# 14. Serial Configuration Devices (EPCS1, EPCS4, EPCS16, EPCS64, and EPCS128) Data Sheet

C51014-3.1

## Introduction

The serial configuration devices provide the following features:

- 1-, 4-, 16-, 64-, and 128-Mbit flash memory devices that serially configure Stratix® III, Stratix II GX, and Stratix II FPGAs, Arria™ GX FPGAs, and the Cyclone® series FPGAs using the active serial (AS) configuration scheme
- Easy-to-use four-pin interface
- Low cost, low-pin count, and non-volatile memory
- Low current during configuration and near-zero standby mode current
- 3.3-V operation
- Available in 8-pin and 16-pin small outline integrated circuit (SOIC) package
- Enables the Nios® processor to access unused flash memory through AS memory interface
- Re-programmable memory with more than 100,000 erase/program cycles
- Write protection support for memory sectors using status register bits
- In-system programming support with SRunner software driver
- In-system programming support with USB Blaster™, EthernetBlaster™, or ByteBlaster™ II download cables
- Additional programming support with the Altera® Programming Unit (APU) and programming hardware from BP Microsystems, System General, and other vendors
- Software design support with the Altera Quartus® II development system for Windows-based PCs as well as Sun SPARC station and HP 9000 Series 700/800
- Delivered with the memory array erased (all the bits set to `1`)

☞ The term "serial configuration devices" used in this document refers to Altera EPCS1, EPCS4, EPCS16, EPCS64, and EPCS128.

## Functional Description

With SRAM-based devices that support active serial configuration, configuration data must be reloaded each time the device powers up, the system reconfigures, or when new configuration data is required. Serial configuration devices are flash memory devices with a serial interface that can store configuration data for FPGA devices that support active serial configuration and reload the data to the device upon power-up or reconfiguration. Table 14–1 lists the serial configuration devices.

*Table 14–1. Serial Configuration Devices (3.3-V Operation)*

| Device | Memory Size (Bits) |
|---|---|
| EPCS1 | 1,048,576 |
| EPCS4 | 4,194,304 |
| EPCS16 | 16,777,216 |
| EPCS64 | 67,108,864 |
| EPCS128 | 134,217,728 |

For an 8-pin SOIC package, you can migrate vertically from the EPCS1 to the EPCS4 or EPCS16 since the EPCS devices are offered in the same device package. Similarly, for a 16-pin SOIC package, you can migrate vertically from the EPCS16 to the EPCS64 or EPCS128.

☞ EPCS16 is available in 8-pin and 16-pin SOIC packages.

Table 14–2 lists the serial configuration device used with each Stratix III FPGA and the configuration file size. Stratix III devices can be used with EPCS16, EPCS64, or EPCS128.

*Table 14–2. Serial Configuration Device Support for Stratix III Devices (Part 1 of 2)*

| Stratix III Device | Raw Binary File Size (Bits) *(1)* | Serial Configuration Device | | | | |
|---|---|---|---|---|---|---|
| | | EPCS1 | EPCS4 | EPCS16 | EPCS64 | EPCS128 |
| EP3SL50 | 22,178,792 | — | — | ✓ *(2)* | ✓ | ✓ |
| EP3SL70 | 22,178,792 | — | — | ✓ *(2)* | ✓ | ✓ |
| EP3SL110 | 47,413,312 | — | — | — | ✓ | ✓ |
| EP3SL150 | 47,413,312 | — | — | — | ✓ | ✓ |
| EP3SL200 | 93,324,656 | — | — | — | ✓ *(2)* | ✓ |
| EP3SL340 | 117,384,664 | — | — | — | — | ✓ |
| EP3SE50 | 25,891,968 | — | — | — | ✓ | ✓ |
| EP3SE80 | 48,225,392 | — | — | — | ✓ | ✓ |

*Table 14–2. Serial Configuration Device Support for Stratix III Devices (Part 2 of 2)*

| Stratix III Device | Raw Binary File Size (Bits) (1) | Serial Configuration Device | | | | |
|---|---|---|---|---|---|---|
| | | EPCS1 | EPCS4 | EPCS16 | EPCS64 | EPCS128 |
| EP3SE110 | 48,225,392 | — | — | — | ✓ | ✓ |
| EP3SE260 | 93,324,656 | — | — | — | ✓ (2) | ✓ |

*Notes to Table 14–2:*

(1) These are uncompressed file sizes.

(2) This is with the Stratix III compression feature enabled.

Table 14–3 lists the serial configuration device used with each Stratix II GX FPGA and the configuration file size. Stratix II GX devices can be used with EPCS16, EPCS64, or EPCS128.

*Table 14–3. Serial Configuration Device Support for Stratix II GX Devices*

| Stratix II GX Device | Raw Binary File Size (Bits) (1) | Serial Configuration Device | | | | |
|---|---|---|---|---|---|---|
| | | EPCS1 | EPCS4 | EPCS16 | EPCS64 | EPCS128 |
| EP2SGX30C<br>EP2SGX30D | 9,640,672 | — | — | ✓ | ✓ | ✓ |
| EP2SGX60C<br>EP2SGX60D<br>EP2SGX60E | 16,951,824 | — | — | ✓ (2) | ✓ | ✓ |
| EP2SGX90E<br>EP2SGX90F | 25,699,104 | — | — | — | ✓ | ✓ |
| EP2SGX130G | 37,325,760 | — | — | — | ✓ | ✓ |

*Notes to Table 14–3:*

(1) These are uncompressed file sizes.

(2) This is with the Stratix II GX compression feature enabled.

Table 14–4 lists the serial configuration device used with each Stratix II FPGA and the configuration file size. Stratix II devices can be used with EPCS4, EPCS16, EPCS64, or EPCS128.

*Table 14–4. Serial Configuration Device Support for Stratix II Devices*

| Stratix II Device | Raw Binary File Size (Bits) *(1)* | Serial Configuration Device | | | |
|---|---|---|---|---|---|
| | | EPCS4 | EPCS16 | EPCS64 | EPCS128 |
| EP2S15 | 4,721,544 | ✓ *(2)* | ✓ | ✓ | ✓ |
| EP2S30 | 9,640,672 | — | ✓ | ✓ | ✓ |
| EP2S60 | 16,951,824 | — | ✓ *(2)* | ✓ | ✓ |
| EP2S90 | 25,699,104 | — | ✓ *(2)* | ✓ | ✓ |
| EP2S130 | 37,325,760 | — | — | ✓ | ✓ |
| EP2S180 | 49,814,760 | — | — | ✓ | ✓ |

*Notes to Table 14–4:*

(1) These are uncompressed file sizes.

(2) This is with the Stratix II compression feature enabled.

Table 14–5 lists the serial configuration device used with each Arria GX FPGA and the configuration file size. Arria GX devices can be used with EPCS16, EPCS64, or EPCS128.

*Table 14–5. Serial Configuration Device Support for Arria GX Devices*

| Arria GX Device | Raw Binary File Size (Bits) *(1)* | Serial Configuration Device | | | | |
|---|---|---|---|---|---|---|
| | | EPCS1 | EPCS4 | EPCS16 | EPCS64 | EPCS128 |
| EP1AGX20C | 9,640,672 | — | — | ✓ | ✓ | ✓ |
| EP1AGX35C<br>EP1AGX35D | 9,640,672 | — | — | ✓ | ✓ | ✓ |
| EP1AGX50C<br>EP1AGX50D | 16,951,824 | — | — | ✓ *(2)* | ✓ | ✓ |
| EP1AGX60C<br>EP1AGX60D<br>EP1AGX60E | 16,951,824 | — | — | ✓ *(2)* | ✓ | ✓ |
| EP1AGX90E | 25,699,104 | — | — | — | ✓ | ✓ |

*Notes to Table 14–5:*

(1) These are uncompressed file sizes.

(2) This is with the Arria GX compression feature enabled.

Table 14–6 lists the serial configuration device used with each Cyclone III FPGA and the configuration file size. Cyclone III devices can be used with EPCS4, EPCS16, EPCS64, or EPCS128.

*Table 14–6. Serial Configuration Device for Cyclone III Devices*

| Cyclone III Device | Raw Binary File Size (Bits) *(1)* | Serial Configuration Device | | | | |
|---|---|---|---|---|---|---|
| | | EPCS1 | EPCS4 | EPCS16 | EPCS64 | EPCS128 |
| EP3C5 | 2,944,088 | — | ✓ | ✓ | ✓ | ✓ |
| EP3C10 | 2,944,088 | — | ✓ | ✓ | ✓ | ✓ |
| EP3C16 | 4,086,848 | — | ✓ | ✓ | ✓ | ✓ |
| EP3C25 | 5,748,552 | — | — | ✓ | ✓ | ✓ |
| EP3C40 | 9,534,304 | — | — | ✓ | ✓ | ✓ |
| EP3C55 | 14,889,560 | — | — | ✓ | ✓ | ✓ |
| EP3C80 | 19,965,752 | — | — | ✓*(2)* | ✓ | ✓ |
| EP3C120 | 28,571,696 | — | — | — | ✓ | ✓ |

*Notes to Table 14–6:*

(1) These are uncompressed file sizes.

(2) This is with the Cyclone III compression feature enabled.

Table 14–7 lists the serial configuration device used with each Cyclone II FPGA and the configuration file size. Cyclone II devices can be used with EPCS1, EPCS4, EPCS16, EPCS64, or EPCS128.

*Table 14–7. Serial Configuration Device for Cyclone II Devices*

| Cyclone II Device | Raw Binary File Size (Bits) *(1)* | Serial Configuration Device | | | | |
|---|---|---|---|---|---|---|
| | | EPCS1 | EPCS4 | EPCS16 | EPCS64 | EPCS128 |
| EP2C5 | 1,265,792 | ✓ *(2)* | ✓ | ✓ | ✓ | ✓ |
| EP2C8 | 1,983,536 | — | ✓ | ✓ | ✓ | ✓ |
| EP2C20 | 3,892,496 | — | ✓ | ✓ | ✓ | ✓ |
| EP2C35 | 6,848,608 | — | — | ✓ | ✓ | ✓ |
| EP2C50 | 9,951,104 | — | — | ✓ | ✓ | ✓ |
| EP2C70 | 14,319,216 | — | — | ✓ | ✓ | ✓ |

*Notes to Table 14–7:*

(1) These are uncompressed file sizes.

(2) This is with the Cyclone II compression feature enabled.

Table 14–8 lists the serial configuration device used with each Cyclone FPGA and the configuration file size. Cyclone devices can be used with EPCS1, EPCS4, EPCS16, EPCS64, or EPCS128.

*Table 14–8. Serial Configuration Device Support for Cyclone Devices*

| Cyclone Device | Raw Binary File Size (Bits) *(1)* | Serial Configuration Device | | | | |
|---|---|---|---|---|---|---|
| | | EPCS1 | EPCS4 | EPCS16 | EPCS64 | EPCS128 |
| EP1C3 | 627,376 | ✓ | ✓ | ✓ | ✓ | ✓ |
| EP1C4 | 924,512 | ✓ | ✓ | ✓ | ✓ | ✓ |
| EP1C6 | 1,167,216 | ✓ *(2)* | ✓ | ✓ | ✓ | ✓ |
| EP1C12 | 2,323,240 | — | ✓ | ✓ | ✓ | ✓ |
| EP1C20 | 3,559,608 | — | ✓ | ✓ | ✓ | ✓ |

*Notes to Table 14–8:*

(1) These are uncompressed file sizes.

(2) This is with the Cyclone compression feature enabled.

With the new data-decompression feature in the Stratix III, Stratix II GX, and Stratix II FPGAs, Arria GX FPGAs, and Cyclone FPGA families, you can use smaller serial configuration devices to configure larger FPGAs.

