System ACE CompactFlash Solution

DS080 (v2.0) October 1, 2008 **Product Specification**

Features

- System-Level Features:
  - High-capacity pre-engineered configuration solution for FPGAs[1]
  - System ACE™ CF Controller XCCACE-TQG144I device
  - Maximum CompactFlash (CF) partition capacity of 2 GB
  - Non-volatile system storage solution
  - Flexible configuration interfaces
  - System configuration rates of up to 30 Mb/s
  - Board space requirement as low as 25 cm^2
- System ACE CF Controller:
  - CompactFlash interface supports most standard third-party CompactFlash (Type I or Type II) cards (up to 8 GB), and Hitachi Microdrives (up to 6 GB)
  - Configuration of a target FPGA chain through IEEE 1149.1 JTAG with a throughput up to 16.7 Mb/s
  - Interfaces include CompactFlash, JTAG, and MPU
  - MPU interface is compatible with various microprocessor and microcontroller bus interfaces, including the Xilinx FPGA-based PowerPC ® and MicroBlaze™ processors
  - IEEE 1149.1 Boundary-Scan Standard Compliant (JTAG)
  - Supports FAT12 and FAT16 file systems
  - Compact 144-pin TQFP package
  - Low power

General Description

Xilinx developed the System Advanced Configuration Environment (System ACE) to address the need for a space-efficient, pre-engineered, high-density configuration solution for systems with multiple FPGAs. System ACE technology is a ground-breaking in-system programmable configuration solution that provides substantial savings in development effort and cost per bit over traditional PROM and embedded solutions for high-capacity FPGA systems.

The System ACE CF solution combines Xilinx expertise in configuration control with industry expertise in commodity memories.

As shown in Figure 1, the System ACE CompactFlash solution is a chipset, consisting of a controller device (System ACE CF controller) and a commercially available CompactFlash storage device.

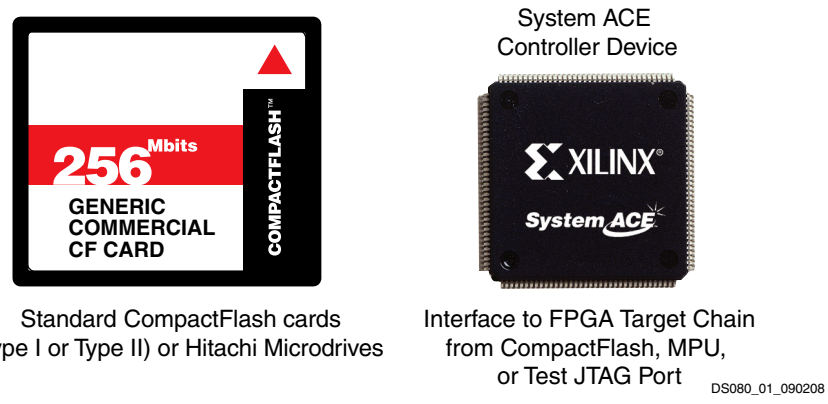


Figure 1: **System ACE CompactFlash Solution**

1. System ACE CF does not support configuration of Xilinx CPLD or PROM devices.

Figure 2 shows that the System ACE CF controller contains multiple interfaces, including CompactFlash, MPU, and JTAG, to allow for a highly flexible configuration solution. For added flexibility, a CompactFlash or Hitachi Microdrive storage device can be used to store multiple bitstreams. The combination of the System ACE CF controller and a standard CompactFlash or Hitachi Microdrive storage device delivers a powerful configuration solution for high-density FPGA systems.

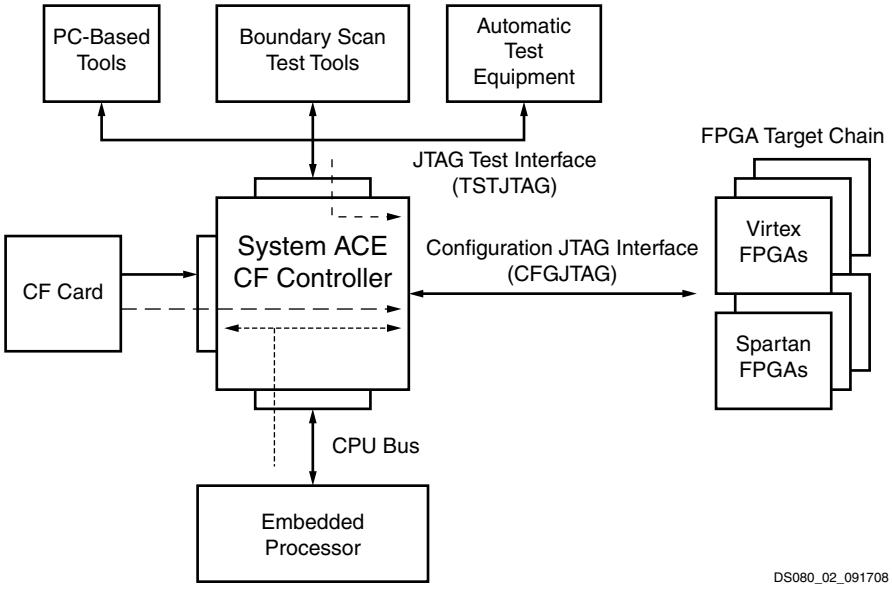


Figure 2: **System ACE CF Controller Interfaces**

System ACE CF Controller

The System ACE CF controller manages FPGA configuration data. The controller provides an intelligent interface between an FPGA target chain and various supported configuration sources; it can target multiple FPGA devices using JTAG at a selectable throughput of up to 16.7 Mbits/sec. As shown in Figure 3, three interfaces are available for configuring a target FPGA chain through the Configuration JTAG Port. These interfaces are: CompactFlash, Microprocessor (MPU), and Test JTAG.

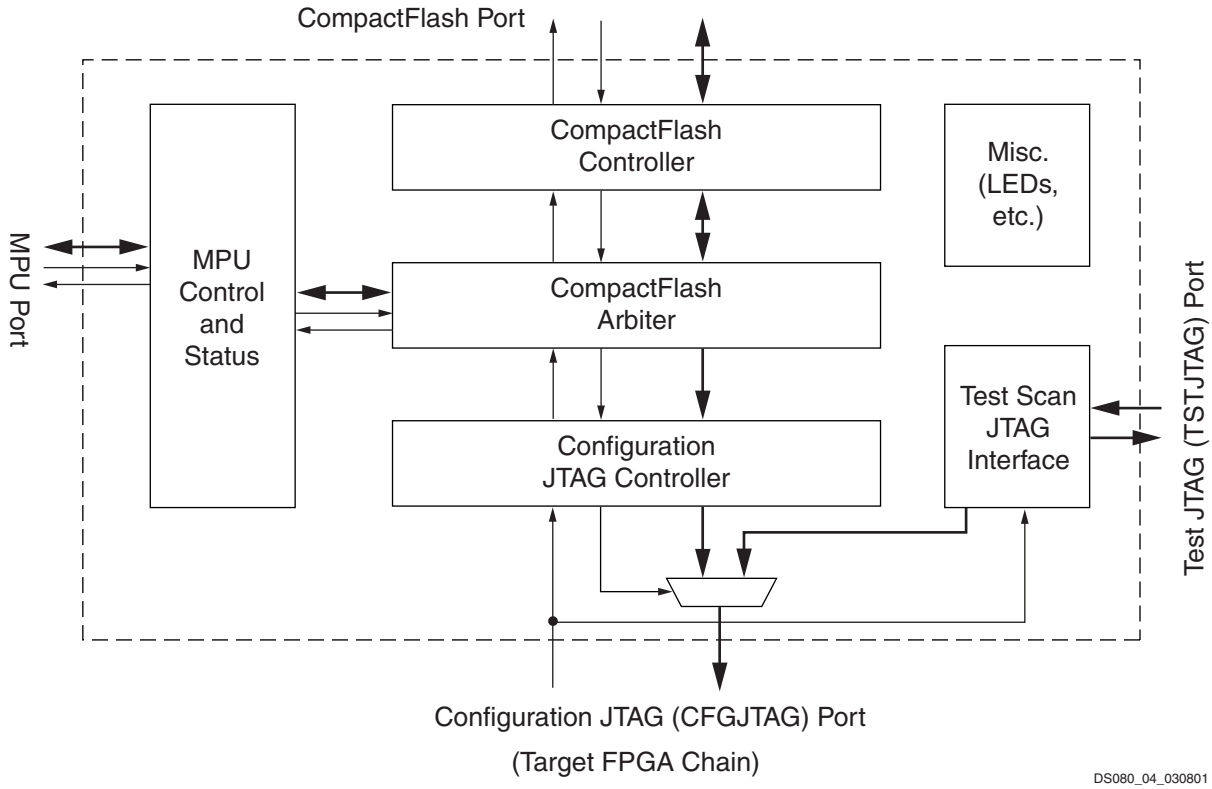


Figure 3: **System ACE CF Controller Block Diagram**

The directory structure used by the System ACE CF controller enables it to support both CompactFlash and Hitachi Microdrive devices through the CompactFlash port.

The MPU interface has access to the CompactFlash port, the Configuration JTAG port, and local control/status features. The Test JTAG port is used when doing Boundary-Scan testing of the target FPGA chain or the System ACE CF controller. Details about each interface are discussed below.

The System ACE CF controller has two main power supplies: the core power supply (V_{CCL}) and a CompactFlash/Test JTAG interface power supply (V_{CCH}). The V_{CCH} power source supplies the Test JTAG and CompactFlash port levels. These two interfaces must be powered at 3.3V. The V_{CCL} core power source supplies the MPU and Configuration JTAG ports, which can be run at 3.3V or 2.5V. It is important to note that the MPU and Configuration JTAG interfaces are always powered at the same voltage. Considerations for the interface voltage are discussed in **Typical Configuration Modes**, page 37. See Figure 4, page 4.

- Shaded output buffers drive $V_{OH} = V_{CCL}$ = 2.5V or 3.3V
- Shaded input buffers sense $V_{IH} = V_{CCL}$ = 2.5V or 3.3V
- All non-shaded output buffers drive $V_{OH} = V_{CCH}$ = 3.3V
- All non-shaded input buffers sense $V_{IH} = V_{CCH}$ = 3.3V
- "LS" denotes level-shifter
- Core voltage level = V_{CCL} = 2.5V or 3.3V

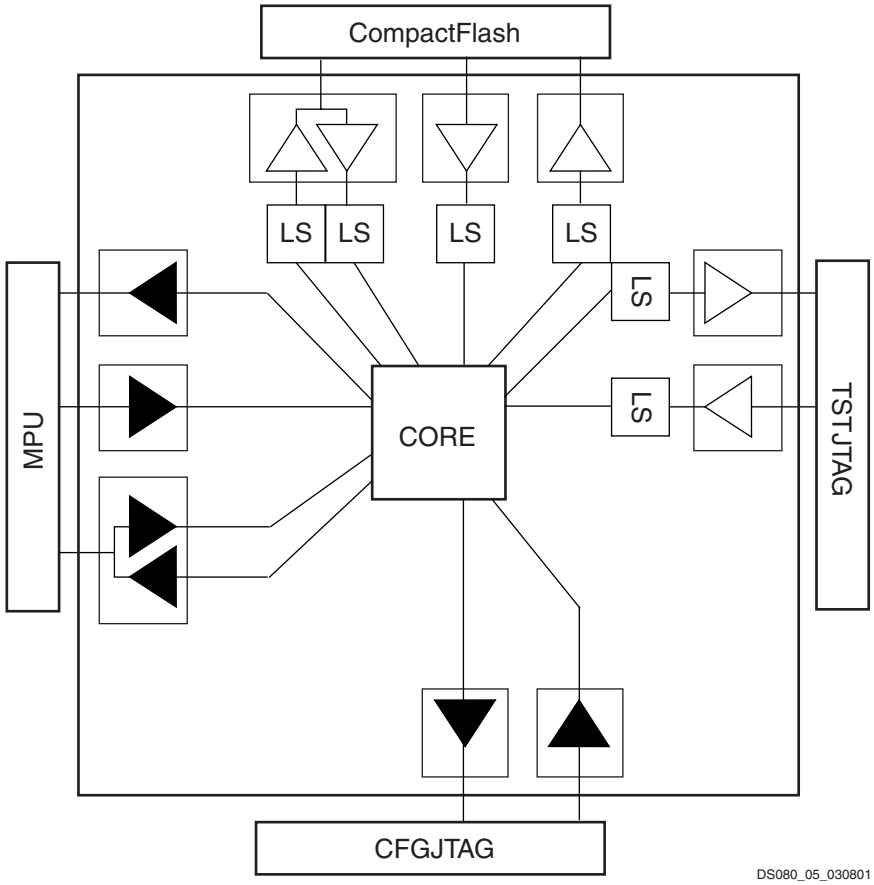


Figure 4: **System ACE CF Controller I/O Requirements**

Status Indicators

The System ACE CF controller has indicator pins (Table 1) to help monitor device status during operation.

Table 1: **System ACE CF Controller Status Indicators**

Name	Pin	Description
$\overline{\text{STATLED}}$	95	• When on, the Status LED indicates that configuration is DONE. • When blinking, this LED indicates that configuration is still in progress. • When off this LED indicates that configuration is in an IDLE state.
$\overline{\text{ERRLED}}$	96	• When on, the ERROR LED indicates that an error occurred. • When blinking, this LED indicates that no CompactFlash device was found when the CompactFlash for the Configuration JTAG interface was enabled. • When off, this LED indicates that no errors are detected.

Resetting the System ACE CF Controller

There are three types of reset of the System ACE CF controller:

1. Power-on-reset (POR)
2. Device reset
3. Configuration controller reset

Power-on-Reset (POR)

The POR circuit is used to reset the entire System ACE CF controller device upon device power up. The built-in POR circuit can be bypassed in order to use an external POR circuit. To bypass the built-in POR circuit, the POR_BYPASS pin should be set to '1' and the POR_RESET pin is used to reset the device (see Table 2).

Note: If the V_{CCL} rail reaches the threshold voltage before the V_{CCH} rail reaches its threshold voltage, then consider using an external POR circuit or $\overline{\text{RESET}}$ pin to hold the device into reset until the V_{CCH} rail reaches the threshold voltage.

Table 2: **POR Functionality**

POR_BYPASS[1]	POR_RESET	Description
'0'	Don't care	Built-in POR circuit is used to reset the device.
'1'	'0'	External POR circuit is selected but the device is not being reset.
'1'	'1' [2]	External POR circuit is selected and the device is being reset.

1. The POR_BYPASS pin should be held at a static '0' or '1' while the System ACE CF controller is receiving power.
2. Hold at '1' for at least one microsecond.

Device-Level Reset

The entire System ACE CF controller device can be reset by asserting the $\overline{\text{RESET}}$ pin of the System ACE CF Controller. The timing associated with this operation is shown in Figure 5, page 6.

CompactFlash Card Reset

The CompactFlash card can be issued a soft reset command by issuing a ResetMemCard command through the CMD[2:0] bits in the **SECCNTCMDREG Register (BYTE address 014h-15h, WORD address 0Ah)**, page 29.

Note: It is important to assert CFGRESET='1' while accessing CompactFlash card sector data via the MPU port, otherwise a CFGERROR condition could result.

Configuration Controller Reset

The configuration controller portion of the System ACE CF device can be reset by asserting CFGRESET = '1' in the CONTROLREG MPU register (CFGRESET is bit 7). Asserting CFGRESET = '1' will reset the portion of the System ACE CF device that controls the reading of ACE file data from the CF card and configuration of the devices connected to the CFGJTAG port. The CFGRESET register is used in conjunction with the CFGMODE and CFGSTART pins/registers to control this configuration process.

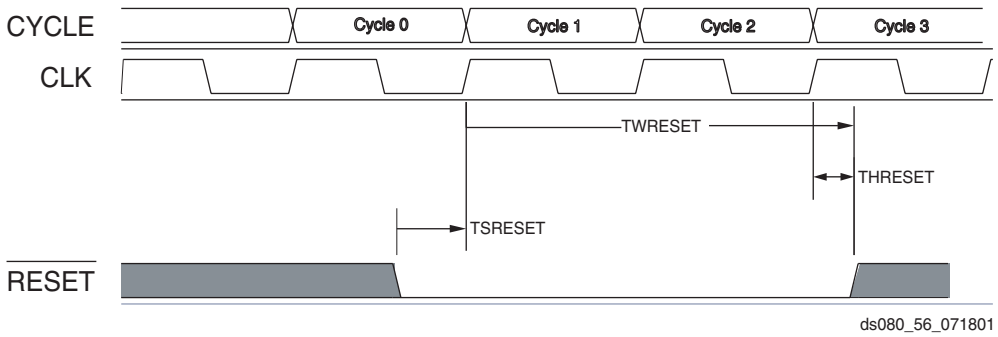


Figure 5: **System ACE RESET Function Timing Diagram**

Table 3: **System ACE RESET**

Symbol	Parameter	Min	Max	Units
TW(RESET)	System ACE CF controller Reset pulse width	3(1)		rising edges
TH(RESET)	Reset hold time after rising edge of CLK	4		ns
TS(RESET)	System ACE CF controller Reset setup up time before rising edge of CLK	7(1)		ns

Notes:
1. When using the System ACE CF controller RESET, TSRESET + TWRESET of three rising edges of CLK is required.

Interfaces Overview

This section discusses the details of each supported System ACE CF controller interface.

CompactFlash Interface (CF)

The CompactFlash interface is the key System ACE CF controller interface for high-capacity systems. The CompactFlash port can accommodate any standard CompactFlash module (up to 8 GB) or Hitachi Microdrives (up to 6 GB), all with the same form factor and board space requirements.

The use of standard CompactFlash devices gives system designers access to high-density Flash memory in a very efficient footprint that does not change with density. CompactFlash is a removable medium which simplifies making changes to the memory contents or upgrading the memory density.

The CompactFlash interface is comprised of two sub-components: a CompactFlash Controller and a CompactFlash Arbiter. The CompactFlash Controller detects the presence and maintains the status of the CompactFlash device. This Controller also handles all CompactFlash device access bus cycles, and abstracts and implements CompactFlash commands such as soft reset, identify drive, and read/write sector(s). The CompactFlash Arbiter controls the interface between the MPU and the Configuration JTAG Controller for access to the CompactFlash data buffer.

When using the CompactFlash card as the configuration source, the CFGTCK output for the System ACE CF controller device is derived from the CLK input to the System ACE CF controller. The operating frequency of the CFGTCK is the same as CLK:

- The minimum clock operating frequency is 0 MHz.
- The maximum clock operating frequency is either 33 MHz *or* the maximum JTAG TCK clock speed dictated by the devices in the JTAG chain and/or the board design. The lowest of these values should be used.

CompactFlash devices are compliant with multiple read and write modes. The System ACE CF controller only supports ATA Common Memory Read and Write functions. Figure 6 and Figure 7, page 8 provide detailed timing information on these functions.

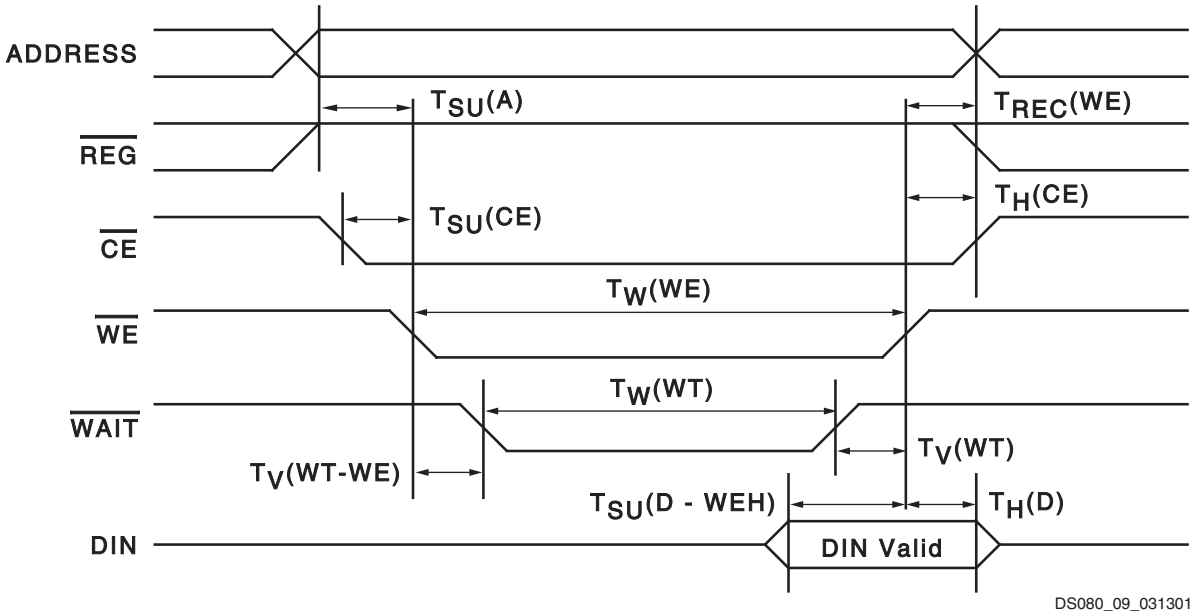


Figure 6: **CompactFlash Common Memory Write Timing Diagram**

Table 4: **Common Memory Write Timing**

Item	Symbol	IEEE Symbol	Min (ns)	Max (ns)
Data Setup before WE	T_{SU}(D-WEH)	tDVWH	80	
Data Hold following WE	T_H(D)	tlWMDX	30	
WE Pulse Width	T_W(WE)	tWLWH	150	
Address Setup Time	T_{SU}(A)	tAVWL	30	
CE Setup before WE	T_{SU}(CE)	tELWL	0	
Write Recovery Time	T_{REC}(WE)	tWMAX	30	

Table 4: **Common Memory Write Timing *(Continued)***

Item	Symbol	IEEE Symbol	Min (ns)	Max (ns)
CE Hold following WE	T_H(CE)	tGHEH	20	
Wait Delay Falling from WE	T_V(WT-WE)	tWLWTV		35
WE HIGH from Wait Release	T_V(WT)	tWTHWH	0	
Wait Width Time (Default Speed)	T_W(WT)	tWTLWTH		350

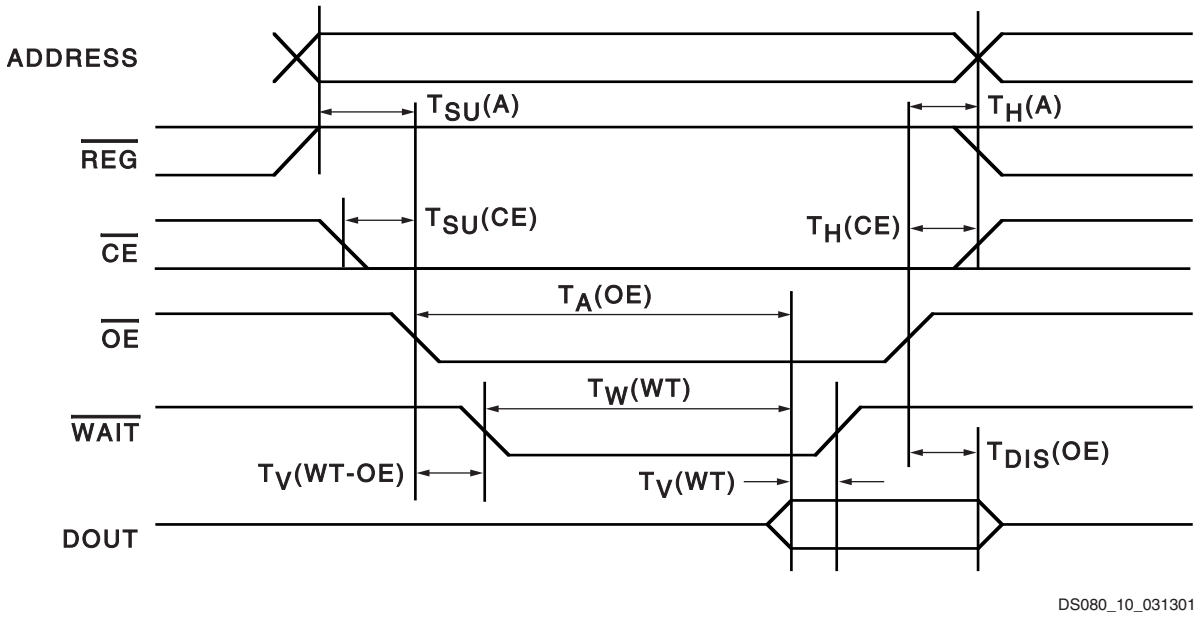


Figure 7: **CompactFlash Common Memory Read Timing Diagram**

Table 5: **Common Memory Read Timing**

Item	Symbol	IEEE Symbol	Min (ns)	Max (ns)
Output Enable Access Time	T_A(OE)	tGLQV		125
Output Disable Time from OE	T_{DIS}(OE)	tGHQZ		100
Address Setup Time	T_{SU}(A)	tAVGL	30	
Address Hold Time	T_H(A)	tGHAX	20	
CE Setup before OE	T_{SU}(CE)	tELGL	0	
CE Hold following OE	T_H(CE)	tGHEH	20	
Wait Delay Falling from OE	T_V(WT-OE)	tGLWTV		35
Data Setup for Wait Release	T_V(WT)	tQVWTH		0
Wait Width Time (Default Speed)	T_W(WT)	tWTLWTH		350

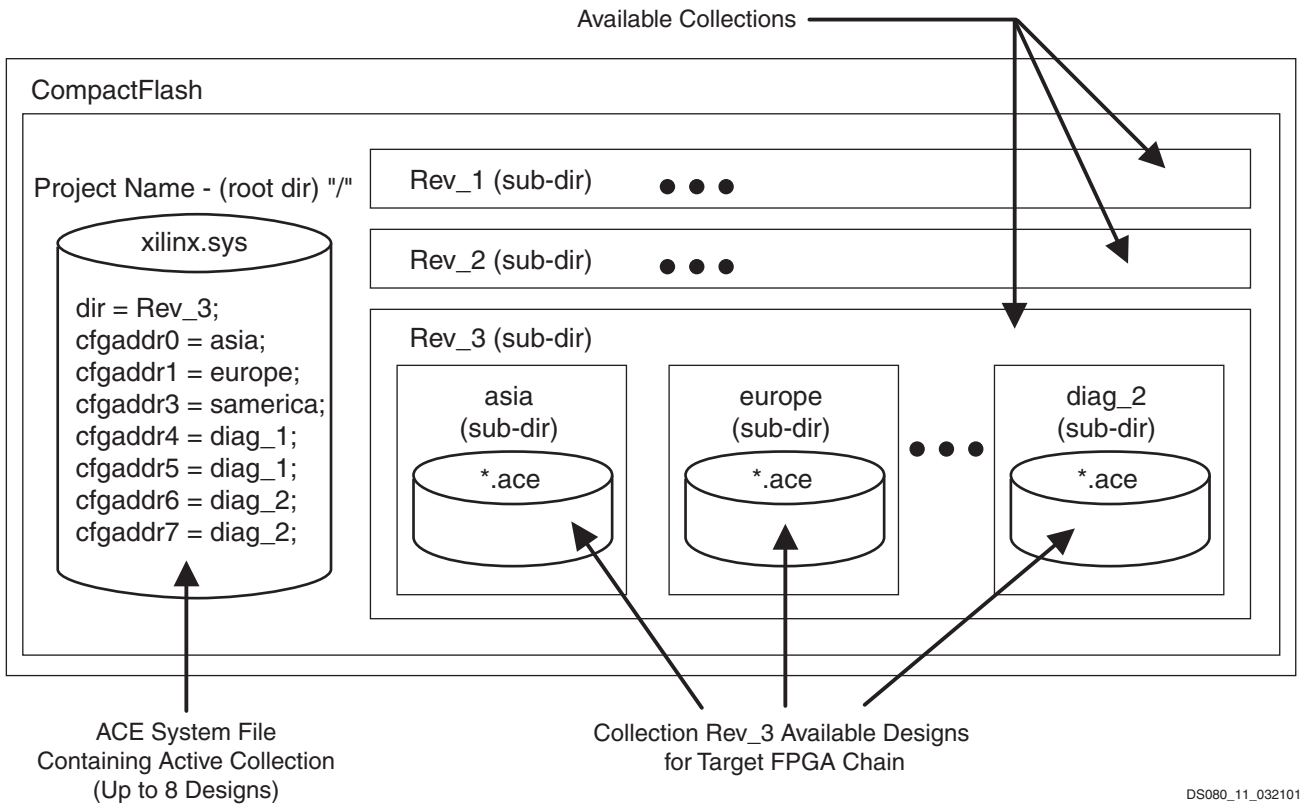


Figure 8: **System ACE Directory Structure**

System ACE CF Directory Structure

A basic understanding of the typical System ACE CF file and directory structure (shown in Figure 8) is useful when programming an FPGA target system with a CompactFlash device in the System ACE solution.

