AN98490

Using CFI to Read and Debug Systems

AN98490 describes how to correctly read the CFI table from a flash device during board development.

1 Introduction

In the course of board development, being able to correctly read the CFI table from a Flash is an important milestone. The objective of this application note is to help development engineers achieve this.

2 CFI Mode

Cypress NOR Flash devices have a command for accessing Common Flash Interface (CFI) information. This command is useful when debugging for the following reasons:

1. *A new Flash comes from the factory erased (all ones).* In the erased state, reading from the array (the Flash) should produce 0xFFh (for 8 bit data busses), 0xFFFFh (for 16 bit data busses) or 0xFFFFFFFFh (for 32 bit data busses). Unfortunately, reading from a device that does not work may yield the same results.
2. *CFI data is read only memory that is only accessible by a specific command (CFI Entry).* If you can read CFI data (look in your Cypress data sheet for the values of the CFI read only memory) it means that your:
    a. Data lines are connected correctly.
    b. Address lines are connected correctly.
    c. The device is being chip enabled correctly.
    d. RE# and WE# are working correctly.
3. *The CFI command is very simple—0x98h—but it must be sent to a specific address.* The address of where to send the command is determine as follows:

    base address of the Flash + (0x55 * X)

    where X is typically 1 for an 8-bit interface to Flash, 2 for a 16-bit interface, or 4 for a 32-bit Flash.

3 CFI Debug Procedure

Use the following procedure to debug code for reading CFI data.

1. Determine the base address of the Flash.
2. Determine the data bus width to the Flash so you can choose the appropriate read/write in step 3.
3. Using a JTAG emulator (or debugger, or other tool), write 0x98h at the appropriate width to the appropriate address. Items a–c are examples of how to put the Flash into CFI mode, and assume the following: base address is 0x2000000h, W8 is an 8-bit write, W16 is a 16-bit write & W32 is a 32-bit write. Your specific read and write commands will be different.
    a. W8 2000055 98
    b. W16 20000AA 98
    c. W32 2000154 98

4. Using the same tool that wrote to the Flash, perform the associated read. Assumptions as above still apply. The purpose of *all* of these read examples is to read the first letter of the “QRY” (‘Q’ which is 0x51h) string.
   a. R8 2000020
   b. R16 2000040
   c. R32 2000080

3.1 Software Troubleshooting

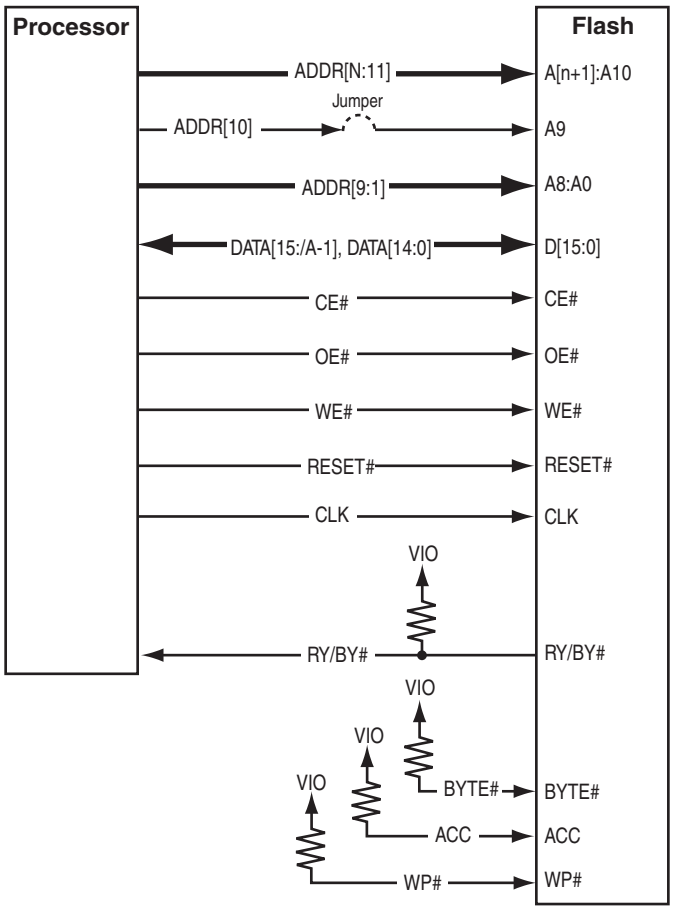
When the hardware is correctly configured, several iterations may be required. The following list provides some hints.

