



XAPP951 (v1.3) September 23, 2010

Configuring Xilinx FPGAs with SPI Serial Flash

Author: Stephanie Tapp

Summary

This application note discusses the Serial Peripheral Interface (SPI) configuration mode introduced in the Virtex®-5 and Spartan®-3E FPGA families. The required connections to configure the FPGA from an SPI serial flash device are discussed and the configuration flow for the SPI mode is shown. Special precautions for configuring from an SPI serial flash are given, and the ISE® Design Suite iMPACT direct SPI programming solution is described.

Note: The ISE Suite iMPACT tool version 11.4 is the last supported release for direct in-system SPI programming and is captured in this application note. For new designs, the iMPACT indirect in-system SPI programming solution is recommended. This solution uses a single JTAG connection to both configure the FPGA and indirectly program the flash. For additional information see *Introduction to Indirect Programming — SPI or BPI Flash Memory* in the iMPACT Help at http://www.xilinx.com/support/documentation/sw_manuals/xilinx11/isehelp_start.htm.

The principles described in this application note apply to the external SPI flash configuration mode of the Extended Spartan-3A family with few differences. See [UG332, Spartan-3 Generation Configuration User Guide](#) for the unique details and requirements of the Extended Spartan-3A family's SPI configuration mode.

Introduction

Xilinx FPGAs are CMOS configurable latch (CCL) based and must be configured at power-up. Traditionally, Xilinx FPGA configuration is accomplished via the IEEE Std 1149.1 (JTAG) interface, a microprocessor, or the Xilinx PROMs (Platform Flash PROMs). In addition to these traditional methods, a direct configuration interface to SPI serial flash is now available.

The direct configuration interface for SPI serial flash memories in the Virtex-5 and Spartan-3E FPGAs broadens the available configuration solutions for Xilinx designers and is the focus of this application note. SPI serial flash memories are popular because they can be easily accessed post-configuration, offering random-access, non-volatile data storage to the FPGA. Systems with SPI serial flash memory already onboard can also benefit from having the option to configure the FPGA from the same memory device.

The SPI protocol does have a few variations among vendors. Variations among some vendors are highlighted along with the connections required between the FPGA and SPI serial flash memory for configuration. The ISE software tools for SPI-formatted PROM file creation and programming during prototyping for select vendors are shown. SPI serial flash memories are not supplied by Xilinx and must be purchased from third-party vendors such as Numonyx.

SPI Basics

SPI serial flash memories use the Serial Peripheral Interface (SPI), a four-wire, synchronous serial data bus. This serial data link was pioneered as a serial communication interface between a microcontroller and its peripherals and is a popular interface in embedded and consumer markets. This interface can now also be used to configure Xilinx FPGAs.

An SPI system typically consists of a master device and a slave device ([Figure 1](#)). When using this four-signal interface to configure a Xilinx FPGA from an SPI serial flash, the FPGA is the master device and the SPI serial flash is the slave device.

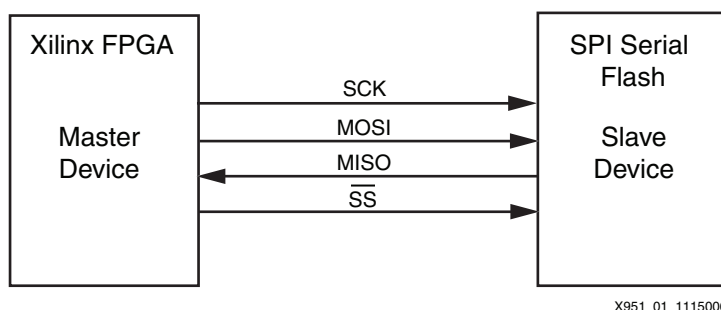


Figure 1: Basic Block Diagram for SPI Configuration Mode

The master FPGA device controls the timing via the SCK clock signal. Data is clocked out of the FPGA master and into the SPI serial flash slave on the MOSI signal after the select signal $\overline{SS}$ goes Low. During the same clock cycle, data is clocked out of the SPI serial flash slave and into the FPGA master using the MISO signal. Data is clocked out of each device on one edge and clocked into each device on the next opposite edge in the period.

In addition to the four-signal interface, each SPI serial flash vendor has unique control signals, such as write protect or hold, that need to be controlled appropriately during programming and configuration (refer to the appropriate vendor's SPI serial flash memory data sheet for additional details on the specific control signals).

A cross reference for the FPGA to SPI interface connections is provided in [Table 1](#).

Table 1: SPI Serial Flash Interface Connections and Pin Naming

SPI Signals	SPI Serial Flash Pins ⁽¹⁾	FPGA Connection (Spartan-3E/Virtex-5 FPGAs)	Signal Description
General SPI Signals			
MOSI	D	MOSI	Master Out Slave In is used by the master to specify the instruction to execute or to send data to the slave device.
MISO	Q	DIN/D_IN	Master In Slave Out is used by the master to collect data transferred from the slave device.
$\overline{SS}$	S	CSO_B/FCS_B ⁽²⁾	Slave Select, active-Low signal; when driven High this signal is used to deselect the slave device and put MISO at high impedance.
SCK	C	CCLK	Serial Clock provides the timing for the serial interface.
Additional Vendor-Specific SPI Control Signals			
Write Protect	$\overline{W}$	Not required for FPGA configuration, but must be High to program or erase SPI serial flash. Optional connection to FPGA user I/O.	Write Protect protects select areas of memory against program or erase instructions.
Hold	$\overline{HOLD}$	Not required for FPGA configuration, but must be High during FPGA configuration and SPI erase or program. Optional connection to FPGA user I/O.	Hold is used to pause any serial communications with the device without deselecting the device.

Notes:

- General SPI serial flash pin names are listed in this table with the most common vendor pin names. The subset of SPI control signals used by each vendor can vary. Refer to the vendor data sheet for specific pin information and descriptions.
- The CSO_B signal is used on Spartan-3E FPGAs and the FCS_B signal is used on the Virtex-5 FPGAs to interface to the SPI serial flash for configuration. On Virtex-5 FPGAs, the CSO_B signal does not control the chip select on the SPI serial flash but is instead used for advanced daisy-chains.

Configuring FPGAs from SPI Serial Flash

Spartan-3E and Virtex-5 FPGAs can be configured from a single SPI serial flash memory. The typical configuration density requirements for these FPGAs are provided in [Table 2](#).