The ACE file is at the lowest level of the directory structure. The Xilinx iMPACT software converts a revision of a design (bitstream) into an ACE file. An ACE file represents a single set of bitstreams for a particular chain of devices.

The next level up in the file structure is a collection. The collection consists of eight ACE files grouped together. All of the ACE files in a collection (directory) can be addressed when in the System ACE CF environment. There can be several collections stored on a CompactFlash device, but only one collection can be active at any given time.

The `xilinx.sys` file determines the collection from which designs can be read.

The hierarchical design of the System ACE CF directory structure provides the ability to maintain multiple revisions or collections of different designs in a single CompactFlash device. Each collection directory can contain one or more designs that reside in different subdirectories. Each design subdirectory should contain a single ACE file that represents a single set of bitstreams for a particular chain of devices. In addition to FPGA configuration information, the collection and design subdirectories can contain other information pertaining to the system design such as system software, documentation, etc.

The `xilinx.sys` file in the root directory of the CompactFlash device is used to control which of the designs within the active collection is to be used to configure the chain of target devices. Only one collection, containing up to eight designs, can be active at one time.

The System ACE CF controller parses the `xilinx.sys` file to determine the active collection designs and uses the three configuration address pins or MPU register bits (CFGADDR) to select the desired design. If no `xilinx.sys` file exists in the root directory of the CompactFlash device, a single ACE file in the root directory is used by System ACE as the active design.

System ACE CF File Structure Requirements

- The System ACE CF file structure must be on the first partition of the CompactFlash device.
- The System ACE CF partition must be formatted as DOS FAT12 or FAT16.
- The `xilinx.sys` file or single ACE file must be in the root directory. The ACE file used only if `xilinx.sys` is not found. The `xilinx.sys` file describes one collection directory with up to eight subdirectories.
- The `xilinx.sys` file must contain the line dir=<*dir_name*>; where <*dir_name*> is the name of the collection directory
- The subsequent 8 lines of the `xilinx.sys` file must consist of the lines cfgaddr<*n*>=<*subdir_name*>; where <*n*> is 1 through 8 and <*subdir_name*> is the name of a design sub-directory in the collection. In the case of fewer than 8 designs in the collection, always start with cfgaddr0 and only use contiguous cfgaddr locations.
-
- Only one ACE file should exist in the ROOT directory and/or in each `\<dir>\<cfgaddr>` folder pointed to by the `xilinx.sys` file.
- When sourcing from the MPU, the total length of the ACE file must be a multiple of 32 bytes. Otherwise, additional dummy bytes (1s or 0s) should be sent to DATABUFREG to flush the last data buffer, allowing the controller to correctly load the final commands in the ACE file.
- All directories accessed by the System ACE CF controller must be formatted in a valid FAT 8.3 file name format.
- ACE file names can be up to eight characters long and must include the `.ace` file extension.
- All directories and ACE file names cannot contain these reserved characters:

left angle bracket	<
right angle bracket	>
colon	:
quote mark	"
forward slash	/
back slash	\
pipe	|

- The Partition Boot Record (PBR) for the first CompactFlash partition that is used by the System ACE CF controller must specify only one reserved sector.
- The CompactFlash card must be formatted with a sector-per-cluster size greater than 1.
- Other files and directories can coexist with System ACE files and directories.
- 2 GB is the maximum capacity partition that the System ACE CF controller can access using the FAT16 file system:

  (65,535 clusters max) X (32 KB per cluster max) = 2,147,123,200 bytes → 2 GB
- 16 MB is the maximum capacity partition that the System ACE CF controller can access using the FAT12 file system:

  (4,086 clusters max) X (4 KB per cluster max) = 16,736,256 bytes → 16 MB

System ACE CF Formatting Requirements

Three potential problem areas arise when formatting the CF card for use with the System ACE CF controller:

1. **Sectors-per-Cluster Size**

   A CF card formatted with only one sector (512 bytes) per cluster can cause problems for the System ACE CF controller.

   When the Windows OS formats the CF card, it uses a formula to determine what it believes to be an optimal sectors-per-cluster value, based on the size of the CF partition and other factors. This can lead some Windows OS versions to specify one sector (512 bytes) per cluster in some CF configurations. For example, this situation is known to occur when formatting 32 MB CF cards with Windows 2000 and Windows XP. Disk formatting utilities (such as mkdosfs, available from http://www1.mager.org/mkdosfs) can be used to avoid this situation.

2. **FAT12 or FAT16 Format**

   The System ACE CF controller does not recognize the FAT32 file system. It was designed to recognize only the FAT12 and FAT16 formats.

3. **Reserved Sectors**

   Reserved sectors are the sectors in the reserved region of the volume starting at the first sector of the volume. The System ACE CF controller can only read a CF card that is formatted with *one reserved sector* in the Partition Boot Record.

Specifying Sectors-per-Cluster and FAT Version

To correct the first two of these formatting issues, the CF card should always be formatted with a sectors-per-cluster size greater than 1 (UnitSize greater than 512), and the FAT file system version should be specified. This can be done using the **`format`** command with the **`/fs:`** and **`/a:`** options in this syntax:

```
format <volume> [/fs:<FileSystem>]
[/a:<UnitSize>]
```

For example:

```
format D: /FS:FAT /A:1024
```

Controlling the Number of Reserved Sectors

Windows 2000, Windows NT, and Windows 98 default to *one reserved sector* when formatting. Therefore, formatting the CF card using these Windows operating systems is not problematical in this regard.

In Windows XP, however, the DOS `format` command automatically formats the CF card with *from two to eight* reserved sectors, depending on the density of the CF card.

Because the DOS format command does not allow specification of the number of reserved sectors, an alternate disk formatting utility (such as mkdosfs, available from http://www1.mager.org/mkdosfs) must be used. When the CF card is correctly formatted, Windows XP can be used to perform normal file access (read/write) operations without causing any additional problems.

Microprocessor Interface (MPU)

The MPU Interface provides a useful means of monitoring the status of and controlling the System ACE CF controller, as well as CompactFlash card READ / WRITE data. The MPU is not required for normal operation, but when used, it provides numerous capabilities. This interface enables communication between an MPU device and a CompactFlash module and the FPGA target system.

The MPU interface is composed of a set of registers that provide a means for communicating with CompactFlash control logic, configuration control logic, and other resources in the System ACE CF controller. Specifically, this interface can be used to read the identity of a CompactFlash device and read/write sectors from or to a CompactFlash device.

The MPU interface can also be used to control configuration flow. The MPU interface enables monitoring of System ACE CF controller configuration status and error conditions. The MPU interface can be used to delay configuration, start configuration, determine the source of configuration (CompactFlash or MPU), control the bitstream version, reset the device, etc.

Two important issues should be understood when using the microprocessor port:

- For the System ACE CF controller to be properly synchronized, the device driving the MPU interface must be synchronized to the CLK signal
- The MPU must comply with System ACE timing requirements

This general-purpose microprocessor interface can update the CompactFlash, read the ACE status, or obtain direct access to the JTAG configuration ports using the ACE Microprocessor commands. This interface supports either 8-bit (default) or 16-bit data transfers. The bus width can be configured dynamically.

All communications between the System ACE CF controller and a host microprocessor involve transfer of data to or from ACE registers. There are 128 addressable registers in 8-bit mode and 64 addressable registers in 16-bit mode. For easy selection of a new configuration from CompactFlash data, the MPU interface allows for easy reconfiguration of an FPGA chain or capability.

When using the MPU interface as the configuration source, the CFGTCK output for the System ACE CF controller device is derived from the CLK input to the System ACE CF controller (supplied by the MPU), and the operating frequency of the CFGTCK is the same as CLK.

- The minimum clock operating frequency is 0 MHz.
- The maximum clock operating frequency is either 33 MHz *or* the maximum JTAG TCK clock speed dictated by the devices in the JTAG chain and/or the board design. The lowest of these values should be used.

The following sections describe supported operations when using the MPU interface.

MPU Port Signal Description

MPU interface port signals are described in Table 6.

Table 6: **MPU Interface Port Signal Description**

Name	Width	Direction	Active	Description
MPA	7	In	N/A	Synchronous address inputs. The internal address register is loaded by MPA by a combination of the rising edge of CLK and $\overline{\text{MPCE}}$ LOW.
MPD	16	In/Out	N/A	Synchronous data input/output pins. Both the data input and output path are registered and triggered by the rising edge of CLK.
$\overline{\text{MPCE}}$	1	In	LOW	Synchronous active LOW chip enable. $\overline{\text{MPCE}}$ LOW is used to enable the MPU interface. $\overline{\text{MPCE}}$ LOW is also used in conjunction with $\overline{\text{MPOE}}$ LOW to enable the MPD output.

Table 6: **MPU Interface Port Signal Description *(Continued)***

Name	Width	Direction	Active	Description
$\overline{\text{MPWE}}$	1	In	LOW	Synchronous active LOW write enable. A high-to-low-to-high transition must occur on $\overline{\text{MPWE}}$ in three consecutive clock cycles in order for the write to take place.During a valid write cycle, $\overline{\text{MPCE}}$ must be LOW and MPD must be valid during the clock cycle that $\overline{\text{MPWE}}$.
$\overline{\text{MPOE}}$	1	In	LOW	Asynchronous active LOW output enable. Both $\overline{\text{MPOE}}$ and $\overline{\text{MPCE}}$ must be LOW to read from the MPU interface. When either $\overline{\text{MPOE}}$ or $\overline{\text{MPCE}}$ is HIGH, the MPD pins of the System ACE CF controller are in a high-impedance state.
MPBRDY	1	Out	HIGH	Synchronous active HIGH buffer ready output. During data buffer read mode MPBRDY is HIGH when the data in the DATABUF buffer is valid. During data buffer write mode MPBRDY is HIGH when data can be written to the DATABUF buffer.
MPIRQ	1	Out	HIGH	Synchronous active HIGH interrupt request output. MPIRQ HIGH indicates that an interrupt condition has occurred in the MPU interface. All interrupt conditions must be manually cleared before MPIRQ will go LOW. MPIRQ is always LOW when interrupts are disabled.

MPU Timing Description

This section contains timing diagrams for the MPU interface. Parameters used in the timing diagrams are described in Table 7.

Table 7: **MPU Interface Timing Parameters**

Symbol	Parameter	Min	Max	Units
tSA	Address setup time	4	--	ns
tSCE	Chip enable setup time	4	--	ns
tSWE	Write enable setup time	12	--	ns
tSOE	Output enable setup time	12	--	ns
tSD	Data setup time	4	--	ns
tDD	Clock HIGH to valid data	--	22	ns
tDOE	Chip/Output enable LOW to valid data	--	13	ns
tDBRDY	Clock HIGH to buffer ready valid	--	22	ns
tH	Hold time	4	--	ns

Single Register Read Cycle

The single register read cycle is shown in Figure 9, page 13. A single register read is accomplished by asserting a valid address (MPA), asserting the chip enable ($\overline{\text{MPCE}}$ = LOW) and de-asserting the write enable ($\overline{\text{MPWE}}$ = HIGH) during the first clock cycle (Cycle 0). These signals should hold these values at least until the rising edge of the fourth clock cycle (Cycle 3).

The output enable signal should be asserted ($\overline{\text{MPOE}}$ = LOW) during the third clock cycle (Cycle 2). Register data associated with the specified address appears on the MPD bus two clock cycles after the falling edge of $\overline{\text{MPCE}}$ during the assertion of $\overline{\text{MPCE}}$. The register read cycle is then completed by de-asserting the output enable during the fourth clock cycle (Cycle 3).

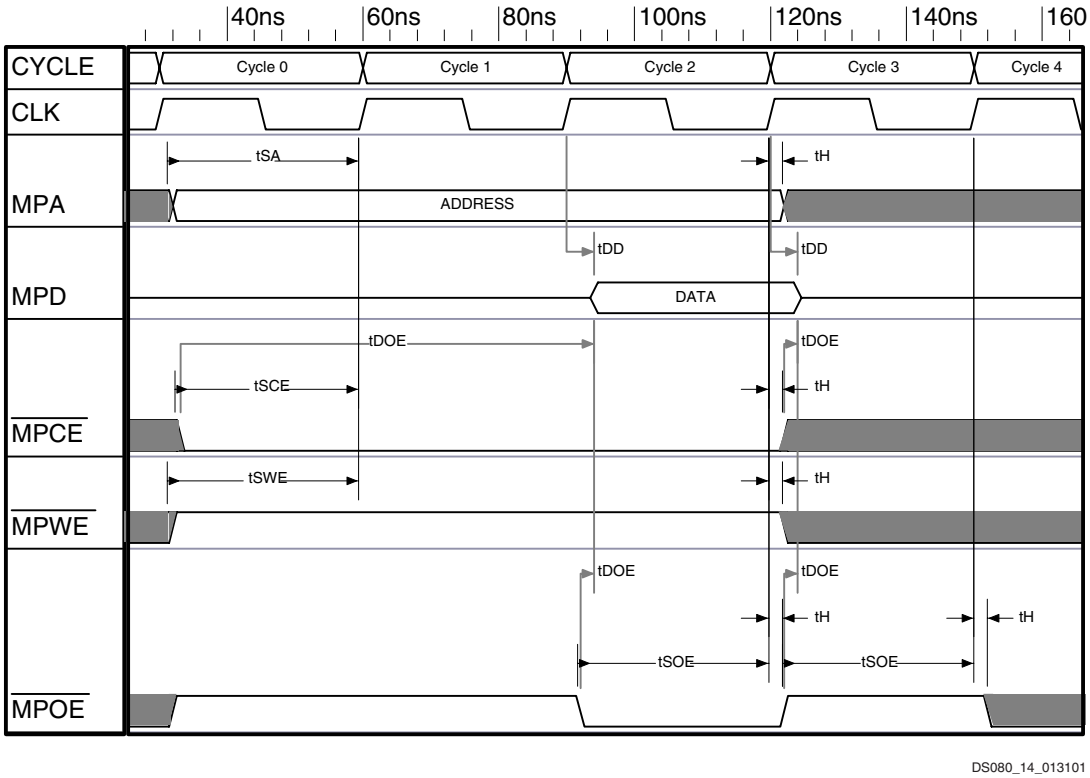


Figure 9: **Single Read From an ACE Register**

Single Register Write Cycle

The single register write cycle is shown in Figure 10. A single register write is accomplished by asserting a valid address (MPA), asserting the chip enable ($\overline{\text{MPCE}}$ = LOW) and de-asserting the output enable ($\overline{\text{MPOE}}$ = HIGH) during the first clock cycle (Cycle 0). These signals should hold these values at least until the rising edge of the third clock cycle (Cycle 2).

The write enable signal should be asserted ($\overline{\text{MPWE}}$ = LOW) during the second clock cycle (Cycle 1). Data (MPD) to be written to the specified address should be asserted during the same clock cycle that the write enable is asserted (Cycle 1). The register write cycle is then completed by de-asserting the write enable during the third clock cycle (Cycle 2).

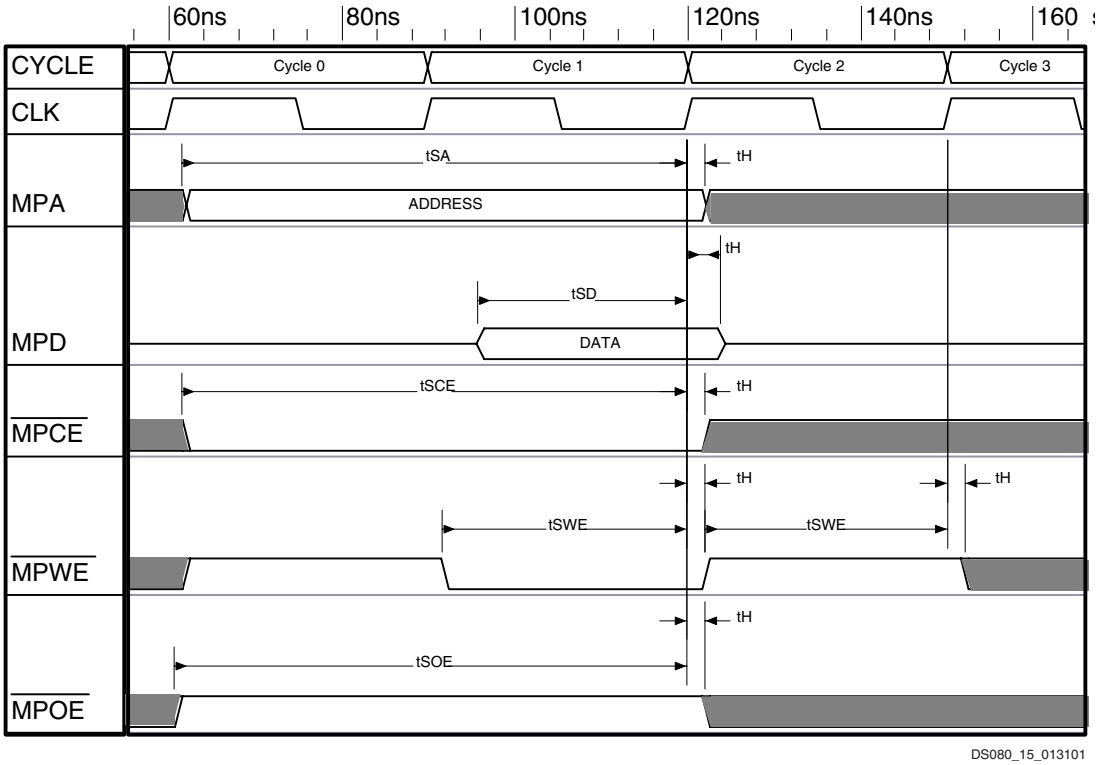


Figure 10: **Single WORD Write to an ACE Register**

Multiple Register Read Timing

The minimum timing requirements for sequential register read cycles are shown in Figure 11. Sequential read cycles are identical to single read cycles, except that the chip enable ($\overline{\text{MPCE}}$) and write enable ($\overline{\text{MPWE}}$) signals do not need to be de-asserted between read cycles.

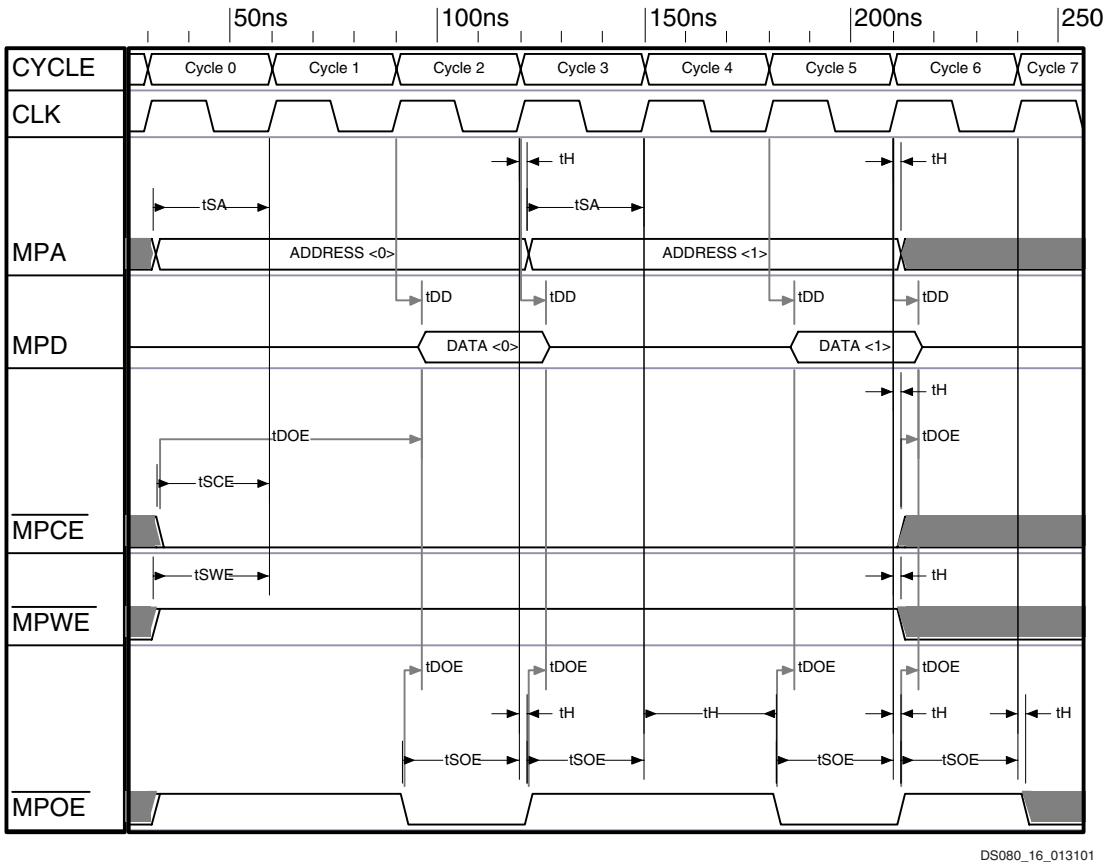


Figure 11: **Multiple WORD Reads From ACE Register(s)**

Multiple Register Write Timing

The minimum timing requirements for sequential write cycles are shown in Figure 12. Sequential write cycles are identical to single write cycles except that the chip enable ($\overline{\text{MPCE}}$) and output enable ($\overline{\text{MPOE}}$) signals do not need to be de-asserted between write cycles.