1. Bus errors and exceptions are the result of
   a. The memory controller not being setup. You must tell the memory controller the equivalent of: “For accesses between 0x2000000 and 0x2FFFFFF” assert CE1 (chip enable one).
   b. The memory controller should not block write cycles (needed for device commands) to the Flash. The Flash is not a read-only device, but some users set their systems to block writes to the Flash device as a method of write protection. If you cannot write to the Flash, then you cannot send commands to the Flash.
   c. Unaligned accesses. For most processors performing reads and writes to a 32-bit device, you are required to give an address that is evenly divisible by four. For 16-bit devices, the requirement is an address that is evenly divisible by two.
2. After writing a 0x98h and performing the appropriate read, if you do not read the expected ‘Q’, you can continue to experiment.
3. If you have a logic analyzer available, attach it to the Flash address lines, the Flash data lines and a few Flash control lines (WE#, OE#, CE#). You will be able to see what the Flash sees when you write 0x98h. You will then be able to deduce the problem much quicker. Note that the logic analyzer must be configured correctly to produce valid results. For example, the logic analyzer must be configured to interpret the voltage of a digital signal the same way the Flash device does. To accomplish this, you would typically set a threshold voltage; signals above the threshold voltage are interpreted as ‘1’s and those below as ‘0’s.
4. If steps 1–3 do not produce the desired result, begin examining the hardware.
   a. Was the board design correct? Flash is unlike some RAM devices, where it does not matter how a processor’s data lines are connected to them.
   b. Is the Flash device soldered down correctly? (check for device orientation, cold solder joints, etc.)
   c. Refer to the *Hardware Troubleshooting* section of this document.

3.2 Hardware Troubleshooting

The following subsections describe a quick check of the circuitry involving the Flash. Figure 1 is an example of a typical implementation of a 16-bit wide data processor/Flash connection.

Figure 1. Typical Implementation of a 16-bit-wide Data Processor/Flash Connection



3.2.1 Address Signals

Connecting a processor address signals to the Flash is dependent upon the data width of the Flash. If the Flash is only an 8-bit (byte-wide) device, then the processor signal address A0 (least significant bit) is connected to the Flash device pin A0. If the Flash is a 16-bit (word-wide) device, then the processor address signal A1 (least significant bit + 1) is connected to the Flash device pin A0. If the Flash is a 32-bit (double-word-wide) device, then the processor address signal A2 (least significant bit + 2) is connected to the Flash device pin A0.

If the Flash is configurable as either a byte-wide device or a word-wide device, then the following is done: 8-bit-wide usage has the BYTE# pin held low and the processor address A0 (least significant bit) signal connected to Flash signal D15/A-1. 16-bit-wide usage has the BYTE# pin held high and the processor A1 (least significant bit + 1) signal connected to Flash signal A0.

3.2.2 Address Input A9

A jumper on ADDR[9] signal allows for the application of V_{ID} to the Flash A9 pin while preventing the V_{ID} voltage from being applied to the system ADDR[9] signal. The jumper on ADDR[9] signal is an optional item and is only used for Autoselect Mode testing. The processor A9 signal can be connected directly to the Flash without an intervening jumper if Autoselect Mode testing is not desired.