Table 2: Typical Configuration Bit Requirements

FPGA	Configuration Bits (Per Device)	Smallest SPI Serial Flash Required
Spartan-3E Family		
XC3S100E	581,344	1 Mb
XC3S250E	1,353,728	2 Mb
XC3S500E	2,270,208	4 Mb
XC3S1200E	3,837,184	4 Mb
XC3S1600E	5,969,696	8 Mb
Virtex-5 Family		
XC5VLX30	8,374,016	8 Mb
XC5VLX50	12,556,672	16 Mb
XC5VLX85	21,845,632	32 Mb
XC5VLX110	29,124,608	32 Mb
XC5VLX155	41,048,064	64 Mb
XC5VLX220	53,139,456	64 Mb
XC5VLX330	79,704,832	128 Mb
XC5VLX20T	6,251,200	8 Mb
XC5VLX30T	9,371,136	16 Mb
XC5VLX50T	14,052,352	16 Mb
XC5VLX85T	23,341,312	32 Mb
XC5VLX110T	31,118,848	32 Mb
XC5VLX155T	43,042,304	64 Mb
XC5VLX220T	55,133,696	64 Mb
XC5VLX330T	82,696,192	128 Mb
XC5VSX35T	13,349,120	16 Mb
XC5VSX50T	20,019,328	32 Mb
XC5VSX95T	35,716,096	64 Mb
XC5VSX240T	79,610,368	128 Mb
XC5VFX30T	13,517,056	16 Mb
XC5VFX70T	27,025,408	32 Mb
XC5VFX100T	39,389,696	64 Mb
XC5VFX130T	49,234,944	64 Mb
XC5VFX200T	70,856,704	128 Mb
XC5VTX150T	43,278,464	64 Mb
XC5VTX240T	65,755,648	128 Mb

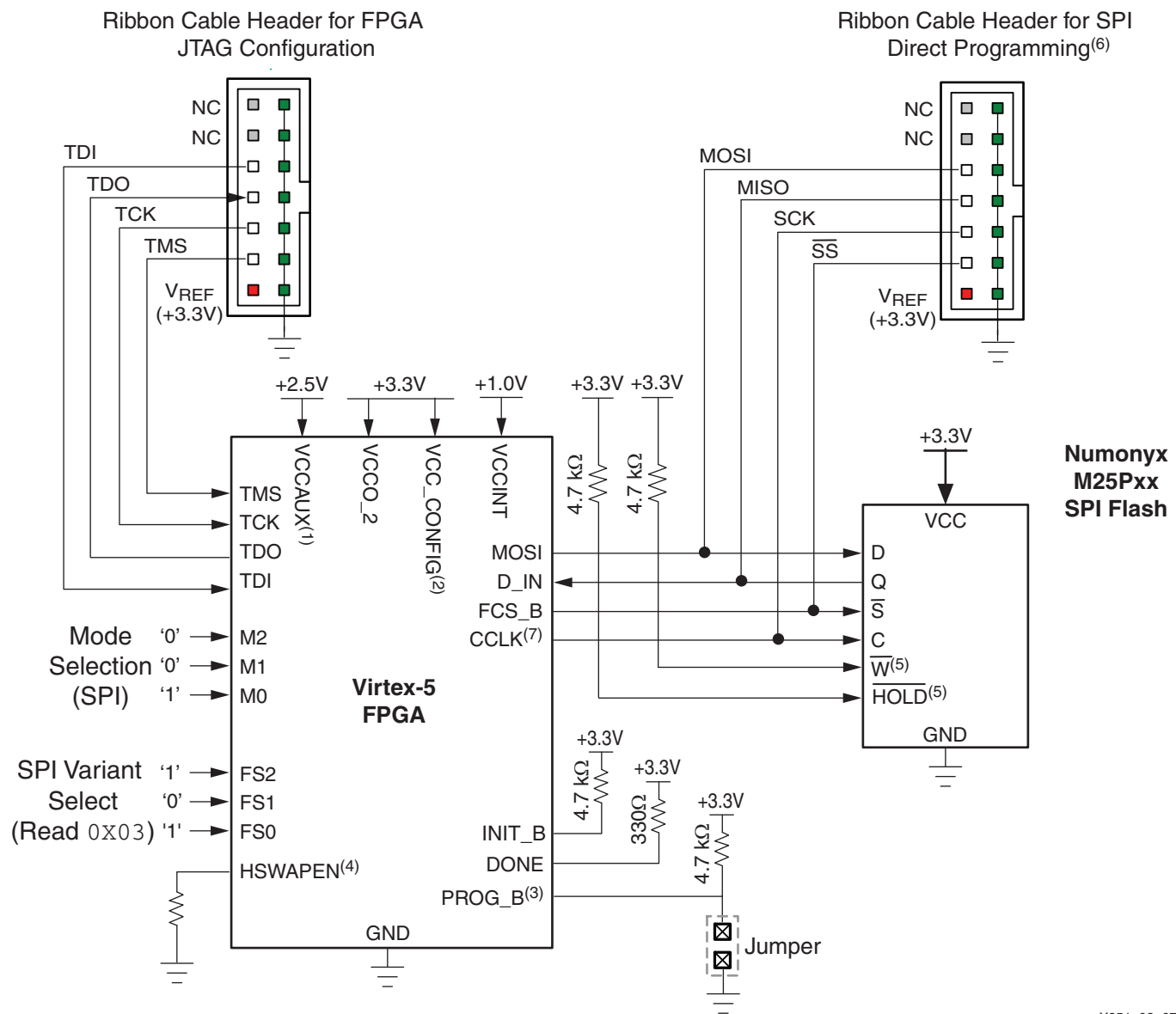
Notes:

1. A larger SPI serial flash device can be used for daisy-chained applications, storing multiple FPGA configuration bitstreams, or for applications storing additional user data, such as code for the embedded MicroBlaze™ or PowerPC™ processors. Daisy-chaining multiple Spartan-3E FPGAs via a single flash is only supported on Stepping 1 and later.

A detailed SPI configuration setup is shown in [Figure 2, page 5](#), where the Virtex-5 FPGA is the master and the Numonyx SPI serial flash is the slave. The configuration connections from the SPI serial flash to the FPGA are highlighted in this diagram. For information on the programming and configuration headers used by the Xilinx cables, refer to [Hardware and Connections for SPI Programming](#).

