Serial Configuration (EPCS) Devices Datasheet

This datasheet describes serial configuration (EPCS) devices.

Supported Devices

Table 1 lists the supported Altera® EPCS devices.

Table 1. Altera EPCS Devices

Device	Memory Size (bits)	On-Chip Decompression Support	ISP Support	Cascading Support	Reprogrammable	Recommended Operating Voltage (V)
EPCS1	1,048,576	No	Yes	No	Yes	3.3
EPCS4	4,194,304	No	Yes	No	Yes	3.3
EPCS16	16,777,216	No	Yes	No	Yes	3.3
EPCS64	67,108,864	No	Yes	No	Yes	3.3
EPCS128	134,217,728	No	Yes	No	Yes	3.3

For more information about programming EPCS devices using the Altera Programming Unit (APU) or Master Programming Unit (MPU), refer to the *Altera Programming Hardware Datasheet*.

The EPCS device can be re-programmed in system with ByteBlaster™ II download cable or an external microprocessor using SRunner. For more information, refer to *AN418: SRunner: An Embedded Solution for Serial Configuration Device Programming*.

Features

EPCS devices offer the following features:

- Supports active serial (AS) x1 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
- 2.7-V to 3.6-V operation
- EPCS1, EPCS4, and EPCS16 devices available in 8-pin small-outline integrated circuit (SOIC) package
- EPCS64 and EPCS128 devices available in 16-pin SOIC package

101 Innovation Drive
San Jose, CA 95134
www.altera.com

- Enables the Nios® processor to access unused flash memory through AS memory interface
- Reprogrammable memory with more than 100,000 erase or program cycles
- Write protection support for memory sectors using status register bits
- In-system programming (ISP) support with SRunner software driver
- ISP support with USB-Blaster™, EthernetBlaster, or ByteBlaster II download cables
- Additional programming support with the APU and programming hardware from BP Microsystems, System General, and other vendors
- By default, the memory array is erased and the bits are set to `1`

Functional Description

To configure a system using an SRAM-based device, each time you power on the device, you must load the configuration data. The EPCS device is a flash memory device that can store configuration data that you use for FPGA configuration purpose after power on. You can use the EPCS device on all FPGA that support AS x1 configuration scheme.

For an 8-pin SOIC package, you can migrate vertically from the EPCS1 device to the EPCS4 or EPCS16 device. For a 16-pin SOIC package, you can migrate vertically from the EPCS64 device to the EPCS128 device.

With the new data decompression feature supported, you can determine using which EPCS device to store the configuration data for configuring your FPGA.

Example 1 shows how you can calculate the compression ratio to determine which EPCS device is suitable for the FPGA.

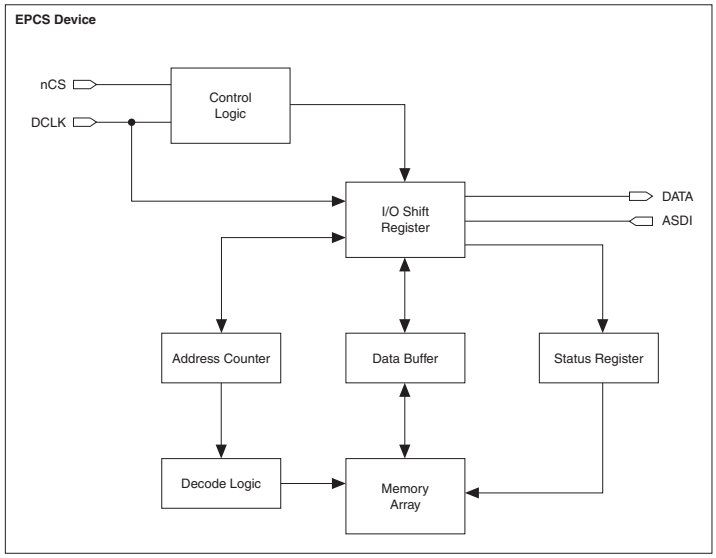
Example 1. Compression Ratio Calculation

```
EP4SGX530 = 189,000,000 bits

EPCS128 = 134,217,728 bits

Preliminary data indicates that compression typically reduces the
configuration bitstream size by 35% to 55%. Assume worst case that is 35%
decompression.

189,000,000 bits x 0.65 = 122,850,000 bits

The EPCS128 device is suitable.
```

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

Figure 1 shows the EPCS device block diagram.

Figure 1. EPCS Device Block Diagram



Accessing Memory in EPCS Devices

You can access the unused memory locations of the EPCS 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 designed specifically to work with the EPCS device. With this core, you can create a system with a Nios embedded processor that allows software access to any memory location within the EPCS device.

Active Serial FPGA Configuration

The following Altera FPGAs support the AS configuration scheme with EPCS devices:

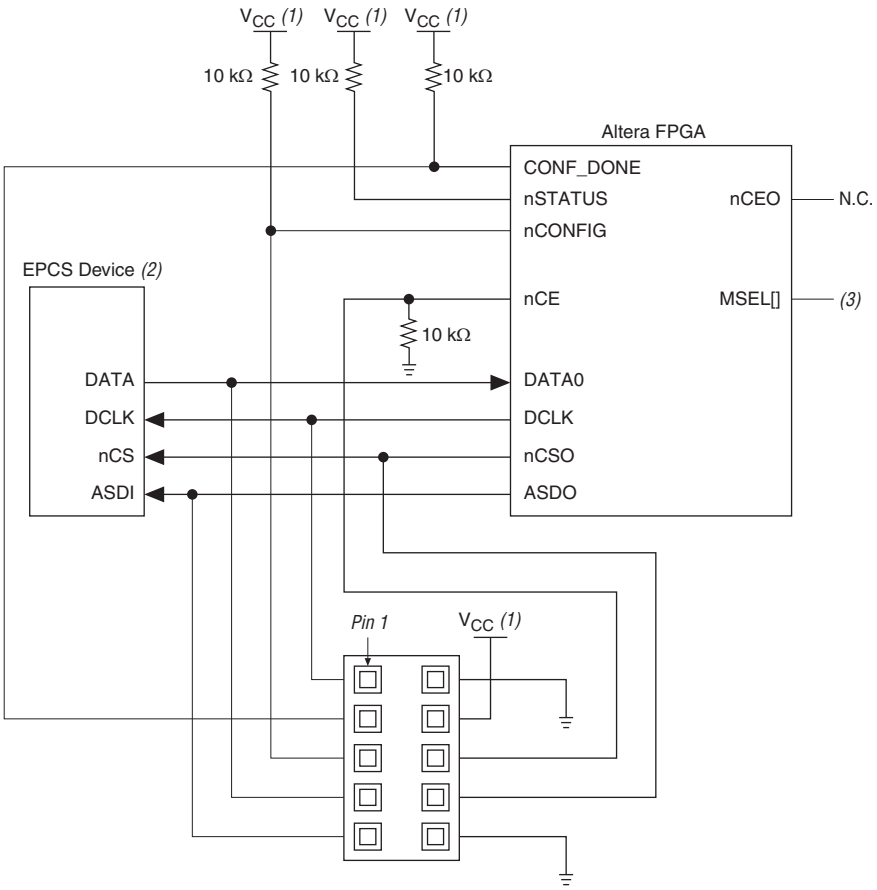
- Arria® series
- Cyclone® series
- All device families in the Stratix® series except the Stratix device family

There are four signals on the EPCS device that interface directly with the FPGA's control signals. The EPCS device signals are DATA, DCLK, ASDI, and nCS interface with the DATA0, DCLK, ASDO, and nCSO control signals on the FPGA, respectively.

For more information about the EPCS device pin description, refer to Table 23 on page 36.

Figure 2 shows the configuration of an FPGA device in the AS configuration scheme with an EPCS device using a download cable.