3.2.3 WP# or WP#/ACC

It is best to use external circuitry to define the state of the WP# pin, as most devices either do not provide internal pull-up resistors, or the internal pull-up resistors are very large (3 to 10 MΩ). External pull-up resistors will prevent unpredictable behavior in the chip. The preceding is also true for Flash devices that combine the WP# and ACC pin into a single WP#/ACC pin.

3.2.4 AC Electrical Check

The following subsections check signals in transition.

3.2.4.1 V_{CC} and V_{CCQ} Check

An incorrect V_{CC} ramp up is a common source for CFI read problems. Check the device data sheet for V_{CC} ramp-up requirements.

For example, the following V_{CC} ramp-up requirement is from the S29WS-N_00_10 data sheet of December 3, 2005.

- All V_{CC} signals must be ramped simultaneously to ensure correct power-up.
- V_{CC} ramp rate is > 1V/100 µs and for V_{CC} ramp rate of < 1V/100 µs a hardware reset is required.

Figure 2. S29WS-N Power-up

11.6 V_{CC} Power-up

Parameter	Description	Test Setup	Speed	Unit
t_{VCS}	V_{CC} Setup Time	Min	1	ms

Notes:

1. *All V_{CC} signals must be ramped simultaneously to ensure correct power-up.*
2. *S29WS128N: V_{CC} ramp rate is > 1V/ 100 µs and for V_{CC} ramp rate of < 1 V / 100 µs a hardware reset is required.*

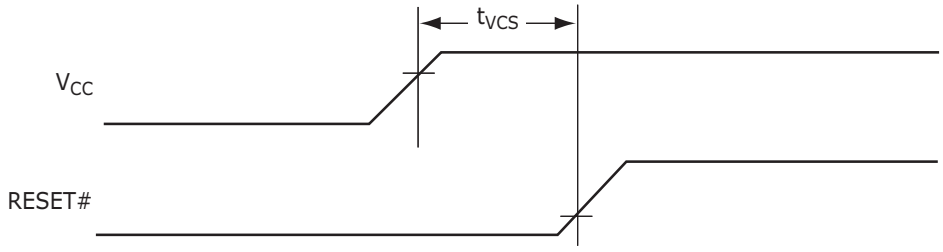
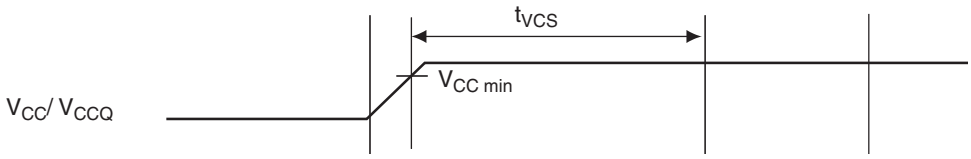


Figure 11.5. V_{CC} Power-up Diagram

Some devices require that V_{CC} and V_{CCQ} ramp up together. For example, the following V_{CC} ramp-up requirement is from the S29WS-P_00_A1 data sheet of March 7, 2004.

Figure 3. S29WS-P Power-up

Note: *V_{CC} and V_{CCQ} must be ramped simultaneously for proper power-up.*

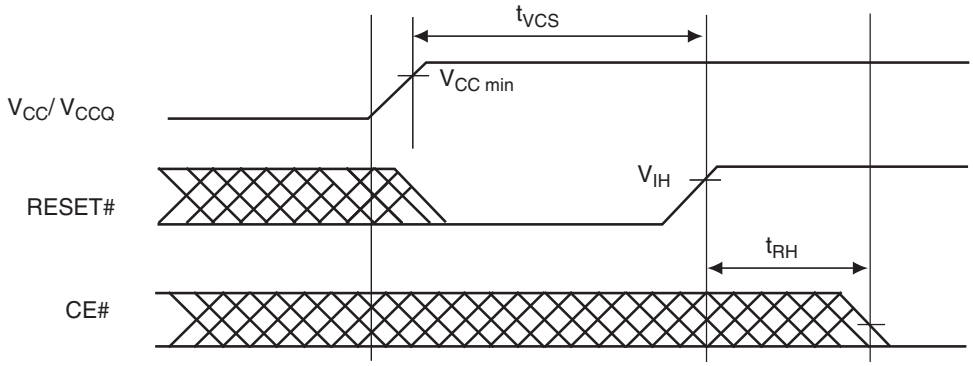


In both cases if the V_{CC} and/or V_{CCQ} ramp rates cannot be met, the Flash requires the RESET# pin to be pulsed low after both V_{CC} and V_{CCQ} are stable.