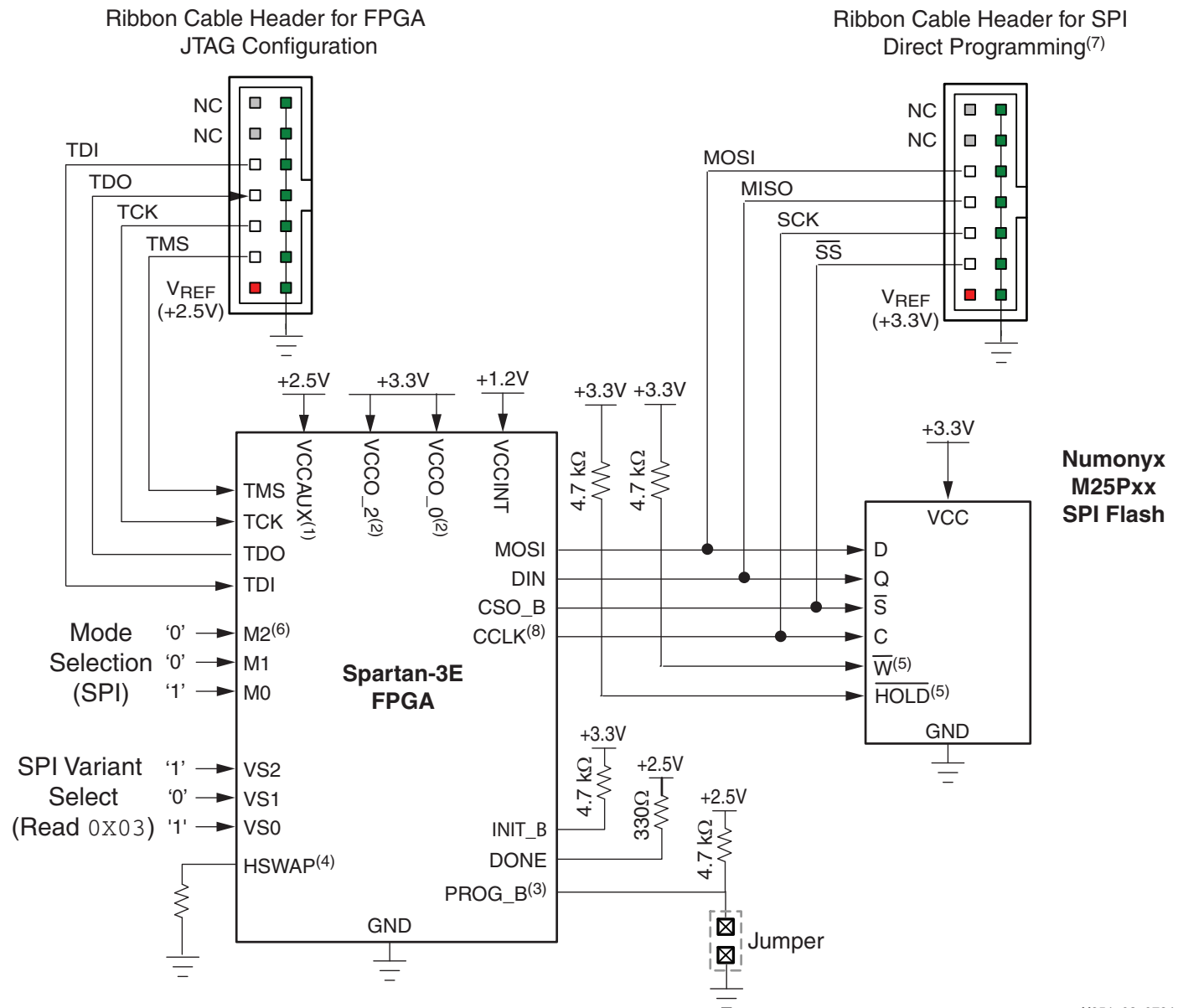
A detailed SPI configuration setup is shown in [Figure 3, page 6](#), where the Spartan-3E FPGA is the master and the Numonyx SPI serial flash is the slave. The configuration connections from the SPI serial flash to the FPGA are highlighted in the diagram. For information on the programming and configuration headers used by the Xilinx cables, refer to [Hardware and Connections for SPI Programming](#).

In addition to the SPI serial flash interface signals discussed in [SPI Basics, page 1](#), there are additional FPGA configuration signals that can influence the successful start and stop of data transfer. These FPGA signals and their descriptions are listed in [Table 3, page 7](#) and are shown in [Figure 2](#) and [Figure 3](#).



X951_02_072410

Figure 2: Virtex-5 FPGA Configuration from Numonyx SPI Serial Flash Connection Diagram (Example for the Read Command 0x03)



X951_03_072410

Notes:

1. VCCAUX supplies the dedicated Spartan-3E FPGA configuration pins: TMS, TDI, TDO, PROG_B, and DONE.
2. VCCO_2 supplies the voltage to the Spartan-3E FPGA configuration, dual-mode pins: M[2:0], VS[2:0], INIT_B, CCLK, CSO_B, DIN, MOSI; and VCCO_0 supplies the dual-mode pin: HSWAP.
3. PROG_B should be held Low during the direct programming of the SPI serial flash. PROG_B can be driven Low to High with external logic to reconfigure the FPGA.
4. HSWAP can be driven low to enable pull-ups on I/O. Refer to [Table 3, page 7](#).
5. Control signals should be driven appropriately when programming the SPI serial flash. Signals such as the $\overline{W}$ and $\overline{HOLD}$ signals should be held High or inactive while programming the SPI serial flash. Refer to the vendor's data sheet for more details.
6. For dual configuration mode usage, it is recommended to have the option to hold the M2 signal High for JTAG configuration mode.
7. Refer to [Table 5, page 11](#) for cable signal cross reference.
8. **Caution!** Care should be taken with the CCLK board layout. The Spartan-3E FPGA drives the internally generated CCLK signal to the FPGA CCLK output pin. The FPGA's internal configuration logic is clocked by the CCLK signal at the FPGA pin, therefore, any noise on the CCLK pin can affect the FPGA configuration. Guidelines and details for CCLK design see the "Configuration Clock:CCLK" section in [\[Ref 5\]](#).