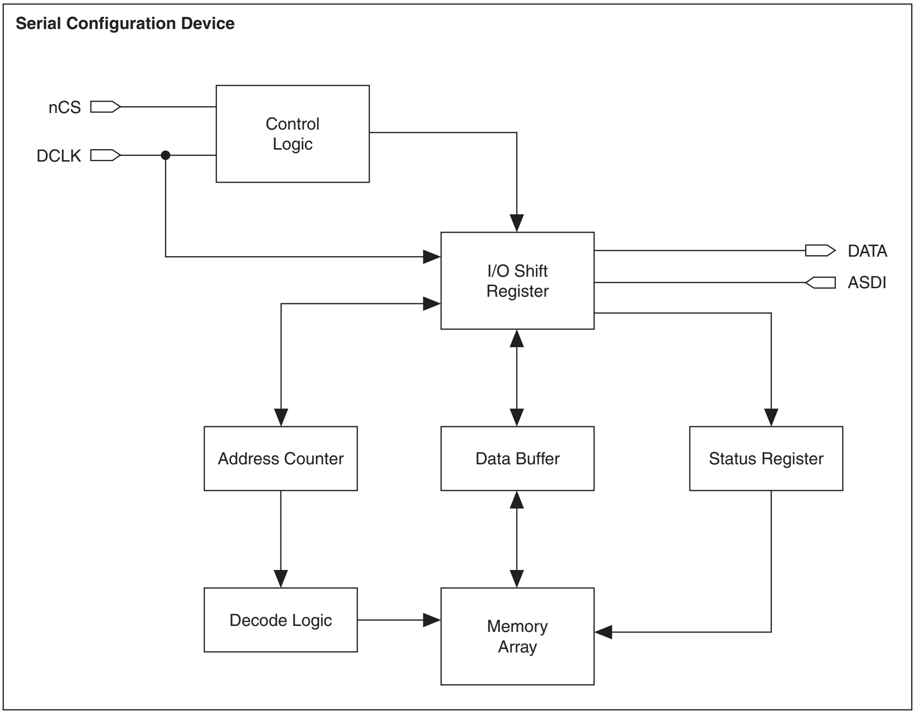
☞ Serial configuration devices cannot be cascaded.

For more information about the FPGA decompression feature, refer to the configuration chapter in the appropriate device handbook.

The serial configuration devices are designed to configure Stratix III, Stratix II GX, and Stratix II FPGAs and the Cyclone series FPGAs and cannot configure other existing Altera FPGA device families.

Figure 14–1 shows the serial configuration device block diagram.

*Figure 14–1. Serial Configuration Device Block Diagram*



## Accessing Memory in Serial Configuration Devices

You can access the unused memory locations of the serial configuration device to store or retrieve data through the Nios processor and SOPC Builder. SOPC Builder is an Altera tool for creating bus-based (especially microprocessor-based) systems in Altera devices. SOPC Builder assembles library components such as processors and memories into custom microprocessor systems.

SOPC Builder includes the EPCS device controller core, which is an interface core specifically designed to work with the serial configuration device. With this core, you can create a system with a Nios embedded processor that allows software access to any memory location within the serial configuration device.

For more information about accessing memory within the serial configuration device, refer to the *Active Serial Memory Interface Data Sheet*.

## Active Serial FPGA Configuration

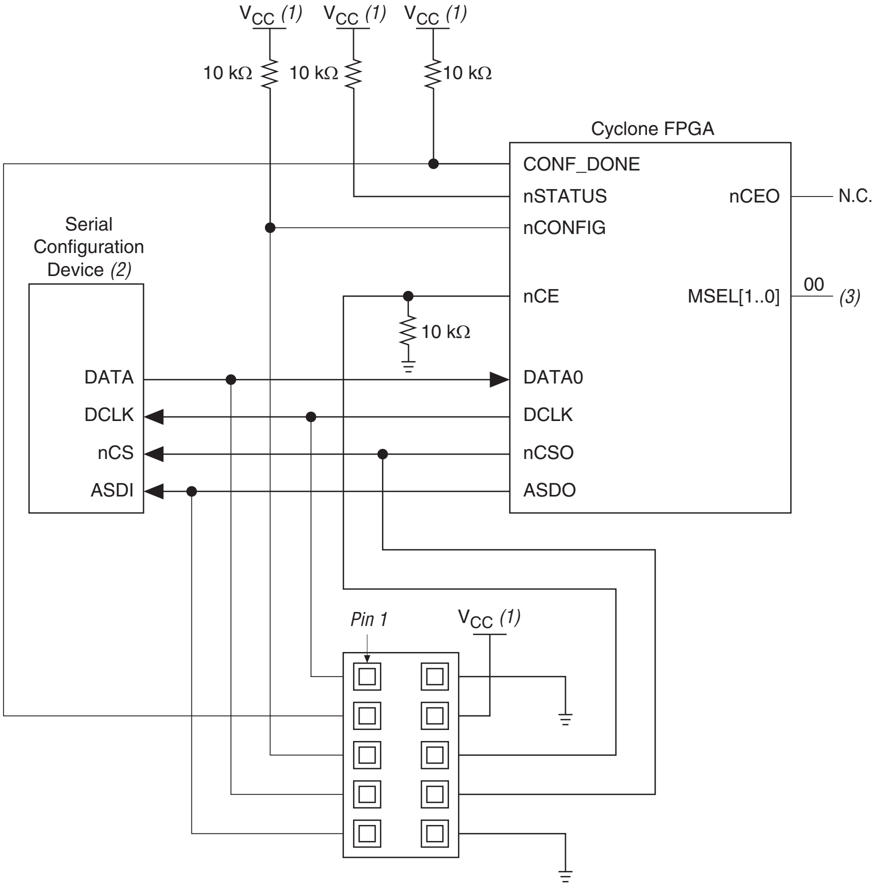
The following Altera FPGAs support Active Serial (AS) configuration scheme with serial configuration devices:

- Stratix III
- Stratix II GX
- Stratix II
- Arria GX
- Cyclone series FPGAs

☞ This section is only relevant for FPGAs that support the AS configuration scheme.

There are four signals on the serial configuration device that interface directly with the FPGA's control signals. The serial configuration device signals `DATA`, `DCLK`, `ASDI`, and `nCS` interface with `DATA0`, `DCLK`, `ASDO`, and `nCSO` control signals on the FPGA, respectively. Figure 14–2 shows a serial configuration device programmed via a download cable, which configures an FPGA in AS mode. Figure 14–3 shows a serial configuration device programmed using the APU or a third-party programmer configuring an FPGA in AS configuration mode.

*Figure 14–2. Cyclone FPGA Configuration in AS Mode (Serial Configuration Device Programmed Using Download Cable)* *Note (4)*



*Notes to Figure 14–2:*

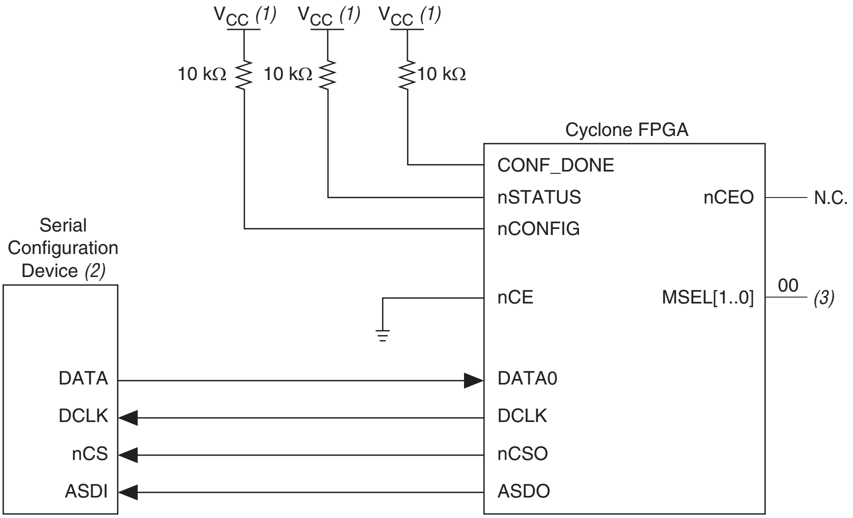
(1) $V_{CC}$ = 3.3 V.

(2) Serial configuration devices cannot be cascaded.

(3) Connect the FPGA `MSEL[]` input pins to select the AS configuration mode. For details, refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

(4) For more information about configuration pin I/O requirements in an AS scheme for a Cyclone III FPGA, refer to the *Configuring Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*.

*Figure 14–3. Cyclone FPGA Configuration in AS Mode (Serial Configuration Device Programmed by APU or Third-Party Programmer) Note (4)*



*Notes to Figure 14–3:*

(1) $V_{CC}$ = 3.3 V.

(2) Serial configuration devices cannot be cascaded.

(3) Connect the FPGA MSEL[] input pins to select the AS configuration mode. For details, refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

(4) For more information about configuration pin I/O requirements in an AS scheme for a Cyclone III FPGA, refer to the *Configuring Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*.

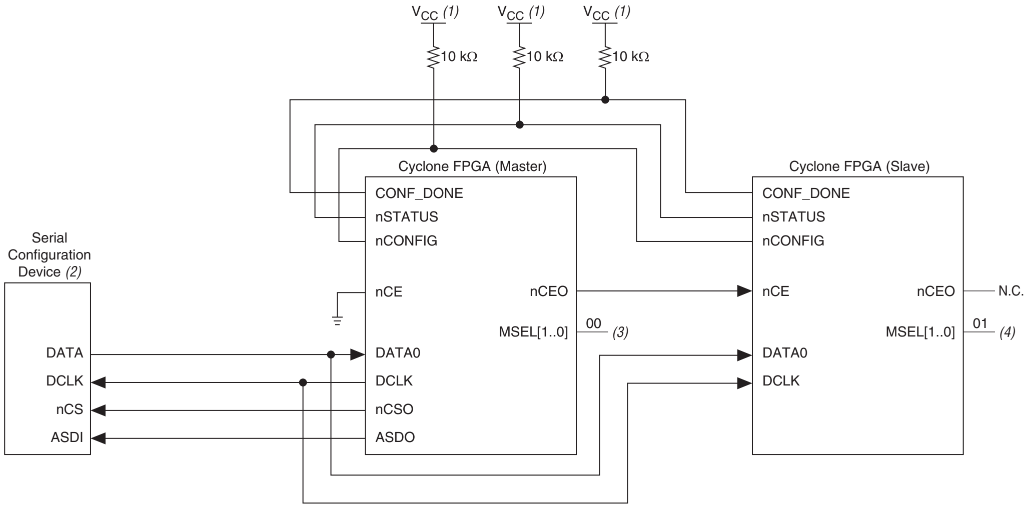
The FPGA acts as the configuration master in the configuration flow and provides the clock to the serial configuration device. The FPGA enables the serial configuration device by pulling the nCS signal low via the nCSO signal (refer to Figures 14–2 and 14–3). Subsequently, the FPGA sends the instructions and addresses to the serial configuration device via the ASDO signal. The serial configuration device responds to the instructions by sending the configuration data to the FPGA's DATA0 pin on the falling edge of DCLK. The data is latched into the FPGA on the DCLK signal's falling edge.

The FPGA controls the nSTATUS and CONF_DONE pins during configuration in AS mode. If the CONF_DONE signal does not go high at the end of configuration or if the signal goes high too early, the FPGA will pulse its nSTATUS pin low to start reconfiguration. Upon successful configuration, the FPGA releases the CONF_DONE pin, allowing the external 10-kΩ resistor to pull this signal high. Initialization begins after the CONF_DONE goes high. After initialization, the FPGA enters user mode.

Refer to the configuration chapter in the appropriate device handbook for more information about configuring the FPGAs in AS mode or other configuration modes.

Multiple devices can be configured by a single EPCS device. However, serial configuration devices cannot be cascaded. Refer to Table 14–1 to ensure the programming file size of the cascaded FPGAs does not exceed the capacity of a serial configuration device. Figure 14–4 shows the AS configuration scheme with multiple FPGAs in the chain. The first FPGA is the configuration master and has its `MSEL[]` pins set to AS mode. The following FPGAs are configuration slave devices and have their `MSEL[]` pins set to PS mode.

