CompactFlash Controller

Reference Design

FPGA-RD-02088-1.4

November 2019

Disclaimers

Lattice makes no warranty, representation, or guarantee regarding the accuracy of information contained in this document or the suitability of its products for any particular purpose. All information herein is provided AS IS and with all faults, and all risk associated with such information is entirely with Buyer. Buyer shall not rely on any data and performance specifications or parameters provided herein. Products sold by Lattice have been subject to limited testing and it is the Buyer's responsibility to independently determine the suitability of any products and to test and verify the same. No Lattice products should be used in conjunction with mission- or safety-critical or any other application in which the failure of Lattice's product could create a situation where personal injury, death, severe property or environmental damage may occur. The information provided in this document is proprietary to Lattice Semiconductor, and Lattice reserves the right to make any changes to the information in this document or to any products at any time without notice.

Contents

Figures

Tables

1. Introduction

CompactFlash is a removable mass storage device that electrically complies with the Personal Computer Memory Card International Association ATA standard. It is used in a wide variety of applications ranging from data storage cards, magnetic disk drives, fax modem cards, and wireless pager cards. CompactFlash cards support both 5V and 3.3V operation and can function at either voltage level. CompactFlash uses standard ATA registers and command set, and has automatic error correction and retry capabilities, including an error correction code.

CompactFlash supports three different modes of operation: PC Card ATA using Memory Mode, PC Card ATA using I/O Mode, and True IDE Mode. This document focuses on PC Card ATA Memory Mode.

2. Design Goals and Limitations

The following goals were considered during the development of this reference design:

- Implement PC Card ATA using Memory Mode
- Support common memory access
- 8-bit wide CompactFlash interface data path
- Sector data transfer with 512 bytes
- WISHBONE host interface
- Two FIFO(s) between the CompactFlash and the host interface
- Read/write cycle access time optimized according to the CompactFlash timing specification
- Support one sector transfer at a time

The reference design does not support the following features:

- Attribute memory access, I/O access and IDE mode access
- 16-bit wide CompactFlash interface data path
- CF-ATA command other than write and read command
- Multiple sector transfer at a time

3. Theory of CompactFlash Operation

3.1. Modes of Operation

CompactFlash supports three different modes of operation. To enter Memory mode, it is mandatory that the /OE (pin 9 of CompactFlash) be high during power-up.

3.2. Data Storage

It is important to understand that the ATA/IDE standard does not permit a single byte to be read or written to or from the CompactFlash at a time. The firmware must read or write data one or more sectors at a time, where one sector equals 512 bytes of data.

Sectors are identified according to two addressing schemes: Cylinder/Head/Sector (C/H/S) and Logical Block Address (LBA).

In LBA mode, the Logical Block Address is interpreted as follows:

- LBA7-LBA0: Sector Number Register
- LBA15-LBA8: Cylinder Low Register
- LBA23-LBA16: Cylinder High Register
- LBA27-LBA24: Drive/Head Register

3.3. ATA Registers

Communication to and from the CompactFlash is accomplished through a set of registers called ATA Registers. It is essential to write the command parameters to these registers before a read/write command because the controller in the CompactFlash executes the command according to the parameters present in these registers. In Memory mode, each register is selected by an access code applied on signals CS0, CS1, A10-A0, and REG. A read or write process is specified with the OE and WE signal respectively.

The ATA registers are grouped into two categories: Control Block Registers and Command Block Registers. The Command Block Registers are used for sending commands to the CompactFlash and returning the status of the CompactFlash. The status conditions include whether the CompactFlash is busy or ready, and whether errors have been detected. The Control Block Registers are used for controlling and providing an alterable means of returning the status of the CompactFlash. The following registers are used in this reference design: Data Register, Sector Count Register, Sector Number Register, Cylinder Low Register, Cylinder High Register, Drive/Head Register, Status Register and Command Register.

To understand the basic functionality of the CompactFlash, the Compact Flash Specification 2.0 document published by the CompactFlash Association (CFA) is recommended. Detailed functionality or electrical characteristics of the CompactFlash specification will not be discussed in this document.