Figure 3: Spartan-3E FPGA Configuration from Numonyx SPI Serial Flash Connection Diagram (Example for the Read Command 0x03)

Table 3: FPGA SPI Configuration Signal Names and Descriptions

Spartan-3E/ Virtex-5 FPGA Pin Name	FPGA Direction During Configuration	Description	During Configuration	After Configuration	
				Spartan-3E Devices	Virtex-5 Devices
PROG_B	Input	Program FPGA. Active Low. When asserted Low, forces the FPGA to restart its configuration process by clearing configuration memory and by resetting the DONE and INIT_B pins. Requires external 4.7 k Ω pull-up resistor. See the corresponding FPGA data sheet for the appropriate pull-up voltage. If driving externally, use an open-drain or open-collector driver.	Must be High to allow configuration to start.	Dedicated. Drive PROG_B Low and release to reprogram FPGA. Hold PROG_B to force FPGA I/O pins into High-Z, allowing direct programming access to SPI serial flash pins.	
INIT_B	Open-drain bidirectional I/O	Initialization Indicator. Active Low. Goes Low at start of configuration during initialization memory clearing process. Released at end of memory clearing, when mode and variant select pins are sampled. In daisy-chain applications, this signal requires an external 4.7 k Ω pull-up resistor to V _{CCO_2} .	Active during configuration. If SPI serial flash requires > 2 ms to awake after powering on, hold INIT_B Low until the flash is ready. If a CRC error is detected during configuration, FPGA drives INIT_B Low.	User I/O. If unused in the application, drive INIT_B High.	Dedicated.
M[2:0]	Input	Mode Select. Selects the FPGA configuration mode.	SPI mode M2=0, M1=0, M0=1. Sampled when INIT_B goes High.	User I/O.	Dedicated.
VS[2:0]/ FS[2:0]	Input	Variant Select. Instructs the FPGA how to communicate with the attached SPI serial flash.	Valid setting options are shown in Table 4. Must be at a valid setting when sampled as INIT_B goes High.	User I/O.	User I/O.
CSO_B/ FCS_B ⁽¹⁾	Output	Chip Select Output. Active Low.	Connects to the SPI serial flash chip-select input. If HSWAP/HSWAPEN=1, connect this signal to a 4.7 k Ω pull-up resistor. See the corresponding FPGA data sheet for the appropriate pull-up voltage.	Drive CSO_B High after configuration to disable the SPI serial flash and reclaim the MOSI, DIN, and CCLK pins as user I/O. Optionally, reuse this pin and MOSI, DIN, and CCLK to continue communicating with SPI serial flash.	Drive FCS_B High after configuration to disable the SPI serial flash and reclaim the pin as user I/O. The D_IN and CCLK are dedicated pins for configuration and cannot be reconfigured as user I/O. The STARTUP_VIRTE X5 primitive can be used to allow access to the pins after configuration.

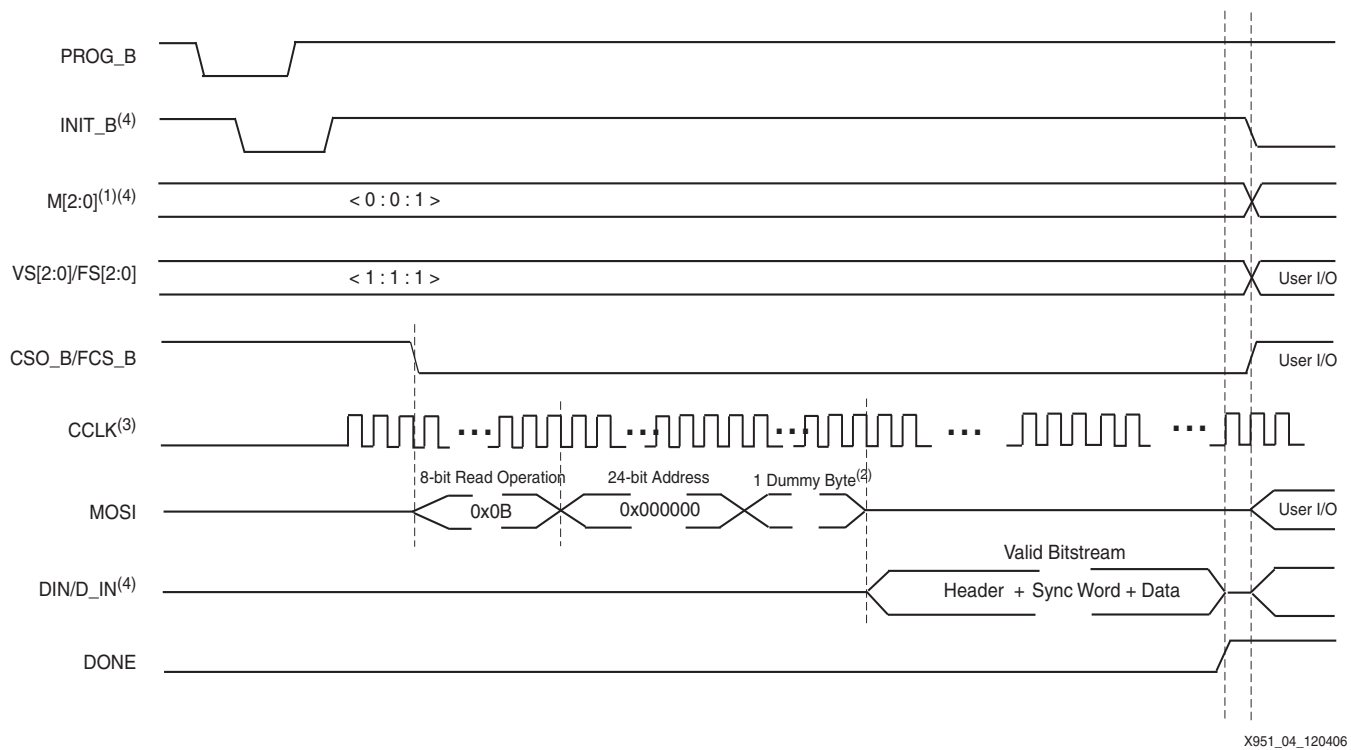
Table 3: FPGA SPI Configuration Signal Names and Descriptions (Cont'd)

Spartan-3E/ Virtex-5 FPGA Pin Name	FPGA Direction During Configuration	Description	During Configuration	After Configuration	
				Spartan-3E Devices	Virtex-5 Devices
CCLK	Output	Configuration Clock. Generated by FPGA internal oscillator. Frequency controlled by ConfigRate bitstream generator option. If CCLK PCB trace is long or has multiple connections, terminate this output to maintain signal integrity.	Drives the SPI serial flash clock input.	User I/O.	Dedicated.
MOSI	Output	Serial Data Output.	FPGA sends SPI serial flash read commands and starting address to the flash serial data input.	User I/O.	User I/O.
DIN/ D_IN	Input	Serial Data Input.	FPGA receives serial data from flash serial data output.	User I/O.	Dedicated.
DONE	Open-drain bidirectional I/O	FPGA Configuration Done. Low during configuration. Goes High when FPGA successfully completes configuration. Requires external 330Ω pull-up resistor. See the corresponding FPGA data sheet for the appropriate pull-up voltage.	Low indicates that the FPGA is not yet configured.	Pulled High via external pull-up. When High, indicates that the FPGA successfully configured.	
DOUT	Output	Serial Data Output.	Actively drives. Not used in single-FPGA configuration. In a daisy-chain configuration, this pin connects to DIN/D_IN input of the next FPGA in the chain. The downstream FPGA is now in Slave Serial mode.	User I/O.	Dedicated.
HSWAP/ HSWAPEN	Input	User I/O Pull-Up Control. When Low during configuration, enables pull-up resistors in all I/O pins to respective I/O bank V_{CCO} input: 0: Pull-ups during configuration 1: No pull-ups	Drive at valid logic level throughout configuration.	User I/O.	Dedicated.