***Figure 14–4. Multiple Devices in AS Mode** Note (5)*



*Notes to Figure 14–4:*

(1) $V_{CC}$ = 3.3 V.

(2) Serial configuration devices cannot be cascaded.

(3) Connect the FPGA `MSEL[]` input pins to select the AS configuration mode. For details, refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

(4) Connect the FPGA `MSEL[]` input pins to select the PS configuration mode. For details, refer to the appropriate FPGA family chapter in the *Configuration Handbook*.

(5) For more information about configuration pin I/O requirements in an AS scheme for a Cyclone III FPGA, refer to the *Configuring Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*.

## Serial Configuration Device Memory Access

This section describes the serial configuration device's memory array organization and operation codes. Timing specifications for the memory are provided in the "Timing Information" section.

### Memory Array Organization

Table 14–9 provides details about the memory array organization in EPCS128, EPCS64, EPCS16, EPCS4, and EPCS1.

*Table 14–9. Memory Array Organization in Serial Configuration Devices*

| Details | EPCS128 | EPCS64 | EPCS16 | EPCS4 | EPCS1 |
|---|---|---|---|---|---|
| Bytes (bits) | 16,777,216 bytes (128 Mbits) | 8,388,608 bytes (64 Mbits) | 2,097,152 bytes (16 Mbits) | 524,288 bytes (4 Mbits) | 131,072 bytes (1 Mbit) |
| Number of sectors | 64 | 128 | 32 | 8 | 4 |
| Bytes (bits) per sector | 262,144 (2 Mbits) | 65,536 bytes (512 Kbits) | 65,536 bytes (512 Kbits) | 65,536 bytes (512 Kbits) | 32,768 bytes (256 Kbits) |
| Pages per sector | 1,024 | 256 | 256 | 256 | 128 |
| Total number of pages | 65,536 | 32,768 | 8,192 | 2,048 | 512 |
| Bytes per page | 256 bytes | 256 bytes | 256 bytes | 256 bytes | 256 bytes |

Tables 14–10 through 14–14 show the address range for each sector in EPCS128, EPCS64, EPCS16, EPCS4, and EPCS1.

*Table 14–10. Address Range for Sectors in EPCS128 (Part 1 of 3)*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 63 | `H'FC0000` | `H'FFFFFF` |
| 62 | `H'F80000` | `H'FBFFFF` |
| 61 | `H'F40000` | `H'F7FFFF` |
| 60 | `H'F00000` | `H'F3FFFF` |
| 59 | `H'EC0000` | `H'EFFFFF` |
| 58 | `H'E80000` | `H'EBFFFF` |
| 57 | `H'E40000` | `H'E7FFFF` |
| 56 | `H'E00000` | `H'E3FFFF` |
| 55 | `H'DC0000` | `H'DFFFFF` |
| 54 | `H'D80000` | `H'DBFFFF` |

*Table 14–10. Address Range for Sectors in EPCS128 (Part 2 of 3)*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 53 | H'D40000 | H'D7FFFF |
| 52 | H'D00000 | H'D3FFFF |
| 51 | H'CC0000 | H'CFFFFF |
| 50 | H'C80000 | H'CBFFFF |
| 49 | H'C40000 | H'C7FFFF |
| 48 | H'C00000 | H'C3FFFF |
| 47 | H'BC0000 | H'BFFFFF |
| 46 | H'B80000 | H'BBFFFF |
| 45 | H'B40000 | H'B7FFFF |
| 44 | H'B00000 | H'B3FFFF |
| 43 | H'AC0000 | H'AFFFFF |
| 42 | H'A80000 | H'ABFFFF |
| 41 | H'A40000 | H'A7FFFF |
| 40 | H'A00000 | H'A3FFFF |
| 39 | H'9C0000 | H'9FFFFF |
| 38 | H'980000 | H'9BFFFF |
| 37 | H'940000 | H'97FFFF |
| 36 | H'900000 | H'93FFFF |
| 35 | H'8C0000 | H'8FFFFF |
| 34 | H'880000 | H'8BFFFF |
| 33 | H'840000 | H'87FFFF |
| 32 | H'800000 | H'83FFFF |
| 31 | H'7C0000 | H'7FFFFF |
| 30 | H'780000 | H'7BFFFF |
| 29 | H'740000 | H'77FFFF |
| 28 | H'700000 | H'73FFFF |
| 27 | H'6C0000 | H'6FFFFF |
| 26 | H'680000 | H'6BFFFF |
| 25 | H'640000 | H'67FFFF |
| 24 | H'600000 | H'63FFFF |
| 23 | H'5C0000 | H'5FFFFF |
| 22 | H'580000 | H'5BFFFF |

*Table 14–10. Address Range for Sectors in EPCS128 (Part 3 of 3)*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 21 | H'540000 | H'57FFFF |
| 20 | H'500000 | H'53FFFF |
| 19 | H'4C0000 | H'4FFFFF |
| 18 | H'480000 | H'4BFFFF |
| 17 | H'440000 | H'47FFFF |
| 16 | H'400000 | H'43FFFF |
| 15 | H'3C0000 | H'3FFFFF |
| 14 | H'380000 | H'3BFFFF |
| 13 | H'340000 | H'37FFFF |
| 12 | H'300000 | H'33FFFF |
| 11 | H'2C0000 | H'2FFFFF |
| 10 | H'280000 | H'2BFFFF |
| 9 | H'240000 | H'27FFFF |
| 8 | H'200000 | H'23FFFF |
| 7 | H'1C0000 | H'1FFFFF |
| 6 | H'180000 | H'1BFFFF |
| 5 | H'140000 | H'17FFFF |
| 4 | H'100000 | H'13FFFF |
| 3 | H'0C0000 | H'0FFFFF |
| 2 | H'080000 | H'0BFFFF |
| 1 | H'040000 | H'07FFFF |
| 0 | H'000000 | H'03FFFF |

*Table 14–11. Address Range for Sectors in EPCS64 (Part 1 of 5)*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 127 | H'7F0000 | H'7FFFFF |
| 126 | H'7E0000 | H'7EFFFF |
| 125 | H'7D0000 | H'7DFFFF |
| 124 | H'7C0000 | H'7CFFFF |
| 123 | H'7B0000 | H'7BFFFF |

*Table 14–11. Address Range for Sectors in EPCS64 (Part 2 of 5)*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 122 | `H'7A0000` | `H'7AFFFF` |
| 121 | `H'790000` | `H'79FFFF` |
| 120 | `H'780000` | `H'78FFFF` |
| 119 | `H'770000` | `H'77FFFF` |
| 118 | `H'760000` | `H'76FFFF` |
| 117 | `H'750000` | `H'75FFFF` |
| 116 | `H'740000` | `H'74FFFF` |
| 115 | `H'730000` | `H'73FFFF` |
| 114 | `H'720000` | `H'72FFFF` |
| 113 | `H'710000` | `H'71FFFF` |
| 112 | `H'700000` | `H'70FFFF` |
| 111 | `H'6F0000` | `H'6FFFFF` |
| 110 | `H'6E0000` | `H'6EFFFF` |
| 109 | `H'6D0000` | `H'6DFFFF` |
| 108 | `H'6C0000` | `H'6CFFFF` |
| 107 | `H'6B0000` | `H'6BFFFF` |
| 106 | `H'6A0000` | `H'6AFFFF` |
| 105 | `H'690000` | `H'69FFFF` |
| 104 | `H'680000` | `H'68FFFF` |
| 103 | `H'670000` | `H'67FFFF` |
| 102 | `H'660000` | `H'66FFFF` |
| 101 | `H'650000` | `H'65FFFF` |
| 100 | `H'640000` | `H'64FFFF` |
| 99 | `H'630000` | `H'63FFFF` |
| 98 | `H'620000` | `H'62FFFF` |
| 97 | `H'610000` | `H'61FFFF` |
| 96 | `H'600000` | `H'60FFFF` |
| 95 | `H'5F0000` | `H'5FFFFF` |
| 94 | `H'5E0000` | `H'5EFFFF` |
| 93 | `H'5D0000` | `H'5DFFFF` |
| 92 | `H'5C0000` | `H'5CFFFF` |
| 91 | `H'5B0000` | `H'5BFFFF` |

*Table 14–11. Address Range for Sectors in EPCS64 (Part 3 of 5)*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 90 | H'5A0000 | H'5AFFFF |
| 89 | H'590000 | H'59FFFF |
| 88 | H'580000 | H'58FFFF |
| 87 | H'570000 | H'57FFFF |
| 86 | H'560000 | H'56FFFF |
| 85 | H'550000 | H'55FFFF |
| 84 | H'540000 | H'54FFFF |
| 83 | H'530000 | H'53FFFF |
| 82 | H'520000 | H'52FFFF |
| 81 | H'510000 | H'51FFFF |
| 80 | H'500000 | H'50FFFF |
| 79 | H'4F0000 | H'4FFFFF |
| 78 | H'4E0000 | H'4EFFFF |
| 77 | H'4D0000 | H'4DFFFF |
| 76 | H'4C0000 | H'4CFFFF |
| 75 | H'4B0000 | H'4BFFFF |
| 74 | H'4A0000 | H'4AFFFF |
| 73 | H'490000 | H'49FFFF |
| 72 | H'480000 | H'48FFFF |
| 71 | H'470000 | H'47FFFF |
| 70 | H'460000 | H'46FFFF |
| 69 | H'450000 | H'45FFFF |
| 68 | H'440000 | H'44FFFF |
| 67 | H'430000 | H'43FFFF |
| 66 | H'420000 | H'42FFFF |
| 65 | H'410000 | H'41FFFF |
| 64 | H'400000 | H'40FFFF |
| 63 | H'3F0000 | H'3FFFFF |
| 62 | H'3E0000 | H'3EFFFF |
| 61 | H'3D0000 | H'3DFFFF |
| 60 | H'3C0000 | H'3CFFFF |
| 59 | H'3B0000 | H'3BFFFF |

*Table 14–11. Address Range for Sectors in EPCS64 (Part 4 of 5)*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 58 | H'3A0000 | H'3AFFFF |
| 57 | H'390000 | H'39FFFF |
| 56 | H'380000 | H'38FFFF |
| 55 | H'370000 | H'37FFFF |
| 54 | H'360000 | H'36FFFF |
| 53 | H'350000 | H'35FFFF |
| 52 | H'340000 | H'34FFFF |
| 51 | H'330000 | H'33FFFF |
| 50 | H'320000 | H'32FFFF |
| 49 | H'310000 | H'31FFFF |
| 48 | H'300000 | H'30FFFF |
| 47 | H'2F0000 | H'2FFFFF |
| 46 | H'2E0000 | H'2EFFFF |
| 45 | H'2D0000 | H'2DFFFF |
| 44 | H'2C0000 | H'2CFFFF |
| 43 | H'2B0000 | H'2BFFFF |
| 42 | H'2A0000 | H'2AFFFF |
| 41 | H'290000 | H'29FFFF |
| 40 | H'280000 | H'28FFFF |
| 39 | H'270000 | H'27FFFF |
| 38 | H'260000 | H'26FFFF |
| 37 | H'250000 | H'25FFFF |
| 36 | H'240000 | H'24FFFF |
| 35 | H'230000 | H'23FFFF |
| 34 | H'220000 | H'22FFFF |
| 33 | H'210000 | H'21FFFF |
| 32 | H'200000 | H'20FFFF |
| 31 | H'1F0000 | H'1FFFFF |
| 30 | H'1E0000 | H'1EFFFF |
| 29 | H'1D0000 | H'1DFFFF |
| 28 | H'1C0000 | H'1CFFFF |
| 27 | H'1B0000 | H'1BFFFF |