4. Functional Description

This CompactFlash Controller reference design is used to interface a CompactFlash with a wishbone bus compatible host. It is located between the CompactFlash and the wishbone bus master and is intended to reduce the amount of effort necessary for the user to deal with the ATA/IDE command. The CompactFlash controller responds to read/write cycles started by the wishbone bus host. The controller functions as a data path controller, transferring data to and from the wishbone bus host onto the CompactFlash. Figure 4.1. shows the relationship of the controller between the host and the CompactFlash.

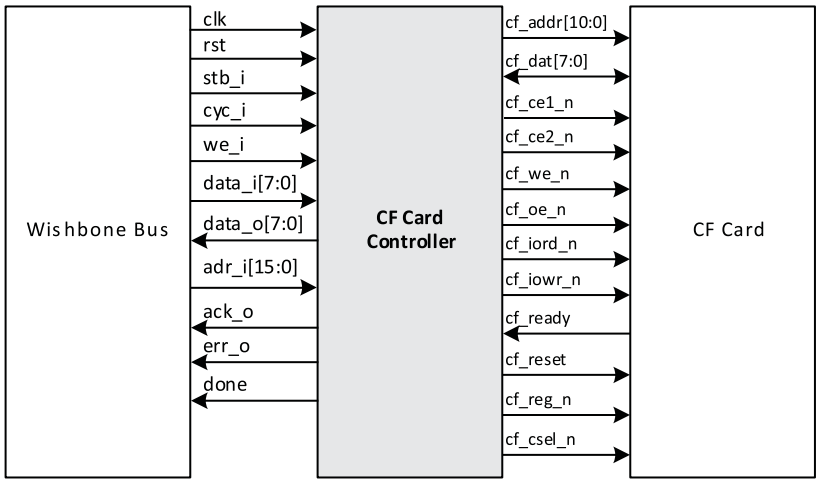

Figure 4.1. CompactFlash Controller System

Table 4.1. CompactFlash Controller Pin Descriptions

Signal Name	Type	Active State	Definition
CompactFlash Interface			
cf_addr	Output	N/A	These address lines, along with the -REG signal, are used to select the ATA regis- ter in the CompactFlash.
cf_dat	In/Out	N/A	These lines carry the Data, Commands and Status information between the Com- pactFlash and the controller.
cf_ce1_n	Output	Low	This signal and the -CE2 signal are used to select the card and to indicate to the card whether a byte or a word operation is being performed. -CE1 accesses the low byte of the word.
cf_ce2_n	Output	Low	This signal and the -CE1 signal are used to select the card and to indicate to the card whether a byte or a word operation is being performed. -CE1 accesses the high byte of the word. In this design, this signal is always assigned to 1.
cf_oe_n	Output	Low	This is an output enable strobe signal. It is used to read data from the Compact- Flash.
cf_iord_n	Output	Low	This signal is not used in Memory mode. In this design, this signal is always assigned to 1.
cf_iowr_n	Output	Low	This signal is not used in Memory mode. In this design, this signal is always assigned to 1.
cf_we_n	Output	Low	This signal is driven by the controller and used for a strobe write data to the Com- pactFlash.
cf_ready	Input	1 = Ready 0 = Busy	In Memory Mode, this signal is set high when the CompactFlash is ready to accept a new data transfer operation and is held low when the card is busy.
cf_cse1_n	Output	Low	This signal is not used in Memory mode. In this design, this signal is always assigned to 0.
cf_reset	Output	High	When this signal is high, it resets the CompactFlash.
cf_reg_n	Output	Low	This signal, along with the address lines, are used to select the ATA register in the CompactFlash.
WISHBONE Interface			
adr_i	Input	N/A	These address lines indicate the Logic Block Address in the CompactFlash that the host initiates to read or write. The low bytes are assigned to LBA7-LBA0 and the high bytes are assigned to LBA15-LBA8. LBA27-LBA16 are all assigned to 0 in this design.
dat_i	Input	N/A	These lines are used to pass data to the CompactFlash from host.
dat_o	Output	N/A	These lines are used to pass data to host from the CompactFlash.
we_i	Input	1 = Write 0 = Read	This signal indicates whether the current local bus cycle is a READ or WRITE cycle. The signal is negated during READ cycles, and is asserted during WRITE cycles.
ack_o	Output	High	The acknowledge output, when asserted indicates the termination of a normal bus cycle.
err_o	Output	High	When this signal is asserted, it indicates that the bit 0 (error bit) of the status regis- ter in CompactFlash is set.
cyc_i	Input	High	When asserted, indicates that a valid bus cycle is in progress.
stb_i	Input	High	When asserted, indicates that the CompactFlash controller is selected.
clk	Input	N/A	The system clock. It coordinates all activities for the internal logic within the con- troller.
rst	Input	High	This signal forces all internal logic within the controller and CompactFlash to reset.
done	Output	High	When asserted, indicates that the CompactFlash completed the write or read pro- cess.