Figure 2. Altera FPGA Configuration in AS Mode Using a Download Cable *(1), (4)*



Notes to Figure 2:

(1) For more information about the V_{CC} value, refer to the configuration chapter in the appropriate device handbook.

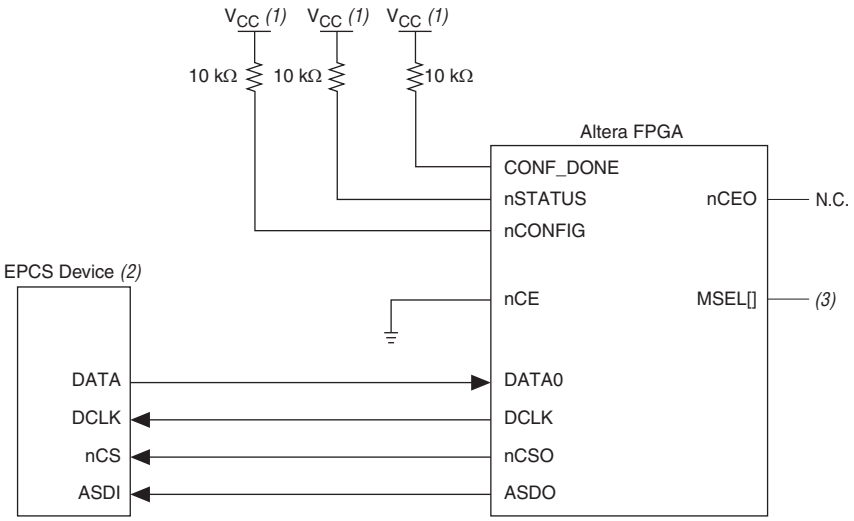
(2) EPCS devices cannot be cascaded.

(3) Connect the `MSEL[]` input pins to select the AS configuration mode. For more information, refer to the configuration chapter in the appropriate device handbook.

(4) For more information about configuration pin I/O requirements in an AS configuration scheme for an Altera FPGA, refer to the configuration chapter in the appropriate device handbook.

Figure 3 shows the configuration of an FPGA device in the AS configuration scheme with an EPCS device using the APU or a third-party programmer.

Figure 3. Altera FPGA Configuration in AS Mode Using APU or a Third-party Programmer *(1), (4)*



Notes to Figure 3:

(1) For more information about the V_{CC} value, refer to the configuration chapter in the appropriate device handbook.

(2) EPCS devices cannot be cascaded.

(3) Connect the MSEL[] input pins to select the AS configuration mode. For more information, refer to the configuration chapter in the appropriate device handbook.

(4) For more information about configuration pin I/O requirements in an AS configuration scheme for an Altera FPGA, refer to the configuration chapter in the appropriate device handbook.

In an AS configuration, the FPGA acts as the configuration master in the configuration flow and provides the clock to the EPCS device. The FPGA enables the EPCS device by pulling the nCS signal low using the nCSO signal as shown in Figure 2 and Figure 3. Then, the FPGA sends the instructions and addresses to the EPCS device using the ASDO signal. The EPCS 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 next DCLK signal's falling edge.

Before the FPGA enters configuration mode, ensure that V_{CC} of the EPCS device is ready. If V_{CC} is not ready, you must hold nCONFIG low until all power rails of EPCS device are ready.

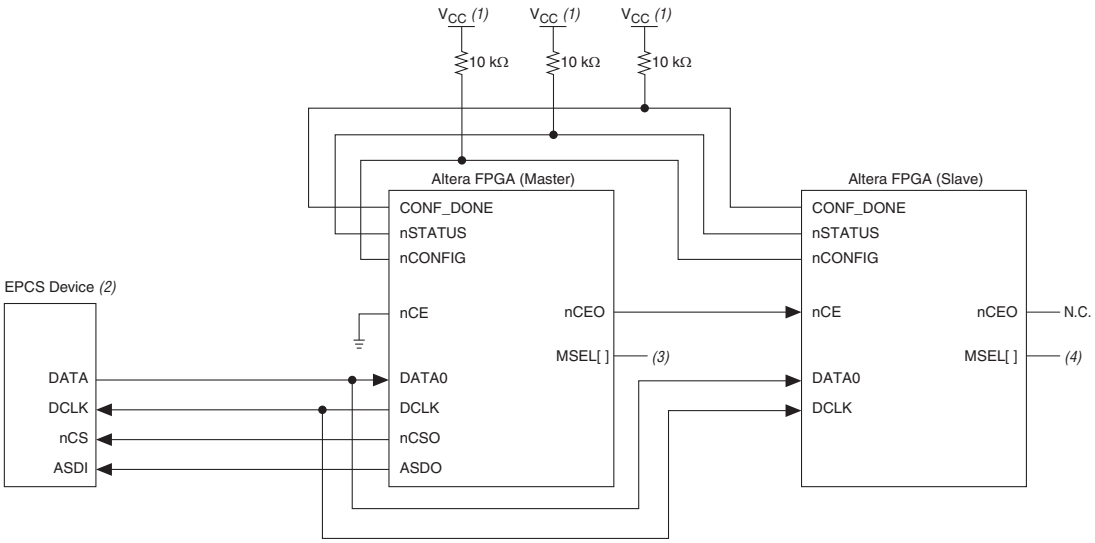
The FPGA controls the nSTATUS and CONF_DONE pins during configuration in the AS mode. If the CONF_DONE signal does not go high at the end of configuration, or if the signal goes high too early, the FPGA pulses its nSTATUS pin low to start a reconfiguration. If the configuration is successful, the FPGA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

For more information about configuring the FPGAs in AS configuration mode or other configuration modes, refer to the configuration chapter in the appropriate device handbook.

You can configure multiple devices with a single EPCS device. However, you cannot cascade EPCS devices. To ensure that the programming file size of the cascaded FPGAs does not exceed the capacity of an EPCS device, refer to Table 1 on page 1.

Figure 4 shows the AS configuration scheme with multiple FPGAs in the chain. The first FPGA is the configuration master and its MSEL[] pins are set to AS mode. The following FPGAs are configuration slave devices and their MSEL[] pins are set to PS mode.

Figure 4. Multiple Devices in AS Mode *(1), (5)*



Notes to Figure 4:

(1) For more information about the V_{CC} value, refer to the configuration chapter in the appropriate device handbook.

(2) EPCS devices cannot be cascaded.

(3) Connect the MSEL[] input pins to select the AS configuration mode. For more information, refer to the configuration chapter in the appropriate device handbook.

(4) Connect the MSEL[] input pins to select the PS configuration mode. For more information, refer to the configuration chapter in the appropriate device handbook.

(5) For more information about configuration pin I/O requirements in an AS configuration scheme for an Altera FPGA, refer to the configuration chapter in the appropriate device handbook.

EPCS Device Memory Access

This section describes the memory array organization and operation codes of the EPCS device. For the timing specifications, refer to "Timing Information" on page 29.

Memory Array Organization

Table 2 lists the memory array organization details in EPCS128, EPCS64, EPCS16, EPCS4, and EPCS1 devices.

Table 2. Memory Array Organization in EPCS Devices

Details	EPCS128	EPCS64	EPCS16	EPCS4	EPCS1
Bytes	16,777,216 bytes (128 Mb)	8,388,608 bytes (64 Mb)	2,097,152 bytes (16 Mb)	524,288 bytes (4 Mb)	131,072 bytes (1 Mb)
Number of sectors	64	128	32	8	4
Bytes per sector	262,144 bytes (2 Mb)	65,536 bytes (512 Kb)	65,536 bytes (512 Kb)	65,536 bytes (512 Kb)	32,768 bytes (256 Kb)
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

Table 3 through Table 7 on page 12 list the address range for each sector in EPCS1, EPCS4, EPCS16, EPCS64, and EPCS128 devices.

