

Core Overview

The CompactFlash core allows you to connect SOPC Builder systems to CompactFlash storage cards in true IDE mode by providing an Avalon[®] Memory-Mapped (Avalon-MM) interface to the registers on the storage cards. The core supports PIO mode 0.

The CompactFlash core also provides an Avalon-MM slave interface which can be used by Avalon-MM master peripherals such as a Nios[®] II processor to communicate with the CompactFlash core and manage its operations.

The CompactFlash core is SOPC Builder-ready and integrates easily into any SOPC Builder-generated systems.

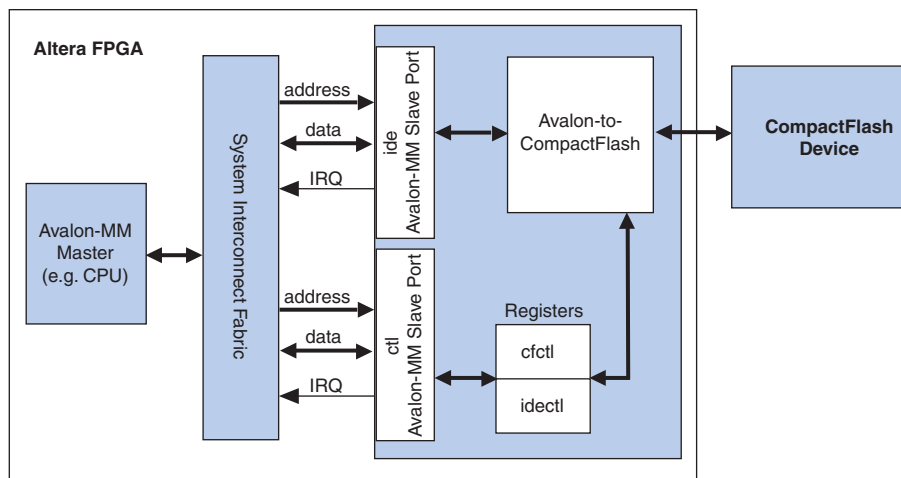
This chapter contains the following sections:

- “Functional Description”
- “Instantiating the Core in SOPC Builder” on page 2-2
- “Device Support” on page 2-3
- “Software Programming Model” on page 2-3

Functional Description

Figure 2-1 shows a block diagram of the CompactFlash core in a typical system configuration.

Figure 2-1. SOPC Builder System With a CompactFlash Core



As shown in [Figure 2-1](#), the CompactFlash core provides two Avalon-MM slave interfaces: the `ide` slave port for accessing the registers on the CompactFlash device and the `ctl` slave port for accessing the core's internal registers. These registers can be used by Avalon-MM master peripherals such as a Nios II processor to control the operations of the CompactFlash core and to transfer data to and from the CompactFlash device.

You can set the CompactFlash core to generate two active-high interrupt requests (IRQs): one signals the insertion and removal of a CompactFlash device and the other passes interrupt signals from the CompactFlash device.

The CompactFlash core maps the Avalon-MM bus signals to the CompactFlash device with proper timing, thus allowing Avalon-MM master peripherals to directly access the registers on the CompactFlash device.



For more information, refer to the CF+ and CompactFlash specifications available at www.compactflash.org.

Instantiating the Core in SOPC Builder

Use the MegaWizard™ interface for the CompactFlash core in SOPC Builder to add the core to a system. There are no user-configurable settings for this core.

Required Connections

[Table 2-1](#) lists the required connections between the CompactFlash core and the CompactFlash device.

Table 2-1. Required Connections (Part 1 of 2)

| CompactFlash Interface Signal Name | Pin Type | CompactFlash Pin Number |
|------------------------------------|--------------|-------------------------|
| <code>addr[0]</code> | Output | 20 |
| <code>addr[1]</code> | Output | 19 |
| <code>addr[2]</code> | Output | 18 |
| <code>addr[3]</code> | Output | 17 |
| <code>addr[4]</code> | Output | 16 |
| <code>addr[5]</code> | Output | 15 |
| <code>addr[6]</code> | Output | 14 |
| <code>addr[7]</code> | Output | 12 |
| <code>addr[8]</code> | Output | 11 |
| <code>addr[9]</code> | Output | 10 |
| <code>addr[10]</code> | Output | 8 |
| <code>atase1_n</code> | Output | 9 |
| <code>cs_n[0]</code> | Output | 7 |
| <code>cs_n[1]</code> | Output | 32 |
| <code>data[0]</code> | Input/Output | 21 |
| <code>data[1]</code> | Input/Output | 22 |