5. Register Transfer Level (RTL) Implementation

The RTL block diagram of the CompactFlash controller is shown in Figure 5.1. It consists of four modules: the toplevel module, the host command state machine module, the CompactFlash interface state machine module and the data buffer module.

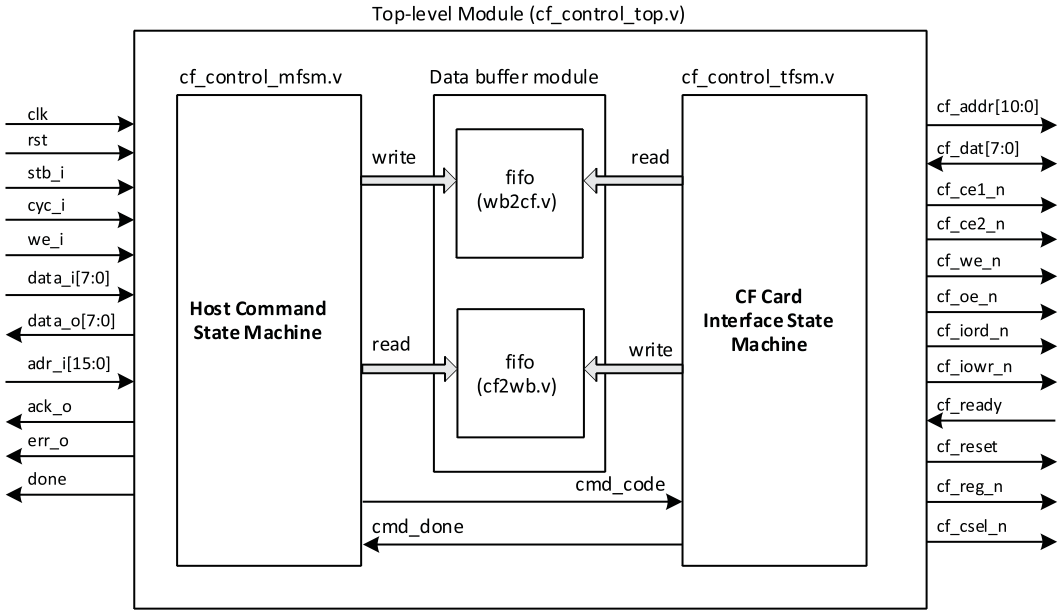

Figure 5.1. Block Diagram

5.1. Top-level Module

This is the top-level HDL block of this reference design. This module is created to instantiate the other lower level modules of the design.

5.2. Host Command State Machine

The host command state machine interprets write/read commands from the host, passing control to the Compact-Flash interface state machine to execute repeated regular tasks with strict timing requirements. Because the ATA/IDE standard does not permit the reading or writing of one byte to or from CompactFlash at a time, the host must read or write data one or more sectors (512 bytes) at a time. Every time the host initiates to write or read the CompactFlash, one sector will be transferred to or from the CompactFlash in this design. The adr_i address lines indicate the sector address (LBA) in the CompactFlash and during the whole transfer, the value of adr_i will be latched until the end of the transferring.