Table 3. Address Range for Sectors in EPCS1 Devices

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

Table 4. Address Range for Sectors in EPCS4 Devices

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 5. Address Range for Sectors in EPCS16 Devices

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 6. Address Range for Sectors in EPCS64 Devices (Part 1 of 4)

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
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
90	H'5A0000	H'5AFFFF

Table 6. Address Range for Sectors in EPCS64 Devices (Part 2 of 4)

Sector	Address Range (Byte Addresses in HEX)	
	Start	End
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
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

Table 6. Address Range for Sectors in EPCS64 Devices (Part 3 of 4)

Sector	Address Range (Byte Addresses in HEX)	
	Start	End
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
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

Table 6. Address Range for Sectors in EPCS64 Devices (Part 4 of 4)

Sector	Address Range (Byte Addresses in HEX)	
	Start	End
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 7. Address Range for Sectors in EPCS128 Devices (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
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

Table 7. Address Range for Sectors in EPCS128 Devices (Part 2 of 3)

Sector	Address Range (Byte Addresses in HEX)	
	Start	End
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
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

Table 7. Address Range for Sectors in EPCS128 Devices (Part 3 of 3)

Sector	Address Range (Byte Addresses in HEX)	
	Start	End
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

Operation Codes

This section describes the operations that you can use to access the memory in EPCS devices. Use the DATA, DCLK, ASDI, and nCS signals to access the memory in EPCS devices. When performing the operation, addresses and data are shifted in and out of the device serially, with MSB first.

The device samples the AS 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, with MSB first, into the EPCS device serially through the AS data input (ASDI) pin. Each operation code bit is latched into the EPCS 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 or data bytes, both address and data bytes, or none of them. The device must drive nCS pin high after the last bit of the operation sequence is shifted in. Table 8 lists the operation sequence for every operation supported by the EPCS devices.

For read operations, the data read is 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 write and erase operations, drive the nCS pin high at a byte boundary that is in a multiple of eight clock pulses. Otherwise, the operation is rejected and not executed.

All attempts to access the memory contents while a write or erase cycle is in progress are rejected, and the write or erase cycle will continue unaffected.

Table 8. EPCS Devices Operation Codes

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)	32
Read bytes	0000 0011	3	0	1 to infinite (2)	20
Read silicon ID (4)	1010 1011	0	3	1 to infinite (2)	32
Fast read	0000 1011	3	1	1 to infinite (2)	40
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

Table 8. EPCS Devices Operation Codes

Operation	Operation Code (1)	Address Bytes	Dummy Bytes	Data Bytes	DCLK f_{MAX} (MHz)
Erase sector	1101 1000	3	0	0	25
Read device identification (5)	1001 1111	0	2	1 to infinite (2)	25

Notes to Table 8:

(1) List MSB first and LSB last.

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

(3) A 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.

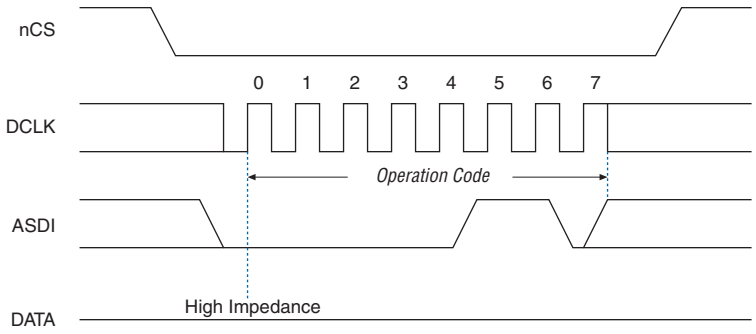
(4) The read silicon ID operation is available only for EPCS1, EPCS4, EPCS16, and EPCS64 devices.

(5) The read device identification operation is available only for EPCS128 devices.

Write Enable Operation

The write enable operation code is b'0000 0110, and it lists the MSB 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 5 shows the instruction sequence of the write enable operation.

Figure 5. Write Enable Operation Timing Diagram



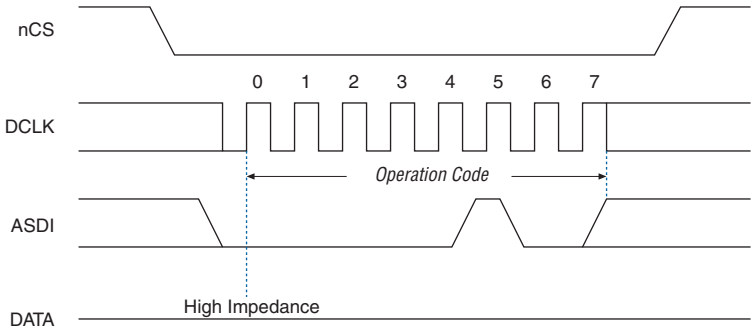
Write Disable Operation

The write disable operation code is b'0000 0100 and it lists the MSB 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, and under the following conditions:

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

Figure 6 shows the instruction sequence of the write disable operation.

Figure 6. Write Disable Operation Timing Diagram



Read Status Operation

The read status operation code is `b'0000 0101` and it lists the MSB first. You can use the read status operation to read the status register. Figure 7 and Figure 8 show the status bits in the status register of the EPCS devices.

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

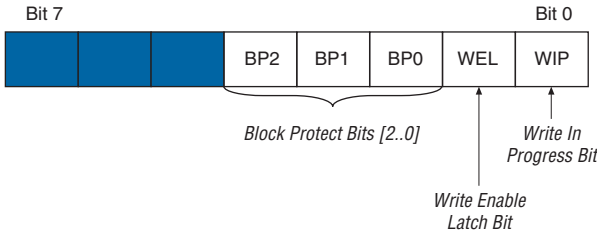
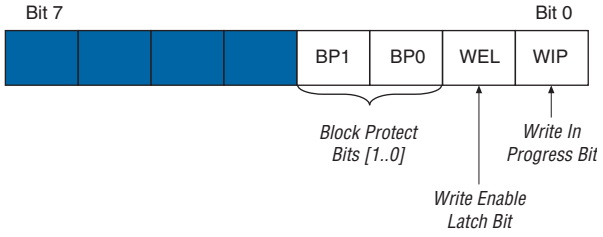


Figure 8. EPCS1 Status Register Status Bits



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

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

The non-volatile block protect bits determine the area of the memory protected from being written or erased unintentionally. Table 9 through Table 13 on page 19 list the protected area in the EPCS devices with reference to the block protect bits. The erase bulk operation is only available when all the block protect bits are set to 0. When any of the block protect bits are set to 1, the relevant area is protected from being written by a write bytes operation or erased by an erase sector operation.

Table 9. Block Protection Bits in the EPCS1 Device

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 10. Block Protection Bits in the EPCS4 Device

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 11. Block Protection Bits in the EPCS16 Device

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 12. Block Protection Bits in the EPCS64 Devices

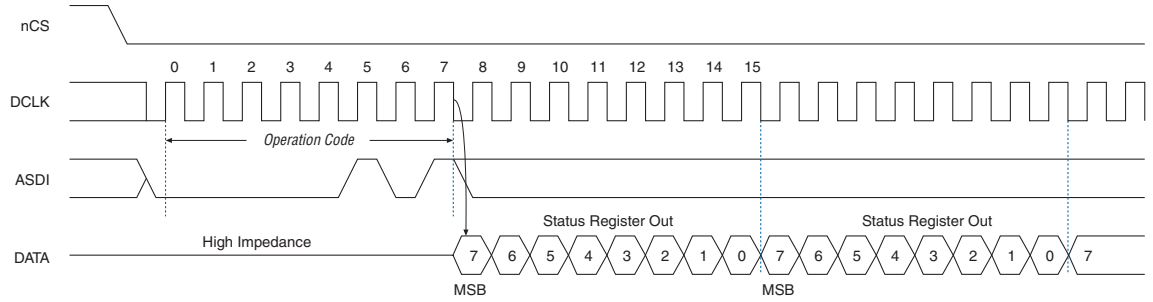
Status Register Content			Memory Content	
BP2 Bit	BP1 Bit	BP0 Bit	Protected Area	Unprotected Area
0	0	0	None	All sectors (128 sectors: 0 to 127)
0	0	1	Upper 64th (2 sectors: 126 and 127)	Lower 63/64ths (126 sectors: 0 to 125)
0	1	0	Upper 32nd (4 sectors: 124 to 127)	Lower 31/32nds (124 sectors: 0 to 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-eights (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 13. Block Protection Bits in the EPCS128 Device

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 during 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 9.

