interval. • Provides the address signals to the DDR device
dm_gen.v	Generates data mask signals for the DDR device
user_interface.v	Contains bridging logic to connect the PCI-X core slave interface to the DDR Controller FIFOs
backend_fifo.v	Instantiates the backend FIFOs for the DDR SDRAM interface, including the Write Address and Data FIFOs and the Read Address and Data FIFOs. The FIFOs are implemented using Virtex-4 FIFO16 primitives or LUT RAM instances.
rd_data_fifo.v	Instantiates the FIFO16 primitive for read data: one FIFO for rising-edge data and the other for falling-edge data
parameter.v	Specifies the values for the DDR SDRAM reference design at 200 MHz
test_tb.v	Specifies the test bench
stimulus.v	Specifies the PCI-X bus driver test bench
ddr.v	Specifies the Micron memory model
ddr_parameters.v	Specifies the DDR memory model timing parameters
ddr_dimm.v	Specifies the DIMM model
4vlx25ff668_64xf.ucf	Specifies the ASCII UCF file for the ML455 board

References

1. UG160, *LogiCORE PCI-X v5.0 User Guide*
2. XAPP709, *DDR SDRAM Controller Using Virtex-4 Devices*
3. UG084, *Virtex-4 ML455 PCI/PCI-X Development Kit*
4. Shanley, Tom, MindShare, Inc. 2000. *PCI-X System Architecture*. USA: Addison-Wesley.
5. XAPP701, *Memory Interfaces Data Capture Using Direct Clocking Technique*
6. PCI-SIG, 2003, *PCI-X Protocol Addendum to the PCI Local Bus Specification v2.0a*

Conclusion

This application note and associated reference design describes a bridge between a 133-MHz, 64-bit PCI-X interface and a 128 MB DDR DIMM interface. The reference design uses the DDR interface described in XAPP709 with the PCI-X LogiCORE, and has been fully tested on the ML455 PCI/PCI-X Development Kit.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
02/14/06	1.0	Initial Xilinx release.