Memory Write Cycle

The cyc_i and stb_i signals will be sampled at the rising edge of every clock cycle. If they are all active and the we_i signal is high, this indicates that the host will initiate write data to the CompactFlash. The host first writes data to FIFO based on ack_o signal until 512 bytes are written to the FIFO. The controller sends a relevant command to the CompactFlash and will check whether the CompactFlash is ready to receive data. If it is ready, the controller enables the CompactFlash to read data from the same FIFO. Otherwise, if the CompactFlash is busy, the controller will wait until the CompactFlash is ready. If an error occurs, the controller sends an err_o signal to the host.

Memory Read Cycle

The cyc_i and stb_i signals will be sampled at the rising edge of every clock cycle. If they are all active and the we_i signal is low, this indicates that the host will initiates read data from the CompactFlash. The controller sends a relevant command to the CompactFlash and will check whether the CompactFlash is ready to send data. If it is ready, the controller enables the CompactFlash to write data to the FIFO. Otherwise, if the CompactFlash is busy, the controller will wait until the CompactFlash is ready. If an error occurs, the controller sends an err_o signal to the host. The host then reads data from the same FIFO based on the ack_o signal until 512 bytes are read from the FIFO.

5.3. CompactFlash Interface State Machine

Based on the sector transfer protocol, the CompactFlash interface state machine writes to or reads from the CompactFlash, and creates all the necessary control signals for the CompactFlash. The sector transfer protocol consists of five steps. Figure 5.2. shows these steps.

For each step, the controller creates the necessary control signals to access the CompactFlash and these control signals must fulfill the requirement of the CompactFlash access timing requirement. Figure 5.3. shows the memory read timing diagram and Figure 5.4. shows the memory write timing diagram.