*Table 14–11. Address Range for Sectors in EPCS64 (Part 5 of 5)*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 26 | `H'1A0000` | `H'1AFFFF` |
| 25 | `H'190000` | `H'19FFFF` |
| 24 | `H'180000` | `H'18FFFF` |
| 23 | `H'170000` | `H'17FFFF` |
| 22 | `H'160000` | `H'16FFFF` |
| 21 | `H'150000` | `H'15FFFF` |
| 20 | `H'140000` | `H'14FFFF` |
| 19 | `H'130000` | `H'13FFFF` |
| 18 | `H'120000` | `H'12FFFF` |
| 17 | `H'110000` | `H'11FFFF` |
| 16 | `H'100000` | `H'10FFFF` |
| 15 | `H'0F0000` | `H'0FFFFF` |
| 14 | `H'0E0000` | `H'0EFFFF` |
| 13 | `H'0D0000` | `H'0DFFFF` |
| 12 | `H'0C0000` | `H'0CFFFF` |
| 11 | `H'0B0000` | `H'0BFFFF` |
| 10 | `H'0A0000` | `H'0AFFFF` |
| 9 | `H'090000` | `H'09FFFF` |
| 8 | `H'080000` | `H'08FFFF` |
| 7 | `H'070000` | `H'07FFFF` |
| 6 | `H'060000` | `H'06FFFF` |
| 5 | `H'050000` | `H'05FFFF` |
| 4 | `H'040000` | `H'04FFFF` |
| 3 | `H'030000` | `H'03FFFF` |
| 2 | `H'020000` | `H'02FFFF` |
| 1 | `H'010000` | `H'01FFFF` |
| 0 | `H'000000` | `H'00FFFF` |

*Table 14–12. Address Range for Sectors in EPCS16*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 31 | H'1F0000 | H'1FFFFF |
| 30 | H'1E0000 | H'1EFFFF |
| 29 | H'1D0000 | H'1DFFFF |
| 28 | H'1C0000 | H'1CFFFF |
| 27 | H'1B0000 | H'1BFFFF |
| 26 | H'1A0000 | H'1AFFFF |
| 25 | H'190000 | H'19FFFF |
| 24 | H'180000 | H'18FFFF |
| 23 | H'170000 | H'17FFFF |
| 22 | H'160000 | H'16FFFF |
| 21 | H'150000 | H'15FFFF |
| 20 | H'140000 | H'14FFFF |
| 19 | H'130000 | H'13FFFF |
| 18 | H'120000 | H'12FFFF |
| 17 | H'110000 | H'11FFFF |
| 16 | H'100000 | H'10FFFF |
| 15 | H'0F0000 | H'0FFFFF |
| 14 | H'0E0000 | H'0EFFFF |
| 13 | H'0D0000 | H'0DFFFF |
| 12 | H'0C0000 | H'0CFFFF |
| 11 | H'0B0000 | H'0BFFFF |
| 10 | H'0A0000 | H'0AFFFF |
| 9 | H'090000 | H'09FFFF |
| 8 | H'080000 | H'08FFFF |
| 7 | H'070000 | H'07FFFF |
| 6 | H'060000 | H'06FFFF |
| 5 | H'050000 | H'05FFFF |
| 4 | H'040000 | H'04FFFF |
| 3 | H'030000 | H'03FFFF |
| 2 | H'020000 | H'02FFFF |
| 1 | H'010000 | H'01FFFF |
| 0 | H'000000 | H'00FFFF |

*Table 14–13. Address Range for Sectors in EPCS4*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 7 | H'70000 | H'7FFFF |
| 6 | H'60000 | H'6FFFF |
| 5 | H'50000 | H'5FFFF |
| 4 | H'40000 | H'4FFFF |
| 3 | H'30000 | H'3FFFF |
| 2 | H'20000 | H'2FFFF |
| 1 | H'10000 | H'1FFFF |
| 0 | H'00000 | H'0FFFF |

*Table 14–14. Address Range for Sectors in EPCS1*

| Sector | Address Range (Byte Addresses in HEX) | |
|---|---|---|
| | Start | End |
| 3 | H'18000 | H'1FFFF |
| 2 | H'10000 | H'17FFF |
| 1 | H'08000 | H'0FFFF |
| 0 | H'00000 | H'07FFF |

## Operation Codes

This section describes the operations that can be used to access the memory in serial configuration devices. The DATA, DCLK, ASDI, and nCS signals access the memory in serial configuration devices. All serial configuration device operation codes, addresses and data are shifted in and out of the device serially, with the most significant bit (MSB) first.

The device samples the active serial data input on the first rising edge of the DCLK after the active low chip select (nCS) input signal is driven low. Shift the operation code (MSB first) serially into the serial configuration device through the active serial data input pin. Each operation code bit is latched into the serial configuration device on the rising edge of the DCLK.

Different operations require a different sequence of inputs. While executing an operation, you must shift in the desired operation code, followed by the address bytes, data bytes, both, or neither. The device must drive nCS high after the last bit of the operation sequence is shifted in. Table 14–15 shows the operation sequence for every operation supported by the serial configuration devices.

For the read byte, read status, and read silicon ID operations, the shifted-in operation sequence is followed by data shifted out on the DATA pin. You can drive the nCS pin high after any bit of the data-out sequence is shifted out.

For the write byte, erase bulk, erase sector, write enable, write disable, and write status operations, drive the nCS pin high exactly at a byte boundary (drive the nCS pin high a multiple of eight clock pulses after the nCS pin is driven low); otherwise, the operation is rejected and is not executed.

All attempts to access the memory contents while a write or erase cycle is in progress will not be granted, and the write or erase cycle will continue unaffected.

*Table 14–15. Operation Codes for Serial Configuration Devices*

| Operation | Operation Code *(1)* | Address Bytes | Dummy Bytes | Data Bytes | DCLK $f_{MAX}$ (MHz) |
|---|---|---|---|---|---|
| Write enable | 0000 0110 | 0 | 0 | 0 | 25 |
| Write disable | 0000 0100 | 0 | 0 | 0 | 25 |
| Read status | 0000 0101 | 0 | 0 | 1 to infinite *(2)* | 25 |
| Read bytes | 0000 0011 | 3 | 0 | 1 to infinite *(2)* | 20 |
| Read silicon ID *(4)* | 1010 1011 | 0 | 3 | 1 to infinite *(2)* | 25 |
| Write status | 0000 0001 | 0 | 0 | 1 | 25 |
| Write bytes | 0000 0010 | 3 | 0 | 1 to 256 *(3)* | 25 |
| Erase bulk | 1100 0111 | 0 | 0 | 0 | 25 |
| Erase sector | 1101 1000 | 3 | 0 | 0 | 25 |
| Read Device Identification *(5)* | 1001 1111 | 0 | 2 | 1 to infinite *(2)* | 25 |

*Notes to Table 14–15:*

(1) The MSB is listed first and the least significant bit (LSB) is listed last.

(2) The status register, data or silicon ID are read out at least once on the DATA pin and will continuously be read out until nCS is driven high.

(3) Write bytes operation requires at least one data byte on the DATA pin. If more than 256 bytes are sent to the device, only the last 256 bytes are written to the memory.

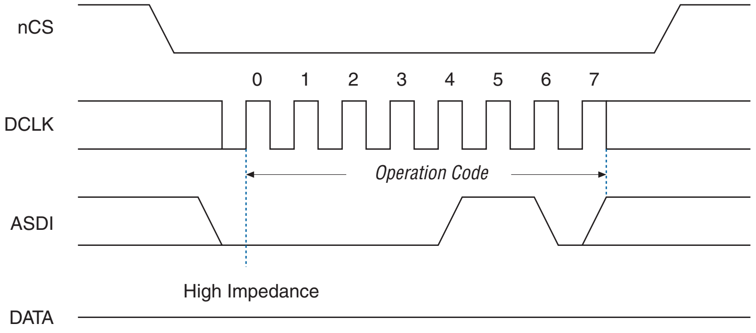
(4) Read silicon ID operation is available only for EPCS1, EPCS4, EPCS16, and EPCS64.

(5) Read Device Identification operation is available only for EPCS128.

### Write Enable Operation

The write enable operation code is `b'0000 0110`, and the MSB is listed first. The write enable operation sets the write enable latch bit, which is bit `1` in the status register. Always set the write enable latch bit before write bytes, write status, erase bulk, and erase sector operations. Figure 14–5 shows the timing diagram for the write enable operation. Figures 14–7 and 14–8 show the status register bit definitions.

**Figure 14–5. Write Enable Operation Timing Diagram**



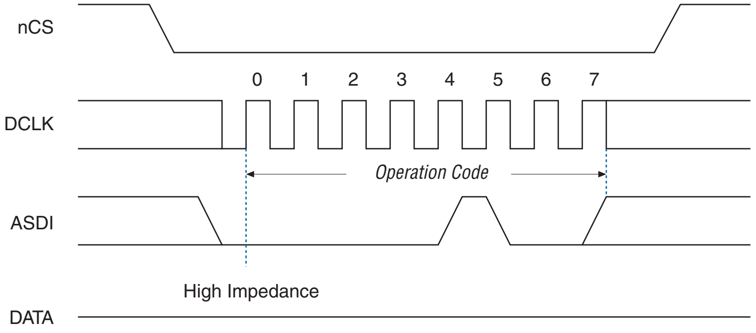
### Write Disable Operation

The write disable operation code is `b'0000 0100`, with the MSB listed first. The write disable operation resets the write enable latch bit, which is bit `1` in the status register. To prevent the memory from being written unintentionally, the write enable latch bit is automatically reset when implementing the write disable operation as well as under the following conditions:

- Power up
- Write bytes operation completion
- Write status operation completion
- Erase bulk operation completion
- Erase sector operation completion

Figure 14–6 shows the timing diagram for the write disable operation.

**Figure 14–6. Write Disable Operation Timing Diagram**



### Read Status Operation

The read status operation code is `b'0000 0101`, with the MSB listed first. You can use the read status operation to read the status register. Figures 14–7 and 14–8 show the status bits in the status register of both serial configuration devices.

**Figure 14–7. EPCS4, EPCS16, EPCS64, and EPCS128 Status Register Status Bits**

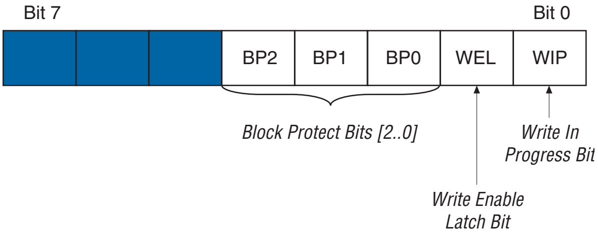
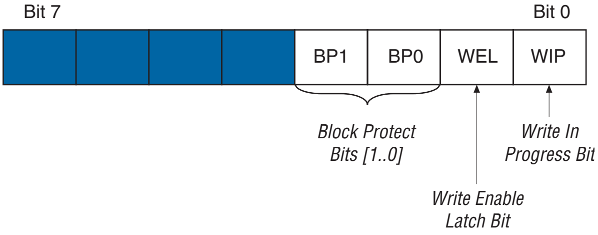


**Figure 14–8. EPCS1 Status Register Status Bits**



Setting the write in progress bit to 1 indicates that the serial configuration device is busy with a write or erase cycle. Resetting the write in progress bit to 0 means no write or erase cycle is in progress.

Resetting the write enable latch bit to 0 indicates that no write or erase cycle will be accepted. Set the write enable latch bit to 1 before every write bytes, write status, erase bulk, and erase sector operation.

The non-volatile block protect bits determine the area of the memory protected from being written or erased unintentionally. Table 14–16 through Table 14–20 show the protected area in the serial configuration devices with reference to the block protect bits. The erase bulk operation is only available when all the block protect bits are 0. When any of the block protect bits are set to 1, the relevant area is protected from being written by write bytes operations or erased by erase sector operations.