Notes:

- For SPI ISP Programming, CSO_B is used for Spartan-3E FPGAs and FCS_B is used on Virtex-5 FPGAs. On Virtex-5 FPGAs, the CSO_B signal is used for the advanced daisy-chaining feature and not for SPI ISP programming. Refer to the FPGA data sheets or FPGA configuration user guides for additional information.

Figure 4 shows an overview of the SPI configuration mode timing diagram for the Xilinx FPGAs. This general timing diagram for the SPI interface applies to the Xilinx FPGA families, which support the new SPI direct interface.

**Notes:**

1. VS[2:0]/FS[2:0] settings can vary depending on which SPI vendor is used. See [Table 3, page 7](#) for selections.
2. The number of MOSI dummy bytes issued vary depending on the variant select (VS[2:0]/FS[2:0]) option. See [Table 4, page 9](#) for reference.
3. Default startup sequence is shown.
4. For Virtex-5 FPGAs these signals are dedicated and not available as user I/O after configuration. For Spartan-3E FPGAs these signals are available as User I/O after configuration.

Figure 4: SPI Configuration Flow for FAST READ (0x0B)

Upon power-up, or when the PROG_B pin is pulsed Low, the FPGA goes through an initialization sequence to clear the internal FPGA configuration memory. At the beginning of this sequence, both the DONE and INIT_B pins go Low. When the initialization has finished, the INIT_B pin goes High and the Mode and Variant Select (VS[2:0] or FS[2:0]) pins are sampled. The mode pins should be set for M[2:0]=<0:0:1> to enable the SPI configuration mode and the FPGA internal clock. In the SPI mode, the FPGA samples the Variant Select pins to determine which SPI command sequence to issue. Both the Mode pins and Variant Select pins must be at proper logic levels when the INIT_B signal is released after initialization to ensure proper sampling.

[Table 4](#) lists the available variant select codes on the Spartan-3E and Virtex-5 devices.

Table 4: Variant Select (VS[2:0] or FS[2:0]) Pin Codes

VS[2:0]/FS[2:0] ⁽¹⁾	SPI Serial Flash Command ⁽²⁾	Address Size	Dummy Bytes ⁽²⁾
<1:1:1>	FAST READ (0x0B)	24 bits	1 byte
<1:0:1>	READ (0x03)	24 bits	0 bytes
<1:1:0>	READ ARRAY (0xEB)	24 bits	4 bytes

Notes:

1. Variant Select pins are denoted as VS[2:0] for Spartan-3E FPGAs, and FS[2:0] for Virtex-5 FPGAs. This table lists the most popular read commands. For a complete listing of read command options, refer to the FPGA data sheets or configuration user guides.
2. SPI serial flash commands and dummy bytes are different between variants. Refer to the SPI serial flash vendor's specific data sheet for supported read commands.

After the SPI command set is selected by the Variant Select pins, the FPGA drives the CSO_B select signal Low and starts clocking the SPI serial flash via the FPGA's CCLK pin. The FPGA then sends an 8-bit read command followed by a 24-bit start address of 0x00_0000 and the appropriate number of dummy bytes for the targeted command set. The FPGA reads the SPI flash array starting from address 0 until the required number of configuration bits is read. If a valid bitstream is read from the memory device, the DONE signal is released indicating a successful configuration of the FPGA. After a successful configuration, all of the FPGA's SPI pins then become available as user I/O.

The FPGA signals used during configuration are listed in [Table 3, page 7](#). When configuring the FPGA in SPI mode, these signals must be tied as specified in the table for a successful configuration.

Power-On Considerations for SPI Serial Flash Configuration

At power-on, a race condition between the FPGA and SPI serial flash can exist. The FPGA sends a read command to the SPI serial flash to acquire the bitstream after the FPGA has completed its power-on-reset sequence. On the other hand, the SPI serial flash is not ready to receive a read command until the SPI serial flash's power-on-reset sequence has completed. Under specific conditions when the 3.3V power supply to the SPI serial flash powers up after the FPGA power supplies, the race condition can cause the SPI serial flash to miss the read command. The system must be designed such that the SPI serial flash is ready to receive the read command before the FPGA sends the read command.

Note: For additional information see the “Power-On Precautions if 3.3V Supply is Last” in the “Sequence” section in [\[Ref 4\]](#), or [\[Ref 3\]](#) for specifics regarding power-on considerations and precautions to ensure successful power-on configuration.

SPI Serial Flash Programming Options Programming

Similar to the traditional configuration memories, SPI serial flash memories must be loaded with the configuration data. SPI serial flash memories have a single interface for programming, but there are multiple methods to deliver the data to this interface. Three primary delivery methods exist to program an SPI serial flash through the SPI interface:

- Third-party programmers (off-board programming)
- Indirect in-system programming (JTAG tool vendor or custom solution)
- Direct in-system programming (SPI direct interface connect)

Production programming is often accomplished via a third-party programmer or JTAG tool vendor, and many distributors offer mass production gang programming. For prototyping, the iMPACT software, included in the Xilinx ISE development software tools, with a Xilinx parallel cable or Platform Cable USB can program select SPI serial flash memories directly ([Table 7, page 19](#)).

However, unlike the Xilinx Platform Flash PROMs ([\[Ref 4\]](#)), which are in-system programmable through a standard JTAG interface, SPI flash devices require an additional cable connector for the SPI direct in-system programming via Xilinx software and cables ([Figure 2, page 5](#)).

The following sections discuss the hardware connections required for the direct in-system programming of SPI serial flash for prototype designs. The Xilinx software tool flows to generate an SPI-formatted file and to program select SPI serial flash memories is also covered.