Table 2-1. Required Connections (Part 2 of 2)

| CompactFlash Interface Signal Name | Pin Type | CompactFlash Pin Number |
|------------------------------------|--------------|---|
| data [2] | Input/Output | 23 |
| data [3] | Input/Output | 2 |
| data [4] | Input/Output | 3 |
| data [5] | Input/Output | 4 |
| data [6] | Input/Output | 5 |
| data [7] | Input/Output | 6 |
| data [8] | Input/Output | 47 |
| data [9] | Input/Output | 48 |
| data [10] | Input/Output | 49 |
| data [11] | Input/Output | 27 |
| data [12] | Input/Output | 28 |
| data [13] | Input/Output | 29 |
| data [14] | Input/Output | 30 |
| data [15] | Input/Output | 31 |
| detect | Input | 25 or 26 |
| intrq | Input | 37 |
| iord_n | Output | 34 |
| iordy | Input | 42 |
| iowr_n | Output | 35 |
| power | Output | CompactFlash power controller, if present |
| reset_n | Output | 41 |
| rfu | Output | 44 |
| we_n | Output | 46 |

Device Support

The CompactFlash interface core supports all Altera® device families.

Software Programming Model

This section describes the software programming model for the CompactFlash core.

HAL System Library Support

The Altera-provided HAL API functions include a device driver that you can use to initialize the CompactFlash core. To perform other operations, use the low-level macros provided. For more information on the macros, refer to the files listed in the section [“Software Files” on page 2-4](#).

Software Files

The CompactFlash core provides the following software files. These files define the low-level access to the hardware. Application developers should not modify these files.

- **altera_avalon_cf_regs.h**—The header file that defines the core's register maps.
- **altera_avalon_cf.h, altera_avalon_cf.c**—The header and source code for the functions and variables required to integrate the driver into the HAL system library.

Register Maps

This section describes the register maps for the Avalon-MM slave interfaces.

Ide Registers

The `ide` port in the CompactFlash core allows you to access the IDE registers on a CompactFlash device. [Table 2-2](#) shows the register map for the `ide` port.

Table 2-2. Ide Register Map

| Offset | Register Names | |
|--------|------------------|------------------|
| | Read Operation | Write Operation |
| 0 | RD Data | WR Data |
| 1 | Error | Features |
| 2 | Sector Count | Sector Count |
| 3 | Sector No | Sector No |
| 4 | Cylinder Low | Cylinder Low |
| 5 | Cylinder High | Cylinder High |
| 6 | Select Card/Head | Select Card/Head |
| 7 | Status | Command |
| 14 | Alt Status | Device Control |

Ctl Registers

The `ctl` port in the CompactFlash core provides access to the registers which control the core's operation and interface. [Table 2-3](#) shows the register map for the `ctl` port.

Table 2-3. Ctl Register Map

| Offset | Register | Fields | | | | |
|--------|----------|----------|------|-----|-----|------|
| | | 31:4 | 3 | 2 | 1 | 0 |
| 0 | cfctl | Reserved | IDET | RST | PWR | DET |
| 1 | idectl | Reserved | | | | IIDE |
| 2 | Reserved | Reserved | | | | |
| 3 | Reserved | Reserved | | | | |

Cfctl Register

The `cfctl` register controls the operations of the CompactFlash core. Reading the `cfctl` register clears the interrupt. [Table 2-4](#) describes the `cfctl` register bits.

Table 2-4. cfctl Register Bits

| Bit Number | Bit Name | Read/Write | Description |
|------------|----------|------------|---|
| 0 | DET | RO | Detect. This bit is set to 1 when the core detects a CompactFlash device. |
| 1 | PWR | RW | Power. When this bit is set to 1, power is being supplied to the CompactFlash device. |
| 2 | RST | RW | Reset. When this bit is set to 1, the CompactFlash device is held in a reset state. Setting this bit to 0 returns the device to its active state. |
| 3 | IDET | RW | Detect Interrupt Enable. When this bit is set to 1, the CompactFlash core generates an interrupt each time the value of the <code>det</code> bit changes. |

Idectl Register

The `idectl` register controls the interface to the CompactFlash device. [Table 2-5](#) describes the `idectl` register bit.

Table 2-5. idectl Register

| Bit Number | Bit Name | Read/Write | Description |
|------------|----------|------------|--|
| 0 | IIDE | RW | IDE Interrupt Enable. When this bit is set to 1, the CompactFlash core generates an interrupt following an interrupt generated by the CompactFlash device. Setting this bit to 0 disables the IDE interrupt. |

Document Revision History

[Table 2-6](#) shows the revision history for this chapter.

Table 2-6. Document Revision History

| Date and Document Version | Changes Made | Summary of Changes |
|---------------------------|--|--------------------|
| November 2009 v9.1.0 | No change from previous release. | — |
| March 2009 v9.0.0 | No change from previous release. | — |
| November 2008 v8.1.0 | Changed to 8-1/2 x 11 page size. No change to content. | — |
| May 2008 v8.0.0 | Added the mode supported by the CompactFlash core. | — |



For previous versions of the *Quartus II Handbook*, refer to the [Quartus II Handbook Archive](#).