*Table 14–16. Block Protection Bits in EPCS1*

| Status Register Content | | Memory Content | |
|---|---|---|---|
| BP1 Bit | BP0 Bit | Protected Area | Unprotected Area |
| 0 | 0 | None | All four sectors: 0 to 3 |
| 0 | 1 | Sector 3 | Three sectors: 0 to 2 |
| 1 | 0 | Two sectors: 2 and 3 | Two sectors: 0 and 1 |
| 1 | 1 | All sectors | None |

*Table 14–17. Block Protection Bits in EPCS4*

| Status Register Content | | | Memory Content | |
|---|---|---|---|---|
| BP2 Bit | BP1 Bit | BP0 Bit | Protected Area | Unprotected Area |
| 0 | 0 | 0 | None | All eight sectors: 0 to 7 |
| 0 | 0 | 1 | Sector 7 | Seven sectors: 0 to 6 |
| 0 | 1 | 0 | Sectors 6 and 7 | Six sectors: 0 to 5 |
| 0 | 1 | 1 | Four sectors: 4 to 7 | Four sectors: 0 to 3 |
| 1 | 0 | 0 | All sectors | None |
| 1 | 0 | 1 | All sectors | None |
| 1 | 1 | 0 | All sectors | None |
| 1 | 1 | 1 | All sectors | None |

*Table 14–18. Block Protection Bits in EPCS16*

| Status Register Content | | | Memory Content | |
|---|---|---|---|---|
| BP2 Bit | BP1 Bit | BP0 Bit | Protected Area | Unprotected Area |
| 0 | 0 | 0 | None | All sectors (32 sectors 0 to 31) |
| 0 | 0 | 1 | Upper 32nd (Sector 31) | Lower 31/32nds (31 sectors: 0 to 30) |
| 0 | 1 | 0 | Upper sixteenth (two sectors: 30 and 31) | Lower 15/16ths (30 sectors: 0 to 29) |
| 0 | 1 | 1 | Upper eighth (four sectors: 28 to 31) | Lower seven-eighths (28 sectors: 0 to 27) |
| 1 | 0 | 0 | Upper quarter (eight sectors: 24 to 31) | Lower three-quarters (24 sectors: 0 to 23) |
| 1 | 0 | 1 | Upper half (sixteen sectors: 16 to 31) | Lower half (16 sectors: 0 to 15) |
| 1 | 1 | 0 | All sectors (32 sectors: 0 to 31) | None |
| 1 | 1 | 1 | All sectors (32 sectors: 0 to 31) | None |

*Table 14–19. Block Protection Bits in EPCS64*

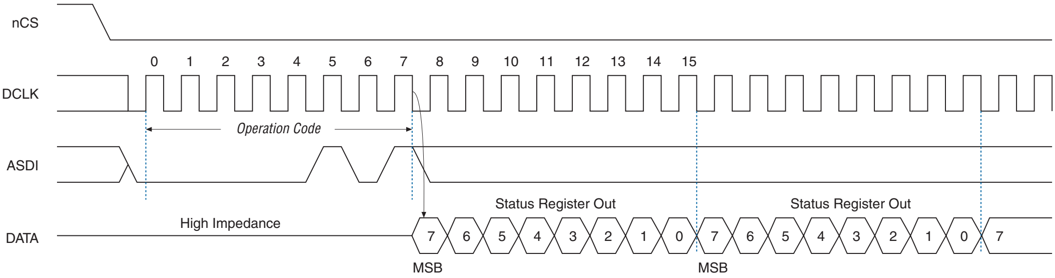
| Status Register Content | | | Memory Content | |
|---|---|---|---|---|
| BP2 Bit | BP1 Bit | BP0 Bit | Protected Area | Unprotected Area |
| 0 | 0 | 0 | None | All sectors (128 sectors: 0 to 127) |
| 0 | 0 | 1 | Upper 64th (2 sectors: 126 and 127) | Lower 63/64ths (126 sectors: 0 to 125) |
| 0 | 1 | 0 | Upper 32nd (4 sectors: 124 to 127) | Lower 31/32nds (124 sectors: 0 to 123) |
| 0 | 1 | 1 | Upper sixteenth (8 sectors: 120 to 127) | Lower 15/16ths (120 sectors: 0 to 119) |
| 1 | 0 | 0 | Upper eighth (16 sectors: 112 to 127) | Lower seven-eighths (112 sectors: 0 to 111) |
| 1 | 0 | 1 | Upper quarter (32 sectors: 96 to 127) | Lower three-quarters (96 sectors: 0 to 95) |
| 1 | 1 | 0 | Upper half (64 sectors: 64 to 127) | Lower half (64 sectors: 0 to 63) |
| 1 | 1 | 1 | All sectors (128 sectors: 0 to 127) | None |

*Table 14–20. Block Protection Bits in EPCS128*

| Status Register Content | | | Memory Content | |
|---|---|---|---|---|
| BP2 Bit | BP1 Bit | BP0 Bit | Protected Area | Unprotected Area |
| 0 | 0 | 0 | None | All sectors (64 sectors: 0 to 63) |
| 0 | 0 | 1 | Upper 64th (1 sector: 63) | Lower 63/64ths (63 sectors: 0 to 62) |
| 0 | 1 | 0 | Upper 32nd (2 sectors: 62 to 63) | Lower 31/32nds (62 sectors: 0 to 61) |
| 0 | 1 | 1 | Upper 16th (4 sectors: 60 to 63) | Lower 15/16ths (60 sectors: 0 to 59) |
| 1 | 0 | 0 | Upper 8th (8 sectors: 56 to 63) | Lower seven-eighths (56 sectors: 0 to 55) |
| 1 | 0 | 1 | Upper quarter (16 sectors: 48 to 63) | Lower three-quarters (48 sectors: 0 to 47) |
| 1 | 1 | 0 | Upper half (32 sectors: 32 to 63) | Lower half (32 sectors: 0 to 31) |
| 1 | 1 | 1 | All sectors (64 sectors: 0 to 63) | None |

You can read the status register at any time, even while a write or erase cycle is in progress. When one of these cycles is in progress, you can check the write in progress bit (bit 0 of the status register) before sending a new operation to the device. The device can also read the status register continuously, as shown in Figure 14–9.

*Figure 14–9. Read Status Operation Timing Diagram*



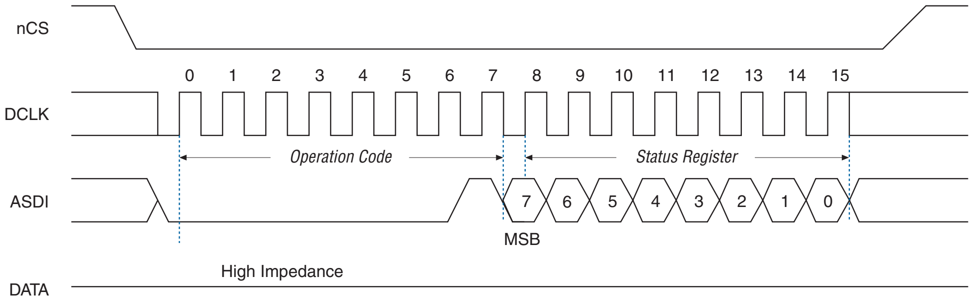
### *Write Status Operation*

The write status operation code is b'0000 0001, with the MSB listed first. Use the write status operation to set the status register block protection bits. The write status operation has no effect on the other bits. Therefore, you can implement this operation to protect certain memory sectors, as defined in Table 14–16 through Table 14–20. After setting the block protect bits, the protected memory sectors are treated as read-only memory. You must execute the write enable operation before the write status operation so the device sets the status register's write enable latch bit to 1.

The write status operation is implemented by driving nCS low, followed by shifting in the write status operation code and one data byte for the status register on the ASDI pin. Figure 14–10 shows the timing diagram for the write status operation. nCS must be driven high after the eighth bit of the data byte has been latched in, otherwise, the write status operation is not executed.

Immediately after nCS drives high, the device initiates the self-timed write status cycle. The self-timed write status cycle usually takes 5 ms for all serial configuration devices and is guaranteed to be less than 15 ms (refer to $t_{WS}$ in Table 14–23). You must account for this delay to ensure that the status register is written with desired block protect bits. Alternatively, you can check the write in progress bit in the status register by executing the read status operation while the self-timed write status cycle is in progress. The write in progress bit is 1 during the self-timed write status cycle, and 0 when it is complete.

*Figure 14–10. Write Status Operation Timing Diagram*



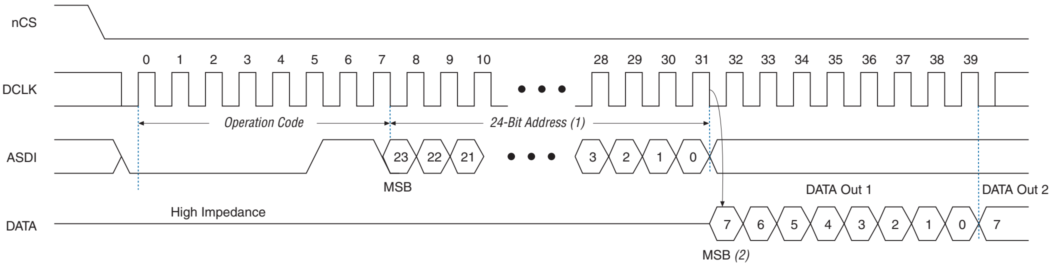
## Read Bytes Operation

The read bytes operation code is b'0000 0011, with the MSB listed first. To read the memory contents of the serial configuration device, the device is first selected by driving nCS low. Then, the read bytes operation code is shifted in followed by a 3-byte address (A[23..0]). Each address bit must be latched in on the rising edge of the DCLK. After the address is latched in, the memory contents of the specified address are shifted out serially on the DATA pin, beginning with the MSB. For reading Raw Programming Data files (**.rpd**), the content is shifted out serially beginning with the LSB. Each data bit is shifted out on the falling edge of

DCLK. The maximum DCLK frequency during the read bytes operation is 20 MHz. Figure 14–11 shows the timing diagram for the read bytes operation.

The first byte address can be at any location. The device automatically increments the address to the next higher address after shifting out each byte of data. Therefore, the device can read the whole memory with a single read bytes operation. When the device reaches the highest address, the address counter restarts at 0x000000, allowing the memory contents to be read out indefinitely until the read bytes operation is terminated by driving nCS high. The device can drive nCS high any time after data is shifted out. If the read bytes operation is shifted in while a write or erase cycle is in progress, the operation is not executed and has no effect on the write or erase cycle in progress.

***Figure 14–11. Read Bytes Operation Timing Diagram***



*Notes to Figure 14–11:*

(1) Address bit A[23] is a don't-care bit in EPCS64. Address bits A[23..21] are don't-care bits in EPCS16. Address bits A[23..19] are don't-care bits in EPCS4. Address bits A[23..17] are don't-care bits in EPCS1.

(2) For RPD files, the read sequence shifts out the LSB of the data byte first.

### *Read Silicon ID Operation*

The read silicon ID operation code is b'1010 1011, with the MSB listed first. Only EPCS1, EPCS4, EPCS16, and EPCS64 support this operation. It reads the serial configuration device's 8-bit silicon ID from the DATA output pin. If this operation is shifted in during an erase or write cycle, it is ignored and has no effect on the cycle that is in progress.

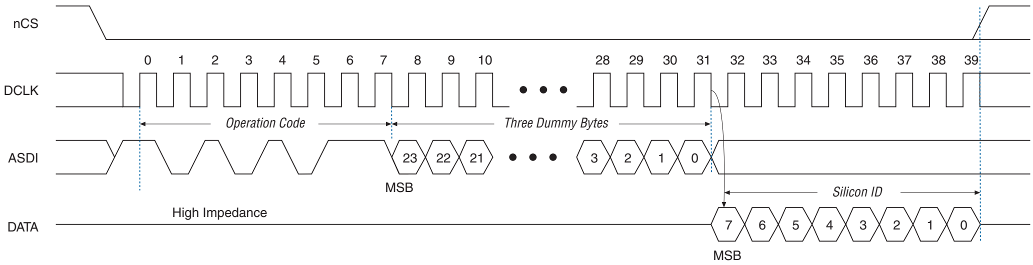
Table 14–21 shows the serial configuration device silicon IDs.