The following are required to successfully program the select SPI Serial Flash In-System:

- A Xilinx Cable (Parallel Cable IV or Platform Cable USB)
- Cable connector onboard
- Properly installed Xilinx ISE 8.2i software (or later — 11.4 is the last supported version and is described in this application note)

Hardware and Connections for SPI Programming

The Xilinx cables listed now support the direct SPI interface for programming SPI serial flash devices in addition to the traditionally supported modes (IEEE Std 1149.1, IEEE Std 1532, Slave Serial, and Slave SelectMAP).

All of these cables use a standard 14-pin ribbon cable. The ribbon cable is advantageous over flying leads because of the ease of connectivity and improved signal quality for programming at higher speeds. To program an SPI serial flash in-system, a ribbon cable connector should be included on the board. It is important that the signals are connected properly as shown in the pinout in [Figure 2, page 5](#) and in the cross reference [Table 5](#) because the existing cable pod labels do not reflect this newly supported programming mode.

Table 5: Download Header Signal Description for SPI Programming Mode

Ribbon Cable Number	SPI Programming Mode	JTAG/Slave Serial Configuration Mode Signal Cross Reference	Type	SPI Header Usage Description
2	V _{REF}	V _{REF}	In	Target Reference Voltage. This pin should be connected to a voltage bus on the target system that serves the JTAG, Slave-serial, or SPI interface. The target reference voltage must be regulated and must not have a current-limiting resistor in series with the V _{REF} pin.
4	$\overline{SS}$	TMS/PROG	Out	Chip Select (S). This pin is used to enable the device to accept an instruction.
6	SCK	TCK/CCLK	Out	SPI Clock (C). SPI flash memory clock provides the timing for the serial interface and is produced by the Xilinx cable.

Table 5: Download Header Signal Description for SPI Programming Mode (Cont'd)

Ribbon Cable Number	SPI Programming Mode	JTAG/Slave Serial Configuration Mode Signal Cross Reference	Type	SPI Header Usage Description
8	MISO	TDO/DONE	In	Serial Data Output (Q). This signal is used to transfer data serially out of the device.
10	MOSI	TDI/DIN	Out	Serial Data Input (D). This input signal is used to transfer data serially into the device. The device receives instructions, addresses, and the data to be programmed from this signal.
12	N/C	N/C	–	Reserved. This pin is reserved for Xilinx diagnostics and should not be connected to any target circuitry.
14	–	– /INIT	BIDIR	–
1, 3, 5, 7, 9, 11, 13	–	GND	GND	Digital Ground.

In addition to the cable connector and SPI serial flash power-up considerations, the designer must ensure the SPI serial flash V_{CC} matches the applied voltage and the CCLK trace is kept short to ensure a clean signal is delivered to the SPI serial flash and is present on the FPGA CCLK pin.

Finally, the FPGA pins driving the MOSI, MISO, SCK, and $\overline{SS}$ signals should also be high-impedance during SPI serial flash programming. Ensuring high-impedance on these inputs prevents any contention on these signals by the FPGA when the SPI serial flash is being programmed directly. Several methods available to place the FPGA SPI signals in high-impedance are listed below:

- Holding the FPGA's PROG_B pin Low 3-states all the I/O pins.
- Changing the FPGA's mode pins to JTAG mode ($M[2:0] = <1:0:1>$) and pulsing PROG_B forces all the FPGA I/Os to high-impedance.
- 3-stating the MOSI, MISO, SCK, and $\overline{SS}$ signals from within a functioning FPGA application when programming the attached SPI serial flash using the Xilinx ISE iMPACT.

Software Flows for SPI File Preparation and Programming

Preparing an SPI PROM File

This section provides guidelines and the software flow to create PROM files for SPI serial flash. Before converting a FPGA bitstream into an SPI-formatted PROM file, the designer must verify the bitstream was generated with the `bitgen -g StartupClk:Cclk` option. This option ensures proper FPGA functionality by synchronizing the startup sequence to the internal FPGA clock.

The Xilinx ISE software tools, PROMGen or iMPACT, generate SPI-formatted PROM files from the FPGA bitstream. The SPI serial flash serially outputs data bytes MSB first, while Xilinx PROMs output data LSB first. An SPI-formatted PROM file is bit-reversed within each byte from a standard Xilinx PROM file.

Preparing an SPI PROM File Using the ISE PROMGen Command-Line Software

The Xilinx ISE PROMGen software takes an FPGA bitstream (`.bit`) file as input and, with the appropriate options, generates a memory image file for the data array of an SPI serial flash. The output memory image file format is chosen through a PROMGen software command-line option. Typical file formats include Intel Hex (`.mcs`) and Motorola Hex (`.exo`).

The ISE PROMGen software utility is easily executed from a command-line (see [Table 5](#), [page 11](#) for ISE PROMGen software options used for SPI PROM file generation). An example PROMGen software command-line to generate an mcs-formatted file for a 64-Mbit (8192 kilobytes) SPI serial flash is:

```
promgen -spi -p mcs -o spi_flash.mcs -s 8192 -u 0 design.bit
```

The `-spi` option is required to ensure proper bit ordering within the SPI PROM file. The `-p mcs` option specifies Intel Hex (`.mcs`) output file format. The `-o spi_flash.mcs` specifies output to the `spi_flash.mcs` file. The `-s 8192` specifies a PROM file image size of 8192 kilobytes. The `-u 0` option specifies the data to start at address zero and fill the data array in the up direction. The `design.bit` is the input bitstream file.

[Table 6](#) list the various PROMGen options and the functions.

Table 6: Example PROMGen SPI PROM File Options

PROMGen Option	Description
<code>-spi</code>	Used to maintain the correct bit ordering required to configure the FPGA from an SPI serial flash device.
<code>-p <format></code>	PROM output file format. Commonly accepted PROM file formats include Intel Hex (<code>.mcs</code>) and Motorola Hex (<code>.exo</code>).
<code>-s <size></code>	Specifies the PROM size in kilobytes. The PROM size must be a power of two for this option, and the default setting is 64 kilobytes.
<code>-u <address></code>	Loads the <code>.bit</code> file from the specified starting address in an upward direction. This option must be specified immediately before the input bitstream file.

Preparing an SPI PROM File Using the ISE iMPACT Graphical Software

The ISE iMPACT 8.2i (or later) software integrates PROM file formatting and in-system programming features behind an intuitive graphical user interface (GUI). The PROMGen file formatting functionality is provided through a step-by-step wizard in the iMPACT software. The wizard steps through the output PROM file options and input bitstream selections. A final step is required for iMPACT to generate the PROM file.