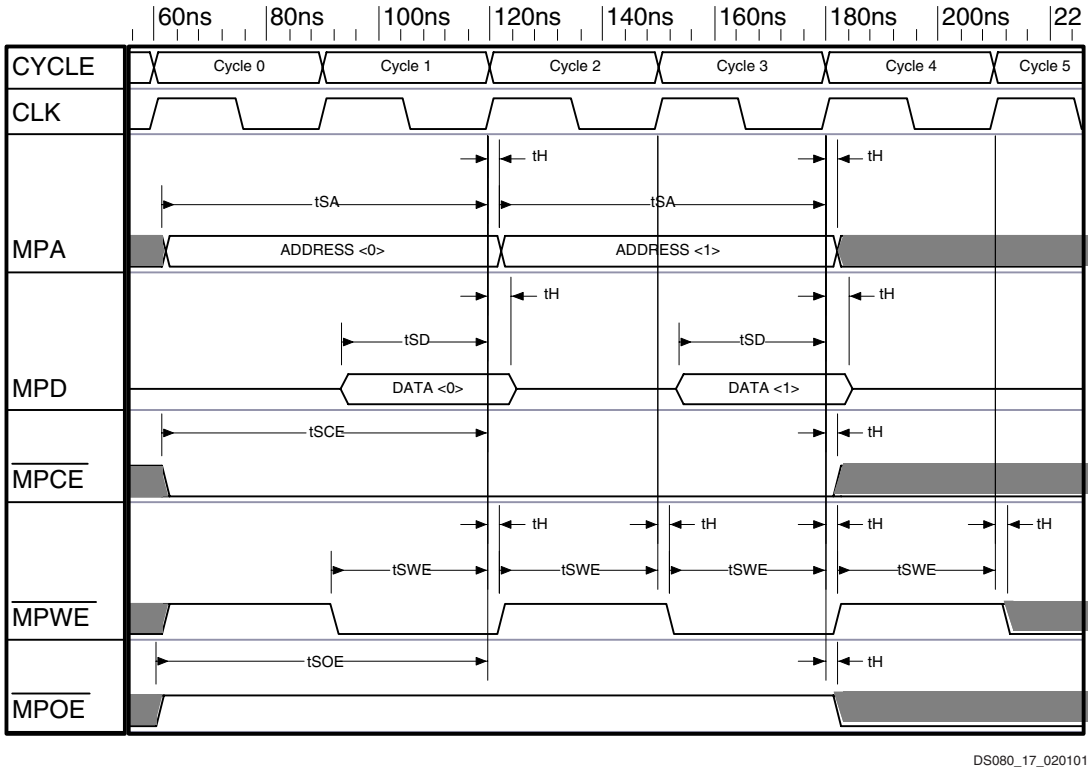


Figure 12: **Multiple WORD Writes to ACE Register(s)**

Data Buffer Ready Timing

The data buffer ready (MPBRDY) signal indicates whether the data buffer is ready to accept new data during a write cycle or whether the data buffer contains valid data to be read during a read cycle. The data buffer itself is sixteen words deep, where each word is 16 bits wide.

The data buffer mode transfer direction is identified by the state of the DATABUFMODE bit in the STATUSREG register:

- DATABUFMODE = 0 indicates data buffer read mode
- DATABUFMODE = 1 indicates data buffer write mode

The data buffer mode depends on the type of command that was issued to the System ACE CF controller. If an IdentifyMemCard or ReadMemCard command was issued, then the data buffer remains in read mode until the command is finished executing (i.e., all sector data has been read from the buffer). If a WriteMemCard command was issued, then the data buffer remains in write mode until the command is finished executing (i.e., all sector data has been written to the buffer).

Data Buffer Read Cycle Ready Timing

When the data buffer is in read mode and the last data word is read from the buffer, the data buffer ready signal will go inactive (MPBRDY = LOW) two clock cycles following the last clock cycle that the output enable is active ($\overline{\text{MPOE}}$ = LOW). Any attempt to read data out of an "empty" data buffer ($\overline{\text{MPOE}}$ = LOW while MPBRDY = LOW) results in invalid data. Valid and invalid data buffer reads are shown in Figure 13.

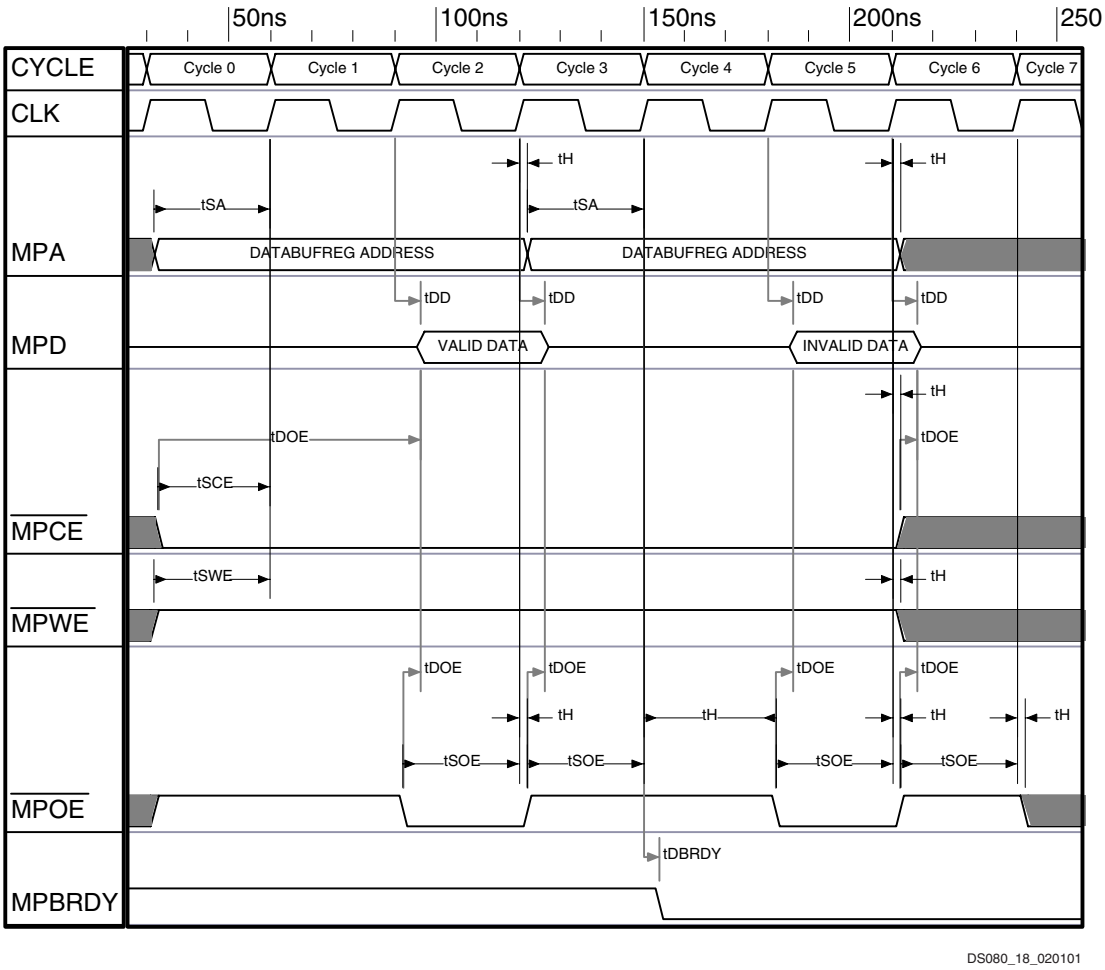


Figure 13: **Valid and Invalid Reads From DATABUFREG Data Buffer**

Data Buffer Write Cycle Ready Timing

When the data buffer is in write mode and the last available space for a data word has been filled, the data buffer ready signal will go inactive (MPBRDY = LOW) two clock cycles following the last clock cycle that the write enable is active ($\overline{\text{MPWE}}$ = LOW). Any attempt to write data to a "full" data buffer ($\overline{\text{MPWE}}$ = LOW while MPBRDY = LOW) does not result in a successful write to the buffer. Valid and invalid data buffer writes are shown in Figure 14.

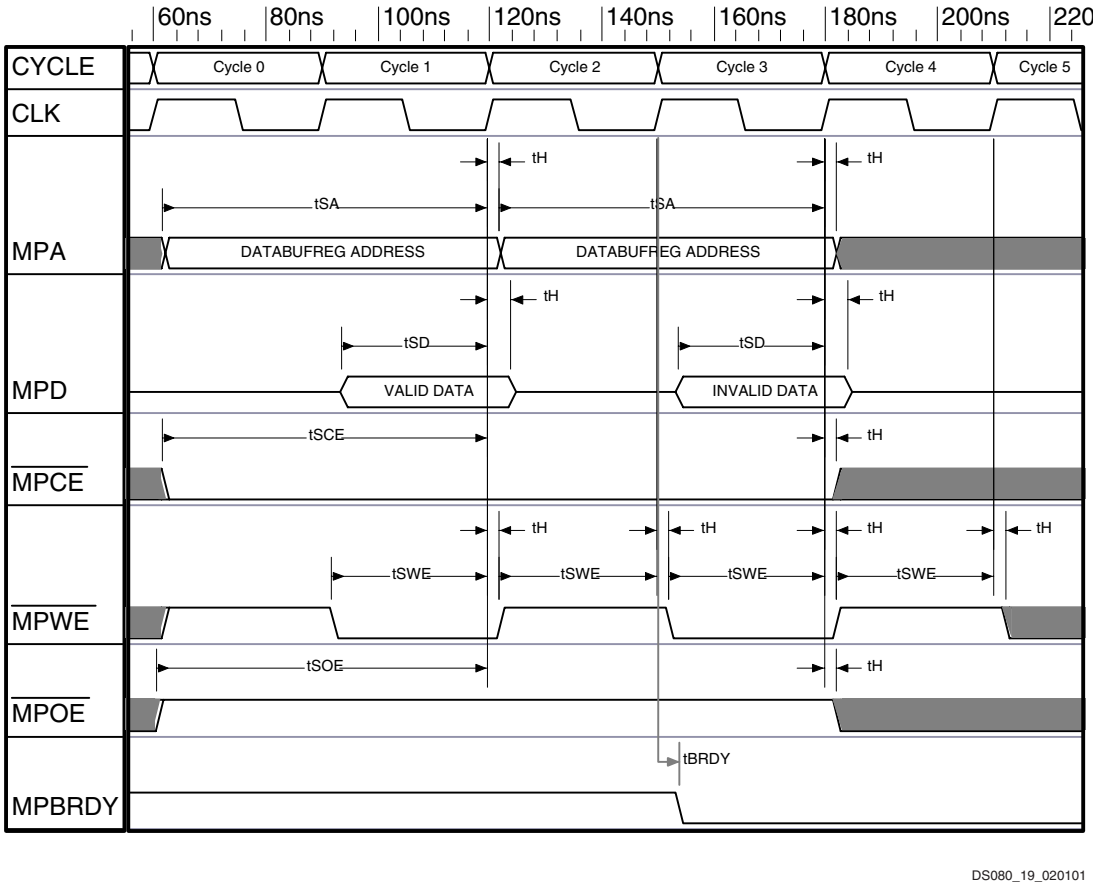


Figure 14: **Valid and Invalid Writes to DATABUFREG Data Buffer**

Interrupt Timing

The interrupt request and clearing cycles are shown in Figure 15. In Figure 15, the interrupt request (MPIRQ = HIGH) occurs sometime before Cycle 0. The interrupt request is cleared by performing a single MPU write cycle that sets RESETIRQ = 1 (bit number 11) in the CONTROLREG(15:0) register (BYTE address 0x19 or WORD address 0x0C).

The MPU interrupt request line (MPIRQ) remains active HIGH until the RESETIRQ bit is set. The MPIRQ line becomes inactive LOW two cycles after the completion of the RESETIRQ write cycle (Cycle 4). For subsequent MPU interrupt requests to be enabled, the RESETIRQ bit must be reset and one of the three IRQ enable bits (DATABUFRDYIRQ, ERRORIRQ, and/or CFGDONEIRQ) in the CONTROLREG register should be set.

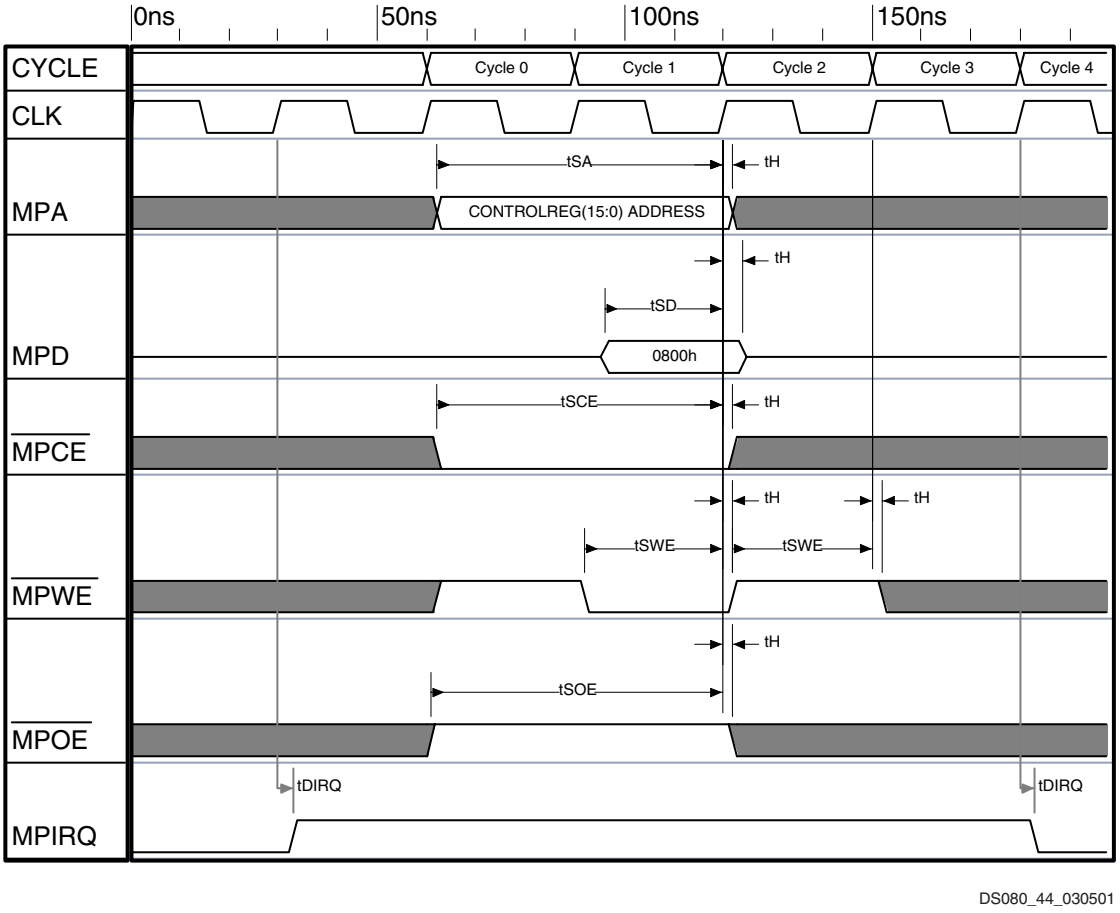


Figure 15: **Interrupt Request Timing**

Register Specification

The BYTE-mode register space of the MPU interface is shown in Table 8.

Table 8: **Register Address Map (BYTE Mode Addresses)**

BYTE Address (MPA [6:0])	Register Name	Width	Mode	Description
0x00	BUSMODEREG	1	RW	Used to control the data bus access mode (8-bit BYTE mode or 16-bit WORD mode)
0x01	BUSMODEREG	1	RW	
0x02	--	--	--	Reserved
0x03	--	--	--	Reserved
0x04	STATUSREG(7:0)	8	R	Used to monitor System ACE CF controller status
0x05	STATUSREG(15:8)	8	R	
0x06	STATUSREG(23:16)	8	R	
0x07	STATUSREG(31:24)	8	R	
0x08	ERRORREG(7:0)	8	R	Used to indicate any existing error condition
0x09	ERRORREG(15:8)	8	R	
0x0A	ERRORREG(23:16)	8	R	
0x0B	ERRORREG(31:24)	8	R	
0x0C	CFGLBAREG(7:0)	8	R	Logical block address used by the Configuration Controller during CompactFlash data transfers
0x0D	CFGLBAREG(15:8)	8	R	
0x0E	CFGLBAREG(23:16)	8	R	
0x0F	CFGLBAREG(27:24)	4	R	
0x10	MPULBAREG(7:0)	8	RW	Logical block address used by the MPU interface during CompactFlash data transfers
0x11	MPULBAREG(15:8)	8	RW	
0x12	MPULBAREG(23:16)	8	RW	
0x13	MPULBAREG(27:24)	4	RW	
0x14	SECCNTCMDREG(7:0)	8	RW	Sector count and CompactFlash command register
0x15	SECCNTCMDREG(15:8)	8	RW	
0x16	VERSIONREG(7:0)	8	R	Version register
0x17	VERSIONREG(15:8)	8	R	
0x18	CONTROLREG(7:0)	8	RW	Used to control System ACE CF controller operations
0x19	CONTROLREG(15:8)	8	RW	
0x1A	CONTROLREG(23:16)	8	RW	
0x1B	CONTROLREG(31:24)	8	RW	
0x1C	FATSTATREG(7:0)	8	R	Contains information about the FAT table of the first valid partition found in the CompactFlash device.
0x1D	FATSTATREG(15:8)	8	R	
0x1E through 0x3F	--	--	--	Reserved
Even Values 0x40 through 0x5E	DATABUFREG(7:0)	8	RW	Address range that provides read and write access to the data buffer.
Odd Values 0x41 through 0x5F	DATABUFREG(15:8)	8	RW	

The 16-bit WORD mode register space of the MPU interface is shown in Table 9.

Table 9: **Register Address Map (WORD Mode Addresses)**

WORD Address (MPA [6:1])	Register Name	Width	Mode	Description
0x00	BUSMODEREG	1	RW	Used to control the data bus access mode (8-bit BYTE mode or 16-bit WORD mode)
0x01	--	--	--	Reserved
0x02	STATUSREG(15:0)	16	R	Used to monitor System ACE CF controller status
0x03	STATUSREG(31:16)	16	R	
0x04	ERRORREG(15:0)	16	R	Used to indicate any existing error condition
0x05	ERRORREG(31:16)	16	R	
0x06	CFGLBAREG(15:0)	16	R	Logical block address used by the Configuration Controller during CompactFlash data transfers
0x07	CFGLBAREG(27:16)	12	R	
0x08	MPULBAREG(15:0)	16	RW	Logical block address used by the MPU interface during CompactFlash data transfers
0x09	MPULBAREG(27:16)	12	RW	
0x0A	SECCNTCMDREG(15:0)	16	RW	Sector count and CompactFlash command register
0x0B	VERSIONREG(15:0)	16	R	Version register
0x0C	CONTROLREG(15:0)	16	RW	Used to control System ACE CF controller operations
0x0D	CONTROLREG(31:16)	16	RW	
0x0E	FATSTATREG(15:0)	16	R	Contains information about the FAT table of the first valid partition found in the CompactFlash device.
0x0F through 0x1F	--	--	--	Reserved
0x20 through 0x2F	DATABUFREG(15:0)	16	RW	Address range that provides read and write access to the data buffer.

BUSMODEREG Register (BYTE address 00h-01h, WORD address 00h)

The BUSMODEREG register is used to control the mode of the MPU address and data bus. The single-bit BUSMODEREG register is aliased across two BYTE addresses (0x00-0x01) and one 16-bit WORD address (0x0). This register aliasing ensures that the MPU bus mode can be set regardless of the mode of the microprocessor that is communicating with the System ACE CF controller. Table 10 provides a description of the BUSMODEREG register bits.

Table 10: **BUSMODEREG Register Bit Descriptions**

Bit	Name	Description
0	BUSMODE0	The BUSMODE bits are used to select the width of the data bus portion of the Microprocessor bus (default is 0): • When 0, the MPU interface is in BYTE mode (all MPU address bits are used, but only MPU data bits 7:0 are used). • When 1, the MPU interface is in WORD mode (all MPU data bits are used, but only MPU address bits 6:1 are used).
1	--	Reserved
2	--	Reserved
3	--	Reserved
4	--	Reserved
5	--	Reserved
6	--	Reserved
7	--	Reserved

STATUSREG Register (BYTE address 04h-07h, WORD address 02h-03h)

The STATUSREG register allows a microprocessor to monitor important System ACE CF controller operating modes. This is also the register that is read upon receiving an IRQ request in order to identify an interrupt source. Table 11 provides a description of the STATUSREG register bits.

Table 11: **STATUSREG Register Bit Descriptions**

Bit	Name	Description
0	CFGLOCK	Configuration controller lock status: • 0 means that the configuration controller does not currently have a lock on the CompactFlash controller resource • 1 means that the configuration controller has successfully locked the CompactFlash controller resource
1	MPULOCK	MPU interface lock status: • 0 means that the MPU interface does not currently have a lock on the CompactFlash controller resource • 1 means that the MPU interface has successfully locked the CompactFlash controller resource
2	CFGERROR	Configuration Controller error status: • 0 means that no Configuration Controller error condition exists • 1 means that an error has occurred in the Configuration Controller (check the ERRORREG register for more information)
3	CFCERROR	CompactFlash Controller error status: • 0 means that no CompactFlash Controller error condition exists • 1 means that an error has occurred in the CompactFlash controller (check the ERRORREG register for more information)

Table 11: **STATUSREG Register Bit Descriptions *(Continued)***

Bit	Name	Description
4	CFDETECT	CompactFlash detect flag: • 0 means that no CompactFlash device is connected to the System ACE CF controller • 1 means that a CompactFlash is connected to the System ACE CF controller
5	DATABUFRDY	Data buffer ready status: • 0 means that the data buffer is not ready for data transfer • 1 means that the data buffer is ready for data to be transferred out of the buffer when reading from the CompactFlash controller or into the buffer when writing to the CompactFlash or Configuration controller
6	DATABUFMODE	Data buffer mode status: • 0 means read-only mode • 1 means write-only mode
7	CFGDONE	Configuration DONE status: • 0 means that the configuration process has not completed • 1 means that the entire System ACE CF controller configuration file has been executed and configuration of all devices in the target Boundary-Scan chain is complete
8	RDYFORCFCMD	Ready for CompactFlash controller command: • 0 means not ready for command • 1 means ready for command
9	CFGMODEPIN	Configuration mode pin (note that this can be overridden by the CFGMODE bit in the CONTROLREG register): • 1 means automatically start the configuration process immediately after System ACE CF controller Reset • 0 means wait for CFGSTART bit in CONTROLREG before starting the configuration process
10	--	Reserved
11	--	Reserved
12	--	Reserved
13	CFGADDRPIN0	Configuration address pins that are used as an offset into the system configuration file in the CompactFlash device used to locate the System ACE CF controller configuration data file (note that these pins can be overridden by the contents of the CFGADDRBIT[2:0] of the CONTROLREG register)
14	CFGADDRPIN1	
15	CFGADDRPIN2	
16	--	Reserved
17	CFBSY	CompactFlash BUSY bit (reflects the state of the BSY bit in the status register of the CompactFlash device): • 0 means that the CompactFlash device is not busy • 1 means that the CompactFlash command register and data buffer cannot be accessed; Bits 18-23 of the STATUSREG register are not valid when this bit is set to 1
18	CFRDY	CompactFlash ready for operation bit (reflects the state of the RDY bit in the status register of the CompactFlash device): • 0 means the CompactFlash device is NOT ready to accept commands • 1 means CompactFlash device is ready to accept commands
19	CFDWF	CompactFlash data write fault bit (reflects the state of the DWF bit in the status register of the CompactFlash device): • 0 means that a write fault has NOT occurred • 1 means that a write fault has occurred

Table 11: **STATUSREG Register Bit Descriptions *(Continued)***

Bit	Name	Description
20	CFDSC	CompactFlash ready bit (reflects the state of the DSC bit in the status register of the CompactFlash device): • 0 means that the CompactFlash device is NOT ready • 1 means that the CompactFlash device is ready
21	CFDRQ	CompactFlash data request bit (reflects the state of the DRQ bit in the status register of the CompactFlash device): • 0 means that no data is ready to be transferred to/from the data buffer of the CompactFlash device • 1 means that information be transferred to/from the data buffer of the CompactFlash device
22	CFCORR	CompactFlash correctable error bit (reflects the state of the CORR bit in the status register of the CompactFlash device): • 0 means that a correctable data error was NOT encountered • 1 means that a correctable data error was encountered (check the ERRORREG register for more information)
23	CFERR	CompactFlash ERROR bit (reflects the state of the ERR bit in the status register of the CompactFlash device): • 0 means that no error has occurred during the execution of the previous command • 1 means that the previous command has ended in some type of error (check the ERRORREG register for more information)
24	--	Reserved
25	--	Reserved
26	--	Reserved
27	--	Reserved
28	--	Reserved
29	--	Reserved
30	--	Reserved
31	--	Reserved

ERRORREG Register (BYTE address 08h-0Bh, WORD address 04h-05h)

The ERRORREG register identifies specific information on any error conditions that might exist in the System ACE CF controller. Table 12 provides a description of the ERRORREG register bits.