*Table 14–21. Serial Configuration Device Silicon ID*

| Serial Configuration Device | Silicon ID (Binary Value) |
|---|---|
| EPCS1 | `b'0001 0000` |
| EPCS4 | `b'0001 0010` |
| EPCS16 | `b'0001 0100` |
| EPCS64 | `b'0001 0110` |

The device implements the read silicon ID operation by driving `nCS` low then shifting in the read silicon ID operation code followed by three dummy bytes on `ASDI`. The serial configuration device's 8-bit silicon ID is then shifted out on the `DATA` pin on the falling edge of `DCLK`, as shown in Figure 14–12. The device can terminate the read silicon ID operation by driving `nCS` high after the silicon ID has been read at least once. Sending additional clock cycles on `DCLK` while `nCS` is driven low can cause the silicon ID to be shifted out repeatedly.

*Figure 14–12. Read Silicon ID Operation Timing Diagram* Note (1)



*Note to Figure 14–12:*

(1) Only EPCS1, EPCS4, EPCS16, and EPCS64 support Read Silicon ID operation.

### *Read Device Identification Operation*

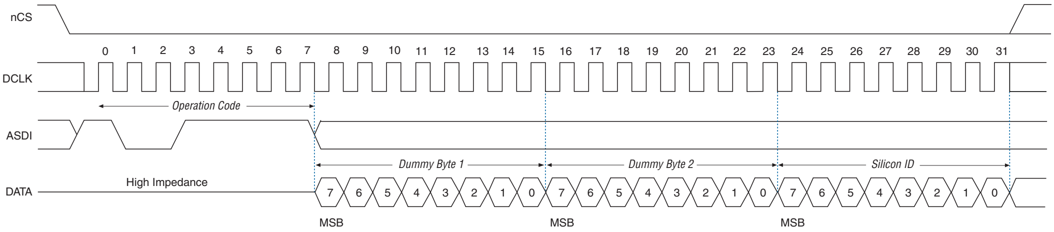
The read device identification operation code is `b'1001 1111`, with the MSB listed first. Only EPCS128 supports this operation. It reads the serial configuration device's 8-bit device identification from the `DATA` output pin. If this operation is shifted in during an erase or write cycle, it is ignored and has no effect on the cycle that is in progress. Table 14–22 shows the serial configuration device identification.

*Table 14–22. Serial Configuration Device Identification*

| Serial Configuration Device | Silicon ID (Binary Value) |
|---|---|
| EPCS128 | b'0001 1000 |

The device implements the read device identification operation by driving nCS low then shifting in the read device identification operation code followed by one dummy byte on ASDI. The serial configuration device's 16-bit device identification is then shifted out on the DATA pin on the falling edge of DCLK, as shown in Figure 14–13. The device can terminate the read device identification operation by driving nCS high after reading the device identification at least once.

*Figure 14–13. Read Device Identification Operation Timing Diagram* *Note (1)*



*Note to Figure 14–13:*

(1) Only EPCS128 supports read device identification operation.

### Write Bytes Operation

The write bytes operation code is b'0000 0010, with the MSB listed first. The write bytes operation allows bytes to be written to the memory. The write enable operation must be executed prior to the write bytes operation to set the write enable latch bit in the status register to 1.

The write bytes operation is implemented by driving nCS low, followed by the write bytes operation code, three address bytes and a minimum one data byte on ASDI. If the eight least significant address bits (A[7..0]) are not all 0, all sent data that goes beyond the end of the current page is not written into the next page. Instead, this data is written at the start address of the same page (from the address whose eight LSBs are all 0). Drive nCS low during the entire write bytes operation sequence, as shown in Figure 14–14.

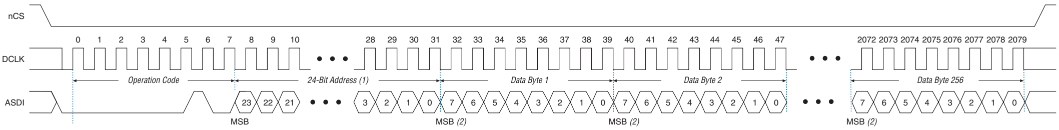
If more than 256 data bytes are shifted into the serial configuration device with a write bytes operation, the previously latched data is discarded and the last 256 bytes are written to the page. However, if less than 256 data bytes are shifted into the serial configuration device, they are guaranteed to be written at the specified addresses and the other bytes of the same page are unaffected.

If the design must write more than 256 data bytes to the memory, it needs more than one page of memory. Send the write enable and write bytes operation codes followed by three new targeted address bytes and 256 data bytes before a new page is written.

`nCS` must be driven high after the eighth bit of the last data byte has been latched in. Otherwise, the device will not execute the write bytes operation. The write enable latch bit in the status register is reset to `0` before the completion of each write bytes operation. Therefore, the write enable operation must be carried out before the next write bytes operation.

The device initiates the self-timed write cycle immediately after `nCS` is driven high. Refer to $t_{WB}$ in Table 14–23 for the self-timed write cycle time for the respective EPCS devices. Therefore, you must account for this amount of delay before another page of memory is written. Alternatively, you can check the status register's write in progress bit by executing the read status operation while the self-timed write cycle is in progress. The write in progress bit is set to `1` during the self-timed write cycle, and `0` when it is complete.

***Figure 14–14. Write Bytes Operation Timing Diagram***



*Notes to Figure 14–14:*

(1) Address bit `A[23]` is a don't-care bit in EPCS64. Address bits `A[23..21]` are don't-care bits in EPCS16. Address bits `A[23..19]` are don't-care bits in EPCS4. Address bits `A[23..17]` are don't-care bits in EPCS1.

(2) For RPD files, write the LSB of the data byte first.

### *Erase Bulk Operation*

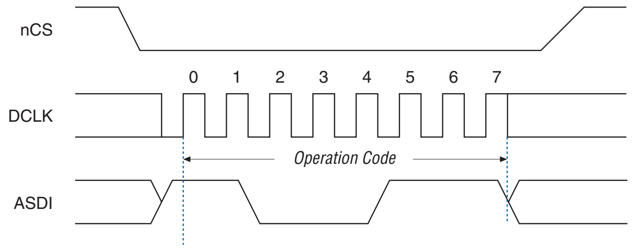
The erase bulk operation code is `b'1100 0111`, with the MSB listed first. The erase bulk operation sets all memory bits to `1` or `0xFF`. Similar to the write bytes operation, the write enable operation must be executed prior to the erase bulk operation so that the write enable latch bit in the status register is set to `1`.

You can implement the erase bulk operation by driving nCS low and then shifting in the erase bulk operation code on the ASDI pin. nCS must be driven high after the eighth bit of the erase bulk operation code has been latched in. Figure 14–15 shows the timing diagram.

The device initiates the self-timed erase bulk cycle immediately after nCS is driven high. Refer to $t_{EB}$ in Table 14–23 for the self-timed erase bulk cycle time for the respective EPCS devices.

You must account for this delay before accessing the memory contents. Alternatively, you can check the write in progress bit in the status register by executing the read status operation while the self-timed erase cycle is in progress. The write in progress bit is 1 during the self-timed erase cycle and 0 when it is complete. The write enable latch bit in the status register is reset to 0 before the erase cycle is complete.

*Figure 14–15. Erase Bulk Operation Timing Diagram*



### Erase Sector Operation

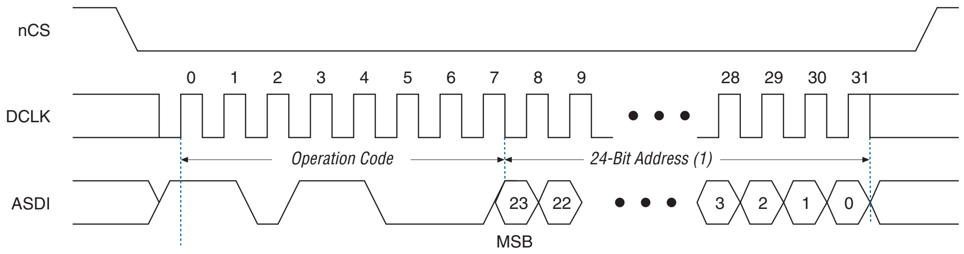
The erase sector operation code is b'1101 1000, with the MSB listed first. The erase sector operation allows the user to erase a certain sector in the serial configuration device by setting all bits inside the sector to 1 or 0xFF. This operation is useful for users who access the unused sectors as general purpose memory in their applications.

The write enable operation must be executed prior to the erase sector operation so that the write enable latch bit in the status register is set to 1.

The erase sector operation is implemented by first driving nCS low, then shifting in the erase sector operation code and the three address bytes of the chosen sector on the ASDI pin. The three address bytes for the erase sector operation can be any address inside the specified sector. (Refer to Tables 14–10 through 14–14 for sector address range information.) Drive nCS high after the eighth bit of the erase sector operation code has been latched in. Figure 14–16 shows the timing diagram.

Immediately after the device drives `nCS` high, the self-timed erase sector cycle is initiated. Refer to $t_{ES}$ in Table 14–23 for the self-timed erase sector cycle time for the respective EPCS devices. You must account for this amount of delay before the memory contents can be accessed. Alternatively, you can check the write in progress bit in the status register by executing the read status operation while the erase cycle is in progress. The write in progress bit is `1` during the self-timed erase cycle and `0` when it is complete. The write enable latch bit in the status register resets to `0` before the erase cycle is complete.

*Figure 14–16. Erase Sector Operation Timing Diagram*



*Note to Figure 14–16:*

(1) Address bit `A[23]` is a don't-care bit in EPCS64. Address bits `A[23..21]` are don't-care bits in EPCS16. Address bits `A[23..19]` are don't-care bits in EPCS4. Address bits `A[23..17]` are don't-care bits in EPCS1.

## Power and Operation

This section describes the power modes, power-on reset (POR) delay, error detection, and initial programming state of serial configuration devices.

### Power Mode

Serial configuration devices support active power and standby power modes. When `nCS` is low, the device is enabled and is in active power mode. The FPGA is configured while in active power mode. When `nCS` is high, the device is disabled but could remain in active power mode until all internal cycles have completed (such as write or erase operations). The serial configuration device then goes into stand-by power mode. The $I_{CC1}$ parameter specifies the $V_{CC}$ supply current when the device is in active power mode and the $I_{CC0}$ parameter specifies the current when the device is in stand-by power mode (refer to Table 14–29).

## Power-On Reset

During initial power-up, a POR delay occurs to ensure the system voltage levels have stabilized. During AS configuration, the FPGA controls the configuration and has a longer POR delay than the serial configuration device.

For the POR delay time, refer to the configuration chapter in the appropriate device handbook.

## Error Detection

During AS configuration with the serial configuration device, the FPGA monitors the configuration status through the `nSTATUS` and `CONF_DONE` pins. If an error condition occurs (`nSTATUS` drives low) or if the `CONF_DONE` pin does not go high, the FPGA will initiate reconfiguration by pulsing the `nSTATUS` and `nCSO` signals, which controls the chip select pin on the serial configuration device (`nCS`).

After an error, configuration automatically restarts if the **Auto-Restart Upon Frame Error** option is turned on in the Quartus II software. If the option is turned off, the system must monitor the `nSTATUS` signal for errors and then pulse the `nCONFIG` signal low to restart configuration.

## Timing Information

Figure 14–17 shows the timing waveform for write operation to the serial configuration device.