Figure 9. Read Status Operation Timing Diagram



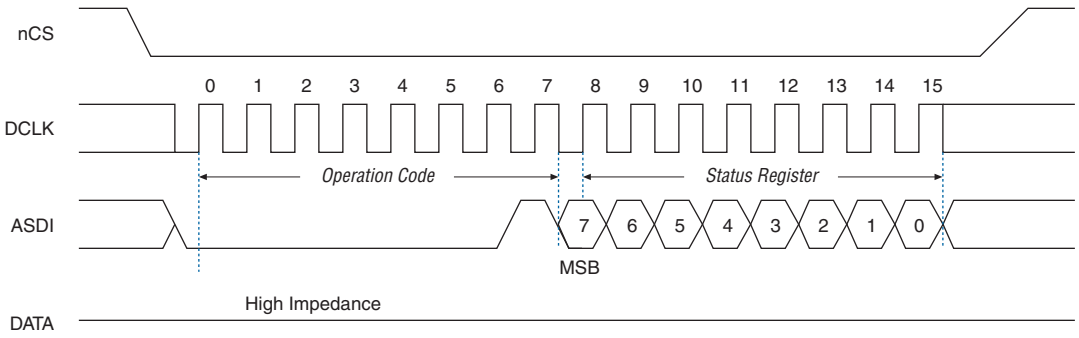
Write Status Operation

The write status operation code is b'0000 0001 and it lists the MSB first. Use the write status operation to set the status register block protection bits. The write status operation does not affect the other bits. Therefore, you can implement this operation to protect certain memory sectors, as listed in Table 9 through Table 13. 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 the nCS signal low, followed by shifting in the write status operation code and one data byte for the status register on the ASDI pin. Figure 10 shows the instruction sequence of the write status operation. The 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 the nCS signal drives high, the device initiates the self-timed write status cycle. The self-timed write status cycle usually takes 5 ms for all EPCS devices and is guaranteed to be less than 15 ms. For more information, refer to the t_{WS} value in Table 16 on page 29. 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 10. Write Status Operation Timing Diagram



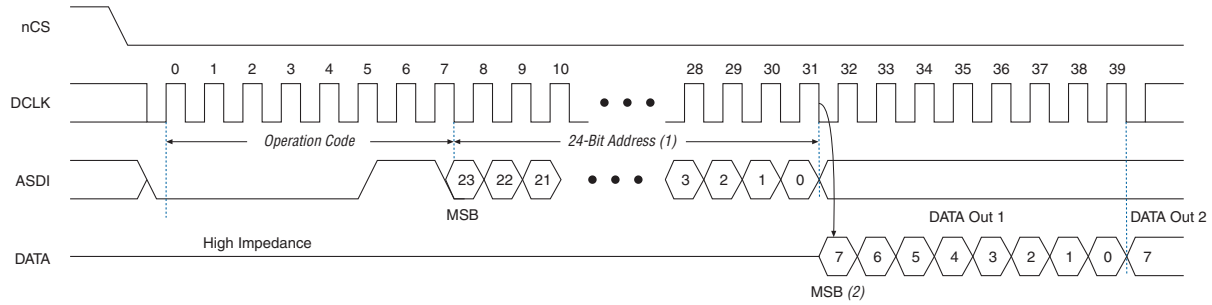
Read Bytes Operation

The read bytes operation code is b'0000 0011 and it lists the MSB first. To read the memory contents of the EPCS device, the device is first selected by driving the nCS signal 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 signal. 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 the DCLK signal. The maximum DCLK frequency during the read bytes operation is 20 MHz.

The first byte address can be at any location. The device automatically increases 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 the nCS signal high. The device can drive the nCS signal high at 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 does not affect the write or erase cycle in progress.

Figure 11 shows the instruction sequence of the read bytes operation.

Figure 11. Read Bytes Operation Timing Diagram



Notes to Figure 11:

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

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

Fast Read Operation

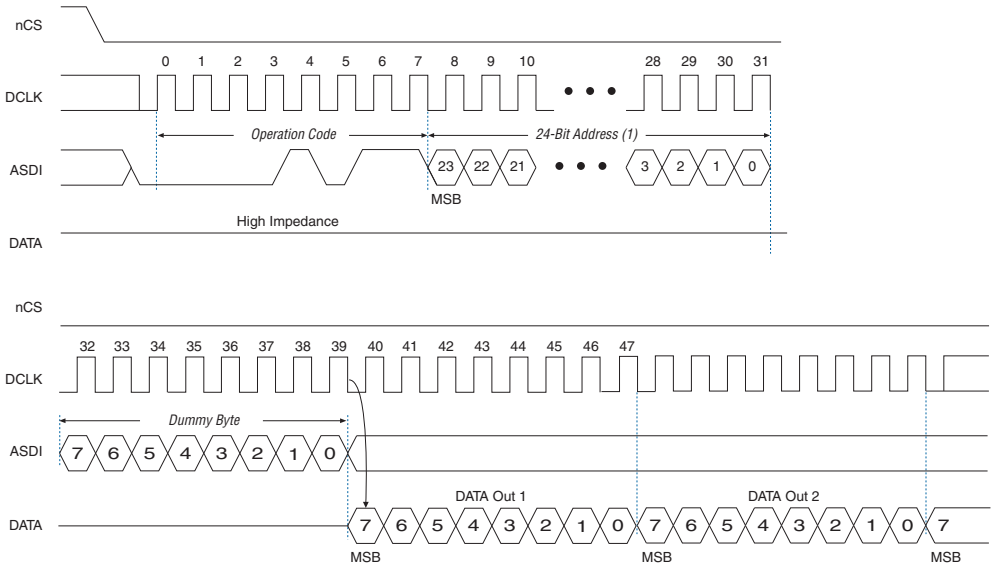
The fast read operation code is b'0000 1011 and it lists the MSB first. You can select the device by driving the nCS signal low. The fast read instruction code is followed by a 3-byte address (A23-A0) and a dummy byte with each bit being latched-in during the rising edge of the DCLK signal. Then, the memory contents at that address is shifted out on DATA with each bit being shifted out at a maximum frequency of 40 MHz during the falling edge of the DCLK signal.

The first addressed byte can be at any location. The address is automatically increased to the next higher address after each byte of data is shifted out. Therefore, the whole memory can be read with a single fast read instruction. When the highest address is reached, the address counter rolls over to 000000h, allowing the read sequence to continue indefinitely.

The fast read instruction is terminated by driving the nCS signal high at any time during data output. Any fast read instruction is rejected during the erase, program, or write operations without affecting the operation that is in progress.

Figure 12 shows the instruction sequence of the fast read operation.

Figure 12. Fast Read Operation Timing Diagram



Note to Figure 12:

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

Read Silicon ID Operation

The read silicon ID operation code is b'1010 1011 and it lists the MSB first. Only EPCS1, EPCS4, EPCS16, and EPCS64 devices support this operation. This operation reads the 8-bit silicon ID of the EPCS device from the DATA output pin. If this operation is shifted in during an erase or write cycle, it is ignored and does not affect the cycle that is in progress.

Table 14 lists the EPCS device silicon IDs.