Table 12: **ERRORREG Register Bit Descriptions**

Bit	Name	Description
0	CARDRESETERR	CompactFlash card reset error: • 0 means no error • 1 means that the CompactFlash card has failed to reset properly before a time-out condition occurred
1	CARDRDYERR	CompactFlash card ready error: • 0 means no error • 1 means that the CompactFlash card has failed to become properly ready for commands before a time-out condition occurred
2	CARDREADERR	CompactFlash card read error: • 0 means no error • 1 means that a CompactFlash data read command (either ReadMemCardData or IdentifyMemCard) has failed

Table 12: **ERRORREG Register Bit Descriptions *(Continued)***

Bit	Name	Description
3	CARDWRITEERR	CompactFlash card write error: • 0 means no error • 1 means that a CompactFlash data write command (WriteMemCardData) has failed
4	SECTORRDYERR	CompactFlash sector ready: • 0 means no error • 1 means that a sector has failed to become properly valid during a CompactFlash read or write command before a time-out condition occurred
5	CFGADDRERR	CFGADDR error: • 0 means no error • 1 means that the CFGADDR (i.e., the CFGADDR(15:0) register or CFGADDR(1:0) pins, depending on the state of the FORCECFGADDR bit in the CONTROLREG register) does not correspond to a valid location in the CompactFlash
6	CFGFAILED	Configuration failure error: • 0 means no error • 1 means that configuration of one or more devices in the target Boundary-Scan chain has failed
7	CFGREADERR	Configuration read error: • 0 means no error • 1 means that an error occurred while reading configuration information from CompactFlash
8	CFGINSTRERR	Configuration instruction error: • 0 means no error • 1 means that an invalid instruction was encountered during configuration
9	CFGINITERR	Configuration INIT monitor error: • 0 means no error • 1 means that the $\overline{\text{CFGINIT}}$ pin did not go HIGH within 500 ms of the start of configuration
10	--	Reserved
11	CFBBK	CompactFlash bad block error (reflects the state of the BBK bit in the error register of the CompactFlash device): • 0 means no error • 1 means that a bad block has been detected
12	CFUNC	CompactFlash uncorrectable error (reflects the state of the UNC bit in the error register of the CompactFlash device): • 0 means no error • 1 means that an uncorrectable error has been encountered
13	CFIDNF	CompactFlash ID not found error (reflects the state of the IDNF bit in the error register of the CompactFlash device): • 0 means no error • 1 means that the requested sector ID is in error or cannot be found
14	CFABORT	CompactFlash command abort error (reflects the state of the ABRT bit in the error register of the CompactFlash device): • 0 means no error • 1 means that the command has been aborted because of a CompactFlash status condition (i.e., Not Ready, Write Fault) or when an invalid command has been issued

Table 12: **ERRORREG Register Bit Descriptions *(Continued)***

Bit	Name	Description
15	CFAMNF	CompactFlash general error (reflects the state of the AMNF bit in the error register of the CompactFlash device): • 0 means no error • 1 means that a general error has occurred
16	--	Reserved
17	--	Reserved
18	--	Reserved
19	--	Reserved
20	--	Reserved
21	--	Reserved
22	--	Reserved
23	--	Reserved
24	--	Reserved
25	--	Reserved
26	--	Reserved
27	--	Reserved
28	--	Reserved
29	--	Reserved
30	--	Reserved
31	--	Reserved

CFGLBAREG Register (BYTE address 0Ch-0Fh, WORD address 06h-07h)

The CFGLBAREG read-only register contains the logical block address used by the System ACE CF controller configuration logic during CompactFlash read/write operations. The CFGLBAREG register affects only transfers between the System ACE CF controller configuration logic and the CompactFlash card. The MPU uses a separate set of registers (MPULBAREG(27:0)) to transfer data to and from the CompactFlash card. Table 13 provides a description of the CFGLBAREG register bits.

Table 13: **CFGLBAREG Register Bit Descriptions**

Bit	Name	Description
0	CFGLBA00	Logical Block Address used during CompactFlash read or write sector commands: each block address points to a sector location which is made up of 512 bytes (i.e., maximum CompactFlash device capacity is up to 128 gigabytes, or 137,438,953,472 bytes)
1	CFGLBA01	
2	CFGLBA02	
3	CFGLBA03	
4	CFGLBA04	
5	CFGLBA05	
6	CFGLBA06	
7	CFGLBA07	
8	CFGLBA08	
9	CFGLBA09	
10	CFGLBA10	
11	CFGLBA11	
12	CFGLBA12	
13	CFGLBA13	
14	CFGLBA14	
15	CFGLBA15	
16	CFGLBA16	
17	CFGLBA17	
18	CFGLBA18	
19	CFGLBA19	
20	CFGLBA20	
21	CFGLBA21	
22	CFGLBA22	
23	CFGLBA23	
24	CFGLBA24	
25	CFGLBA25	
26	CFGLBA26	
27	CFGLBA27	
28	--	Reserved
29	--	Reserved
30	--	Reserved
31	--	Reserved

MPULBAREG Register (BYTE address 10h-13h, WORD address 08h-09h)

The MPULBAREG read-write register contains the logical block address that is used by the MPU interface during CompactFlash read/write operations. The MPULBAREG register affects only transfers between the MPU interface and the CompactFlash card. System ACE CF controller configuration logic maintains a separate set of registers (CFGLBAREG(27:0)) for use when transferring data to and from the CompactFlash card. Table 14 provides a description of MPULBAREG register bits.

Table 14: **MPULBAREG Register Bit Descriptions**

Bit	Name	Description
0	MPULBA00	Logical Block Address used during CompactFlash read or write sector commands: each block address points to a sector location which is made up of 512 bytes (i.e., maximum CompactFlash device capacity is up to 128 gigabytes, or 137,438,953,472 bytes)
1	MPULBA01	
2	MPULBA02	
3	MPULBA03	
4	MPULBA04	
5	MPULBA05	
6	MPULBA06	
7	MPULBA07	
8	MPULBA08	
9	MPULBA09	
10	MPULBA10	
11	MPULBA11	
12	MPULBA12	
13	MPULBA13	
14	MPULBA14	
15	MPULBA15	
16	MPULBA16	
17	MPULBA17	
18	MPULBA18	
19	MPULBA19	
20	MPULBA20	
21	MPULBA21	
22	MPULBA22	
23	MPULBA23	
24	MPULBA24	
25	MPULBA25	
26	MPULBA26	
27	MPULBA27	
28	--	Reserved
29	--	Reserved
30	--	Reserved
31	--	Reserved

SECCNTCMDREG Register (BYTE address 014h-15h, WORD address 0Ah)

The SECCNTCMDREG register provides the means for an MPU interface to set the sector count and execute CompactFlash Controller commands. Table 15 provides a description of the SECCNTCMDREG register bits.

The SECCNT bits of the SECCNTCMDREG register specify the number of sectors to transfer during each ReadMemCardData or WriteMemCardData command:

- A SECCNT value of 1 to 255 indicates to the CompactFlash device that 1 to 255 sectors should be transferred.
- A SECCNT value of 0 indicates that 256 sectors should be transferred.

The CMD bits of the SECCNTCMDREG register identify a specific command to be executed:

- If the MPU has NOT successfully locked access to the CompactFlash Controller, then writes to the CMD bits of the SECCNTCMDREG register do not change the value of the register.
- If the MPU has successfully locked access to the CompactFlash Controller and a non-zero value is written to the CMD bits of the SECCNTCMDREG register, then the specified command is executed by the CompactFlash Controller.
- If the MPU has successfully locked access to the CompactFlash Controller and a zero value is written to the CMD bits of the SECCNTCMDREG register, there is no effect on the value of the CMD bits. The only way to clear the CMD bits is to issue the cfAbort command, which aborts the currently executing command and waits until the CompactFlash Controller clears the CMD bits.

Table 15: **SECCNTCMDREG Register Bit Descriptions**

Bit	Name	Description
0	SECCNT0	Sector Count used during CompactFlash read or write sector commands: each sector is made up of 512 bytes
1	SECCNT1	
2	SECCNT2	
3	SECCNT3	
4	SECCNT4	
5	SECCNT5	
6	SECCNT6	
7	SECCNT7	
8	CMD0	Command value: 0x0 : Reserved 0x1 : ResetMemCard command 0x2 : IdentifyMemCard command 0x3 : ReadMemCardData command 0x4 : WriteMemCardData command 0x5: Reserved 0x6 : Abort command 0x7 : Reserved
9	CMD1	
10	CMD2	
11	--	Reserved
12	--	Reserved
13	--	Reserved
14	--	Reserved
15	--	Reserved

VERSIONREG Register (BYTE address 16h-17h, WORD address 0Bh)

The VERSIONREG register holds the System ACE CF controller version number in the form of a 4-bit major version field, a 4-bit minor version field, and an 8-bit revision/build number field. Table 16 provides a description of the VERSIONREG register bits.

Table 16: **VERSIONREG Register Bit Descriptions**

Bit	Name	Description
0	VERSION0	Revision / build number: MSB is bit 7, LSB is bit 0
1	VERSION1	
2	VERSION2	
3	VERSION3	
4	VERSION4	
5	VERSION5	
6	VERSION6	
7	VERSION7	
8	VERSION8	Minor version number: MSB is bit 11, LSB is bit 8
9	VERSION9	
10	VERSION10	
11	VERSION11	
12	VERSION12	Major version number: MSB is bit 15, LSB is bit 12
13	VERSION13	
14	VERSION14	
15	VERSION15	

CONTROLREG Register (BYTE address 18h-1Bh, WORD address 0Ch-0Dh)

The CONTROLREG register provides the means for the MPU interface to control System ACE CF controller functionality. Table 17 provides a description of the CONTROLREG register bits.

Table 17: **CONTROLREG Register Bit Descriptions**

Bit	Name	Description
0	FORCELOCKREQ	Forces the CompactFlash arbitration logic to grant a lock to the MPU interface based on the value of the LOCKREQ bit of the CONTROLREG register (default is 0): • 0 means do not force MPU lock request (i.e., arbitrate between Configuration Controller and MPU interface) • 1 means force MPU lock request (i.e., do not perform arbitration: grant lock request based only on MPU requests)
1	LOCKREQ	CF arbitration lock request signal; Once a lock is granted, the LOCKREQ must be de-asserted before the lock is removed (default is 0): • 0 means do not request CompactFlash access lock • 1 means request CompactFlash access lock
2	FORCECFGADDR	Forces the overriding of the CFGADDR(1:0) pins in favor of using the CFGADDRBIT(2:0) bits of the CONTROLREG(15:13) register (default is 0): • 0 means use the CFGADDR(1:0) pins • 1 means use the CONTROLREG(15:13) register bits
3	FORCECFGMODE	Forces the overriding of CFGMODEPIN in favor of using the CFGMODE bit of the CONTROLREG register (default is 0): • 0 means use CFGMODEPIN • 1 means use the CFGMODE bit of the CONTROLREG register
4	CFGMODE	Configuration mode (default is 0): • 1 means automatically start the configuration process immediately after System ACE CF controller Reset • 0 means wait for CFGSTART bit in CONTROLREG before starting the configuration process
5	CFGSTART	Configuration start bit (default is 0): • 0 means do not start configuration • 1 means start configuration process
6	CFGSEL	Configuration select (default is 0): • 0 means configure from CompactFlash • 1 means configure from MPU interface
7	CFGRESET	Configuration/CompactFlash controller reset (default is 0): • 0 means do not reset • 1 means reset the Configuration and CompactFlash controllers (this also causes a “soft-reset” of the CompactFlash device)
8	DATABUFRDYIRQ	Data buffer ready IRQ enable (default is 0): • 1 means interrupts are enabled for when data buffer is ready for transfer of data into or out of the buffer • 0 means data buffer ready interrupts are disabled
9	ERRORIRQ	Error IRQ enable (default is 0): • 1 means interrupts are enabled for when an error occurs • 0 means error interrupts are disabled

Table 17: **CONTROLREG Register Bit Descriptions *(Continued)***

Bit	Name	Description
10	CFGDONEIRQ	Configuration DONE IRQ enable (default is 0): • 1 means interrupts are enabled for when configuration is DONE • 0 means configuration DONE interrupts are disabled
11	RESETIRQ	Resets the interrupt request line when a '1' is written to this register bit. Note that a '0' must be written to this register bit in order to re-arm for subsequent interrupt conditions.
12	--	Reserved
13	CFGADDRBIT0	Configuration address register bits that are used as an offset into the system configuration file in the CompactFlash device used to locate the System ACE CF controller configuration data file (note that these register bits can be used to override the CFGADDR[2:0] pins of the System ACE CF controller)
14	CFGADDRBIT1	
15	CFGADDRBIT2	
16	CFGRSVD0	Reserved for future use. These bits must be set to zero at all times.
17	CFGRSVD1	
18	CFGRSVD2	
19	--	Reserved
20	--	Reserved
21	--	Reserved
22	--	Reserved
23	--	Reserved
24	--	Reserved
25	--	Reserved
26	--	Reserved
27	--	Reserved
28	--	Reserved
29	--	Reserved
30	--	Reserved
31	--	Reserved

FATSTATREG Register (BYTE address 1Ch-1Dh, WORD address 0Eh)

The FATSTATREG register contains information about the first valid partition of the CompactFlash device such as the boot record and FAT types found. Table 18 provides a description of the FATSTATREG register bits.

Table 18: **FATSTATREG Register Bit Descriptions**

Bit	Name	Description
0	MBRVALID	Master boot record (MBR) valid flag: • 0 means no MBR was detected • 1 means a valid MBR was found
1	PBRVALID	Partition boot record (PBR) valid flag: • 0 means no PBR was detected • 1 means a valid PBR was found
2	MBRFAT12	Master boot record (MBR) FAT12 flag: • 0 means FAT12 flag is not set in MBR • 1 means FAT12 flag is set in MBR
3	PBRFAT12	Partition boot record (PBR) FAT12 flag: • 0 means FAT12 flag is not set in PBR • 1 means FAT12 flag is set in PBR
4	MBRFAT16	Master boot record (MBR) FAT16 flag: • 0 means FAT16 flag is not set in MBR • 1 means FAT16 flag is set in MBR
5	PBRFAT16	Partition boot record (PBR) FAT16 flag: • 0 means FAT16 flag is not set in PBR • 1 means FAT16 flag is set in PBR
6	CALCFAT12	Calculated FAT12 flag (based on cluster count): • 0 means not FAT12 (cluster count > 4085) • 1 means FAT12 (cluster count < 4085)
7	CALCFAT16	Calculated FAT12 flag (based on cluster count): • 0 means not FAT16 (cluster count > 65525) • 1 means FAT16 (4085 < cluster count < 65535)
8	--	Reserved
9	--	Reserved
10	--	Reserved
11	--	Reserved
12	--	Reserved
13	--	Reserved
14	--	Reserved
15	--	Reserved

DATABUFREG Register (BYTE address 40h-5Fh, WORD address 20h-2Fh)

The DATABUFREG register is the portal register to the data buffer that is used to transfer data between the MPU interface and the CompactFlash and/or Configuration controllers. The description of the DATABUFREG register bits are shown in Table 19.

Table 19: **DATABUFREG Register Bit Descriptions**

Bit	Name	Description
0	DATA00	Data buffer portal register: • Data register bits are read-only when the DATABUFMODE bit in the STATUSREG register is a 0, otherwise they are write-only when the DATABUFMODE bit is a 1. • DATABUFREG(07:00) are accessible in BYTE and WORD bus modes.
1	DATA01	
2	DATA02	
3	DATA03	
4	DATA04	
5	DATA05	
6	DATA06	
7	DATA07	
8	DATA08	Data register: • Data register bits are read-only when the DATABUFMODE bit in the STATUSREG register is a 0, otherwise they are write-only when the DATABUFMODE bit is a 1. • DATABUFREG(15:08) are accessible in BYTE and WORD bus modes. • During BYTE bus write mode, if the data buffer is ready, any writes to the DATABUFREG(15:08) bits cause the DATABUFREG(15:00) contents to be written to the data buffer. • During BYTE bus read mode, if the data buffer is ready, the DATABUFREG(15:00) register will hold the current value until the DATABUFREG(15:08) bits are read. After DATABUFREG(15:08) is read, the DATABUFREG(15:00) register is loaded with any pending new data.
9	DATA09	
10	DATA10	
11	DATA11	
12	DATA12	
13	DATA13	
14	DATA14	
15	DATA15	

Test JTAG Interface (TSTJTAG)

The Test JTAG Interface (TSTJTAG) supports IEEE 1149.1 Boundary-Scan operations on the System ACE CF controller and all chained FPGA devices connected to the Configuration JTAG (CFGJTAG) port. This interface can also be used to program the target FPGA chain on the CFGJTAG port, using Xilinx or third-party JTAG programming tools.

The System ACE CF controller is fully compliant with the IEEE 1149.1 Boundary-Scan standard, commonly referred to as JTAG. As shown in Figure 16, page 35, a Test Access Port (TAP), instruction decoder, and the required IEEE 1149.1 Registers are included in the System ACE CF controller to support the mandatory Boundary-Scan instructions. In addition, the Controller also supports an optional 32-bit identification register. Refer to the IEEE 1149.1 Boundary-Scan standard specification for a complete description of the required instructions and detailed information on JTAG.

Table 20: **System ACE CF Controller TAP Pins**

Pins	Description
TSTTDI (TDI)	Test Data In
TSTTDO (TDO)	Test Data Out
TSTTMS (TMS)	Test Mode Select
TSTTCK (TCK)	Test Clock

When using the TSTJTAG interface as the configuration source, the CFGTCK output of the System ACE CF controller device is derived from the TSTTCK input. The operating frequency of the CFGTCK is the same as TSTTCK.

- The minimum clock operating frequency is 0 MHz.
- The maximum clock operating frequency is either 16.7 MHz or the maximum JTAG TCK clock speed dictated by the devices in the JTAG chain and/or the board design. The lowest of these values should be used.

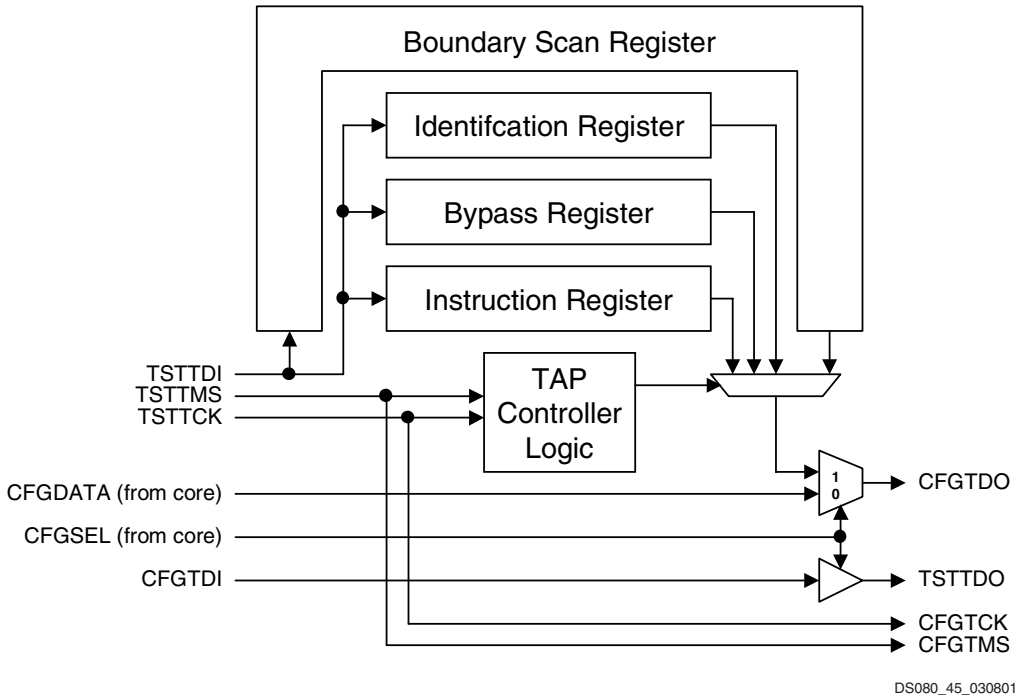


Figure 16: **Test JTAG Interface Block Diagram**

The JTAG signals are directly multiplexed from the respective configuration source. The TSTJTAG logic is connected to the CFGJTAG port as long as the CompactFlash and MPU interfaces are not connected to the CFGJTAG port. Outlined in the following sections are the details of the JTAG interface for the System ACE CF controller.

The available Boundary-Scan registers for the System ACE CF controller are shown in Table 21.

Table 21: **System ACE CF Controller Boundary-Scan Registers**

Register Name	Register Length	Description
Instruction Register	8 bits	Holds current instruction OPCODE and captures internal device status.
Boundary-Scan Register	109 bits	Controls and observes input, output, and output enable.
Identification Register	32 bits	Captures device IDCODE.
Bypass Register	1 bit	Device bypass.

Instruction Register

The Instruction Register (IR) for the System ACE CF controller is eight bits wide and is connected between TDI and TDO during an instruction scan sequence. The Instruction Register is parallel loaded with a fixed instruction capture pattern in preparation for an instruction sequence. This pattern is shifted out onto TDO (LSB first), while an instruction is shifted into the instruction register from TDI. This pattern is illustrated in Table 22.

Table 22: **Instruction Register Values Loaded into IR During Instruction Scan Sequence**

IR[7]	IR[6]	IR[5]	IR[4]	IR[3]	IR[2]	IR[1:0]
CFGINSTRERR (MPU ERRORREG register bit)	CFGFAILED (MPU ERRORREG register bit)	CFGREADERR (MPU ERRORREG register bit)	CFCERROR (MPU STATUSREG register bit)	CFGERROR (MPU STATUSREG register bit)	CFGDONE	01

The optional IDCODE instruction is supported in addition to the mandatory instructions (BYPASS, SAMPLE/PRELOAD, and EXTEST). The binary values for these instructions are listed in Figure 23, page 36.

Table 23: **System ACE CF Controller Boundary-Scan Instructions**

Boundary-Scan Instruction	Binary Code [7:0]	Description
BYPASS	11111111	Enables BYPASS
SAMPLE/PRELOAD	00000001	Enables boundary-scan SAMPLE/PRELOAD Operation
IDCODE	00001001	Enables shifting out 32-bit IDCODE
EXTEST	00000000	Enables boundary-scan EXTEST operation

Boundary-Scan Register

The Boundary-Scan register, which is the primary test data register, is used to control and observe the state of device pins during EXTEST and SAMPLE/PRELOAD instructions. For more information on the System ACE Boundary-Scan register (such as bit sequence, 3-state control, and so forth), refer to the System ACE Boundary-Scan Description Language (BSDL) file available from the software download area at: www.xilinx.com.

Bit Sequence

The bit sequence of the device is obtainable from the Boundary-Scan Description Language (BSDL) Files. These files are available from the software download area at: www.xilinx.com.

Identification Register

The Identification Register known as the IDCODE is a fixed, vendor-assigned value that is used to electronically identify the type of device and the manufacturer for a specific device being tested. The System ACE CF controller IDCODE register is 32 bits wide. The contents of this register can be shifted out for examination by selecting the IDCODE instruction. The IDCODE is available to any other system component via JTAG. The IDCODE register has the following binary format, described in Table 24.

Table 24: **System ACE CF Controller Identification Register**

Version	Family	Array Size	Manufacturer	Required by IEEE 1149.1
0000	0000001	00000000	00001001001	1

Bypass Register

The last standard 1149.1 Boundary-Scan data register in the System ACE CF controller is the single flip-flop BYPASS register. It directly passes data serially from the TDI pin to the TDO pin during a bypass instruction. This register is initialized to zero when the TAP controller is in the UPDATE-DR state.

TAP Timing Characteristics

IEEE 1149.1 boundary-scan (JTAG) testing is performed via the standard 4-wire Test Access Port (TAP). The Boundary Scan timing waveforms and switching characteristics of the TAP are described in Figure 17 and Table 25, respectively.

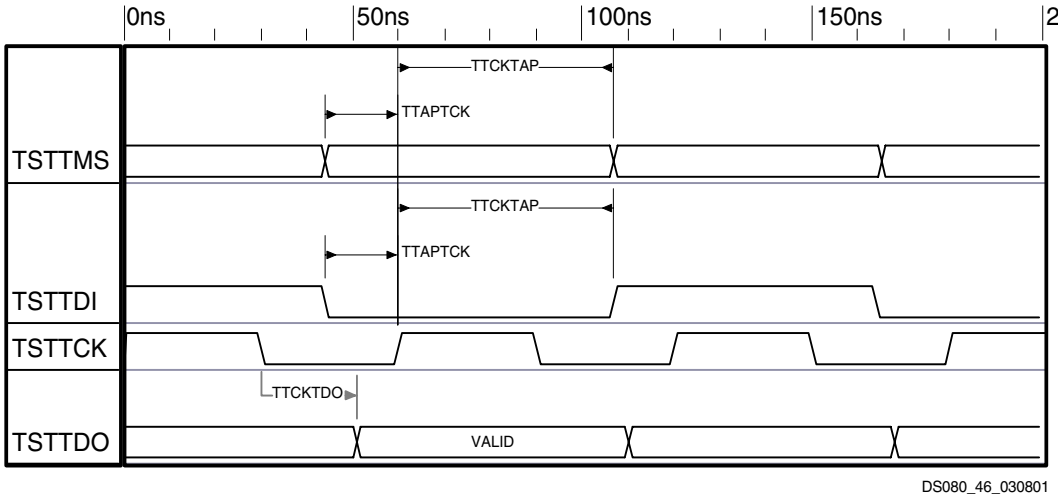


Figure 17: **Test JTAG Boundary-Scan Port Timing Waveforms**

Table 25: **System ACE CF Controller TAP Characteristics**

Symbol	Parameter	Min	Max	Units
$T_{(TAPTCK)}$	TSTTMS and TSTTDI setup time before rising edge of TSTTCK	4		ns
$T_{(TCKTAP)}$	TSTTMS and TSTTDI hold times after TSTTCK	4		ns
$T_{(TCKTDO)}$	TSTTCK falling edges to TSTTDO output valid		16	ns
$F_{(TSTTCK)}$	Maximum TSTTCK clock frequency		16.7	MHz

Configuration JTAG Interface (CFGJTAG)

Configuration JTAG Port is the interface between the System ACE CF controller and the target FPGA chain. This port is accessed when configuring the target FPGA chain of devices via any of the System ACE CF controller interfaces (Test JTAG, MPU, or CompactFlash). To program or test the FPGA target chain, the data from these interfaces is converted to IEEE 1149.1 Boundary-Scan (JTAG) serial data.