*Figure 14–17. Write Operation Timing*

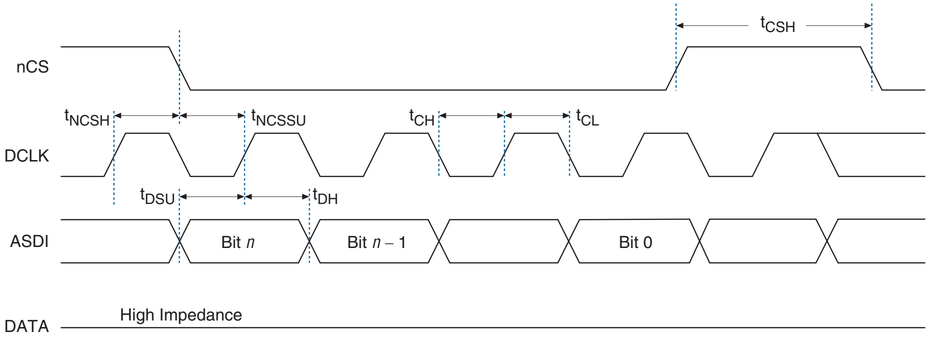


Table 14–23 defines the serial configuration device timing parameters for write operation.

*Table 14–23. Write Operation Parameters (Part 1 of 2)*

| Symbol | Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| $f_{WCLK}$ | Write clock frequency (from FPGA, download cable, or embedded processor) for write enable, write disable, read status, read silicon ID, write bytes, erase bulk, and erase sector operations | — | — | 25 | MHz |
| $t_{CH}$ | DCLK high time | 20 | — | — | ns |
| $t_{CL}$ | DCLK low time | 20 | — | — | ns |
| $t_{NCSSU}$ | Chip select (nCS) setup time | 10 | — | — | ns |
| $t_{NCSH}$ | Chip select (nCS) hold time | 10 | — | — | ns |
| $t_{DSU}$ | Data (ASDI) in setup time before rising edge on DCLK | 5 | — | — | ns |
| $t_{DH}$ | Data (ASDI) hold time after rising edge on DCLK | 5 | — | — | ns |
| $t_{CSH}$ | Chip select high time | 100 | — | — | ns |
| $t_{WB}$ *(1)* | Write bytes cycle time for EPCS1, EPCS4, EPCS16, and EPCS64 | — | 1.5 | 5 | ms |
| | Write bytes cycle time for EPCS128 | — | 2.5 | 7 | ms |
| $t_{WS}$ *(1)* | Write status cycle time | — | 5 | 15 | ms |

*Table 14–23. Write Operation Parameters (Part 2 of 2)*

| Symbol | Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| $t_{EB}$ *(1)* | Erase bulk cycle time for EPCS1 | — | 3 | 6 | s |
| | Erase bulk cycle time for EPCS4 | — | 5 | 10 | s |
| | Erase bulk cycle time for EPCS16 | — | 17 | 40 | s |
| | Erase bulk cycle time for EPCS64 | — | 68 | 160 | s |
| | Erase bulk cycle time for EPCS128 | — | 105 | 250 | s |
| $t_{ES}$ *(1)* | Erase sector cycle time for EPCS1, EPCS4, EPCS16, and EPCS64 | — | 2 | 3 | s |
| | Erase sector cycle time for EPCS128 | — | 2 | 6 | s |

*Note to Table 14–23:*

(1) These parameters are not shown in Figure 14–17.

Figure 14–18 shows the timing waveform for the serial configuration device's read operation.

*Figure 14–18. Read Operation Timing*

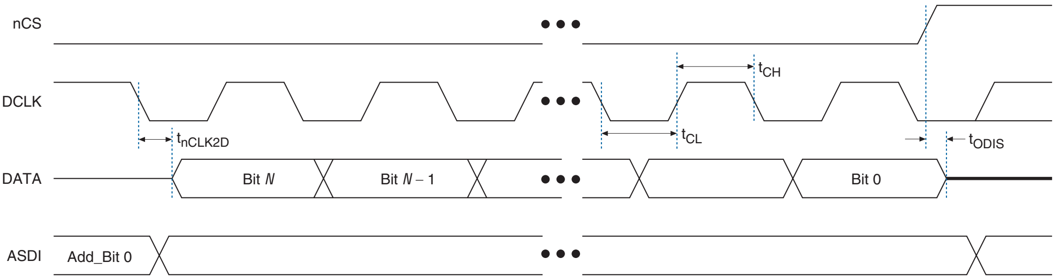


Table 14–24 defines the serial configuration device timing parameters for read operation.

*Table 14–24. Read Operation Parameters (Part 1 of 2)*

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| $f_{RCLK}$ | Read clock frequency (from FPGA or embedded processor) for read bytes operation | — | 20 | MHz |
| $t_{CH}$ | `DCLK` high time | 25 | — | ns |
| $t_{CL}$ | `DCLK` low time | 25 | — | ns |

*Table 14–24. Read Operation Parameters (Part 2 of 2)*

| Symbol | Parameter | Min | Max | Unit |
|---|---|---|---|---|
| $t_{ODIS}$ | Output disable time after read | — | 15 | ns |
| $t_{nCLK2D}$ | Clock falling edge to data | — | 15 | ns |

Figure 14–19 shows the timing waveform for FPGA AS configuration scheme using a serial configuration device.

*Figure 14–19. AS Configuration Timing*

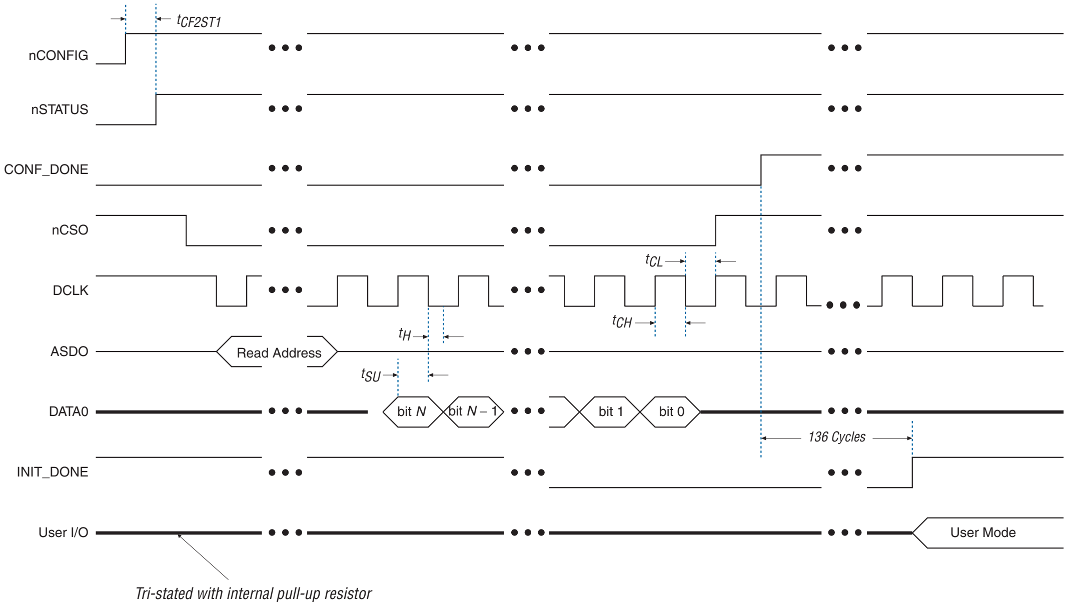


Table 14–25 shows the timing parameters for AS configuration mode.

*Table 14–25. Timing Parameters for AS Configuration*

| Symbol | Parameter | Min | Typ | Max | Unit |
|---|---|---|---|---|---|
| $f_{CLK}$ | `DCLK` frequency from Cyclone FPGA | 14 | 17 | 20 | MHz |
| | `DCLK` frequency from Stratix II or Cyclone II FPGA (40 MHz) *(1)* | 20 | 26 | 40 | MHz |
| | `DCLK` frequency from Stratix II or Cyclone II FPGA (20 MHz) | 10 | 13 | 20 | MHz |
| | `DCLK` frequency from Cyclone III FPGA *(1)* | 20 | 30 | 40 | MHz |
| | `DCLK` frequency from Stratix III FPGA *(1)* | 15 | 25 | 40 | MHz |
| $t_H$ | Data hold time after rising edge on `DCLK` | 0 | — | — | ns |
| $t_{SU}$ | Data set up time before rising edge on `DCLK` | 5 | — | — | ns |

*Note to Table 14–25:*

(1) Existing batches of EPCS1 and EPCS4 manufactured on 0.15 µm process geometry supports AS configuration up to 40 MHz. However, batches of EPCS1 and EPCS4 manufactured on 0.18 µm process geometry support only up to 20 MHz. EPCS16, EPCS64, and EPCS128 are not affected. For information about product traceability and transition date to differentiate between 0.15 µm process geometry and 0.18 µm process geometry EPCS1 and EPCS4, refer to PCN 0514 Manufacturing Changes on EPCS Family process change notification on the Altera website at www.altera.com.

## Programming and Configuration File Support

The Quartus II design software provides programming support for serial configuration devices. After selecting the serial configuration device, the Quartus II software automatically generates the Programmer Object File (**.pof**) to program the device. The software allows users to select the appropriate serial configuration device density that most efficiently stores the configuration data for a selected FPGA.

The serial configuration device can be programmed in-system by an external microprocessor using SRunner. SRunner is a software driver developed for embedded serial configuration device programming that designers can customize to fit in different embedded systems. The SRunner can read RPD file and write to the serial configuration devices. The programming time is comparable to the Quartus II software programming time. Note that writing and reading the RPD file to the EPCS is different from other data and address bytes. The LSB of RPD bytes must be shifted out first during the read bytes instruction and the LSB of RPD bytes must be shifted in first during the write bytes instruction. This is because the FPGA reads the LSB of the RPD data first during the configuration process.

For more information about SRunner, refer to the *AN 418: SRunner: An Embedded Solution for Serial Configuration Device Programming User Guide* and the source code on the Altera website (www.altera.com).

Serial configuration devices can be programmed using the APU with the appropriate programming adapter (PLMSEPC-8) via the Quartus II software, USB Blaster, EthernetBlaster, or the ByteBlaster II download cable via the Quartus II software. In addition, many third-party programmers, such as BP Microsystems and System General, offer programming hardware that supports serial configuration devices.

During in-system programming of a serial configuration device via the USB Blaster, EthernetBlaster, or ByteBlaster II download cable, the cable pulls `nCONFIG` low to reset the FPGA and overrides the 10-kΩ pull-down resistor on the FPGA's `nCE` pin (refer to Figure 14–2). The download cable then uses the four interface pins (`DATA`, `nCS`, `ASDI`, and `DCLK`) to program the serial configuration device. Once the programming is complete, the download cable releases the serial configuration device's four interface pins and the FPGA's `nCE` pin, and pulses `nCONFIG` to start configuration.

The FPGA can program the serial configuration device in-system using the JTAG interface with the Serial FlashLoader. This solution allows you to indirectly program the serial configuration device using the same JTAG interface that is used to configure the FPGA.

For more information about the Serial FlashLoader, refer to *AN 370: Using the Serial FlashLoader with the Quartus II Software*.

For more information on programming and configuration support, refer to the following documents:

- *Altera Programming Hardware Data Sheet*
- *Programming Hardware Manufacturers*
- *USB-Blaster Download Cable User Guide*
- *ByteBlaster II Download Cable User Guide*
- *EthernetBlaster Communications Cable User Guide*

## Operating Conditions

Tables 14–26 through 14–30 provide information on absolute maximum ratings, recommended operating conditions, DC operating conditions, and capacitance for serial configuration devices.