3.2.5 RESET#

An incorrect RESET# pulse is also common source for CFI read problems. Some devices require a delay between the ramp-up of V_{CC}/V_{CCQ} and RESET#. They can also require a delay before the first access to the Flash is performed. Please refer to the data sheet for the proper power-up and reset requirement of your device.

Figure 4. RESET# and V_{CC} Power-up



3.2.6 Clock

Flash devices reset to the asynchronous read mode. The CFI can be read in asynchronous mode, so the clock is not required for reading the CFI. However, the clock signal should be observed with an oscilloscope to verify that it is not outside of the voltage or frequency specifications.

3.2.7 Check Overshoots and Undershoots

Cypress devices can tolerate overshoot and undershoot, but the user should keep them within the device specification.

3.2.8 Signal Glitch / Non Monotonic / Too Slow Slew Rate

Verify that the signals into the Flash are monotonic and transition quickly between states. Some signals like CE# and WE# have a glitch rejection feature that will ignore glitches shorter then 5 ns. Input signals into the Flash should not be left floating and the output signals RY/BY# and RDY# are open-drain outputs that require a pull-up resistor if they are being used by the system.

3.2.9 Bus Loading

Verify that the Flash and the devices connected to the Flash bus can handle the bus loading of your system.

3.3 Autoselect Command Sequence Testing

The following subsections show a procedure for testing using the Autoselect command sequence. If the Flash device used can be configured for different data widths, there will be different versions of the Memory Array commands for each of these data widths. Use the table for the Flash data width of your system. Table 1 is for a Flash configured for 16-bit wide data.

Table 1. Typical Cypress Flash Memory Array Commands

Command Sequence		Cycles	Bus Cycles											
			First		Second		Third		Fourth		Fifth		Sixth	
			Addr	Data	Addr	Data	Addr	Data	Addr	Data	Addr	Data	Addr	Data
Asynchronous Read		1	RA	RD										
Reset		1	XXX	F0										
Autoselect	Manufacturer ID	4	555	AA	2AA	55	555	90	X00	01				
	Device ID	6	555	AA	2AA	55	555	90	X01	227E	X0E	Data	X0F	Data
	Sector Protect Verify	4	555	AA	2AA	55	555	90	[SA]X02	Data				
	Secure Device Verify	4	555	AA	2AA	55	555	90	X03	Data				
CFI Query		1	55	98										

Table 1 shows the four-cycle command sequence used to obtain the Manufacturer ID. There are three write cycles and one read cycle. The first write cycle is to Flash address 0x555 with data 0xAA. The second write cycle is to Flash address 0x2AA with data 0x55. The third write cycle is to Flash address 0x555 with data 0x90. The read cycle is from Flash address 0xx and should be data 0x01.

3.3.1 Configuring the Memory Controller

Using the Autoselect command sequence usually requires configuring the system processor's memory controller. If applicable, configure the system's memory controller for the chip select, data width, and timing requirements of the Flash. The most relaxed timing setting usually provides the best conditions for a successful CFI read.

3.3.2 Autoselect Command Debug Actions

If an attempted autoselect command fails, try the following actions to gain more information.