Typical Configuration Modes

The four System ACE CF controller interfaces are designed to work together in a number of different combinations. This section discusses typical user configuration modes. A handful of signals determine which interface provides the configuration data source. Table 26 describes these important signals, and Table 27 shows how they work together to determine which interface will be used. This is especially important when using multiple interfaces in a design, or when not using the default values of these signals. The default values of these signals set the CompactFlash interface as the source of configuration data.

Table 26: **Configuration Signals Used for Selecting Configuration Modes and Active Design**

Configuration Signal	Description	Default
CFGMODE	Pin or MPU register bit	CFGMODEPIN = 1 CFGMODE Register Bit = 0
CFGADDR[2:0]	Pins or MPU register bits	0
CFGSEL	MPU register bit	0
CFGSTART	MPU register bit	0
CFGRESET	MPU register bit (CFGRESET is a subset of the $\overline{\text{RESET}}$ pin)	0
FORCECFGADDR	MPU register bit (Overrides value on CFGADDR [2:0] pins)	0
FORCECFGMODE	MPU register bit (Overrides value on CFGMODEPIN)	0

Table 27: **Active Configuration Modes**

Configuration Interface	CFGMODE(1)	CFGSEL	CFGSTART	CFGRESET
CompactFlash (Configure from CF immediately after CFGRESET)	1	0	X(2)	0
CompactFlash (Configure from CF after receiving MPU start signal)	0	0	1	0
Microprocessor (Configure from MPU after receiving MPU start signal)	1	1	1	0
Microprocessor (Configure from MPU)	1	1	X	0
Test JTAG (Configure using the TSTJTAG port)(3)	X	X	X	X

Notes:
1. The FORCECFGMODE bit in the CONTROLREG register of the MPU interface can be used to force the CFGMODE register bit to override the System ACE CF controller CFGMODEPIN.
2. An X entry indicates “don’t care”.
3. The Test JTAG configuration mode is active regardless of the pin settings as long as none of the other configuration modes are in operation.

CompactFlash (CF) to Configuration JTAG (CFGJTAG) Setup

This setup provides a standard CompactFlash interface for high-density FPGA systems. The CompactFlash interface is the source of configuration data. The data configures the Xilinx FPGA chain through Boundary-Scan (JTAG) using the Configuration JTAG port, as shown in Figure 18.

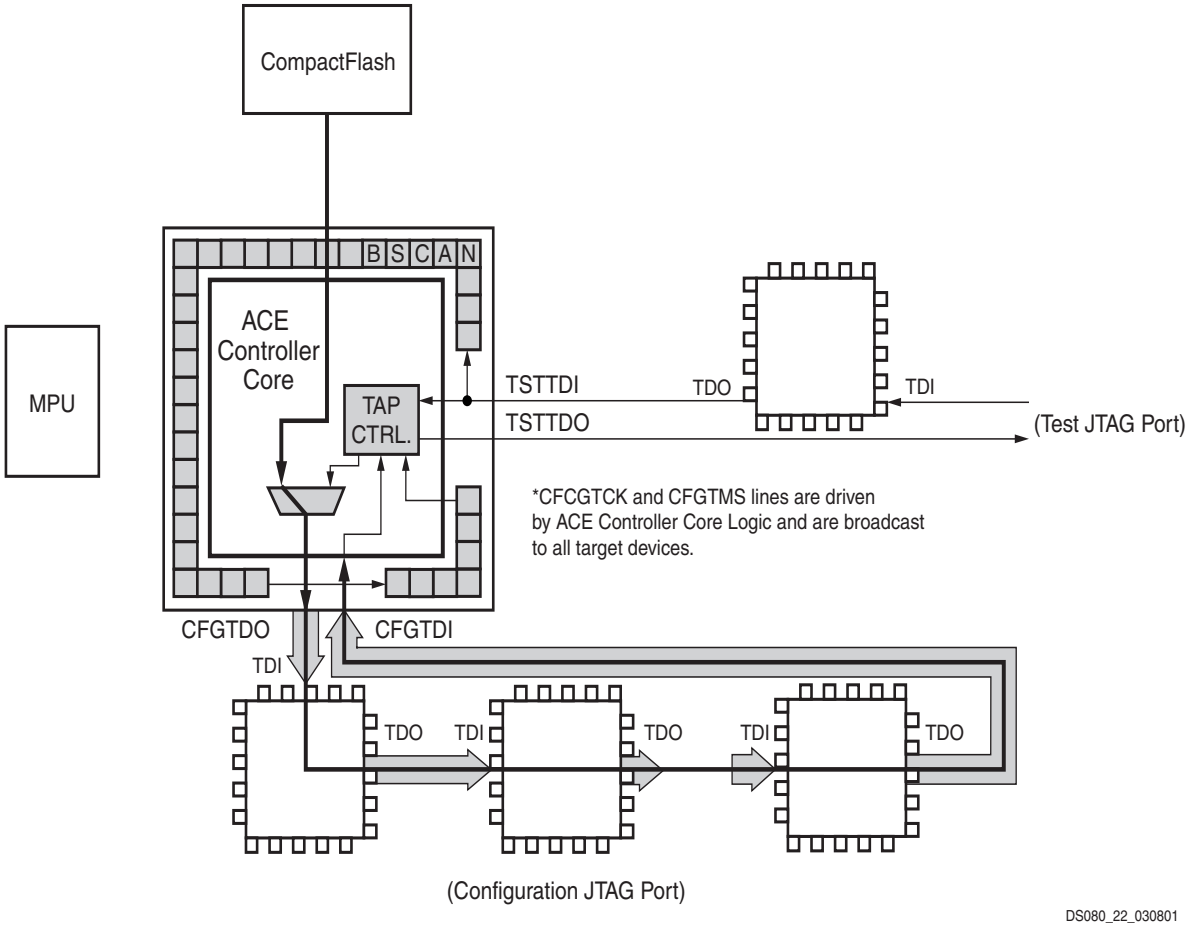


Figure 18: **Data Flow Diagram of CF to CFGJTAG**

The System ACE CF controller handles all necessary steps to perform configuration from the CF to the target system. The appropriate signal connections for this setup are shown in Figure 19, page 39. This setup can be used in conjunction with any of the other interfaces.

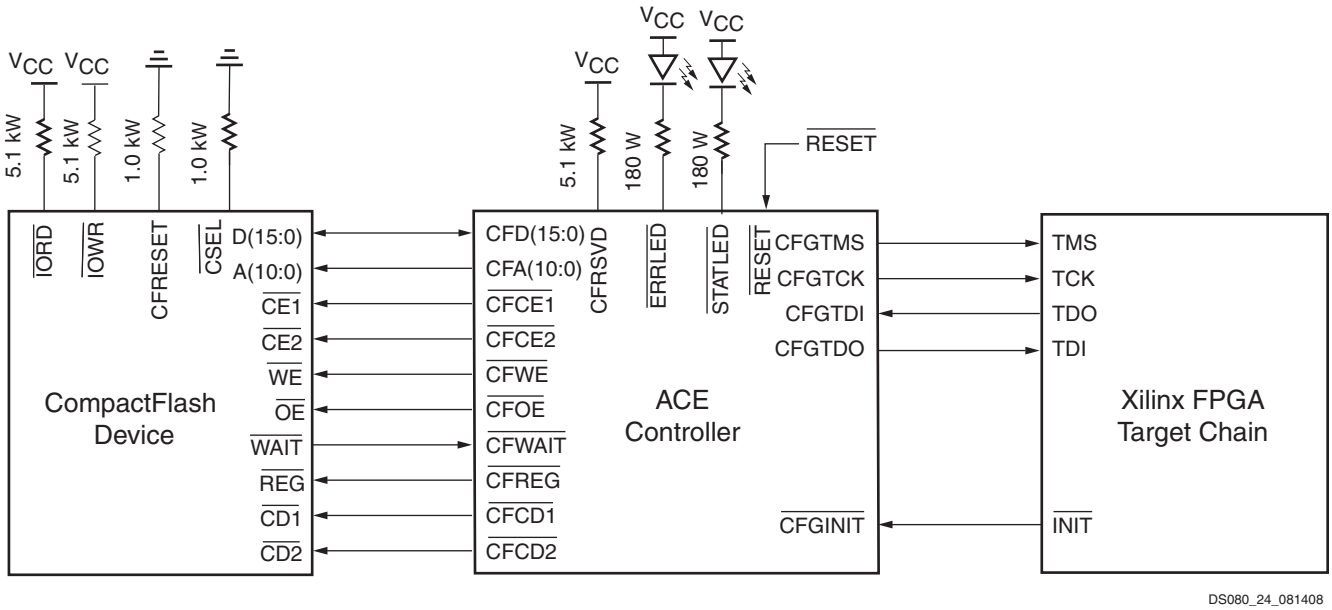


Figure 19: **Wiring Diagram for CF to CFGJTAG**

CompactFlash (CF) to Microprocessor (MPU) Setup

This setup provides a standard CompactFlash to MPU interface for high-density FPGA systems. The ability to communicate with the CF through the MPU port allows the user to perform many operations, such as being able to access application data or microprocessor programming information from the CompactFlash device.

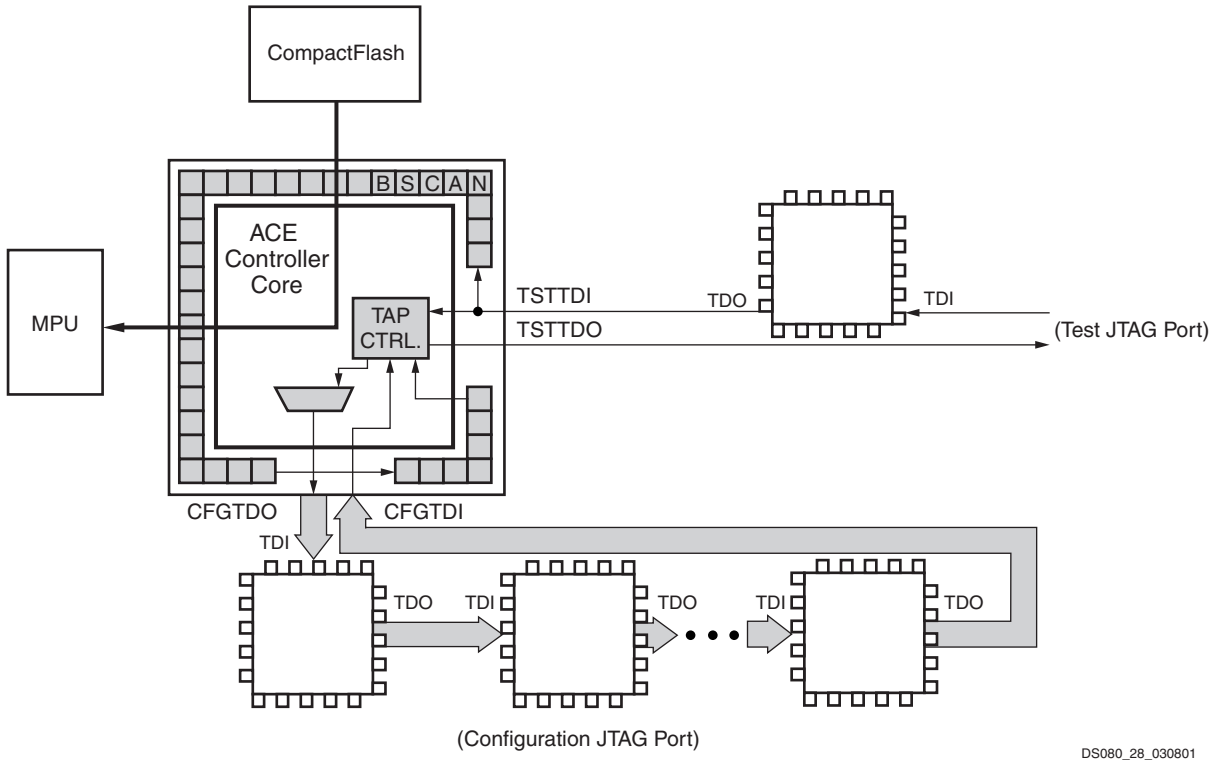


Figure 20: **Data Flow Diagram of CF to MPU**

The System ACE CF controller handles all necessary steps to perform configuration from the CF to the target system. The appropriate signal connections for this setup are shown in Figure 19. This setup can be used in conjunction with any of the other interfaces.

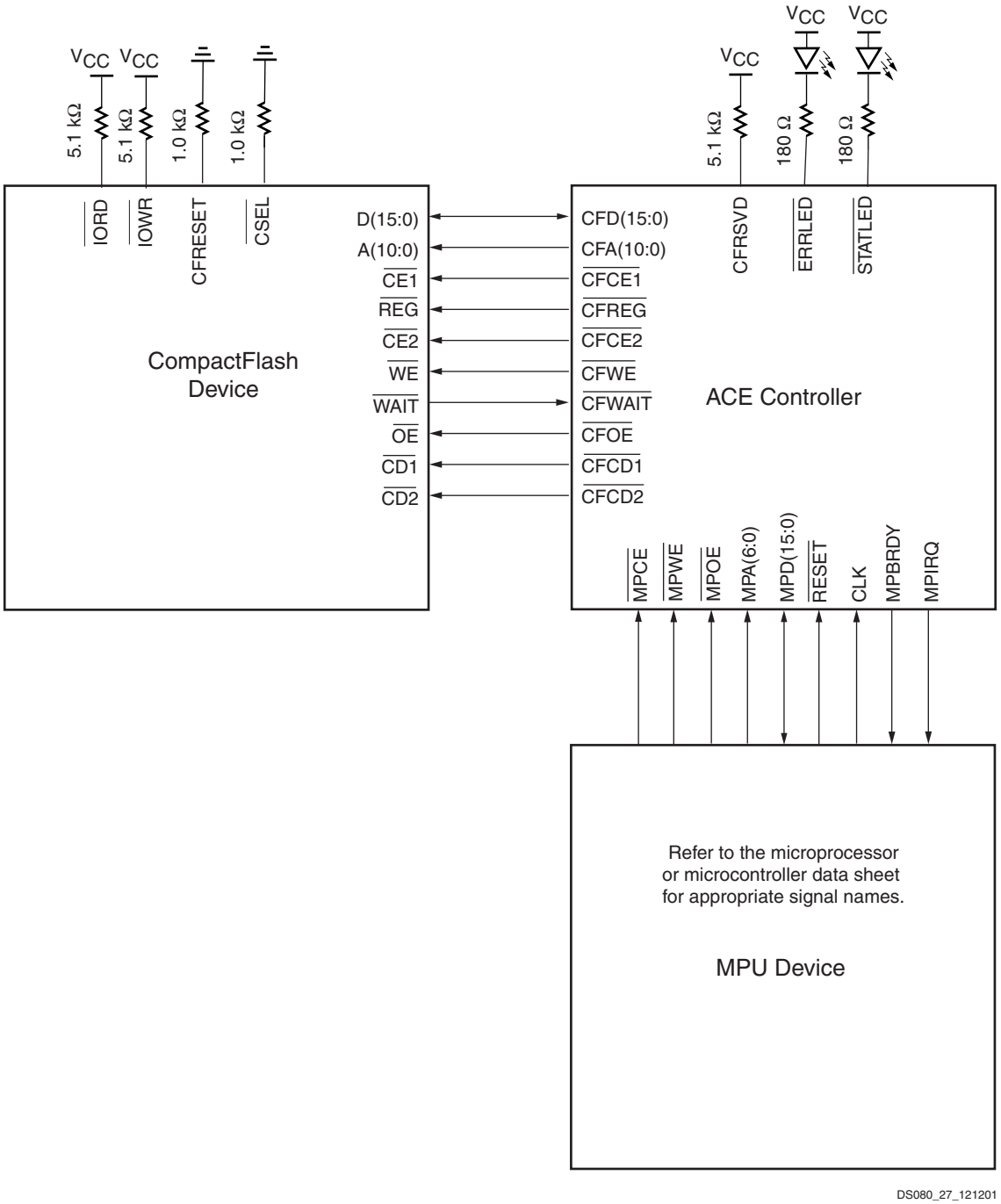


Figure 21: **Wiring Diagram CF to MPU**

Reading Sector Data from CompactFlash Control Flow Process

Sector data can be read from the CompactFlash device via the MPU interface of the System ACE CF controller by following the control flow sequence shown in Figure 22. The first step in the sequence of accessing the CompactFlash interface is to arbitrate for a lock. The control flow process for obtaining a CompactFlash resource lock is shown in Figure 23, page 43. Once the MPU interface has been granted a CompactFlash lock, the MPU interface needs to make sure that the CompactFlash device is ready to receive a command. The process for polling the command readiness indicator is shown in Figure 24, page 44.

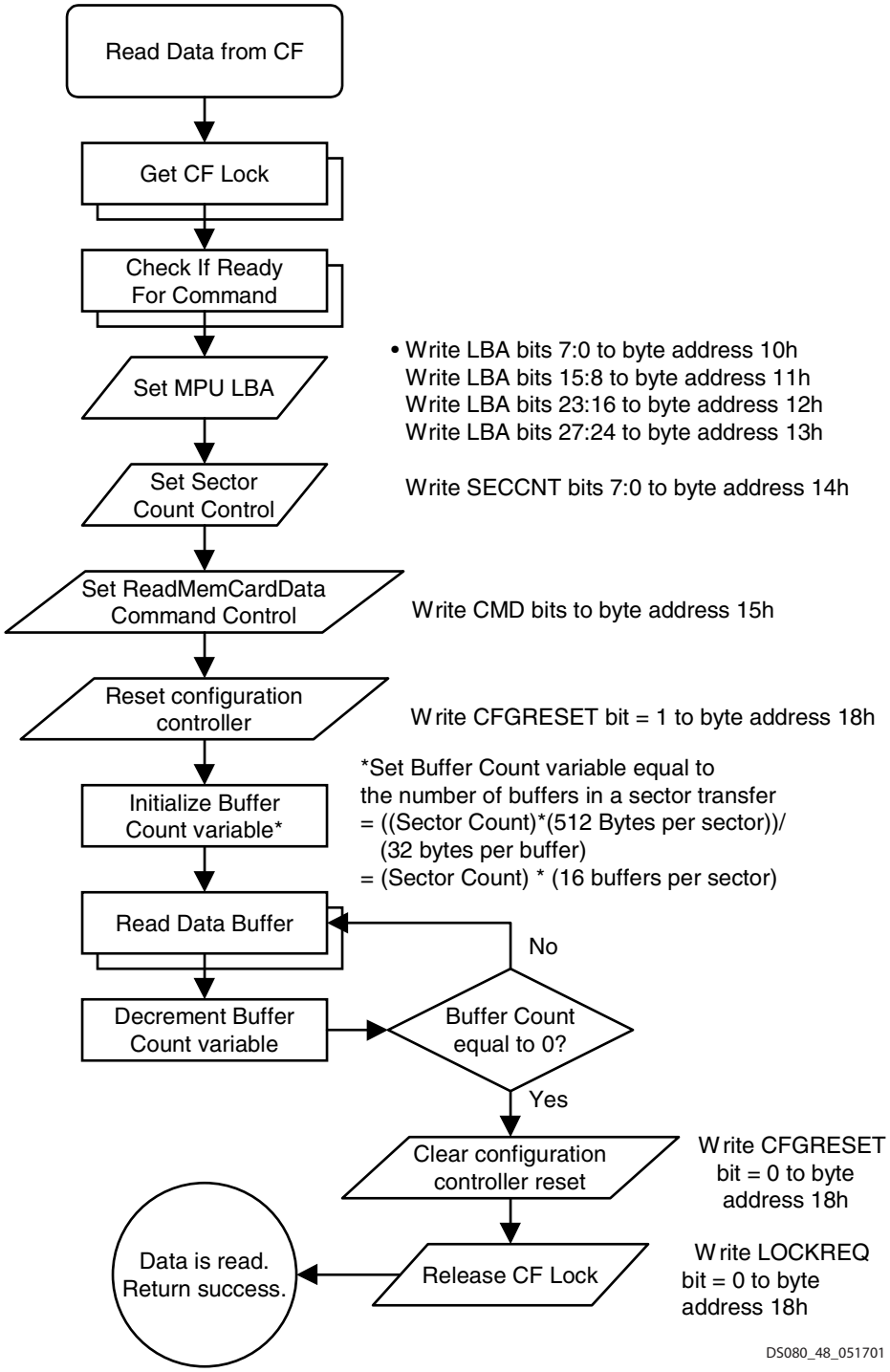


Figure 22: **Reading Sector Data from CompactFlash Control Flow Process**

Once the CompactFlash device is ready to receive a new command, the following information needs to be written to the MPU interface:

1. The sector address or logical block address (LBA) of the first sector to be transferred should be written to the following MPU address locations:
   - LBA[7:0] @ MPU byte address 10h
   - LBA[15:8] @ MPU byte address 11h
   - LBA[23:16] @ MPU byte address 12h
   - LBA[27:24] @ MPU byte address 13h (note that only four bits are used in the most significant LBA byte)
2. The number of sectors to be read should be written to the low byte of the SECCNTCMDREG register (MPU byte address 14h)
3. The ReadMemCardData command (03h) should be written to the high byte of the SECCNTCMDREG register (MPU byte address 15h)
4. Reset the CFGJTAG controller by setting the CFGRESET bit (bit 7) of the CONTROLREG register (MPU address 18h) to a 1.

Immediately after writing the command to the MPU interface, the CFGJTAG controller should be reset before reading the sector data from the data buffer.

The control flow process for reading the sector data from the data buffer is shown in Figure 25, page 45.

After all of the requested sector data has been read, the CFGJTAG controller should be taken out of reset and the CompactFlash lock should be released by setting the LOCKREQ bit (bit 1) and CFGRESET bit (bit 7) of the low byte of the CONTROLREG register (MPU byte address 18h) to a 0. Note that all requested sector data should be read from the data buffer in order to avoid a deadlock situation with the CompactFlash device.

Get CompactFlash Lock Control Flow Process

The CompactFlash resource must be arbitrated for before it can be accessed via the MPU interface. The CompactFlash arbitration process is shown in Figure 23. A CompactFlash lock is requested by setting the LOCKREQ bit (bit 1) to a 1 in the CONTROLREG register (MPU address 18h) and polling the MPULOCK bit (bit 1) in the STATUSREG register (MPU byte address 04h).

Note that if the CFGLOCK bit (bit 0) in the STATUSREG register (MPU byte address 04h) is set, then the CFGJTAG controller has locked the CompactFlash resource. In this case, the MPU interface must either wait for the CFGJTAG interface to release the lock or it can force the lock to be released. This is done by resetting the CFGJTAG controller by setting the CFGRESET bit (bit 7) and the FORCELOCKREQ bit (bit 0) in the CONTROLREG register (MPU byte address 18h). The lock request process can be started again after forcing the CFGJTAG controller to release the lock.

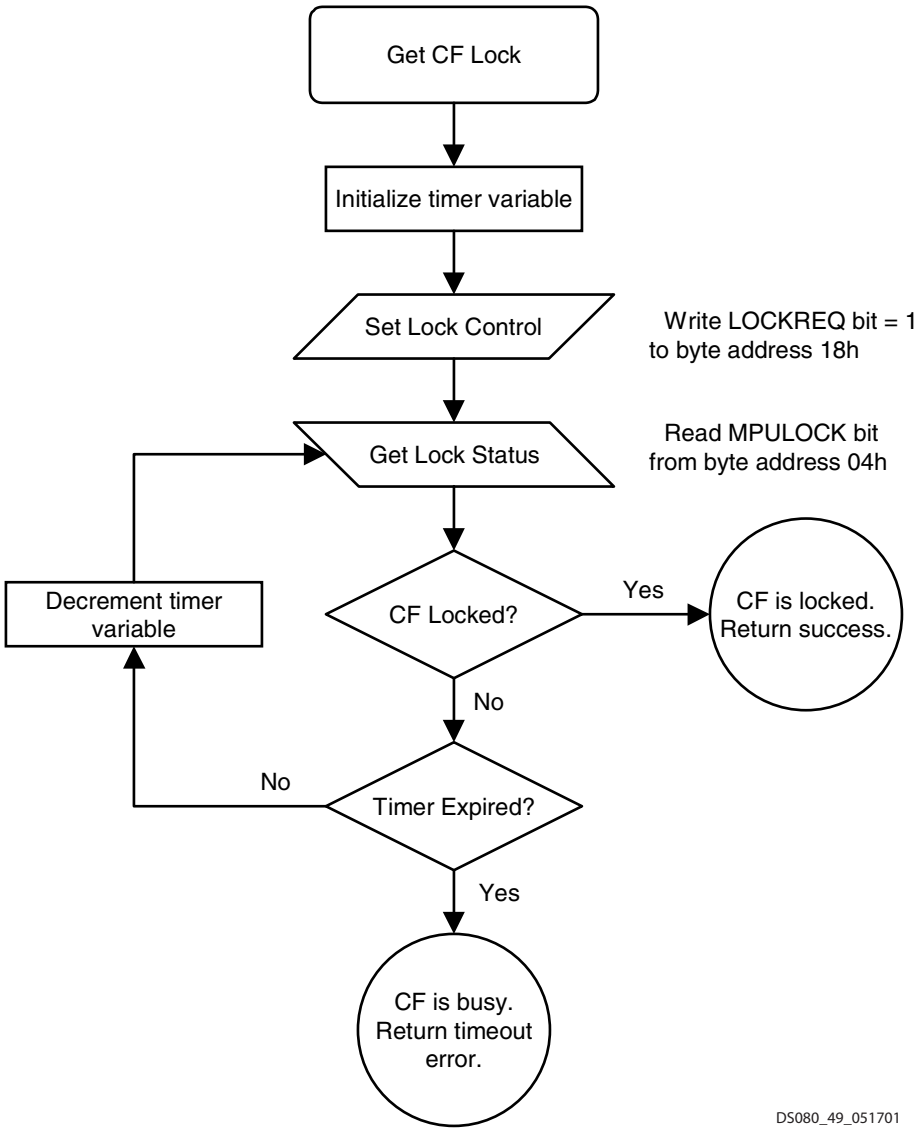


Figure 23: **Get CompactFlash Lock Control Flow Process**

Check if Ready for a Command Control Flow Process

Before reading or writing sector data, it is important to make sure that the CompactFlash device is ready for a command. This is done by polling the RDYFORCFCMD bit (bit 0) in the second byte of the STATUSREG register (MPU byte address 05h) until it is set to a 1. This control flow process is shown in Figure 24.