***Table 14–26. Absolute Maximum Ratings** Note (1) **(Part 1 of 2)***

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage for EPCS1, EPCS4, and EPCS16 | With respect to ground | –0.6 | 4.0 | V |
| | Supply voltage for EPCS64 and EPCS128 | With respect to ground | –0.2 | 4.0 | V |

*Table 14–26. Absolute Maximum Ratings  Note (1)  (Part 2 of 2)*

| Symbol | Parameter | Condition | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_I$ | DC input voltage for EPCS1, EPCS4, and EPCS16 | With respect to ground | –0.6 | 4.0 | V |
| | DC input voltage for EPCS64 and EPCS128 | With respect to ground | –0.5 | 4.0 | V |
| $I_{MAX}$ | DC $V_{CC}$ or GND current | — | — | 15 | mA |
| $I_{OUT}$ | DC output current per pin | — | –25 | 25 | mA |
| $P_D$ | Power dissipation | — | — | 54 | mW |
| $T_{STG}$ | Storage temperature | No bias | –65 | 150 | °C |
| $T_{AMB}$ | Ambient temperature | Under bias | –65 | 135 | °C |
| $T_J$ | Junction temperature | Under bias | — | 135 | °C |

*Table 14–27. Recommended Operating Conditions*

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | *(2)* | 2.7 | 3.6 | V |
| $V_I$ | Input voltage | Respect to GND | –0.3 | 0.3 + $V_{CC}$ | V |
| $V_O$ | Output voltage | — | 0 | $V_{CC}$ | V |
| $T_A$ | Operating temperature | For commercial use | 0 | 70 | °C |
| | | For industrial use | –40 | 85 | °C |
| $t_R$ | Input rise time | — | — | 5 | ns |
| $t_F$ | Input fall time | — | — | 5 | ns |

*Table 14–28. DC Operating Conditions*

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $V_{IH}$ | High-level input voltage for EPCS1, EPCS4, and EPCS16 | — | 0.6 × $V_{CC}$ | $V_{CC}$ + 0.4 | V |
| | High-level input voltage for EPCS64 and EPCS128 | — | 0.6 × $V_{CC}$ | $V_{CC}$ + 0.2 | V |
| $V_{IL}$ | Low-level input voltage | — | –0.5 | 0.3 × $V_{CC}$ | V |
| $V_{OH}$ | High-level output voltage | $I_{OH}$ = –100 µA *(3)* | $V_{CC}$ – 0.2 | — | V |
| $V_{OL}$ | Low-level output voltage | $I_{OL}$ = 1.6 mA *(3)* | — | 0.4 | V |
| $I_I$ | Input leakage current | $V_I$ = $V_{CC}$ or GND | –10 | 10 | µA |
| $I_{OZ}$ | Tri-state output off-state current | $V_O$ = $V_{CC}$ or GND | –10 | 10 | µA |

*Table 14–29. $I_{CC}$ Supply Current*

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $I_{CC0}$ | $V_{CC}$ supply current (standby) for EPCS1, EPCS4, and EPCS16 | — | — | 50 | µA |
| | $V_{CC}$ supply current (standby) for EPCS64 and EPCS128 | — | — | 100 | µA |
| $I_{CC1}$ | $V_{CC}$ supply current (during active power mode) for EPCS1, EPCS4, and EPCS16 | — | 5 | 15 | mA |
| | $V_{CC}$ supply current (during active power mode) for EPCS64 and EPCS128 | — | 5 | 20 | mA |

*Table 14–30. Capacitance* *Note (4)*

| Symbol | Parameter | Conditions | Min | Max | Unit |
|---|---|---|---|---|---|
| $C_{IN}$ | Input pin capacitance | $V_{IN}$ = 0 V | — | 6 | pF |
| $C_{OUT}$ | Output pin capacitance | $V_{OUT}$ = 0 V | — | 8 | pF |

***Notes to Table 14–26 through 14–30:***

(1) Refer to the *Operating Requirements for Altera Devices Data Sheet*.

(2) Maximum $V_{CC}$ rise time is 100 ms.

(3) The $I_{OH}$ parameter refers to high-level TTL or CMOS output current; the $I_{OL}$ parameter refers to low-level TTL or CMOS output current.

(4) Capacitance is sample-tested only at $T_A$ = 25 °C and at a 20-MHz frequency.

## Pin Information

As shown in Figures 14–20 and 14–21, the serial configuration device is an 8-pin or 16-pin device. The control pins on the serial configuration device are: serial data output (`DATA`), active serial data input (`ASDI`), serial clock (`DCLK`), and chip select (`nCS`). Table 14–31 shows the serial configuration device's pin descriptions.

Figure 14–20 shows the Altera serial configuration device 8-pin SOIC package and its pin-out diagram.

*Figure 14–20. Altera Serial Configuration Device 8-Pin SOIC Package Pin-Out Diagram*

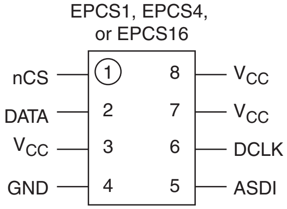
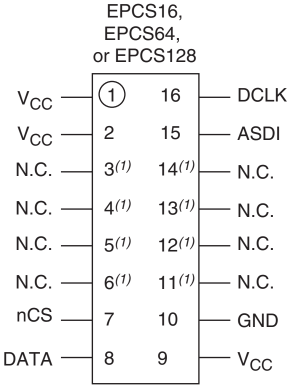


Figure 14–21 shows the Altera serial configuration device 16-pin SOIC package and its pin-out diagram.

*Figure 14–21. Altera Serial Configuration Device 16-Pin SOIC Package Pin-Out Diagram*



*Note to Figure 14–21:*

(1) These pins can be left floating or connected to $V_{CC}$ or GND, whichever is more convenient on the board.

*Table 14–31. Serial Configuration Device Pin Description*

| Pin Name | Pin Number in 8-Pin SOIC Package | Pin Number in 16-Pin SOIC Package | Pin Type | Description |
|---|---|---|---|---|
| DATA | 2 | 8 | Output | The DATA output signal transfers data serially out of the serial configuration device to the FPGA during read/configuration operation. During a read/configuration operations, the serial configuration device is enabled by pulling nCS low. The DATA signal transitions on the falling edge of DCLK. |
| ASDI | 5 | 15 | Input | The AS data input signal is used to transfer data serially into the serial configuration device. It receives the data that should be programmed into the serial configuration device. Data is latched on the rising edge of DCLK. |
| nCS | 1 | 7 | Input | The active low chip select input signal toggles at the beginning and end of a valid instruction. When this signal is high, the device is deselected and the DATA pin is tri-stated. When this signal is low, it enables the device and puts the device in an active mode. After power up, the serial configuration device requires a falling edge on the nCS signal before beginning any operation. |
| DCLK | 6 | 16 | Input | DCLK is provided by the FPGA. This signal provides the timing of the serial interface. The data presented on ASDI is latched to the serial configuration device on the falling edge of DCLK. Data on the DATA pin changes after the falling edge of DCLK and is latched into the FPGA on the falling edge. |
| $V_{CC}$ | 3, 7, 8 | 1, 2, 9 | Power | Power pins connect to 3.3 V. |
| GND | 4 | 10 | Ground | Ground pin. |

## Package

All serial configuration devices are available in 8-pin or 16-pin plastic SOIC package.

For more information on Altera device packaging including mechanical drawing and specifications for this package, refer to the *Altera Device Package Information Data Sheet*.

## Ordering Code

Table 14–32 shows the ordering codes for serial configuration devices.

*Table 14–32. Serial Configuration Device Ordering Codes*

| Device | Ordering Code *(1)* |
|---|---|
| EPCS1 | EPCS1SI8<br>EPCS1SI8N |
| EPCS4 | EPCS4SI8<br>EPCS4SI8N |
| EPCS16 | EPCS16SI16N<br>EPCS16SI8N |
| EPCS64 | EPCS64SI16N |
| EPCS128 | EPCS128SI16N |

***Note to Table 14–32:***

(1) N: Lead free.

## Referenced Documents

This chapter references the following documents:

- *Active Serial Memory Interface Data Sheet*
- *Altera Device Package Information Data Sheet*
- *Altera Programming Hardware Data Sheet*
- *AN 370: Using the Serial FlashLoader with the Quartus II Software*
- *AN 418: SRunner: An Embedded Solution for Serial Configuration Device Programming User Guide*
- *ByteBlaster II Download Cable User Guide*
- *Configuring Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*
- *EthernetBlaster Communications Cable User Guide*
- *Operating Requirements for Altera Devices Data Sheet*
- *Programming Hardware Manufacturers*
- *USB-Blaster Download Cable User Guide*

## Document Revision History

Table 14–33 shows the revision history for this chapter.

*Table 14–33. Document Revision History (Part 1 of 2)*

| Date and Document Version | Changes Made | Summary of Changes |
|---|---|---|
| May 2008<br>v3.1 | ● Updated Tables 14–2, 14–5, 14–6, 14–27, and 14–28.<br>● Deleted Note 5 to Table 14–30.<br>● Added "Referenced Documents" section. | — |
| August 2007<br>v3.0 | ● Updated "Introduction" section.<br>● Updated "Functional Description" section.<br>● Updated Tables 14–1 through 14–3 and Tables 14–6 through 14–8 to with EPCS128 information.<br>● Added Table 14–5 on Arria GX.<br>● Added *Note (4)* to Figure 14–3.<br>● Added *Note (5)* to Figure 14–4.<br>● Updated Table 14–9 with EPCS128 information.<br>● Added new Table 14–10 on address range for sectors in EPCS128.<br>● Updated Table 14–15 with information on "Read Device Identification" and added *Note (5)*.<br>● Added new Table 14–20 on block protection bits in EPCS128.<br>● Added *Note (1)* to Figure 14–12.<br>● Added new section "Read Device Identification Operation" with Table 14–22 and Figure 14–13.<br>● Updated "Write Bytes Operation", "Erase Bulk Operation" and "Erase Sector Operation" sections.<br>● Updated Table 14–23 to include EPCS128 information.<br>● Updated *Note (1)* to Table 14–25.<br>● Updated $V_{CC}$ and $V_I$ information to include EPCS128 in Table 14–26.<br>● Updated $V_{IH}$ information to include EPCS128 in Table 14–28.<br>● Updated $I_{CC0}$ and $I_{CC1}$ information to include EPCS128 in Table 14–29.<br>● Updated Figure 14–21 and Table 14–32 with EPCS128 information. | ● Updated document to include EPCS128.<br>● Updated document to include Arria GX. |
| April 2007<br>v2.0 | ● Updated "Introduction" section.<br>● Updated "Functional Description" section and added handpara note.<br>● Added Tables 14–3, 14–5, and 14–6.<br>● Updated "Active Serial FPGA Configuration" section and its handpara note.<br>● Added *Note (4)* to Figure 14–2.<br>● Updated Table 14–25 and added *Note (1)*.<br>● Updated Figure 14–20.<br>● Updated Table 14–32. | ● Updated chapter to include Stratix II GX, Stratix III, and Cyclone III support for EPCS devices.<br>● Added information about EPCS16SI8N. |
| January 2007<br>v1.7 | ● Removed reference to PLMSEPC-16 in "Programming and Configuration File Support".<br>● Updated `DCLK` pin information in Table 14–31. | — |

*Table 14–33. Document Revision History (Part 2 of 2)*

| Date and Document Version | Changes Made | Summary of Changes |
|---|---|---|
| October 2006<br>v1.6 | ● Updated Figure 14–19.<br>● Updated Table 14–29 and Table 14–31. | — |
| August 2005<br>v1.5 | Updated table 4-4 to include EPCS64 support for Cyclone devices. | — |
| August 2005<br>v1.4 | ● Updated tables.<br>● Minor text updates. | — |
| February 2005<br>v1.3 | Updated hot socketing AC specifications. | — |
| October 2003<br>v1.2 | ● Added Serial Configuration Device Memory Access section.<br>● Updated timing information in Tables 4–10 and 4–11.section.<br>● Updated timing information in Tables 4-16 and 4-17. | — |
| July 2003<br>v1.1 | Minor updates. | — |
| May 2003<br>v1.0 | Added document to the *Cyclone Device Handbook.* | — |