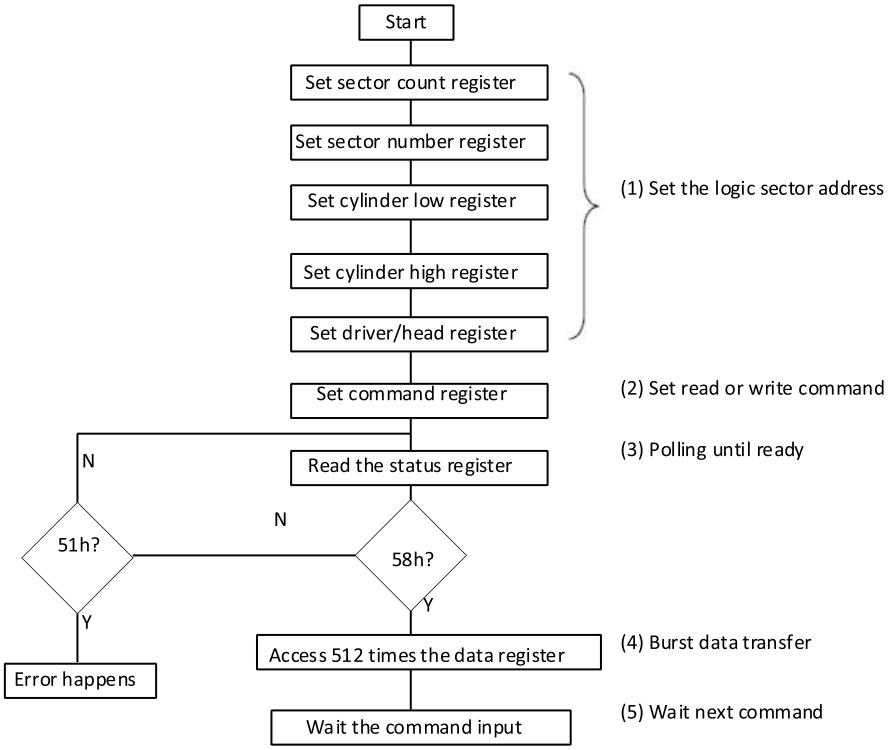

Figure 5.2. Sector Transfer Protocol State Machine

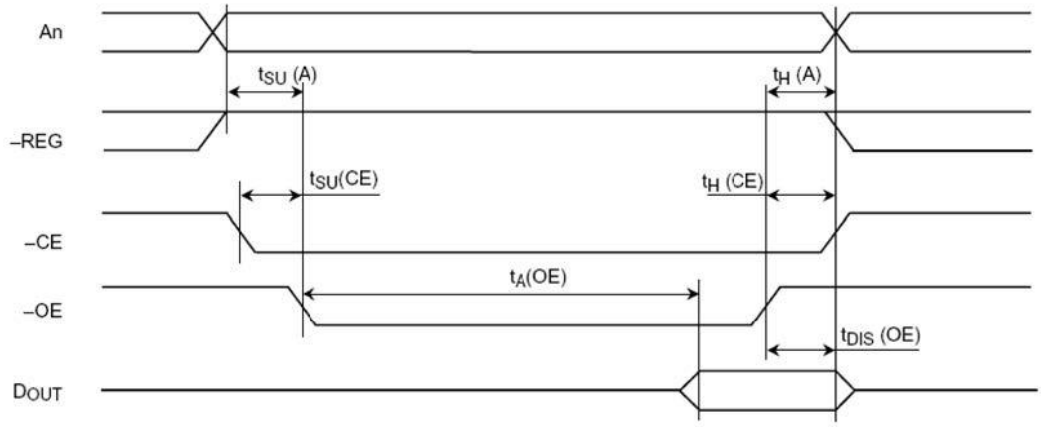

Figure 5.3. Memory Read Timing

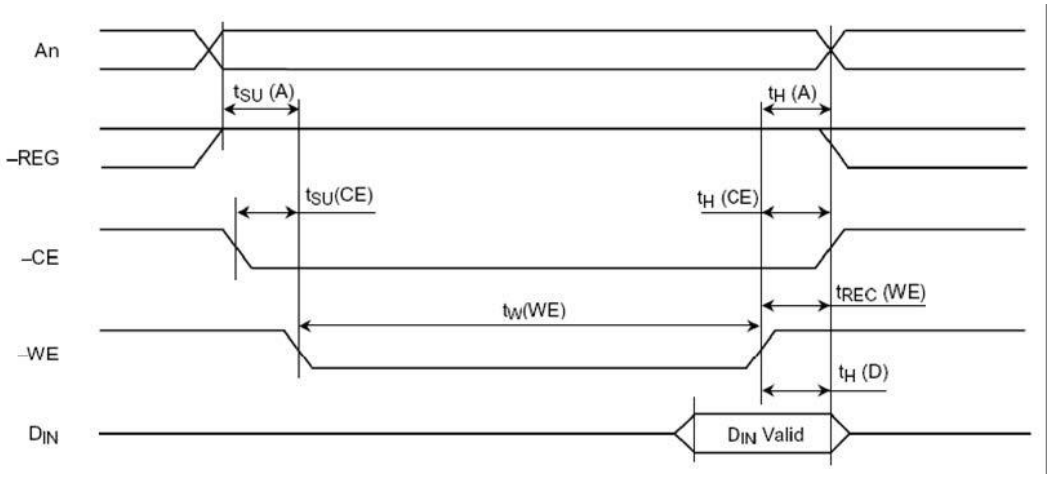

Figure 5.4. Memory Write Timing

The access timing requirement is probably different for different CompactFlash devices and these timing parameters are defined in the file par.v. The designer can modify the value of these parameters if necessary.

5.4. Data Buffer Module

The data buffer module is implemented by two FIFO(s) which are generated by IPexpress™ in the ispLEVER® design tool. When the host initiates a write command, the CompactFlash controller first stores the data in the FIFO. If the CompactFlash is ready, then the controller enables the CompactFlash to read data from the same FIFO. On the other hand, when the host initiates a read command, the controller will check whether the CompactFlash is ready. If it is ready, then the controller enables the CompactFlash to write data to the FIFO, and informs the host to read data from the same FIFO.