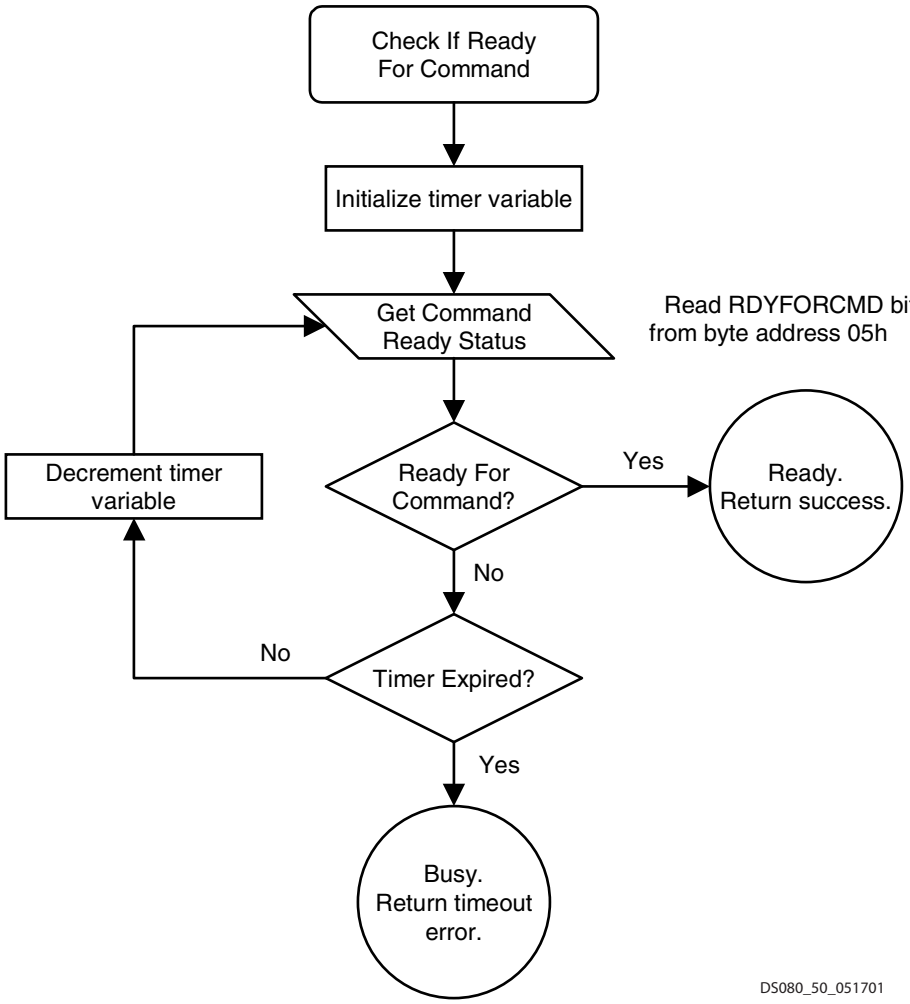


Figure 24: **Check if Ready for a Command Control Flow Process**

Read Data Buffer Control Flow Process

The control flow process for reading from the data buffer is shown in Figure 25. The System ACE data buffer is implemented as a 32-byte (16-word) deep FIFO that is aliased across a range of MPU byte addresses (40h through 7Fh) in order to facilitate burst transfers across the MPU interface. Sector data is read from the data buffer by first waiting for the buffer to become ready (i.e., full of sector data), as shown in Figure 26, page 46. Once the buffer is ready, then all 32 bytes can be read from the buffer from alternating even and odd byte addresses. Reading from an odd byte address while in BYTE mode causes the FIFO to increment the data word to the next available word in the FIFO. Reading from any data buffer address while in WORD mode will cause the FIFO to increment.

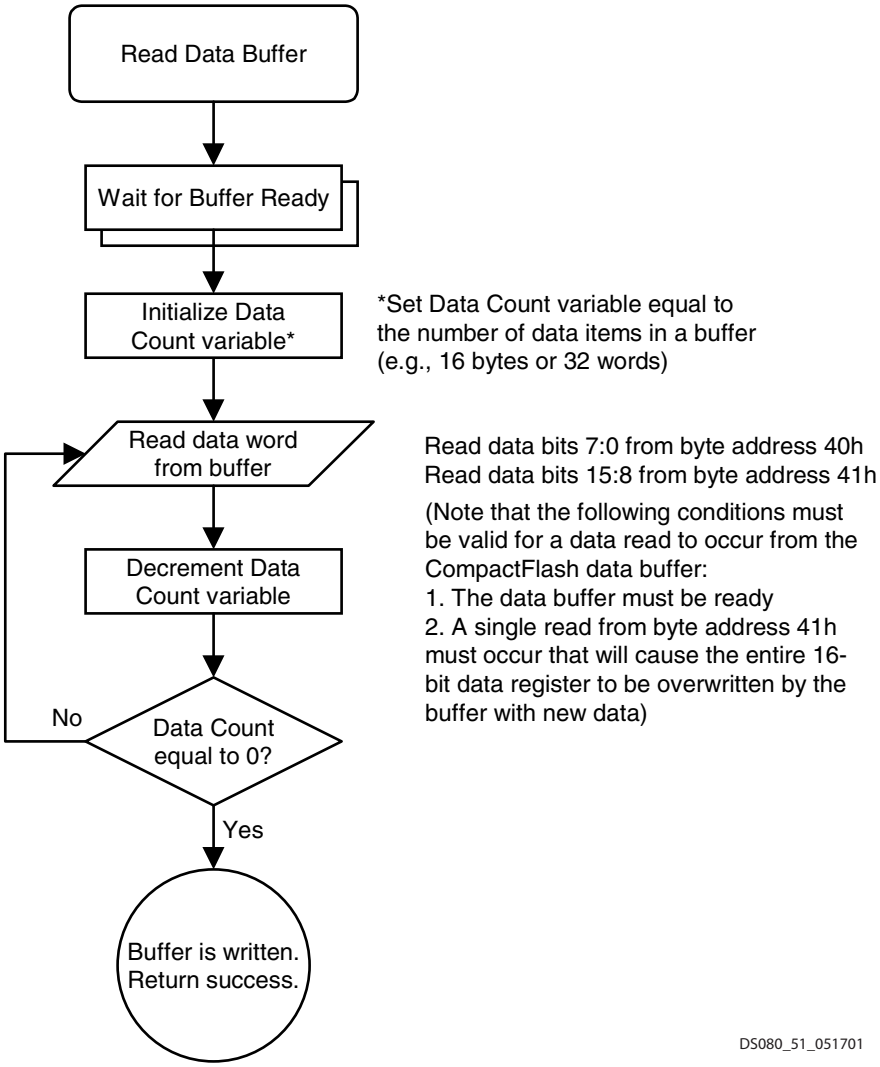


Figure 25: **Read Data Buffer Control Flow Process**

Wait for Buffer Ready Control Flow Process

The readiness of the System ACE data buffer indicates that the buffer is either full during a ReadMemCardData command execution or empty during a WriteMemCardData command execution. The control flow process for waiting for the buffer to become ready is shown in Figure 26. The buffer ready status can be obtained from either the DATABUFRDY bit (bit 5) of the STATUSREG register (MPU byte address 04h) or from the MPBRDY pin of the System ACE CF controller.

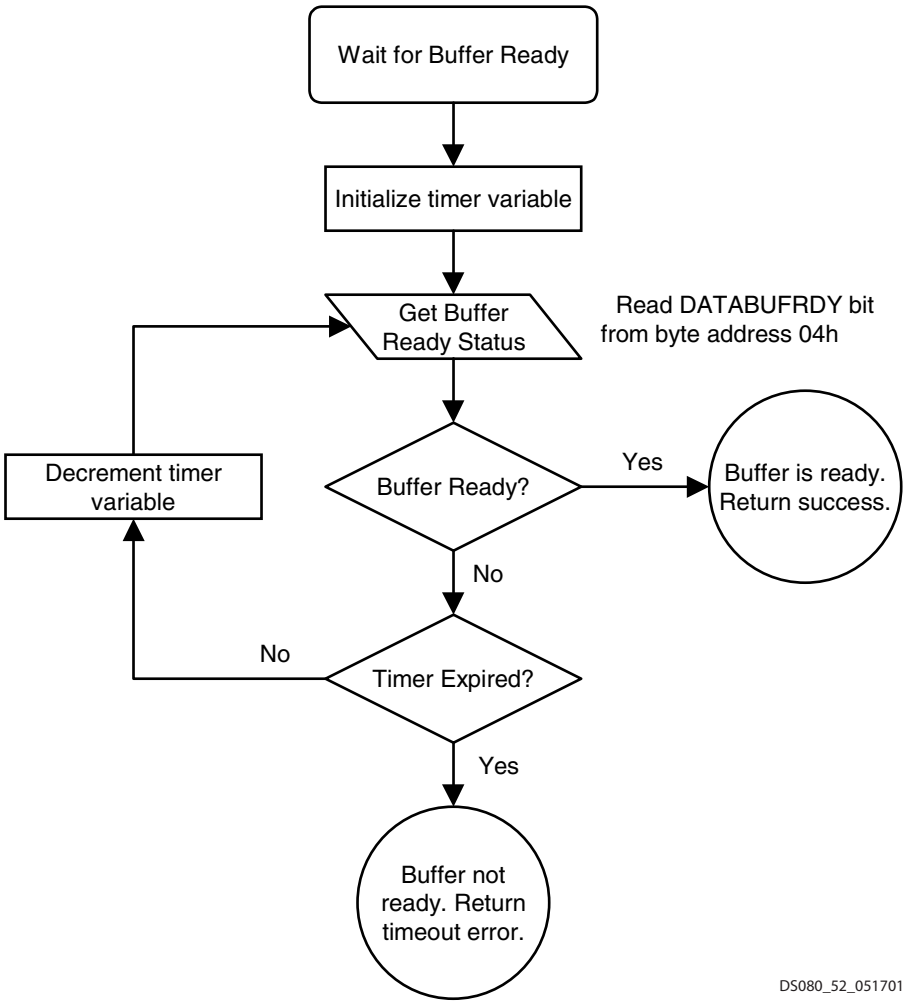


Figure 26: **Wait for Buffer Ready Control Flow Process**

Microprocessor (MPU) to CompactFlash (CF) Setup

This setup provides a communication path from the MPU to the CF device (Figure 27). The CompactFlash is the source of the configuration data, and this path enables users to read the contents of the CF device.

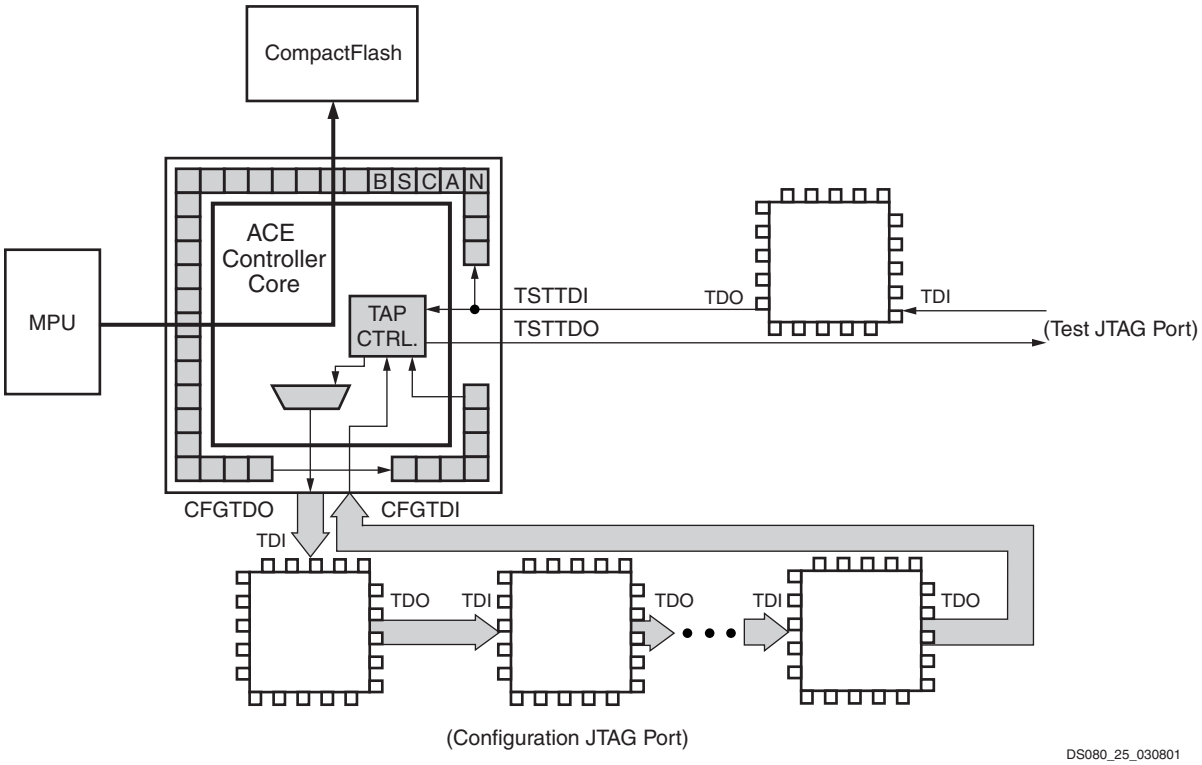


Figure 27: **Data Flow Diagram of MPU to CF**

The System ACE CF controller handles all necessary steps to perform an MPU to CF operation. The necessary signals for this setup are shown in Figure 21, page 40.

Writing Sector Data to CompactFlash Control Flow Process

Sector data can be written to the CompactFlash device via the MPU interface of the System ACE CF controller by following the control flow sequence shown in Figure 28. The first step in the sequence of accessing the CompactFlash interface is to arbitrate for a lock. The control flow process for obtaining a CompactFlash resource lock is shown in Figure 23, page 43. Once the MPU interface has been granted a CompactFlash lock, the MPU interface needs to make sure that the CompactFlash device is ready to receive a command. The process for polling the command readiness indicator is shown in Figure 24, page 44.

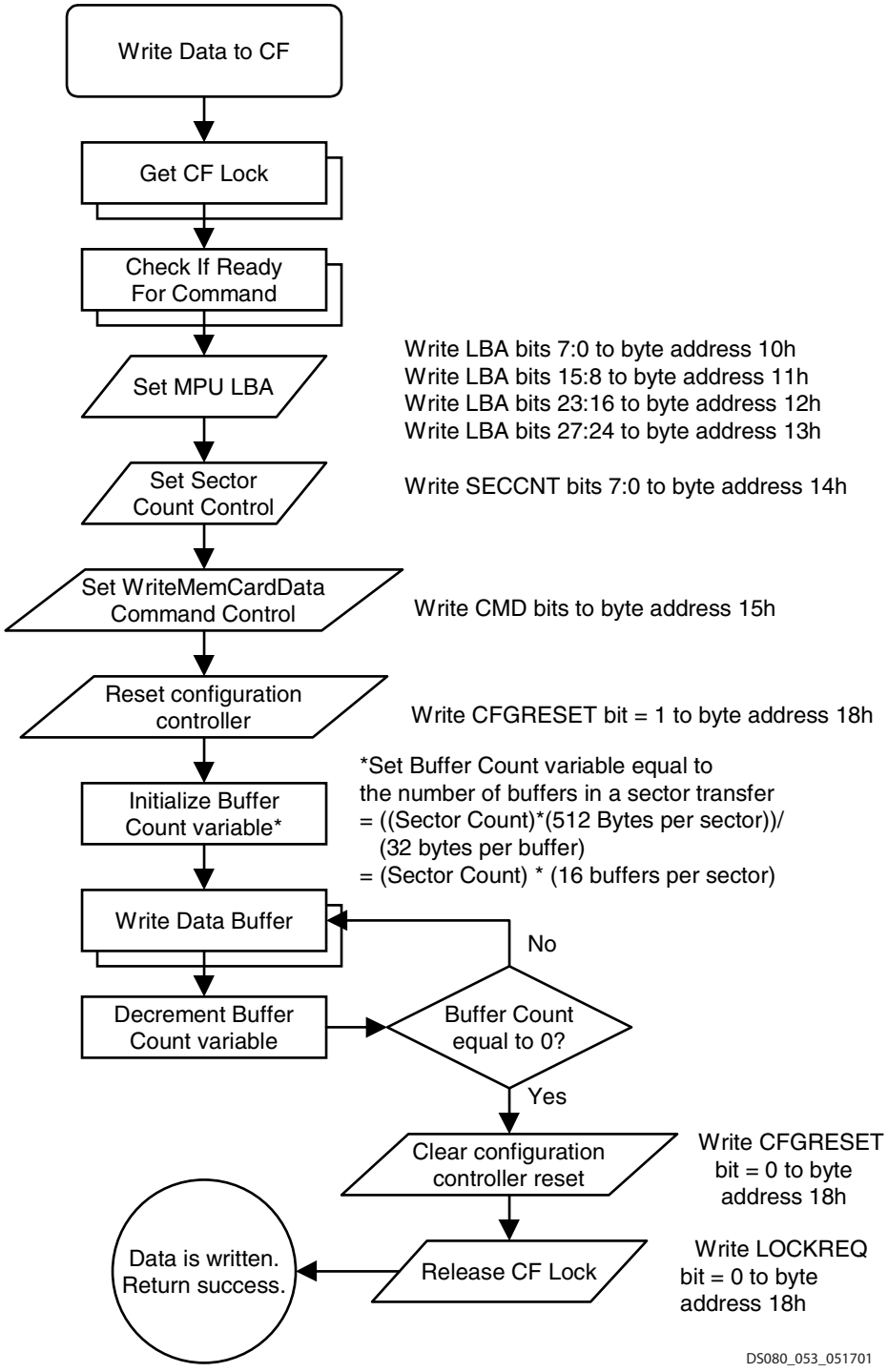


Figure 28: **Write Data to CompactFlash Control Flow Process**

Once the CompactFlash device is ready to receive a new command, the following information needs to be written to the MPU interface:

1. The sector address or logical block address (LBA) of the first sector to be transferred should be written to the following MPU address locations:
   - LBA[7:0] @ MPU byte address 10h
   - LBA[15:8] @ MPU byte address 11h
   - LBA[23:16] @ MPU byte address 12h
   - LBA[27:24] @ MPU byte address 13h (note that only four bits are used in the most significant LBA byte)
2. The number of sectors that will be written should be loaded into the low byte of the SECCNTCMDREG register (MPU byte address 14h)
3. The WriteMemCardData command (04h) should be written to the high byte of the SECCNTCMDREG register (MPU byte address 15h)
4. Reset the CFGJTAG controller by setting the CFGRESET bit (bit 7) of the CONTROLREG register (MPU address 18h) to a 1.

Immediately after writing the command to the MPU interface, the CFGJTAG controller should be reset before writing the sector data to the data buffer.

The control flow process for writing the sector data from the data buffer is shown in Figure 29.

After all of the required sector data has been written, the CFGJTAG controller should be taken out of reset and the CompactFlash lock should be released. This is done by setting the CFGRESET (bit 7) and LOCKREQ (bit 1) bits of the low byte of the CONTROLREG register (MPU byte address 18h) to a 0, respectively. Note that all requested sector data should be written to the data buffer in order to avoid a deadlock situation with the CompactFlash device.

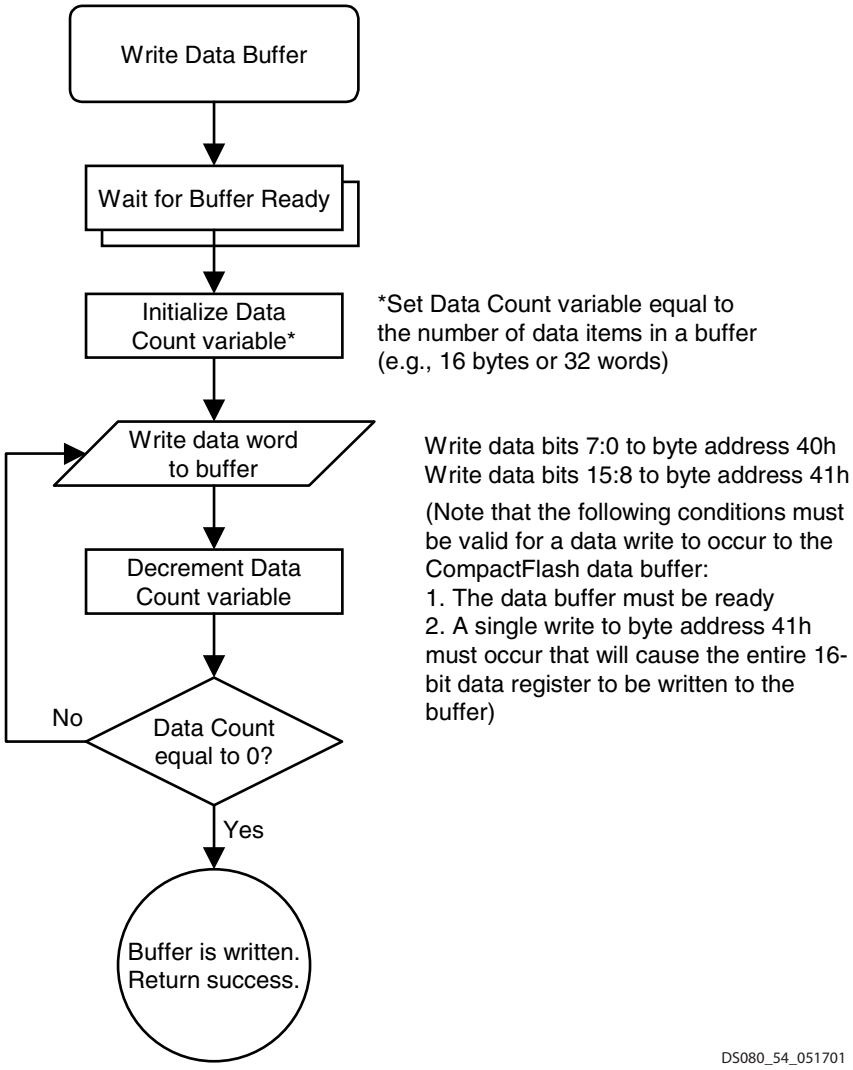


Figure 29: **Write Data Buffer Control Flow Process**

Write Data Buffer Control Flow Process

The control flow process for writing to the data buffer is shown in Figure 29, page 49. The System ACE data buffer is implemented as a 32-byte (16-word) deep FIFO that is aliased across a range of MPU byte addresses (40h through 7Fh) in order to facilitate burst transfers across the MPU interface. Sector data is written to the data buffer by first waiting for the buffer to become ready (i.e., empty of any sector data), as shown in Figure 26, page 46. Once the buffer is ready, then all 32 bytes can be written to the buffer to alternating even and odd byte addresses. Writing to an odd byte address while in BYTE mode causes the FIFO to increment the data word to the next available word in the FIFO. Writing to any data buffer address while in WORD mode will cause the FIFO to increment.

Microprocessor (MPU) to Configuration JTAG (CFGJTAG) Setup

This setup provides an MPU to CFGJTAG communication path. The data configures the FPGA system through JTAG via the Configuration JTAG Port.