In the iMPACT software, an SPI PROM file can be generated from a Xilinx FPGA bitstream through a simple process. A prescribed sequence of dialog boxes (also known as a wizard) acts as a guide through most of the PROM file generation process.

The following section demonstrates the iMPACT 11.4 software process for generating an SPI-formatted PROM file in the MCS file format for a 64 Mb SPI serial flash. The demonstrated process takes the `design.bit` FPGA bitstream file as input and generates a PROM file named `spi_flash.mcs`.

Step 1: Create a New Project for PROM File Generation

After launching the iMPACT software, the iMPACT project dialog box is displayed (Figure 5). Choose the “create a new project (.ipf)” option. Optionally, specify a project location via the **Browse...** button. Then, click **OK** to continue to step 2 in the process.

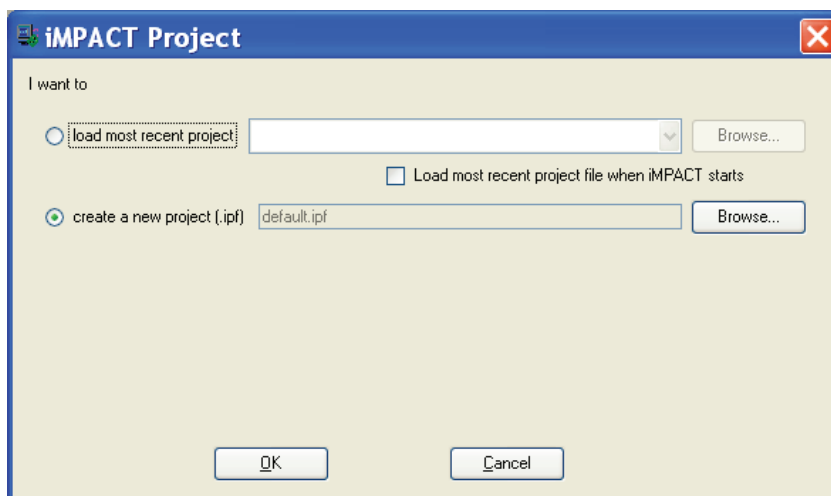


Figure 5: Create a New Project for PROM File Generation

Step 2: Choose to Prepare a PROM File

The first dialog box of the wizard displays the available kinds of projects that can be created (Figure 6). Check **Prepare a PROM File** and select **OK** to proceed to step 3 of the process.

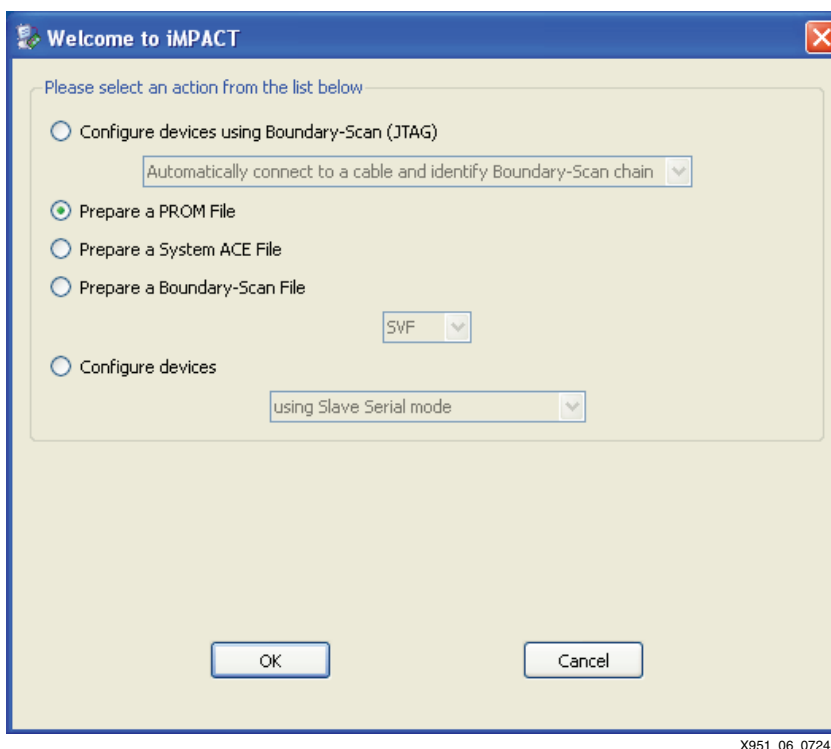
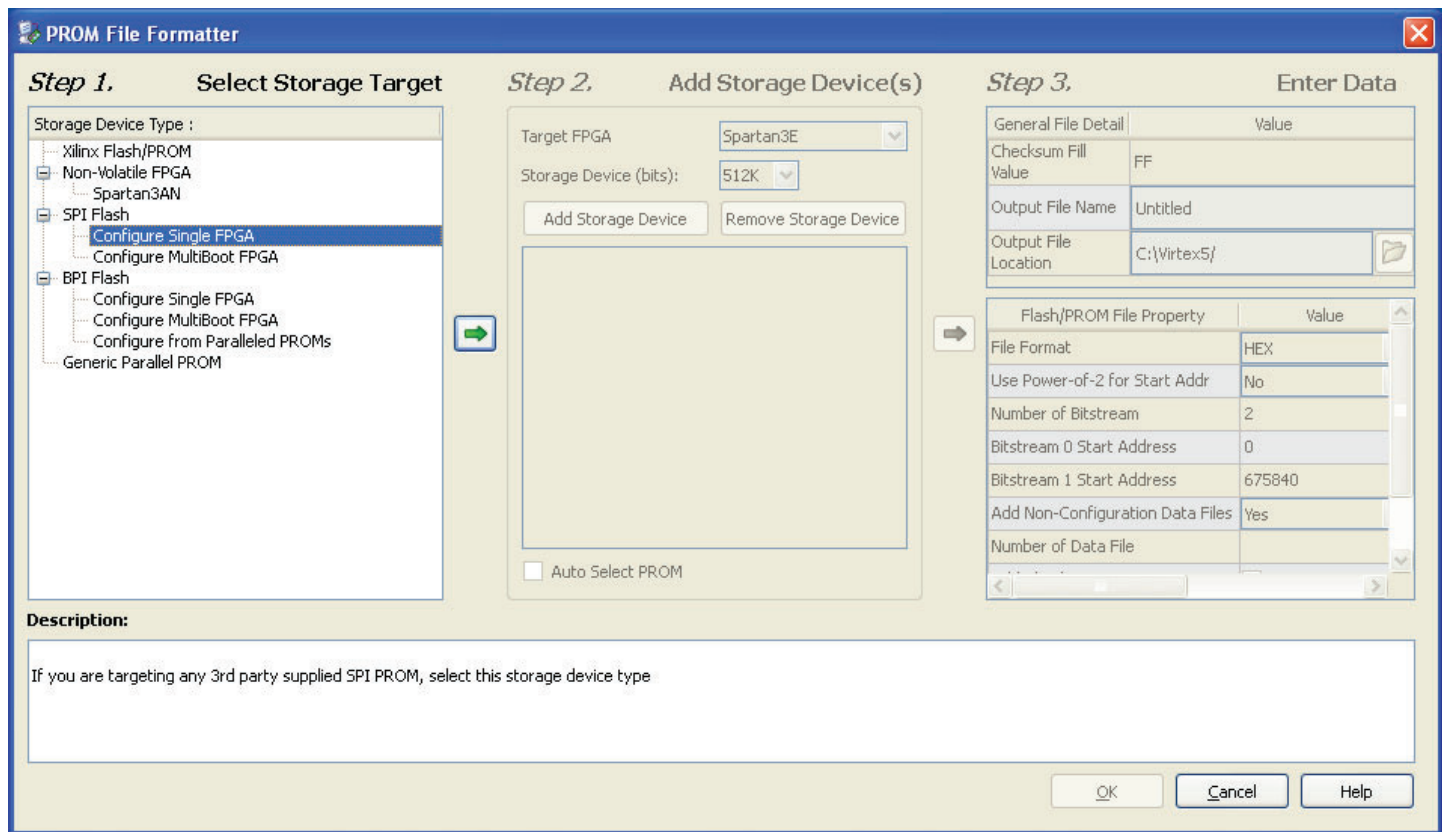