- Add Cypress software tools to your system software to help track the processor activity (see www.cypress.com).
- Disable complier code optimizations that could eliminate CFI accesses with the Flash. Assign data reads from the Flash to "violate" variables so that the compiler will not optimize them out.
- Use a logic analyzer or oscilloscope (if practical) to verify control over the Flash signals.
- Verify that these signals can move independently of each other.
- Disable data caching with the Flash.
- Disable interrupts and run the Autoselect command sequence together as a single uninterrupted sequence of bus cycles at the Flash.
- Write a Flash software reset command before reading the CFI.
- Assert the RESET# signal immediately before reading the Flash.
- Relax the timing of the signals at the Flash.
- Use single-cycle bus operations for CFI accesses. A single-cycle bus operation includes the assertion and negation of the control signals CE# and OE#, or CE# and WE#.
- Pre-program a Flash with a sequence pattern of data and install it into your system. Read the Flash to verify the memory.
- Access other devices that share signals with the Flash. Depending on how they work or do not work, information on the integrity of the shared signals with the Flash.
- Move the Flash between "working" and "non-working" circuit boards and see it the problem moves with the Flash or stays with the board.
- Check the terminations of signals to the Flash. Change the terminations if needed.
- Heat or cool the Flash or other components in your system to see if the problem is related to temperature.
- Vary the V_{CC} and V_{IO} voltages to the Flash within the specifications range.

- Use an address is that outside of the expected values of the Memory Array Command table. Shift the address bits left or right. Do this with the data bit too.
- Vary the CFI Command sequence. Send some commands twice or in a different order.
- Check power margins for the devices involved with the Flash at their device pins. Use a FET oscilloscope probe with minimal probe length and a persistence display.
- Use Autoselect Mode to read CFI values.

3.4 Autoselect Mode Testing

If the Autoselect command sequence and debug actions were not successful then using the Autoselect Mode can provide useful debug information about the system. The advantage of the Autoselect Mode is that bus cycles are not needed to read out the CFI values. Just applying the DC voltages to the Flash inputs as defined in the table will provide the CFI value on the Flash data pins. In this way the user can show that the Flash device itself is working at least to the level of Autoselect Mode.

The advantage of Autoselect Mode over the Autoselect Command Sequence is that only static DC levels (high and low) pin levels are needed. In Autoselect Mode the Flash data signals are used only for output and not input.

3.4.1 Using Autoselect Mode

To use Autoselect Mode in a circuit, some features are required of the circuit design. To enter Autoselect mode V_{ID} needs to be applied to the Flash A9 pin. Note that V_{ID} could be too high a voltage for other devices in the system which is why the isolation jumper is on the ADDR[9] signal. Also, high or low DC voltages must be driven into the Flash control pins. To avoid bus contention, the other circuitry must be made passive on signals shared with the Flash.

Using the Autoselect Mode requires the following:

1. Isolate the Flash A9 pin to prevent damage to other parts of the circuit.
2. Disable the outputs of other devices sharing the address, data, and control bus with the Flash.
3. Apply V_{ID} only to the Flash A9 pin.
4. Apply V_{IH} and V_{IO} to the Flash inputs as defined by the Autoselect Code table.
5. Measure the voltage levels on Flash data pins D[15:0].

Note that the above steps must be reversed to return to normal board operation. Table 1 shows the autoselect codes obtained using the high voltage method (A9 held at V_{ID}).

Table 2. Autoselect Codes (High Voltage Method) (Sheet 1 of 2)

Description		CE#	OE#	WE#	A22 to A15	A14 to A10	A9	A8 to A7	A6	A5 to A4	A3 to A2	A1	A0	DQ8 to DQ15		DQ7 to DQ0
														BYTE# = V_{IH}	BYTE# = V_{IL}	
Manufacturer ID: Cypress Product		L	L	H	X	X	V_{ID}	X	L	X	L	L	L	00	X	01h
Device ID S29GL512N	Cycle 1	L	L	H	X	X	V_{ID}	X	L	X	L	L	H	22	X	7Eh
	Cycle 2										H	H	L	22	X	23h
	Cycle 3										H	H	H	22	X	01h
Device ID S29GL256N	Cycle 1	L	L	H	X	X	V_{ID}	X	L	X	L	L	H	22	X	7Eh
	Cycle 2										H	H	L	22	X	22h
	Cycle 3										H	H	H	22	X	01h
Device ID S29GL128N	Cycle 1	L	L	H	X	X	V_{ID}	X	L	X	L	L	H	22	X	7Eh
	Cycle 2										H	H	L	22	X	21h
	Cycle 3										H	H	H	22	X	01h
Sector Group Protection Verification		L	L	H	SA	X	V_{ID}	X	L	X	L	H	L	X	X	01h (protected), 00h (unprotected)