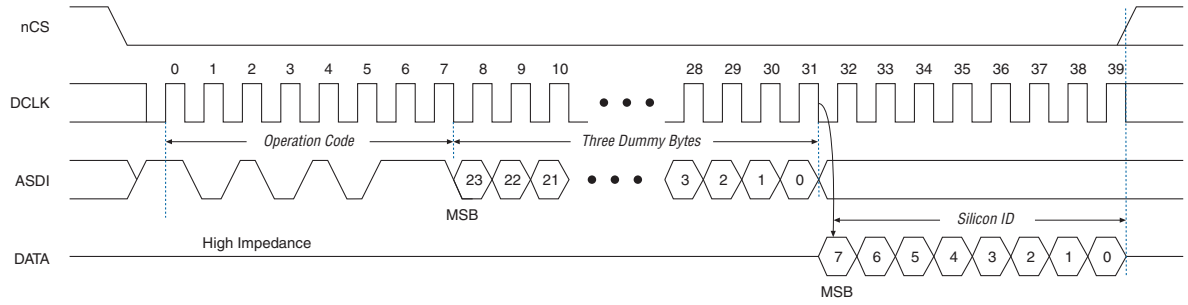
Table 14. EPCS Device Silicon ID

EPCS 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 the nCS signal low and then shifting in the read silicon ID operation code, followed by three dummy bytes on the ASDI pin. The 8-bit silicon ID of the EPCS device is then shifted out on the DATA pin on the falling edge of the DCLK signal. The device can terminate the read silicon ID operation by driving the nCS signal high after reading the silicon ID at least one time. Sending additional clock cycles on DCLK while nCS is driven low can cause the silicon ID to be shifted out repeatedly.

Figure 13 shows the instruction sequence of the read silicon ID operation.

Figure 13. Read Silicon ID Operation Timing Diagram (1)



Note to Figure 13:

(1) Only EPCS1, EPCS4, EPCS16, and EPCS64 devices support the read silicon ID operation.

Read Device Identification Operation

The read device identification operation code is `b'1001 1111` and it lists the MSB first. Only EPCS128 device supports this operation. This operation reads the 8-bit device identification of the EPCS device from the `DATA` output pin. If this operation is shifted in during an erase or write cycle, it is ignored and does not affect the cycle that is in progress. Table 15 lists the EPCS device identification.

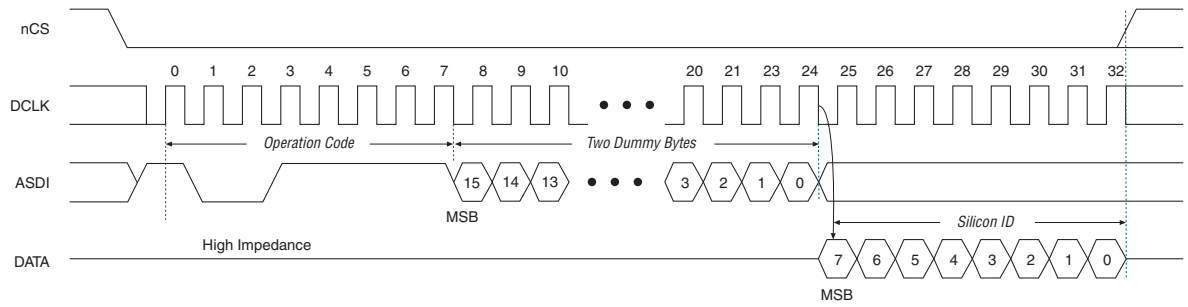
Table 15. EPCS Device Identification

EPCS Device	Silicon ID (Binary Value)
EPCS128	`b'0001 1000`

The device implements the read device identification operation by driving the `nCS` signal low and then shifting in the read device identification operation code, followed by two dummy bytes on the `ASDI` pin. The 16-bit device identification of the EPCS device is then shifted out on the `DATA` pin on the falling edge of the `DCLK` signal. The device can terminate the read device identification operation by driving the `nCS` signal high after reading the device identification at least one time.

Figure 14 shows the instruction sequence of the read device identification operation.

Figure 14. Read Device Identification Operation Timing Diagram (1)



Note to Figure 14:

(1) Only EPCS128 device supports the read device identification operation.

Write Bytes Operation

The write bytes operation code is b'0000 0010 and it lists the MSB first. This operation allows bytes to be written to the memory. You must execute the write enable operation before 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 the nCS signal low, followed by the write bytes operation code, three address bytes, and at least one data byte on the ASDI pin. If the eight LSBs (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). You must ensure the nCS signal is set low during the entire write bytes operation.

If more than 256 data bytes are shifted into the EPCS 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 EPCS device, they are guaranteed to be written at the specified addresses and the other bytes of the same page are not affected.

If your design requires writing more than 256 data bytes to the memory, more than one page of memory is required. 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.

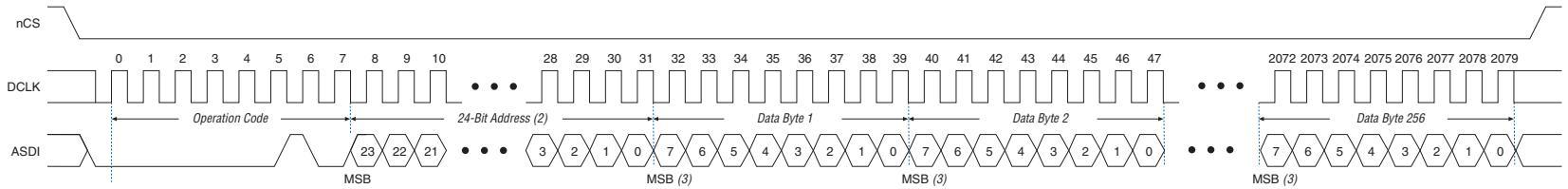
The nCS signal must be driven high after the eighth bit of the last data byte has been latched in. Otherwise, the device does not execute the write bytes operation. The write enable latch bit in the status register is reset to 0 before the completion of each write bytes operation. Therefore, the write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

The device initiates a self-timed write cycle immediately after the nCS signal is driven high. For more information about the self-timed write cycle time, refer to the t_{WB} value in Table 16 on page 29. You must account for this amount of delay before another page of memory is written. Alternatively, you can check the write in progress bit in the status register 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.

You must erase all the memory bytes of the EPCS devices to all 1 or 0xFF before you implement the write bytes operation. You can erase all the memory bytes by executing the erase sector operation in a sector or the erase bulk operation throughout the entire memory.

Figure 15 shows the instruction sequence of the write bytes operation.

Figure 15. Write Bytes Operation Timing Diagram *(1)*



Notes to Figure 15:

(1) Use the erase sector operation or the erase bulk operation to initialize the memory bytes of the EPCS devices to all `1` or `0xFF` before implementing the write bytes operation.

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

(3) For **.rpd** files, write the LSB of the data byte first.

Erase Bulk Operation

The erase bulk operation code is `b'1100 0111` and it lists the MSB first. This operation sets all the memory bits to `1` or `0xFF`. Similar to the write bytes operation, you must execute the write enable operation before the erase bulk operation so that the write enable latch bit in the status register is set to `1`.

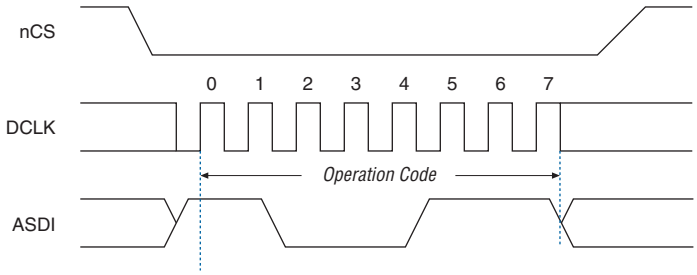
You can implement the erase bulk operation by driving the `nCS` signal low and then shifting in the erase bulk operation code on the `ASDI` pin. The `nCS` signal must be driven high after the eighth bit of the erase bulk operation code has been latched in.