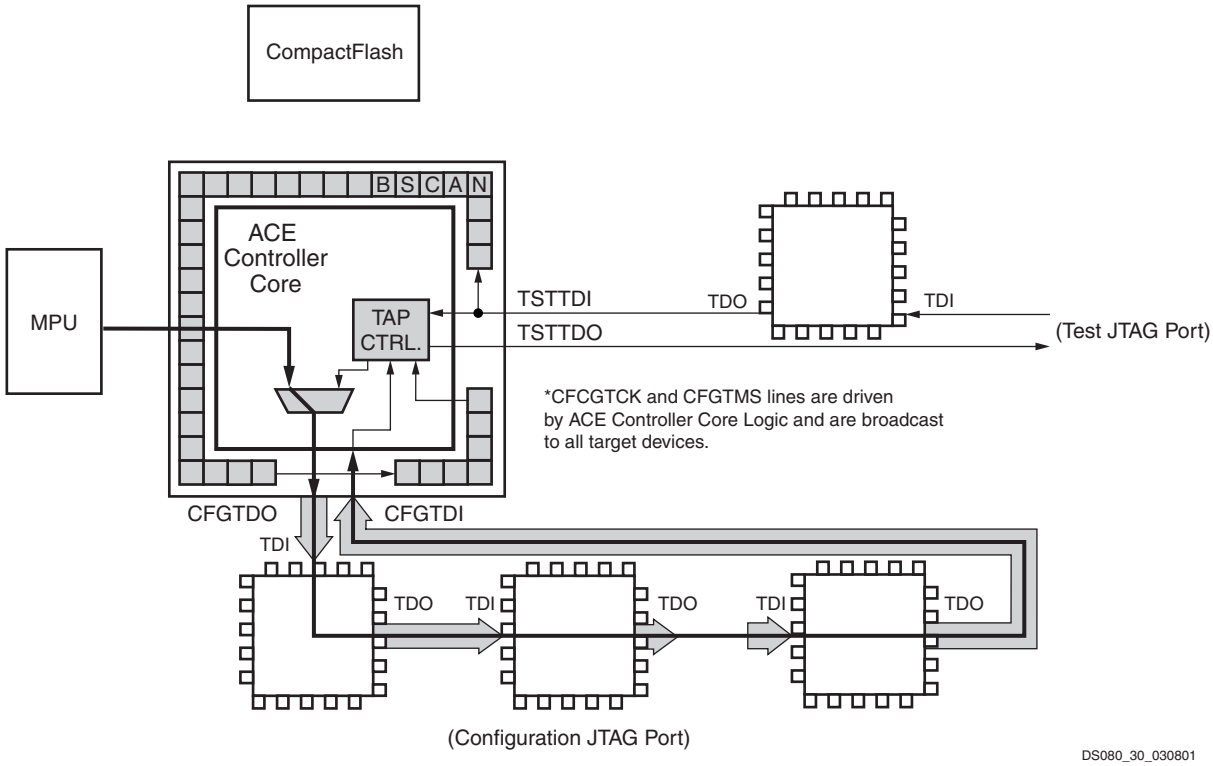


Figure 30: **Data Flow Diagram of MPU to CFGJTAG**

The System ACE CF controller handles all necessary steps to perform configuration using the MPU communication path to the target system. Figure 31, page 51 shows the connections required for this setup.

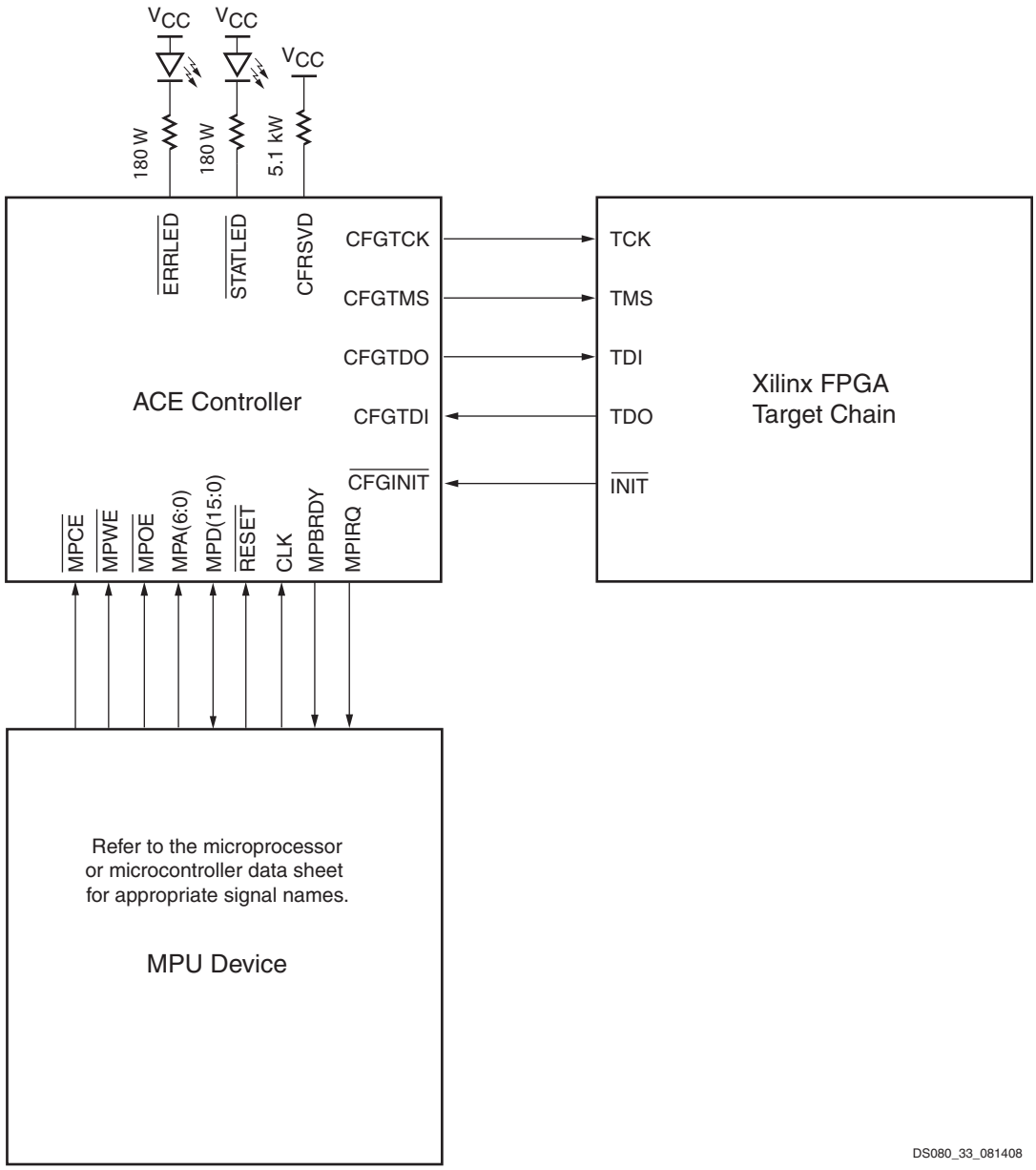


Figure 31: **Wiring Diagram of MPU to CFGJTAG**

Write Data to CFGJTAG Interface Control Flow Process

The target devices in the CFGJTAG chain can also be programmed via the MPU interface as shown in Figure 32, page 53. The following steps should be taken to write configuration data to the CFGJTAG controller:

1. Arbitrate for the data buffer by requesting a CompactFlash lock as shown in Figure 23, page 43. Once the lock has been granted, go to step 2.
2. Put the CFGJTAG controller into the reset state by setting CFGRESET=1 (bit 7 of the CONTROLREG register, MPU byte address 18h).
3. Direct the CFGJTAG controller to wait for CFGSTART=1 to begin configuration by setting FORCECFGMODE=1 (bit 3 of the CONTROLREG register, MPU byte address 18h) and CFGMODE=0 (bit 4 of the CONTROLREG register, MPU byte address 18h).
4. Directs the CFGJTAG controller to start receiving ACE configuration information from the MPU port when CFGRESET is released by setting CFGSTART=1 (bit 5 of the CONTROLREG register, MPU byte address 18h) and CFGSEL=1 (bit 6 of the CONTROLREG register, MPU byte address 18h).
5. Release the CFGJTAG controller from the Reset state and cause it to wait for ACE configuration data from the MPU port by setting CFGRESET=0 (bit 7 of the CONTROLREG register, MPU byte address 18h).
6. Initialize the Buffer Count variable.
7. Perform the Write Data Buffer process. All ACE file information should be sent with the exception of the first 512 bytes of the file. Note that an entire buffer's worth of data should be written to the buffer to ensure that it gets sent to the CFGJTAG controller.

Notes:

Note: The first 512 bytes of the ACE file comprise a comment header and do not contain valid ACE instructions and therefore should not be written to the CFGJTAG controller via the MPU port. The configuration engine does this automatically when processing the ACE file from CF, but it does not do this for ACE information coming from the MPU port.Failure to strip off the first 512 bytes will result in CFGFAILED=1 and CFGINSTRERR=1 in the ERRORREG register.

8. Decrement the Buffer Count variable.
9. Check configuration status. If a configuration error exists, stop writing data to the MPU port and return the error condition. If no error, go to step 10.
10. Check the Buffer Count variable. If Buffer Count is not 0, go back to step 7. If Buffer Count is 0, then go to step 11.
11. Check to see if the configuration process has completed successfully by checking for CFGDONE=1 (bit 7 of the STATUSREG register, MPU byte address 04h). If this is not the case, then other bits of the STATUSREG and ERRORREG register should indicate the status of the configuration process. If CFGDONE=1, then go to step 12.
12. Set CFGRESET=1 (bit 7 of the CONTROLREG register, MPU byte address 18h) and CFGSTART=0 (bit 5 of the CONTROLREG register, MPU byte address 18h). This puts the configuration engine into the Reset state and directs it not to start again if CFGRESET is subsequently released.

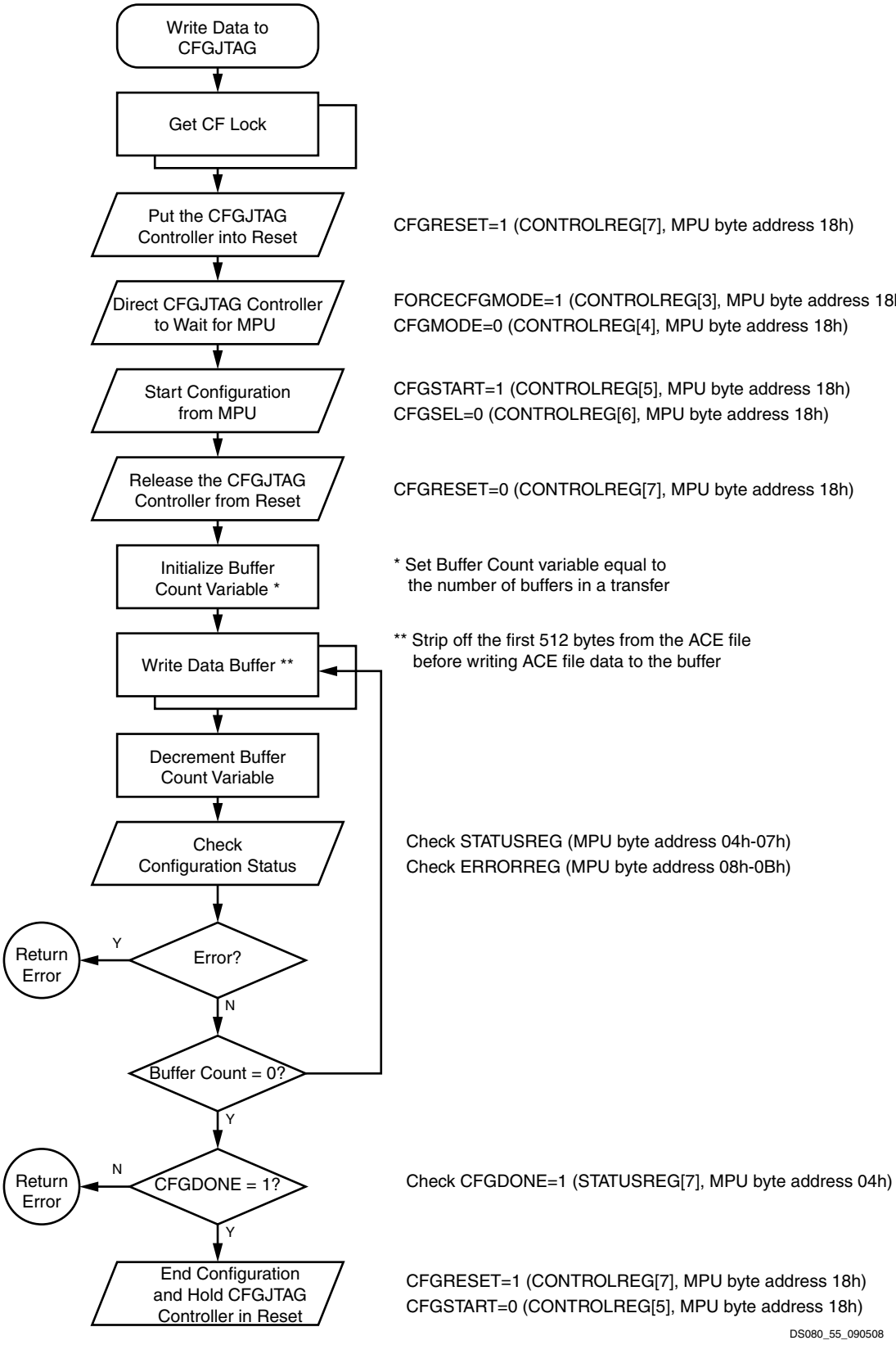


Figure 32: **Write Data to CFGJTAG Interface Control Flow Process**

Test JTAG (TSTJTAG) to Configuration JTAG (CFGJTAG) Setup

This setup provides a 1149.1 Boundary-Scan communication path to the target FPGA system. Using this setup, the target system can be configured via JTAG from a JTAG compliant tool.

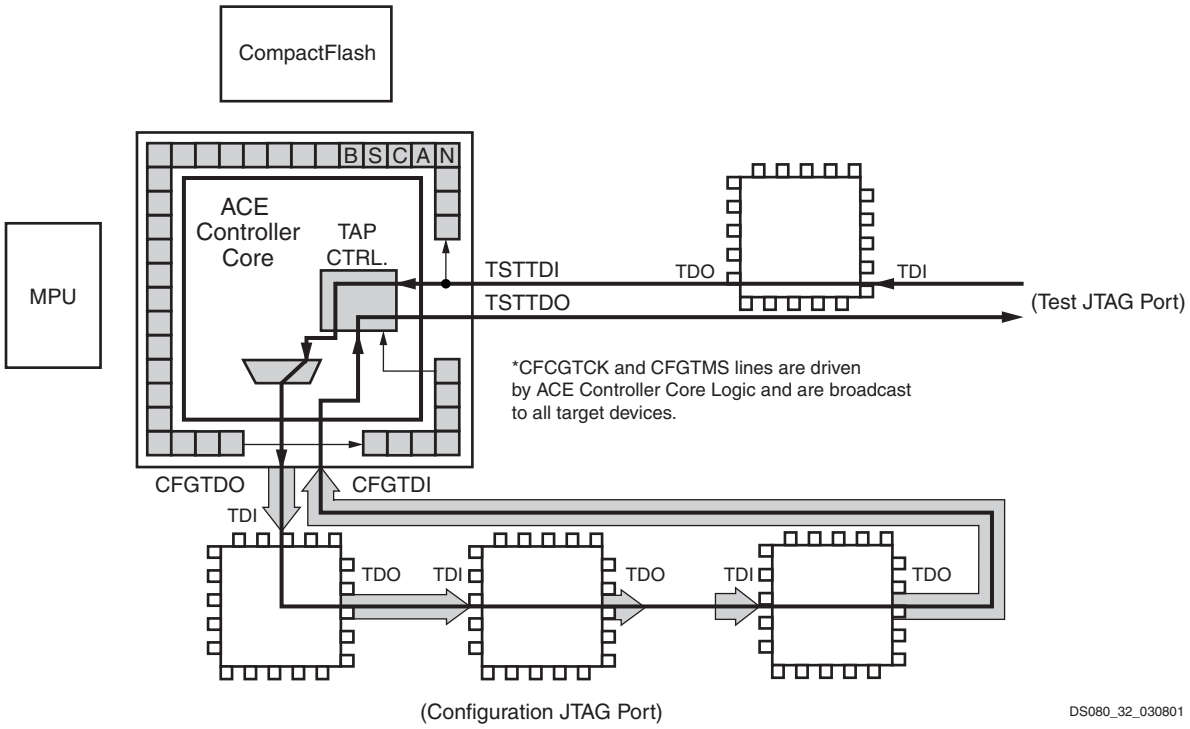


Figure 33: **Data Flow Diagram of TSTJTAG to CFGJTAG (Using Bypass Path)**

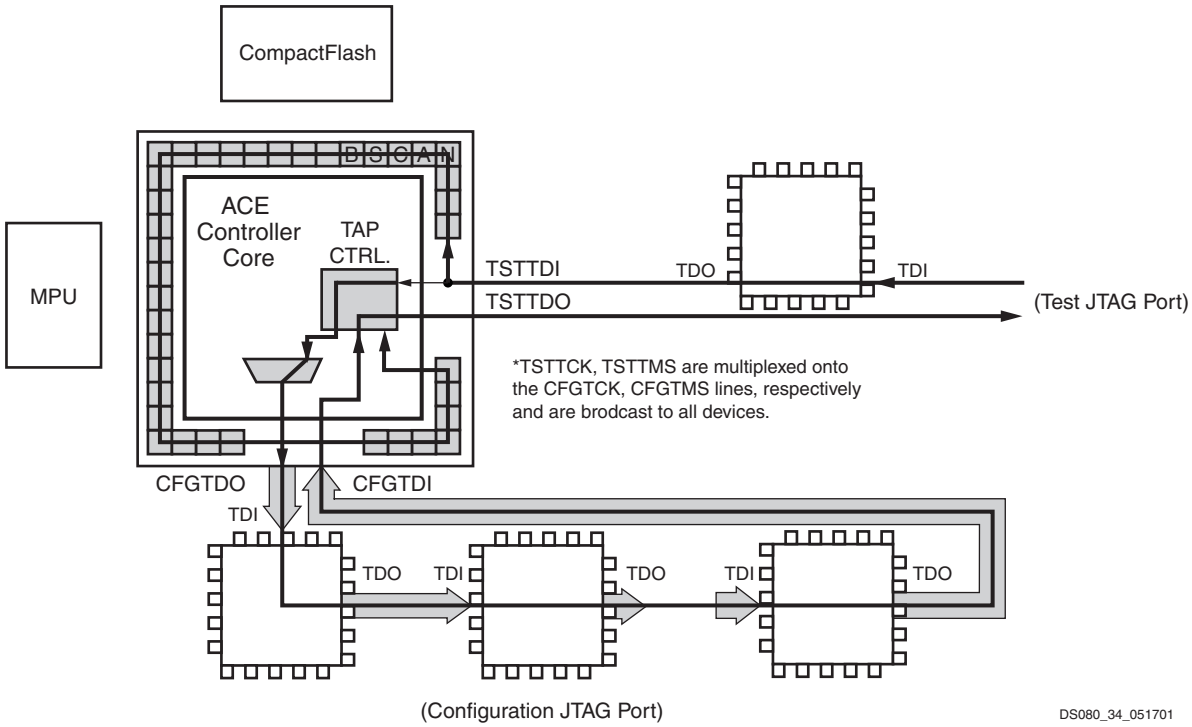


Figure 34: **Data Flow Diagram of TSTJTAG to CFGJTAG (Using Boundary-Scan Path)**

The System ACE CF controller handles all necessary steps to perform a configuration from the TSTJTAG to the target system via the CFGJTAG interface. When using the TSTJTAG to CFGJTAG setup, the signals in Figure 35, page 56 should be connected.

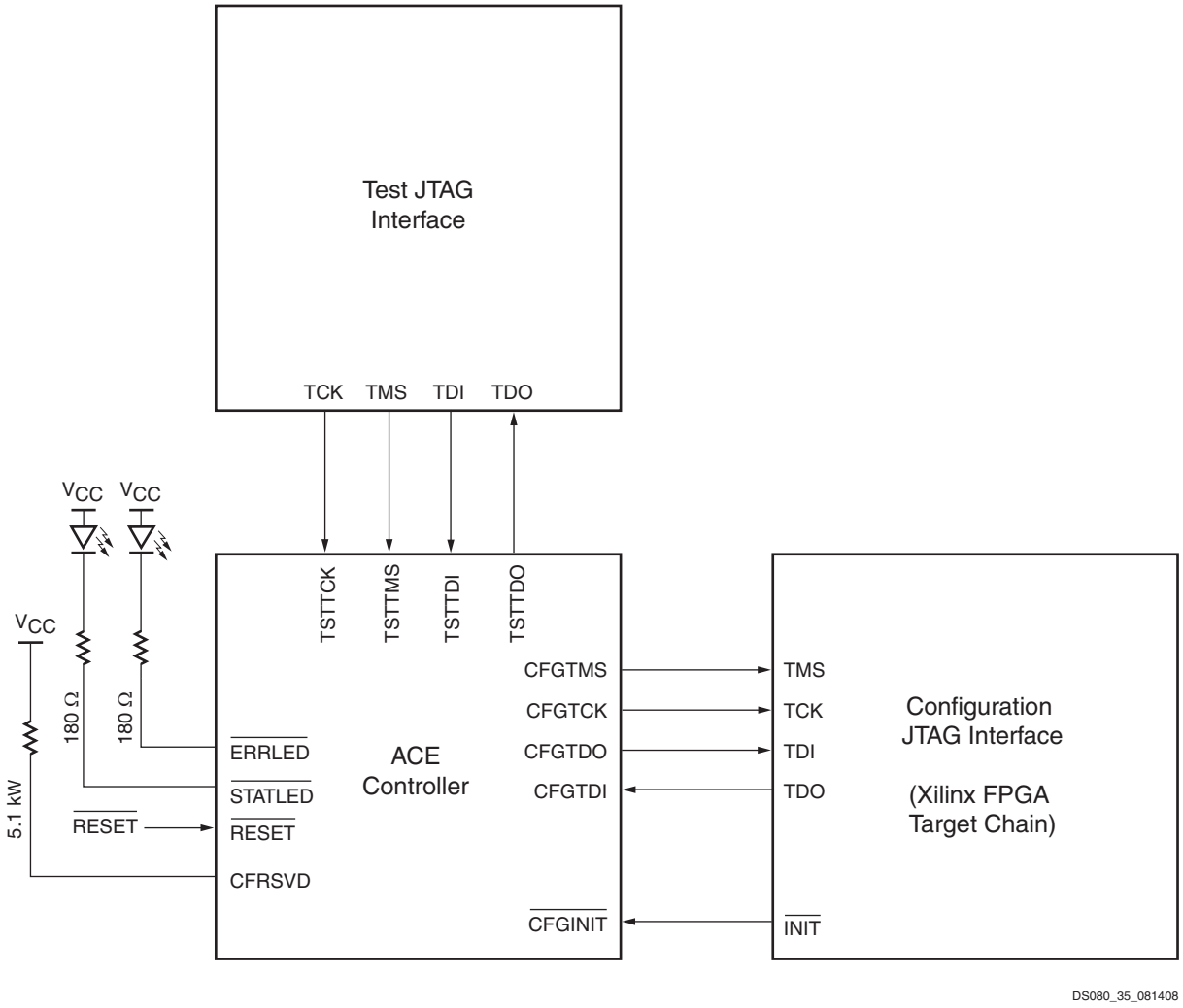


Figure 35: **Wiring Diagram of TSTJTAG to CFGJTAG**

General Timing Specifications

Table 28: **Clock Frequency Characteristics**

Symbol	Parameter	Min	Max	Units
F(CLK)	System ACE clock frequency		33	MHz
F(TSTTCK)	Test JTAG clock frequency		16.7	MHz

MPU Interface Timing Characteristics

Table 29: **MPU Interface Timing Characteristics**

Symbol	Parameter	Min	Max	Units
TS(MPACLK)	MPA[6:0] setup time before rising edge of CLK	4		ns
TS(MPCECLK)	$\overline{\text{MPCE}}$ setup time before rising edge of CLK	4		ns
TS(MPDCLK)	MPD[15:0] setup time before rising edge of CLK	4		ns
TS(MPOECLK)	$\overline{\text{MPOE}}$ setup time before rising edge of CLK	12		ns
TS(MPWECLK)	$\overline{\text{MPWE}}$ setup time before rising edge of CLK	12		ns
TH(CLKMPA)	MPA hold time after rising edge of CLK	4		ns
TH(CLKMPCE)	$\overline{\text{MPCE}}$ hold time after rising edge of CLK	4		ns
TH(CLKMPD)	MPD[15:0] hold time after rising edge of CLK	4		ns
TH(CLKMPOE)	$\overline{\text{MPOE}}$ hold time after rising edge of CLK	4		ns
TH(CLKMPWE)	$\overline{\text{MPWE}}$ hold time after rising edge of CLK	4		ns
TD(CLKMPD)	CLK rising edge to MPD		22	ns
TD(CLKMPBRDY)	CLK rising edge to MPBRDY		22	ns
TD(CLKMPIRQ)	CLK rising edge to MPIRQ		22	ns
TD(MPCEMPD)	Propagation delay from $\overline{\text{MPCE}}$ to MPD		13	ns
TD(MPOEMPD)	Propagation delay from $\overline{\text{MPOE}}$ to MPD		13	ns

CompactFlash Interface Timing Characteristics

Table 30: **CompactFlash Interface Timing Characteristics**

Symbol	Parameter	Min	Max	Units
$T_{S(CFCDCLK)}$	$\overline{CFCD1}$ and $\overline{CFCD2}$ setup time before rising edge of CLK	4		ns
$T_{S(CFDCLK)}$	CFD[15:0] setup time before rising edge of CLK	4		ns
$T_{S(CFWAITCLK)}$	$\overline{CFWAIT}$ setup time before rising edge of CLK	4		ns
$T_{H(CLKCFCD)}$	$\overline{CFCD1}$ and $\overline{CFCD2}$ hold time after rising edge of CLK	5		ns
$T_{H(CLKCFD)}$	CFD[15:0] hold time after rising edge of CLK	5		ns
$T_{H(CLKCFWAIT)}$	$\overline{CFWAIT}$ hold time after rising edge of CLK	4		ns
$T_{D(CLKCFA)}$	CLK rising edge to CFA[10:0]		19	ns
$T_{D(CLKCFCE)}$	CLK rising edge to $\overline{CFCE1}$ and $\overline{CFCE2}$		16	ns
$T_{D(CLKCFD)}$	CLK rising edge to CFD[15:0]		19	ns
$T_{D(CLKCFOE)}$	CLK rising edge to $\overline{CFOE}$		16	ns
$T_{D(CLKCFWE)}$	CLK rising edge to $\overline{CFWE}$		16	ns