To write the CompactFlash sequence:

1. The adr_i address lines are loaded with the LBA address.
2. Write 512 bytes sequentially to the CompactFlash with the WISHBONE Classic Cycle.
3. Wait for at least 150 µs.

To read the CompactFlash sequence:

1. The adr_i address lines are loaded with the LBA address.
2. Read 512 bytes sequentially from the CompactFlash with the WISHBOE Classic Cycle.
3. Wait for at least 150 µs.

6. HDL Verification

6.1. Functional Blocks of the HDL Test Suite

The HDL code for the CompactFlash controller reference design test suite contains the following blocks:

- Top level test bench
- Host write and read task
- CompactFlash interface
- CompactFlash controller under test

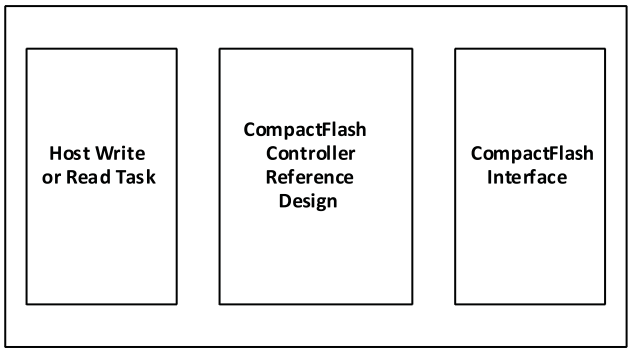

Figure 6.1. Test Bench Block Diagram

6.2. Test Bench Top Level

The CompactFlash controller test bench is used to verify the timing and functionality of this reference design. It simulates the host read or write transfers and the interface with the CompactFlash device.

6.3. Host Write and Read Task

These two stimulus generators create the read or write cycles as a host initiator on the wishbone bus and check the data from the CompactFlash.

6.3.1. CompactFlash Interface

CompactFlash interface module responds to the host initiate write or read commands.

7. Timing Diagrams

The following timing diagrams show the major timing milestones in the simulation.