The device initiates a self-timed erase bulk cycle immediately after the `nCS` signal is driven high. For more information about the self-timed erase bulk cycle time, refer to the t_{EB} value in Table 16 on page 29.

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 set to `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 16 shows the instruction sequence of the erase bulk operation.

Figure 16. Erase Bulk Operation Timing Diagram



Erase Sector Operation

The erase sector operation code is b'1101 1000 and it lists the MSB first. This operation allows you to erase a certain sector in the EPCS device by setting all the bits inside the sector to 1 or 0xFF. This operation is useful if you want to access the unused sectors as general purpose memory in your applications. You must execute the write enable operation before the erase sector operation so that the write enable latch bit in the status register is set to 1.

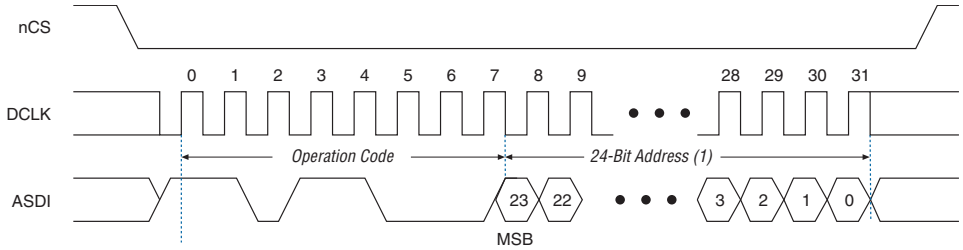
You can implement the erase sector operation by first driving the nCS signal low, then you shift in the erase sector operation code, followed by the three address bytes of the chosen sector on the ASDI pin. The three address bytes for the erase sector operation can be any address inside the specified sector. For more information about the sector address range, refer to Table 3 on page 7 through Table 7 on page 12. Drive the nCS signal high after the eighth bit of the erase sector operation code has been latched in.

The device initiates the self-timed erase sector cycle immediately after the nCS signal is driven high. For more information about the self-timed erase sector cycle time, refer to the t_{ES} value in Table 16 on page 29.

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 sector cycle is in progress. The write in progress bit is set to 1 during the self-timed erase sector 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 17 shows the instruction sequence of the erase sector operation.

Figure 17. Erase Sector Operation Timing Diagram



Note to Figure 17:

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

Power and Operation

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

Power Mode

EPCS devices support active and standby power modes. When the nCS signal is low, the device is enabled and is in active power mode. The FPGA is configured while the EPCS device is in active power mode. When the nCS signal is high, the device is disabled but remains in active power mode until all internal cycles are completed, such as write or erase operations. The EPCS device then goes into standby power mode. The I_{CC1} and I_{CC0} parameters list the V_{CC} supply current when the device is in active and standby power modes. For more information, refer to Table 21 on page 34.

Power-On Reset

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

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

Error Detection

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

After an error, the 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 18 shows the timing waveform for the write operation of the EPCS device.

Figure 18. Write Operation Timing Diagram

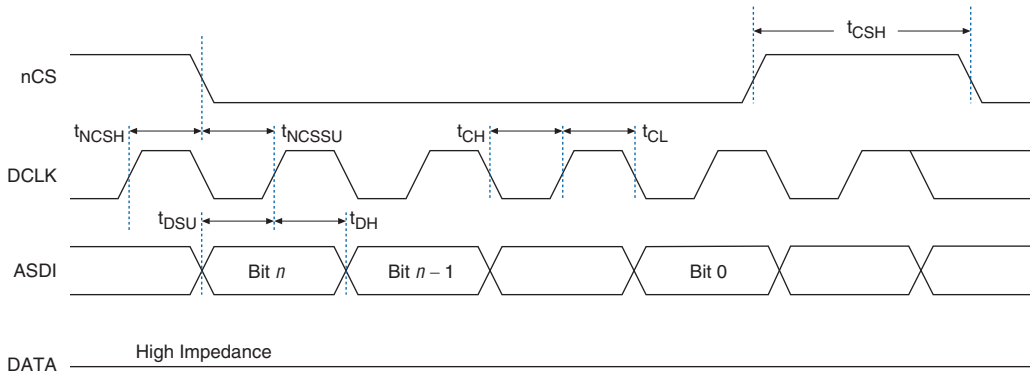


Table 16 lists the EPCS device timing parameters for the write operation.

Table 16. Write Operation Parameters

Symbol	Parameter	Min	Typ	Max	Unit
f_{WCLK}	Write clock frequency (from the 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 the rising edge on DCLK	5	—	—	ns
t_{DH}	Data (ASDI) hold time after rising edge on DCLK	5	—	—	ns
t_{CSH}	Chip select (nCS) high time	100	—	—	ns
t_{WB} (1)	Write bytes cycle time for EPCS1, EPCS4, EPCS16, and EPCS64 devices	—	1.5	5	ms
	Write bytes cycle time for the EPCS128 device	—	2.5	7	ms
t_{WS} (1)	Write status cycle time	—	5	15	ms
t_{EB} (1)	Erase bulk cycle time for the EPCS1 device	—	3	6	s
	Erase bulk cycle time for the EPCS4 device	—	5	10	s
	Erase bulk cycle time for the EPCS16 device	—	17	40	s
	Erase bulk cycle time for the EPCS64 device	—	68	160	s
	Erase bulk cycle time for the EPCS128 device	—	105	250	s
t_{ES} (1)	Erase sector cycle time for EPCS1, EPCS4, EPCS16, and EPCS64 devices	—	2	3	s
	Erase sector cycle time for the EPCS128 device	—	2	6	s

Note to Table 16:

(1) Figure 18 does not show these parameters.

Figure 19 shows the timing waveform for the read operation of the EPCS device.

Figure 19. Read Operation Timing Diagram

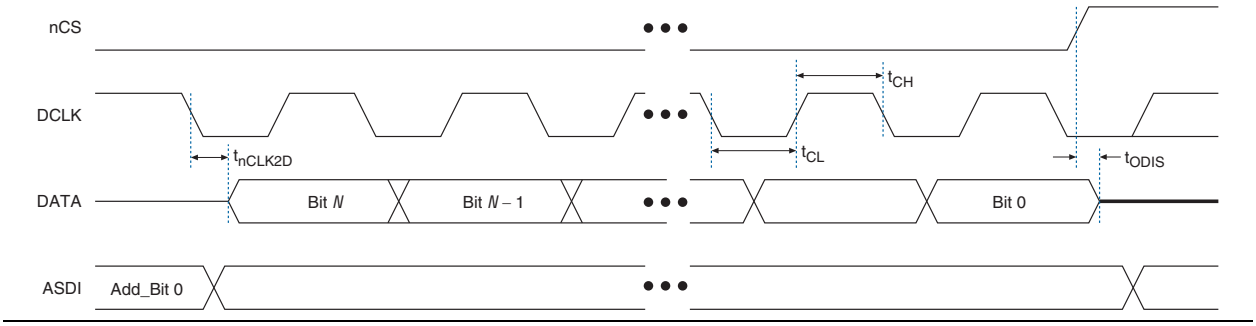


Table 17 lists the EPCS device timing parameters for the read operation.