Configuration JTAG Interface Timing Characteristics

Table 31: **Configuration JTAG Interface Timing Characteristics**

Symbol	Parameter	Min	Max	Units
$T_{S(CFGADDRCLK)}$	CFGADDR[2:0] setup time before rising edge of CLK	6		ns
$T_{S(CFGINITCLK)}$	$\overline{CFGINIT}$ setup time before rising edge of CLK	11		ns
$T_{S(CFGMODEPINCLK)}$	CFGMODEPIN setup time before rising edge of CLK	7		ns
$T_{S(CFGTDICLK)}$	CFGTDI setup time before falling edge of CLK	4		ns
$T_{H(CLKCFGADDR)}$	CFGADDR[2:0] hold time after rising edge of CLK	5		ns
$T_{H(CLKCFGINIT)}$	$\overline{CFGINIT}$ hold time after rising edge of CLK	0		ns
$T_{H(CLKCFGMODEPIN)}$	CFGMODEPIN hold time after rising edge of CLK	5		ns
$T_{H(CLKCFGTDI)}$	CFGTDI hold time after falling edge of CLK	4		ns
$T_{D(CLKCFGTDO)}$	CLK falling edge to CFGTDO		16	ns
$T_{D(CLKCFGTMS)}$	CLK falling edge to CFGTMS		20	ns
$T_{D(CLKCFGTCK)}$	Propagation delay from CLK to CFGTCK		15	ns

Test JTAG Interface Timing Characteristics

Table 32: **Test JTAG Interface Timing Characteristics**

Symbol	Parameter	Min	Max	Units
$T_{S(TSTTDITSTTCK)}$	TSTTDI setup time before rising edge of TSTTCK	4		ns
$T_{S(TSTTMSTSTTCK)}$	TSTTMS setup time before rising edge of TSTTCK	4		ns
$T_{S(INTSTTCK)}$	All other inputs setup time before rising edge of TSTTCK	5		ns
$T_{H(TSTTCKTSTTDI)}$	TSTTDI hold time after rising edge of TSTTCK	4		ns
$T_{H(TSTTCKTSTTMS)}$	TSTTMS hold time after rising edge of TSTTCK	4		ns
$T_{H(TSTTCKIN)}$	All other inputs hold time after rising edge of TSTTCK	4		ns
$T_{D(TSTTCKOUT)}$	TSTTCK falling edge to all other outputs		24	ns
$T_{D(TSTTCKCFGTCK)}$	Propagation delay from TSTTCK to CFGTCK		14	ns
$T_{D(CFGTDITSTTDO)}$	Propagation delay from CFGTDI to TSTTDO		11	ns
$T_{D(TSTTMSCFGTMS)}$	Propagation delay from TSTTMS to CFGTMS		13	ns

Miscellaneous Timing Characteristics

Table 33: **Miscellaneous Timing Characteristics**

Symbol	Parameter	Min	Max	Units
$T_{S(RESETCLK)}$	$\overline{\text{RESET}}$ setup time before rising edge of CLK	7		ns
$T_{H(CLKRESET)}$	$\overline{\text{RESET}}$ hold time after rising edge of CLK	4		ns
$T_{H(CLKERRLED)}$	CLK rising edge to $\overline{\text{ERRLED}}$		17	ns
$T_{H(CLKSTATLED)}$	CLK rising edge to $\overline{\text{STATLED}}$		17	ns

Electrical Characteristics

Table 34: **System ACE CF Controller Absolute Maximum Ratings (for V_{CCL} = 2.5 [V] or V_{CCL} = 3.3 [V])**

Description	Symbol	Limits	Units
Power Supply Voltage	V_{CCH}(1)	GND – 0.3 to 7.0	V
	V_{CCL}(1)	GND – 0.3 to 4.0	
Input Voltage	V_{IH}	GND – 0.3 to V_{CCH} + 0.5	V
	V_{IL}	GND – 0.3 to V_{CCL} + 0.5	
Output Voltage	V_{OH}	GND – 0.3 to V_{CCH} + 0.5	V
	V_{OL}	GND – 0.3 to V_{CCL} + 0.5	
Output Current/Pin	I_{OUT}	±30	mA
Storage Temperature	T_{STG}	–65 to 150	°C

Notes:
1. V_{CCH} is greater than or equal to V_{CCL}.

Table 35: **System ACE CF Controller Recommended Operating Conditions (for V_{CCL} = 2.5 [V])**

Description	Symbol	Min	Typ	Max	Units
Power Supply Voltage	V_{CCH}	3.0	3.3	3.6	V
	V_{CCL}	2.25	2.5	2.75	
Input Voltage	V_{IH}	GND	–	V_{CCH}	V
	V_{IL}	GND	–	V_{CCL}	
Ambient Temperature	T_A	–40	–	85(1)	°C

Notes:
1. The ambient temperature range is recommended for T_J = –40 to 125 °C.

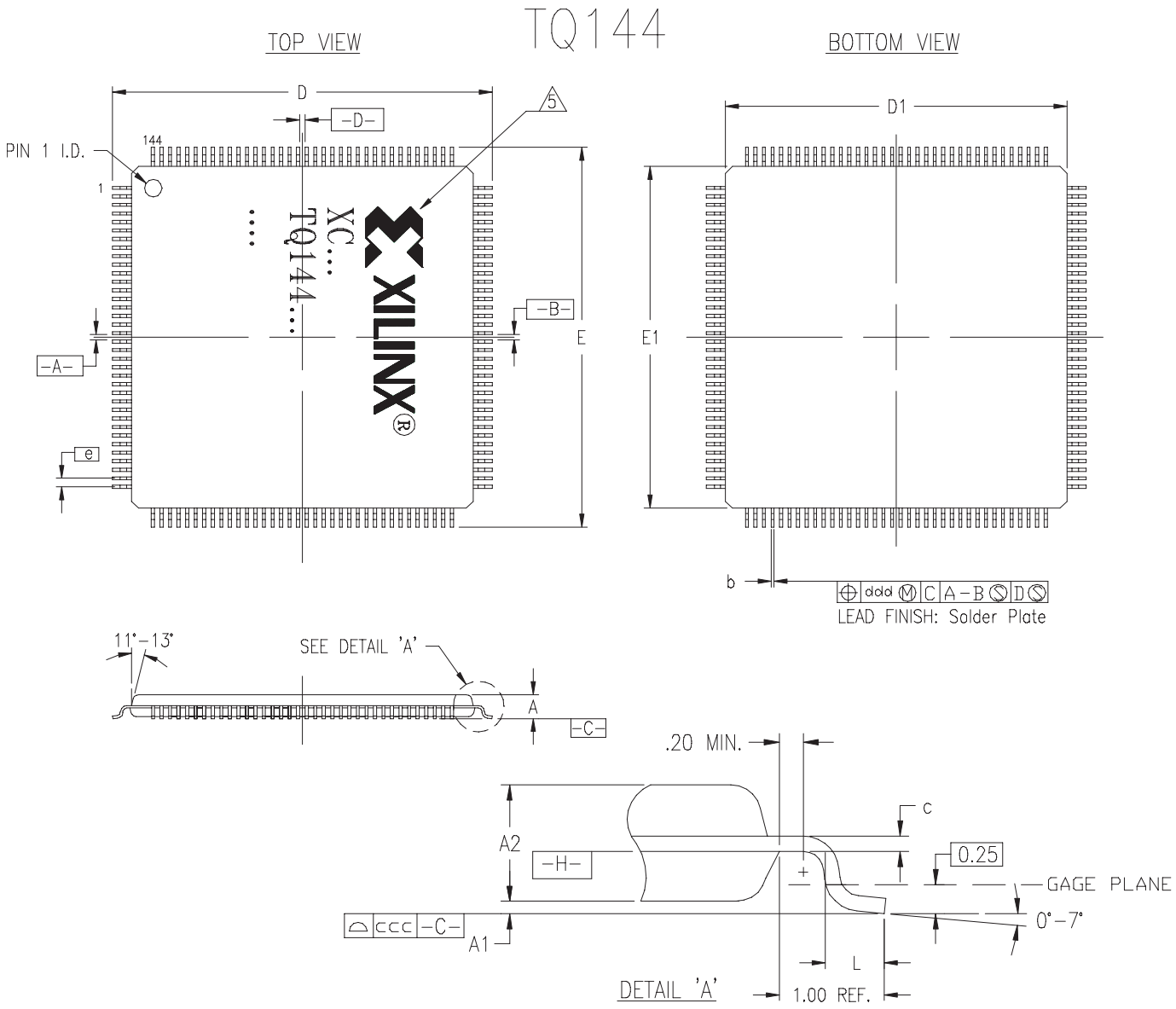
Table 36: **System ACE CF Controller Recommended Operating Conditions (for V_{CCL} = 3.3 [V])**

Description	Symbol	Min	Typ	Max	Units
Power Supply Voltage	V_{CCH}	3.0	3.3	3.6	V
	V_{CCL}	3.0	3.3	3.6	
Input Voltage	V_{IH}	GND	–	V_{CCH}	V
	V_{IL}	GND	–	V_{CCL}	
Ambient Temperature	T_A	–40	–	85(1)	°C

Notes:
1. The ambient temperature range is recommended for T_J = –40 to 125 °C.

Table 37: **System ACE CF Controller Characteristics**

Description	Symbol	Min	Typ	Max	Units	Conditions
Quiescent Current (between V_{CCH} and GND)	I_{CCSH}	--	--	300	µA	$V_I = V_{CCH}$ or V_{CCL} or GND, V_{CCH} = Max, V_{CCL} = Max, $I_{OH} = I_{OL} = 0$
Quiescent Current (between V_{CCL} and GND)	I_{CCSL}	--	--	420	µA	$V_I = V_{CCH}$ or V_{CCL} or GND, V_{CCH} = Max, V_{CCL} = Max, $I_{OH} = I_{OL} = 0$
Input Leakage Current	I_{LI}	–1	--	1	µA	V_{CCH} = Max, V_{CCL} = Max, $V_{IHH} = V_{CCH}$, $V_{IHL} = V_{CCL}$, V_{IL} = GND
High-Level Input Voltage	V_{IH1H}	2.0	--	--	V	Input Characteristics for I/O Supply Rail V_{CCH} = Max
Low-Level Input Voltage	V_{IL1H}	--	--	0.8	V	Input Characteristics for I/O Supply Rail V_{CCH} = Min
High-Level Input Voltage	V_{IH1L}	2.0	--	--	V	Input Characteristics for I/O Supply Rail V_{CCL} = Max
Low-Level Input Voltage	V_{IL1L}	--	--	0.8	V	Input Characteristics for I/O Supply Rail V_{CCL} = Min
Pull-Up Resistance	R_{PU1H}	40	100	240	kΩ	V_I = GND
Pull-Down Resistance	R_{PD1H}	40	100	240	kΩ	$V_I = V_{CCH}$
Pull-Up Resistance	R_{PU1L}	20	50	120	kΩ	V_I = GND
Pull-Down Resistance	R_{PD1L}	20	50	120	kΩ	$V_I = V_{CCL}$
High-Level Output Voltage	V_{OH3H}	V_{CCH} – 0.4	--	--	V	V_{CCH} = Min, I_{OH} = –12 mA
Low-Level Output Voltage	V_{OL3H}	--	--	GND + 0.4	V	V_{CCH} = Min, I_{OL} = 12 mA
High-Level Output Voltage	V_{OH3L}	V_{CCL} – 0.4	--	--	V	V_{CCL} = Min, I_{OH} = –12 mA
Low-Level Output Voltage	V_{OL3L}	--	--	GND + 0.4	V	V_{CCL} = Min, I_{OL} = 12 mA
Off-State Leakage Current	I_{OZ}	–1	--	1	µA	V_{CCH} = Max, V_{CCL} = Max, $V_{OHH} = V_{CCH}$, $V_{OHL} = V_{CCL}$, V_{OL} = GND
Input Terminal Capacitance	C_I	--	--	--	pF	--
Output Terminal Capacitance	C_O	--	--	--	pF	--
Input/Output Terminal Capacitance	C_{IO}	--	--	--	pF	--



SYMBOL	MILLIMETERS		
	MIN.	NOM.	MAX.
A			1.60
A_1	0.05	0.10	0.15
A_2	1.35	1.40	1.45
D/E	22.00 BSC		
D_1/E_1	20.00 BSC		
b	0.17	0.22	0.27
c	0.09		0.20
e	0.50 BSC		
L	0.45	0.60	0.75
ccc			0.08
ddd			0.08

NOTES:

1. ALL DIMENSIONS AND TOLERANCES CONFORM TO ANSI Y14.5M-1982.
2. DIMENSIONS "D1" AND "E1" DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION SHALL NOT EXCEED 0.25mm EACH SIDE.
3. THE TOP OF PACKAGE MAYBE SMALLER THAN THE BOTTOM OF PACKAGE BY 0.10mm.
4. THIS OUTLINE CONFORMS TO JEDEC PUBLICATION 95 REGISTRATION MS-026-BFB
5. MARK ORIENTATION WITH RESPECT TO PIN-1 I.D. IF 2 OR MORE CIRCLES EXIST ON TOP OF THE PACKAGE, USE THE SMALLEST CIRCLE AS PIN-1 I.D.

DS080_47_030801

Figure 36: **System ACE CF Controller TQ144 Package Drawing**

Pin Descriptions

This section provides System ACE CF controller pinout information.

System ACE CF Controller I/O Pins

Table 38 lists System ACE CF controller active pins.

Table 38: **System ACE CF Controller Pin Table (IN = input, OUT2 = 2-State Output, OUT3 = 3-State Output)**

Pin Name	Pin #	I/O Type	I/O Supply Rail	Termination	Description
CLK	93	IN	V_{CCL}	N/A	System ACE CF controller system clock
$\overline{\text{RESET}}$(2)	33	IN	V_{CCL}	Int. Pull-up	System ACE CF controller reset (active Low; needs to be active for three clock cycles). This also resets the CONTROLREG register to its default state.
$\overline{\text{STATLED}}$	95	OUT3 (Open-drain)	V_{CCL}	Ext. Pull-up	System ACE CF controller status LED
$\overline{\text{ERRLED}}$	96	OUT3 (Open-drain)	V_{CCL}	Ext. Pull-up	System ACE CF controller error LED; when LOW, this pin indicates that an error has occurred in the System ACE CF controller.
$\overline{\text{MPCE}}$	42	IN	V_{CCL}	Int. Pull-up	Chip enable (active LOW)
$\overline{\text{MPWE}}$	76	IN	V_{CCL}	Int. Pull-up	Write enable (active LOW)
$\overline{\text{MPOE}}$	77	IN	V_{CCL}	Int. Pull-up	Output enable (active LOW)
MPIRQ	41	OUT2	V_{CCL}	N/A	Interrupt request flag
MPBRDY	39	OUT2	V_{CCL}	N/A	Data buffer ready flag
MPA00	70	IN	V_{CCL}	N/A	MPU address line 0
MPA01	69	IN	V_{CCL}	N/A	MPU address line 1
MPA02	68	IN	V_{CCL}	N/A	MPU address line 2
MPA03	67	IN	V_{CCL}	N/A	MPU address line 3
MPA04	45	IN	V_{CCL}	N/A	MPU address line 4
MPA05	44	IN	V_{CCL}	N/A	MPU address line 5
MPA06	43	IN	V_{CCL}	N/A	MPU address line 6
MPD00	66	IN/OUT3	V_{CCL}	N/A	MPU data line 0
MPD01	65	IN/OUT3	V_{CCL}	N/A	MPU data line 1
MPD02	63	IN/OUT3	V_{CCL}	N/A	MPU data line 2
MPD03	62	IN/OUT3	V_{CCL}	N/A	MPU data line 3
MPD04	61	IN/OUT3	V_{CCL}	N/A	MPU data line 4
MPD05	60	IN/OUT3	V_{CCL}	N/A	MPU data line 5
MPD06	59	IN/OUT3	V_{CCL}	N/A	MPU data line 6
MPD07	58	IN/OUT3	V_{CCL}	N/A	MPU data line 7
MPD08	56	IN/OUT3	V_{CCL}	N/A	MPU data line 8
MPD09	53	IN/OUT3	V_{CCL}	N/A	MPU data line 9
MPD10	52	IN/OUT3	V_{CCL}	N/A	MPU data line 10

Table 38: **System ACE CF Controller Pin Table (IN = input, OUT2 = 2-State Output, OUT3 = 3-State Output)**

Pin Name	Pin #	I/O Type	I/O Supply Rail	Termination	Description
MPD11	51	IN/OUT3	V_{CCL}	N/A	MPU data line 11
MPD12	50	IN/OUT3	V_{CCL}	N/A	MPU data line 12
MPD13	49	IN/OUT3	V_{CCL}	N/A	MPU data line 13
MPD14	48	IN/OUT3	V_{CCL}	N/A	MPU data line 14
MPD15	47	IN/OUT3	V_{CCL}	N/A	MPU data line 15
CFA00	4	OUT2	V_{CCH}	N/A	CompactFlash address line 0
CFA01	142	OUT2	V_{CCH}	N/A	CompactFlash address line 1
CFA02	141	OUT2	V_{CCH}	N/A	CompactFlash address line 2
CFA03	139	OUT2	V_{CCH}	N/A	CompactFlash address line 3
CFA04	137	OUT2	V_{CCH}	N/A	CompactFlash address line 4
CFA05	135	OUT2	V_{CCH}	N/A	CompactFlash address line 5
CFA06	134	OUT2	V_{CCH}	N/A	CompactFlash address line 6
CFA07	132	OUT2	V_{CCH}	N/A	CompactFlash address line 7
CFA08	130	OUT2	V_{CCH}	N/A	CompactFlash address line 8
CFA09	125	OUT2	V_{CCH}	N/A	CompactFlash address line 9
CFA10	121	OUT2	V_{CCH}	N/A	CompactFlash address line 10
CFD00	5	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 0
CFD01	6	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 1
CFD02	8	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 2
CFD03	104	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 3
CFD04	106	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 4
CFD05	113	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 5
CFD06	115	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 6
CFD07	117	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 7
CFD08	7	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 8
CFD09	11	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 9
CFD10	12	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 10
CFD11	105	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 11
CFD12	107	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 12
CFD13	114	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 13
CFD14	116	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 14
CFD15	118	IN/OUT3	V_{CCH}	N/A	CompactFlash data line 15
$\overline{\text{CFCE1}}$	119	OUT2	V_{CCH}	N/A	CompactFlash chip enable 1 (active LOW);
$\overline{\text{CFCE2}}$	138	OUT2	V_{CCH}	N/A	CompactFlash chip enable 2 (active LOW);

Table 38: **System ACE CF Controller Pin Table (IN = input, OUT2 = 2-State Output, OUT3 = 3-State Output)**

Pin Name	Pin #	I/O Type	I/O Supply Rail	Termination	Description
$\overline{\text{CFREG}}$	3	OUT2	V_{CCH}	N/A	CompactFlash register select line (active LOW); this pin is always driven to a 1 but is provided here for future compatibility.
$\overline{\text{CFWE}}$	131	OUT2	V_{CCH}	N/A	CompactFlash write enable line (active LOW)
$\overline{\text{CFOE}}$	123	OUT2	V_{CCH}	N/A	CompactFlash output enable line (active LOW)
$\overline{\text{CFWAIT}}$	140	IN	V_{CCH}	N/A	CompactFlash memory cycle wait flag (active LOW)
CFRSVD	133	IN	V_{CCH}	Ext. Pull-up	This pin must be pulled up to V_{CCH} using an external pull-up resistor.
$\overline{\text{CFCD1}}$	103	IN	V_{CCH}	Int. Pull-up	CompactFlash card detect line 1 (active LOW)
$\overline{\text{CFCD2}}$	13	IN	V_{CCH}	Int. Pull-up	CompactFlash card detect line 2 (active LOW)
CFGADDR0	86	IN	V_{CCL}	Int. Pull-down	Configuration address select pin 0
CFGADDR1	87	IN	V_{CCL}	Int. Pull-down	Configuration address select pin 1
CFGADDR2	88	IN	V_{CCL}	Int. Pull-down	Configuration address select pin 2
CFGMODEPIN	89	IN	V_{CCL}	Int. Pull-up	Configuration mode pin: • When 0, this pin instructs the System ACE CF controller to start the configuration process when the CFGSTART bit is set in the CONTROLREG register in the MPU interface. • When 1, this pin instructs the System ACE CF controller to start the configuration process immediately following reset.
TSTTDI	102	IN	V_{CCH}	Int. Pull-up	Test JTAG port test data input
TSTTCK	101	IN	V_{CCH}	N/A	Test JTAG port test clock
TSTTMS	98	IN	V_{CCH}	Int. Pull-up	Test JTAG port test mode select
TSTTDO	97	OUT3	V_{CCH}	Ext. Pull-up(1)	Test JTAG port test data output
CFGTDO	82	OUT3	V_{CCL}	Ext. Pull-up(1)	Configuration JTAG test data output
CFGTDI	81	IN	V_{CCL}	Int. Pull-up	Configuration JTAG test data input
CFGTCK	80	OUT2	V_{CCL}	N/A	Configuration JTAG test clock
CFGTMS	85	OUT3	V_{CCL}	Ext. Pull-up(1)	Configuration JTAG test mode select
$\overline{\text{CFGINIT}}$	78	IN	V_{CCL}	Int. Pull-up	Configuration JTAG INIT pin (active LOW); this pin is used to sense when all devices are ready to be programmed (i.e., INIT = 1 indicates target device(s) are ready to receive configuration data and INIT = 0 indicates that the target device(s) are being cleared and are not ready to be configured)

Table 38: **System ACE CF Controller Pin Table (IN = input, OUT2 = 2-State Output, OUT3 = 3-State Output)**

Pin Name	Pin #	I/O Type	I/O Supply Rail	Termination	Description
POR_BYPASS	108	IN	V_{CCH}	Int. Pull-down	Power-on-reset (POR) bypass input; used in conjunction with POR_RESET to bypass the internal POR circuit in favor of using an external board-level POR circuit; the internal POR circuit is bypassed when POR_BYPASS = 1; the POR_BYPASS pin should be held at a static 0 or 1 while the System ACE CF controller is receiving power.
POR_RESET(2)	72	IN	V_{CCH}	Int. Pull-down	Power-on-reset bypass input; can be used in conjunction with POR_BYPASS to bypass the internal POR circuit in favor of using an external board-level POR circuit; all internal circuitry is reset when POR_BYPASS = 1 and POR_RESET = 1; The POR_RESET pulse duration should be at least 1 microsecond long.
$\overline{\text{POR\_TEST}}$	74	OUT2	V_{CCH}	N/A	Power-on-reset test output; this pin should be a true No Connect on the board (see Table 40, page 68) but is listed here for informational purposes. POR_TEST is Low during power up and when POR_BYPASS and POR_RESET are asserted High (causing a device reset). After power up is complete, POR_TEST is High when POR_BYPASS is Low, or when POR_BYPASS is High and POR_RESET is Low. An internal timer circuit in the POR component determines when to release POR_TEST after power reaches the ON threshold.

Notes:
1. JTAG 1149.1 requires a pull-up resistor on potentially undriven TDO/TMS signals.
2. If not using the $\overline{\text{RESET}}$ signal prior to CFGJTAG configuration, ensure that the V_{CCH} and V_{CCL} are brought all the way back to 0V when power-cycling the board. Failure to do this, could cause the Power-On-Reset (POR) circuitry to operate incorrectly.

Table 39 lists System ACE CF controller voltage and ground pins.

Table 39: **System ACE CF Controller Voltage and Ground Pins**

Pin Name	Pin Number	Description
VCCH	1	High-voltage (3.3V) source pins
	17	
	37	
	55	
	73	
	92	
	109	
	128	
VCCL	10	Low-voltage (2.5V or 3.3V) source pins
	15	
	25	
	57	
	84	
	94	
	99	
	126	
GND	9	Ground pins
	18	
	26	
	35	
	46	
	54	
	64	
	75	
	83	
	91	
	100	
	110	
	111	
	112	
	120	
	129	
	136	
	144	

Table 40 lists System ACE CF controller no-connect pins.

Table 40: **System ACE CF Controller No-Connect Pins**

Pin Name	Pin Number	Description
NC	2	Pins that must not be connected to any board-level signals, including ground and power planes.
	14	
	16	
	19	
	20	
	21	
	22	
	23	
	24	
	27	
	28	
	29	
	30	
	31	
	32	
	34	
	36	
	38	
	40	
	71	
	74	
	79	
	90	
	122	
	124	
	127	
	143	

Ordering Information

System ACE Valid Ordering Combinations	Description	Package	Operating Range
XCCACE — TQG144I[1]	System ACE CF Controller Chip	TQ144	(T_A = -40 to +85 °C)

1.This device is Pb-free. The non Pb-free version of this device was discontinued as noted by XCN06006 (http://www.xilinx.com/support/documentation/customer_notices/xcn06006.pdf).

Revision History

Version No.	Date	Description
1.0	05/18/01	Initial Xilinx release.
1.1	06/04/01	Corrected Table 27, page 37. CFGMODE is 1 after Reset, 0 after MPU start signal.
1.2	07/18/01	Updated.
1.3	12/12/01	Updated.
1.4	01/03/02	Updated Table 1, Figure 19, Figure 21, Figure 31, Figure 35, and Table 38 (last row only).
1.5	04/05/02	Fixed the note numbers in Table 27.
2.0	10/01/08	Major update.

Notice of Disclaimer