Table 2. Autoselect Codes (High Voltage Method) (Sheet 2 of 2)

Description	CE#	OE#	WE#	A22 to A15	A14 to A10	A9	A8 to A7	A6	A5 to A4	A3 to A2	A1	A0	DQ8 to DQ15		DQ7 to DQ0
													BYTE# = V_{IH}	BYTE# = V_{IL}	
Secured Silicon Sector Indicator Bit (DQ7), WP# protects highest address sector	L	L	H	X	X	V_{ID}	X	L	X	L	H	H	X	X	98h (factory locked), 18h (not factory locked)
Secured Silicon Sector Indicator Bit (DQ7), WP# protects lowest address sector	L	L	H	X	X	V_{ID}	X	L	X	L	H	H	X	X	88h (factory locked), 08h (not factory locked)

Legend

L = Logic Low = V_{IL}, H = Logic High = V_{IH}, SA = Sector Address, X = Don't care.

If BYTE# = V_{IL}, and all of the Autoselect Mode CFI reads correctly, then D7, D4, D3, and D0 are proven to be independent. Note that D5/D1 (italic) and D6/D2 (bold/italic) are pairs whose values remain the same within each pair for all of the Autoselect Codes.

Table 3. Proven Independence of D[7:0] from Autoselect Mode Testing

	D7	D6	D5	D4	D3	D2	D1	D0
01h	0	***0***	*0*	0	0	***0***	*0*	1
7Eh	0	***1***	*1*	1	1	***1***	*1*	0
23h	0	***0***	*1*	0	0	***0***	*1*	1
7Eh	0	***1***	*1*	1	1	***1***	*1*	0
22h	0	***0***	*1*	0	0	***0***	*1*	0
98h	1	***0***	*0*	1	1	***0***	*0*	0
88h	1	***0***	*0*	0	1	***0***	*0*	0
18h	0	***0***	*0*	1	1	***0***	*0*	0

Verifying the Autoselect Mode table also verifies the functionality of the CE#, OE#, WE#, A9, A6, A3-A0, and BYTE# pins.

4 Contact Cypress Support

If after these efforts the CFI still can not be read, contact Cypress technical support. Provide as much of acquired data has possible. This will help to resolve these problem faster since fewer communications will be needed. When you contact Cypress, please provide the following information:

- What is the full part number?
- Is the CFI read error consistent across all units?
- Is it 100% repeatable?
- Is there something that triggers it?
- Is there something that prevents it?
- What percentage of units them have this problem and how many units were tested?
- What changes in hardware or software are related to this problem?

See the Cypress website (www.cypress.com) for the latest information on Cypress support.

5 Conclusion

The procedures in this document should enable your system to read CFI data, and will also provide basic knowledge of your system. This information can prove useful for when encountering other Flash tasks in the future.

Document History Page

Document Title: AN98490 - Using CFI to Read and Debug Systems Document Number: 001-98490				
Rev.	**ECN No.**	**Orig. of Change**	**Submission Date**	**Description of Change**
**	–	–	03/20/2007	Initial version
*A	4980741	MSWI	10/22/2015	Updated in Cypress template
*B	5842966	AESATMP8	08/03/2017	Updated logo and Copyright.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

ARM® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP | PSoC 6

Cypress Developer Community

Forums | WICED IOT Forums | Projects | Video | Blogs | Training | Components

Technical Support

cypress.com/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709