Table 17. Read Operation Parameters

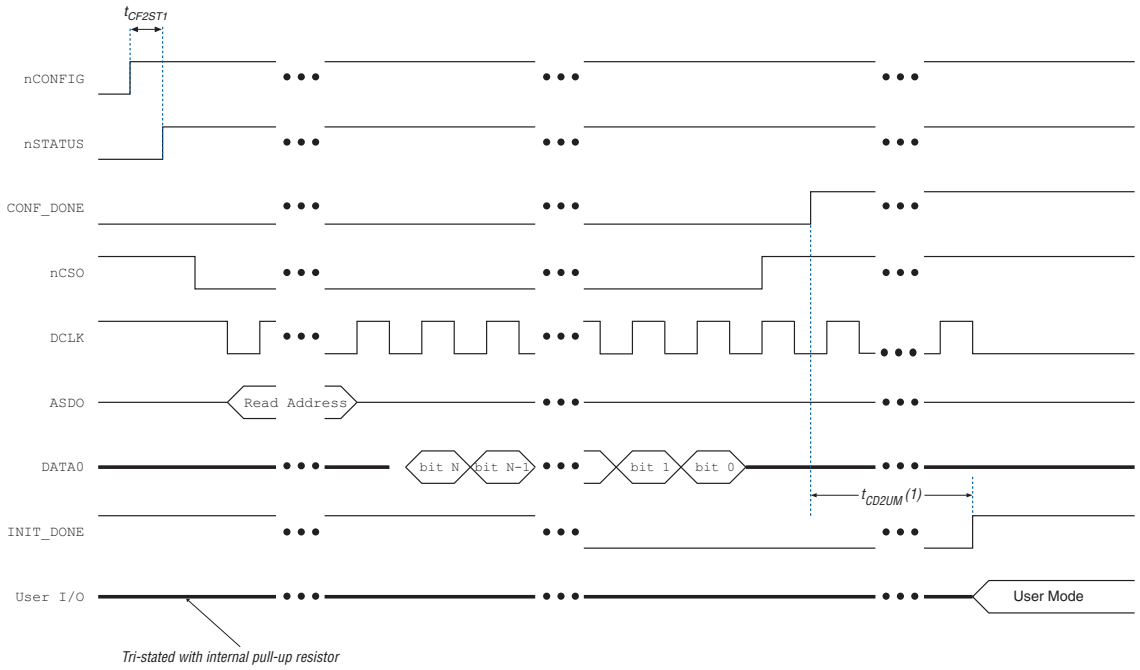
Symbol	Parameter	Min	Max	Unit
f_{RCLK}	Read clock frequency (from the FPGA or embedded processor) for the read bytes operation	—	20	MHz
	Fast read clock frequency (from the FPGA or embedded processor) for the fast read bytes operation	—	40	MHz
t_{CH}	DCLK high time	11	—	ns
t_{CL}	DCLK low time	11	—	ns
t_{ODIS}	Output disable time after read	—	8	ns
t_{nCLK2D}	Clock falling edge to DATA	—	8	ns

Existing batches of EPCS1 and EPCS4 devices manufactured on 0.15 µm process geometry support the AS configuration up to 40 MHz. However, batches of EPCS1 and EPCS4 devices manufactured on 0.18 µm process geometry support the AS configuration only up to 20 MHz. EPCS16, EPCS64, and EPCS128 devices are not affected.

For more information about product traceability and transition date to differentiate between 0.15 µm process geometry and 0.18 µm process geometry of the EPCS1 and EPCS4 devices, refer to the *PCN 0514: Manufacturing Changes on EPCS Family*.

Figure 20 shows the timing waveform for the AS configuration scheme of the FPGA using an EPCS device.

Figure 20. AS Configuration Timing Diagram



Note to Figure 20:

(1) t_{CD2UM} is an FPGA-dependent parameter. For more information, refer to the configuration chapter in the appropriate device handbook.

For more information about the timing parameters in Figure 20, refer to the configuration chapter in the appropriate device handbook.

Programming and Configuration File Support

The Quartus II software provides programming support for EPCS devices. When you select an EPCS device, the Quartus II software automatically generates the Programmer Object File (**.pof**) to program the device. The software allows you to select the appropriate EPCS device density that most efficiently stores the configuration data for the selected FPGA.

You can program the EPCS device in-system by an external microprocessor using the SRunner software driver. The SRunner software driver is developed for embedded EPCS device programming that you can customize to fit in different embedded systems. The SRunner software driver reads **.rpd** files and writes to the EPCS devices. The programming time is comparable to the Quartus II software programming time. Because the FPGA reads the LSB of the **.rpd** data first during the configuration process, the LSB of **.rpd** bytes must be shifted out first during the read bytes operation and shifted in first during the write bytes operation.

Writing and reading the **.rpd** file to and from the EPCS device is different from the other data and address bytes.

For more information about the SRunner software driver, refer to *AN 418: SRunner: An Embedded Solution for Serial Configuration Device Programming*.

You can program EPCS devices using the APU with the appropriate programming adapter, such as PLMSEPC-8, using the Quartus II software or the USB-Blaster, EthernetBlaster, or ByteBlaster II download cable. In addition, many third-party programmers, such as the BP Microsystems and System General programmers, offer programming hardware that supports EPCS devices.

During the ISP of an EPCS device using the USB-Blaster, EthernetBlaster, or ByteBlaster II download cable, the cable pulls the `nCONFIG` signal low to reset the FPGA and overrides the 10-kΩ pull-down resistor on the `nCE` pin of the FPGA, as shown in Figure 2 on page 4. The download cable then uses the four interface pins—`DATA`, `nCS`, `ASDI`, and `DCLK`—to program the EPCS device. When programming is complete, the download cable releases the four interface pins of the EPCS device and the `nCE` pin of the FPGA and pulses the `nCONFIG` signal to start the configuration process.

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

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

For more information about 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

Table 18 through Table 22 list information about the absolute maximum ratings, recommended operating conditions, DC operating conditions, and capacitance for EPCS devices.

Table 18. Absolute Maximum Ratings *(1)*

Symbol	Parameter	Condition	Min	Max	Unit
V_{CC}	Supply voltage for EPCS1, EPCS4, and EPCS16 devices	With respect to GND	–0.6	4.0	V
	Supply voltage for EPCS64 and EPCS128 devices	With respect to GND	–0.2	4.0	V
V_I	DC input voltage for EPCS1, EPCS4, and EPCS16 devices	With respect to GND	–0.6	4.0	V
	DC input voltage for EPCS64 and EPCS128 devices	With respect to GND	–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 19. Recommended Operating Conditions

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

Table 20. DC Operating Conditions

Symbol	Parameter	Conditions	Min	Max	Unit
V_{IH}	High-level input voltage for EPCS1, EPCS4, and EPCS16 devices	—	0.6 x V_{CC}	V_{CC} + 0.4	V
	High-level input voltage for EPCS64 and EPCS128 devices	—	0.6 x V_{CC}	V_{CC} + 0.2	V
V_{IL}	Low-level input voltage	—	–0.5	0.3 x 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 21. I_{CC} Supply Current

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

Table 22. Capacitance (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 18 through Table 22:

(1) For more information, 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 the high-level TTL or CMOS output current and the I_{OL} parameter refers to the 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

Figure 21 and Figure 22 show the EPCS device in an 8-pin or 16-pin device. The following lists the control pins on the EPCS device:

- Serial data output (DATA)
- AS data input (ASDI)
- Serial clock (DCLK)
- Chip select (nCS)

Figure 21 shows the 8-pin SOIC package of the EPCS device.

Figure 21. Altera EPCS Device 8-Pin SOIC Package Pin-Out Diagram

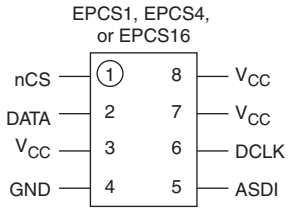
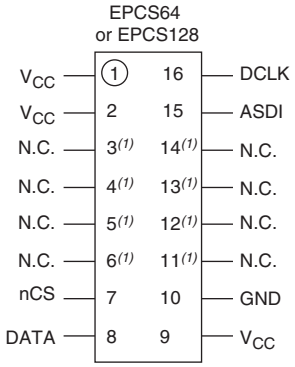


Figure 22 shows the 16-pin SOIC package of the EPCS device.