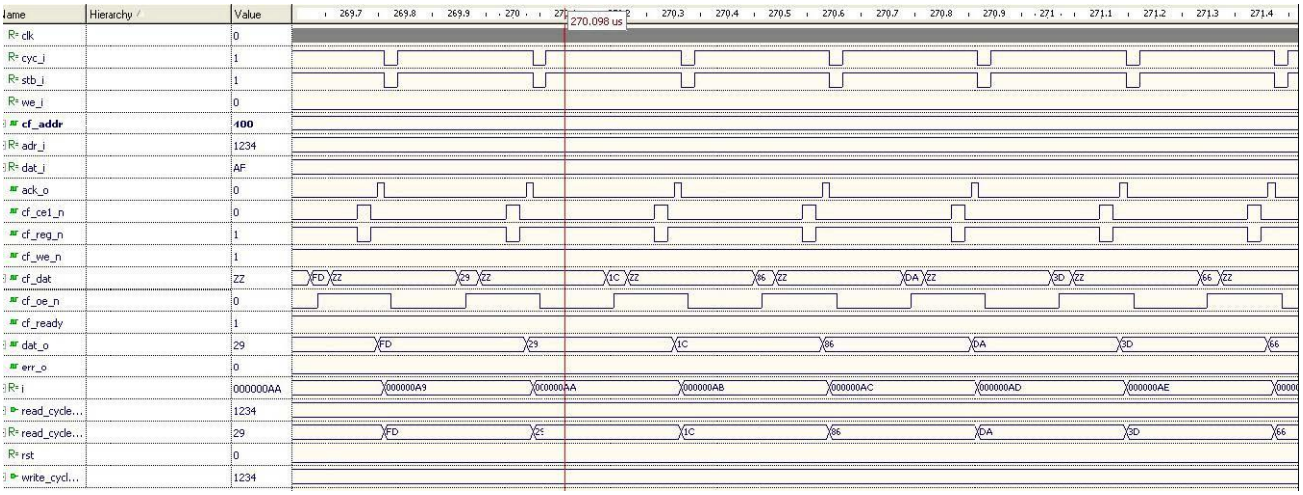

Figure 7.1. Verilog Read Timing

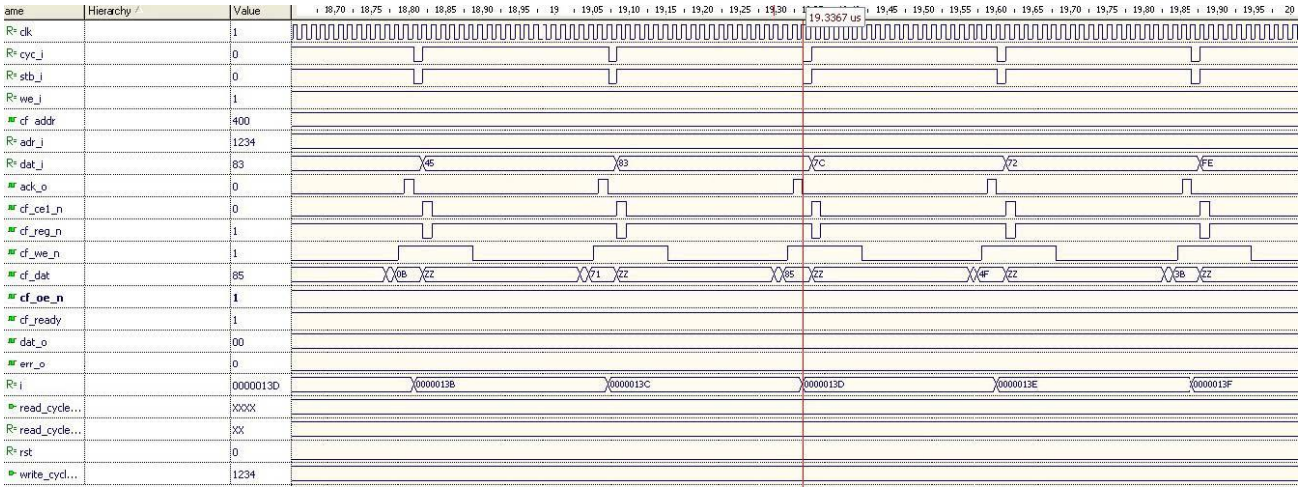

Figure 7.2. Verilog Write Timing

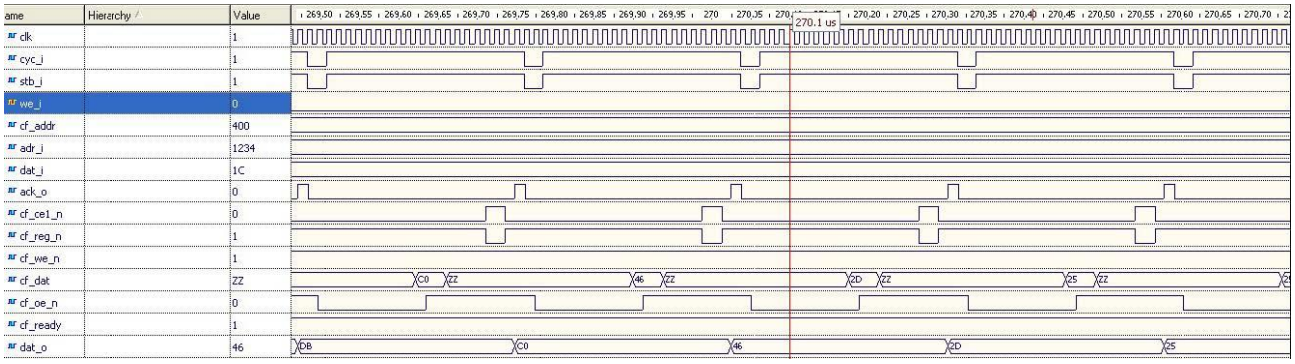

Figure 7.3. VHDL Read Timing

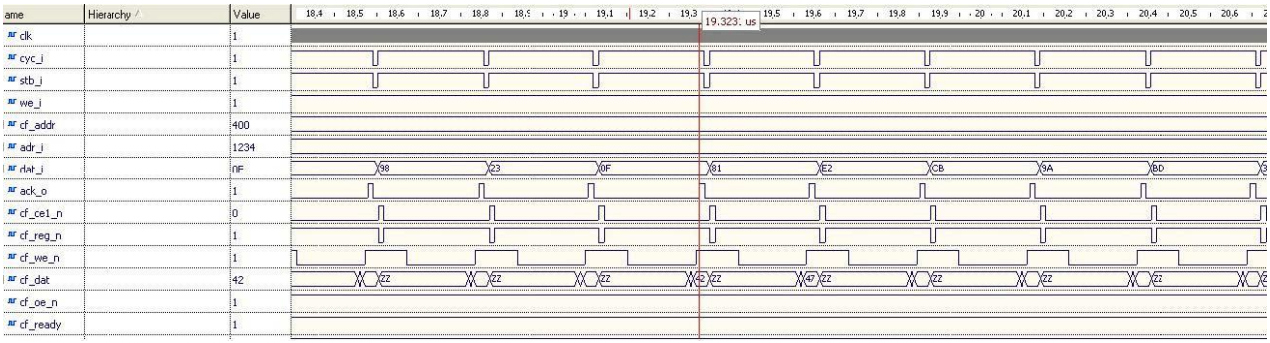

Figure 7.4. VHDL Write Timing

8. Implementation

This design is available in Verilog HDL and VHDL languages, and the Lattice ispLEVER design tool is used for implementation.

Table 8.1. Performance and Resource Utilization

Device Family	Language	Speed Grade	Utilization (LUTs)	f_{MAX} (MHz)	I/Os	Architecture Resources
MachXO2™ [1]	Verilog	-6	173	>100	68	2 EBRs
	VHDL	-6	174	>100	68	2 EBRs
MachXO™ [2]	Verilog	-5	629	>100	68	1 EBR
	VHDL	-5	626	>100	68	1 EBR
LatticeXP2™ [3]	Verilog	-5	732	>100	68	1 EBR
	VHDL	-5	730	>100	68	1 EBR

Notes:

1. Performance and utilization characteristics are generated using LCMXO2-1200HC-6TG144CES, with Lattice Diamond™ 1.1 or ispLEVER® 8.1 SP1. When using this design in a different device, density, speed, or grade, performance and utilization may vary.
2. Performance and utilization characteristics are generated using LCMXO2280C-5T100C, with Lattice Diamond 1.1 or ispLEVER 8.1 SP1 software. When using this design in a different device, density, speed, or grade, performance and utilization may vary.
3. Performance and utilization characteristics are generated using LFXP2-5E-5TN144C, with Lattice Diamond 1.1 or ispLEVER 8.1 SP1. When using this design in a different device, density, speed, or grade, performance and utilization may vary.

References

- Compact Flash Specification 2.0 document, CompactFlash Association (CFA)

Technical Support Assistance

Submit a technical support case through www.latticesemi.com/techsupport.

Revision History

Revision 1.4, November 2019

Section	Change Summary
All	• Changed document number from RD1040 to FPGA-RD-02088. • Updated document template.
Disclaimers	Added this section.

Revision 1.3, November 2010

Section	Change Summary
Implementation	Added support for MachXO2 device family and Lattice Diamond design software.

Revision 1.2, December 2009

Section	Change Summary
Implementation	Added support for LatticeXP2 device family.

Revision 1.1, November 2009

Section	Change Summary
Implementation	Added VHDL support..

Revision 1.0, February 2009

Section	Change Summary
All	Initial release.

www.latticesemi.com