Figure 22. Altera EPCS Device 16-Pin SOIC Package Pin-Out Diagram



Note to Figure 22:

(1) You can leave these pins floating or you can connect them to V_{CC} or GND.

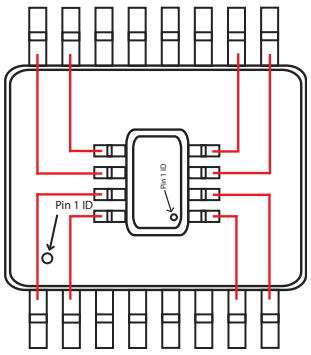
Table 23 lists the pin description of the EPCS device.

Table 23. EPCS 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 EPCS device to the FPGA during the read operation or configuration. During the read operation or configuration, the EPCS device is enabled by pulling the nCS signal low. The DATA signal transitions on the falling edge of the DCLK signal.
ASDI	5	15	Input	The ASDI signal is used to transfer data serially into the EPCS device. This pin are also receiving data that are programmed into the EPCS device. Data is latched on the rising edge of the DCLK signal.
nCS	1	7	Input	The nCS signal toggles at the beginning and the end of a valid instruction. When this signal goes high, the device is deselected and the DATA pin is tri-stated. When this signal goes low, the device is enabled and in an active mode. After power up, the EPCS device requires a falling edge on the nCS signal before the EPCS device begins any operation.
DCLK	6	16	Input	The FPGA provides the DCLK signal. This signal provides the timing for the serial interface. The data presented on the ASDI pin is latched to the EPCS device on the rising edge of the DCLK signal. The data on the DATA pin changes after the falling edge of the DCLK signal and is latched into the FPGA on the next falling edge of the DCLK signal.
V_{CC}	3, 7, 8	1, 2, 9	Power	Connect the power pins to a 3.3-V power supply.
GND	4	10	GND	Ground pin.

Figure 23 shows the layout recommendation for vertical migration from the EPCS1 device to the EPCS128 device.

Figure 23. Layout Recommendation for Vertical Migration from the EPCS1 Device to the EPCS128 Device



Device Package and Ordering Code

This section describes the package offered in EPCS devices and the ordering codes for each EPCS device.

Package

The EPCS1, EPCS4, and EPCS16 devices are available in 8-pin SOIC package. The EPCS64 and EPCS128 devices are available in 16-pin SOIC package.

If you use the AS x1 configuration scheme, you can migrate EPCS64 and EPCS128 devices to EPCQ64, EPCQ128, or EPCQ256 devices.

For more information, refer to the Package and Thermal Resistance page.

For more information about migration to EPCQ, refer to the *Quad-Serial Configuration (EPCQ) Devices Datasheet*.

Ordering Code

Table 24 lists the ordering codes for EPCS devices.

Table 24. EPCS Device Ordering Codes

Device	Ordering Code (1)
EPCS1	EPCS1SI8 EPCS1SI8N
EPCS4	EPCS4SI8 EPCS4SI8N
EPCS16	EPCS16SI8N
EPCS64	EPCS64SI16N
EPCS128	EPCS128SI16N

Note to Table 24:

(1) N indicates that the device is lead free.

Document Revision History

Table 25 lists the revision history for this document.

Table 25. Document Revision History (Part 1 of 4)

Date	Version	Changes
April 2014	5.1	Removed the operating temperature for commercial use in Table 19.
January 2014	5.0	■ Added Table 12 to include the block protection bits for EPCS64 devices. ■ Updated DCLK f_{MAX} for the read status and read silicon ID operations to 32 MHz in Table 8. ■ Updated t_{CH}, t_{CL}, and t_{ODIS} values in the read operation parameters in Table 17. ■ Updated the “Package” section to include device migration information.

Table 25. Document Revision History (Part 2 of 4)

Date	Version	Changes
January 2012	4.0	■ Updated “Package” and “Ordering Code” sections. ■ Updated Figure 5, Figure 6, and Figure 22. ■ Updated Table 16 and Table 18. ■ Minor text edits.
June 2011	3.4	■ Updated Table 3–19. ■ Updated Figure 3–20.
December 2009	3.3	■ Updated “Features” and “Functional Description” sections. ■ Added “Fast Read Operation” section. ■ Removed Table 4–2 to Table 4–9, Table 4-26, and Table 4–33. ■ Updated Table 3–1. ■ Updated Figure 3–2. ■ Removed “Referenced Documents” section.
October 2008	3.2	■ Updated “Introduction”, “Active Serial FPGA Configuration”, “Operation Codes”, “Read Status Operation”, “Read Device Identification Operation”, and “Package” sections. ■ Updated Table 4–10, Table 4–25, Table 4–26, and Table 4–32. ■ Updated Figure 4–5, Figure 4–13, and Figure 4–19. ■ Added Figure 4–22. ■ Added Table 4–33. ■ Updated new document format.
May 2008	3.1	■ Updated Table 4–3, Table 4–6, Table 4–7, Table 4–28, and Table 4–29. ■ Deleted Note 5 to Table 4–31. ■ Added “Referenced Documents” section.

Table 25. Document Revision History (Part 3 of 4)

Date	Version	Changes
August 2007	3.0	■ Updated “Introduction” section. ■ Updated “Functional Description” section. ■ Updated Table 4–1 through Table 4–4 and Table 4–7 through Table 4–9 to with EPCS128 information. ■ Added Table 4–6 on Arria GX. ■ Added notes to Figure 4–3. ■ Added notes to Figure 4–4. ■ Updated Table 4–10 with EPCS128 information. ■ Added new Table 4–11 on address range for sectors in EPCS128 device. ■ Updated Table 4–16 with information about “Read Device Identification” and added (Note 5). ■ Added new Table 4–21 on block protection bits in EPCS128. ■ Added notes to Figure 4–12. ■ Added new section “Read Device Identification Operation” with Table 4–23 and Figure 4–13. ■ Updated “Write Bytes Operation”, “Erase Bulk Operation” and “Erase Sector Operation” sections. ■ Updated Table 4–24 to include EPCS128 information. ■ Updated (Note 1) to Table 4–26. ■ Updated VCC and VI information to include EPCS128 in Table 4–27. ■ Updated VIH information to include EPCS128 in Table 4–29. ■ Updated ICC0 and ICC1 information to include EPCS128 in Table 4–30. ■ Updated Figure 4–21 and Table 4–34 with EPCS128 information.
April 2007	2.0	■ Updated “Introduction” section. ■ Updated “Functional Description” section and added handpara note. ■ Added Table 4–4, Table 4–6, and Table 4–7. ■ Updated “Active Serial FPGA Configuration” section and its handpara note. ■ Added notes to Figure 4–2. ■ Updated Table 4–26 and added (Note 1). ■ Updated Figure 4–20. ■ Updated Table 4–34.
January 2007	1.7	■ Removed reference to PLMSEPC-16 in “Programming and Configuration File Support”. ■ Updated DCLK pin information in Table 4–32.
October 2006	1.6	■ Updated Figure 4–19. ■ Updated Table 4–30 and Table 4–32.
August 2005	1.5	■ Updated table 4-4 to include EPCS64 support for Cyclone devices.
August 2005	1.4	■ Updated tables. ■ Minor text updates.
February 2005	1.3	Updated hot socketing AC specifications.

Table 25. Document Revision History (Part 4 of 4)

Date	Version	Changes
October 2003	1.2	■ Added Serial Configuration Device Memory Access section. ■ Updated timing information in Tables 4–10 and 4–11 section. ■ Updated timing information in Tables 4-16 and 4-17.
July 2003	1.1	Minor updates.
May 2003	1.0	■ Added document to the *Cyclone Device Handbook.* ■ Initial release.