Figure 6: Choose to Prepare a PROM File

Step 3: Specify the Select Storage Target Option

The third step of the process is to specify the targeted storage type (see Figure 7). Choose to target the **SPI Flash** → **Configure Single FPGA** type and then click the green arrow to the right of the left-hand window to proceed to step 4.



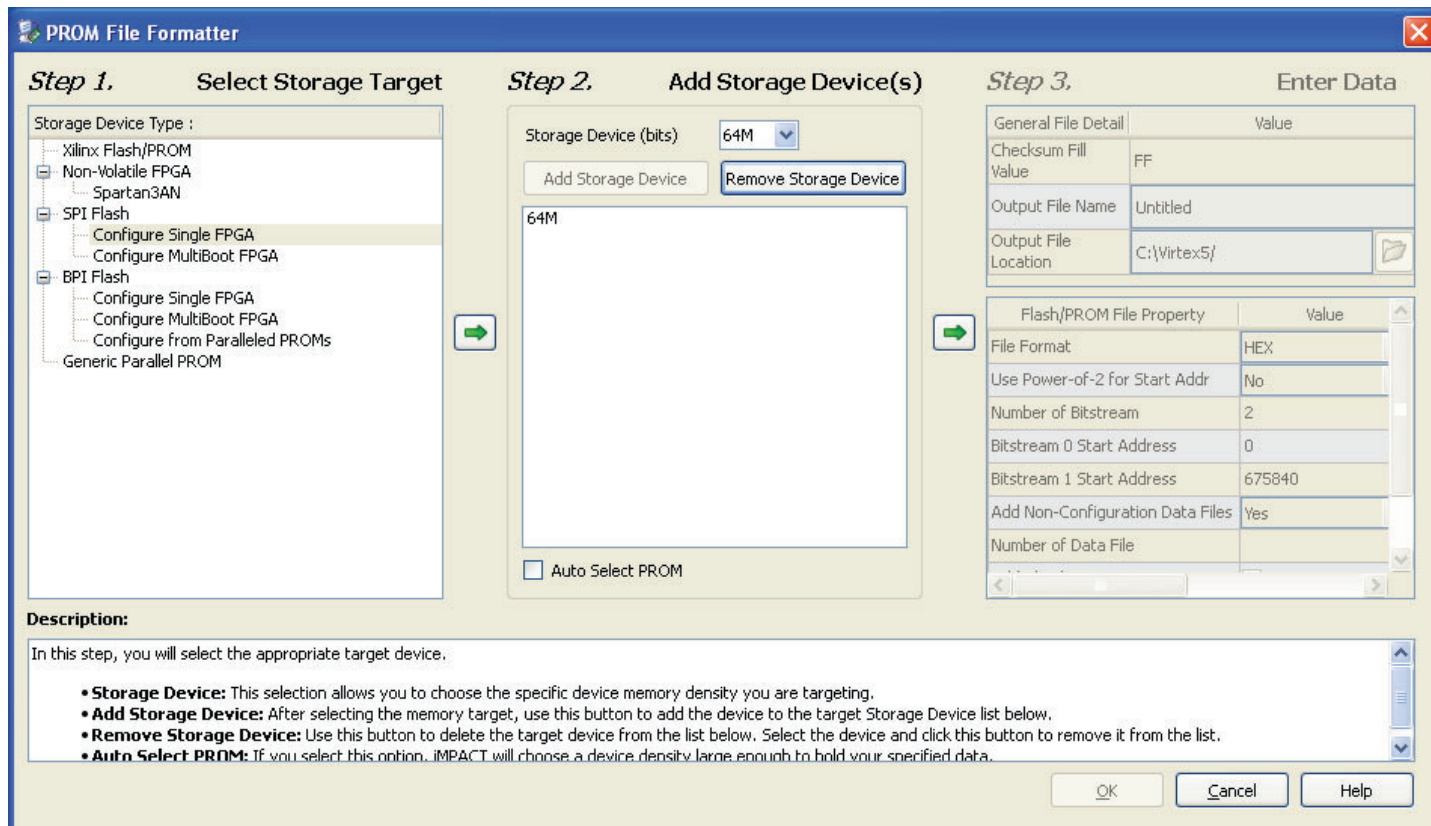
X951_07_072410

Figure 7: Specify the Select Storage Target Options

Step 4: Specify an SPI PROM Density

The fourth step specifies the SPI flash density. Select the **Storage Device (bits)** option, and for this example select the SPI PROM density **64M(bits)**. Next, click **Add Storage Device**.

Figure 8 displays the proper settings. Click the green arrow to the right of the center window to proceed to step 5.

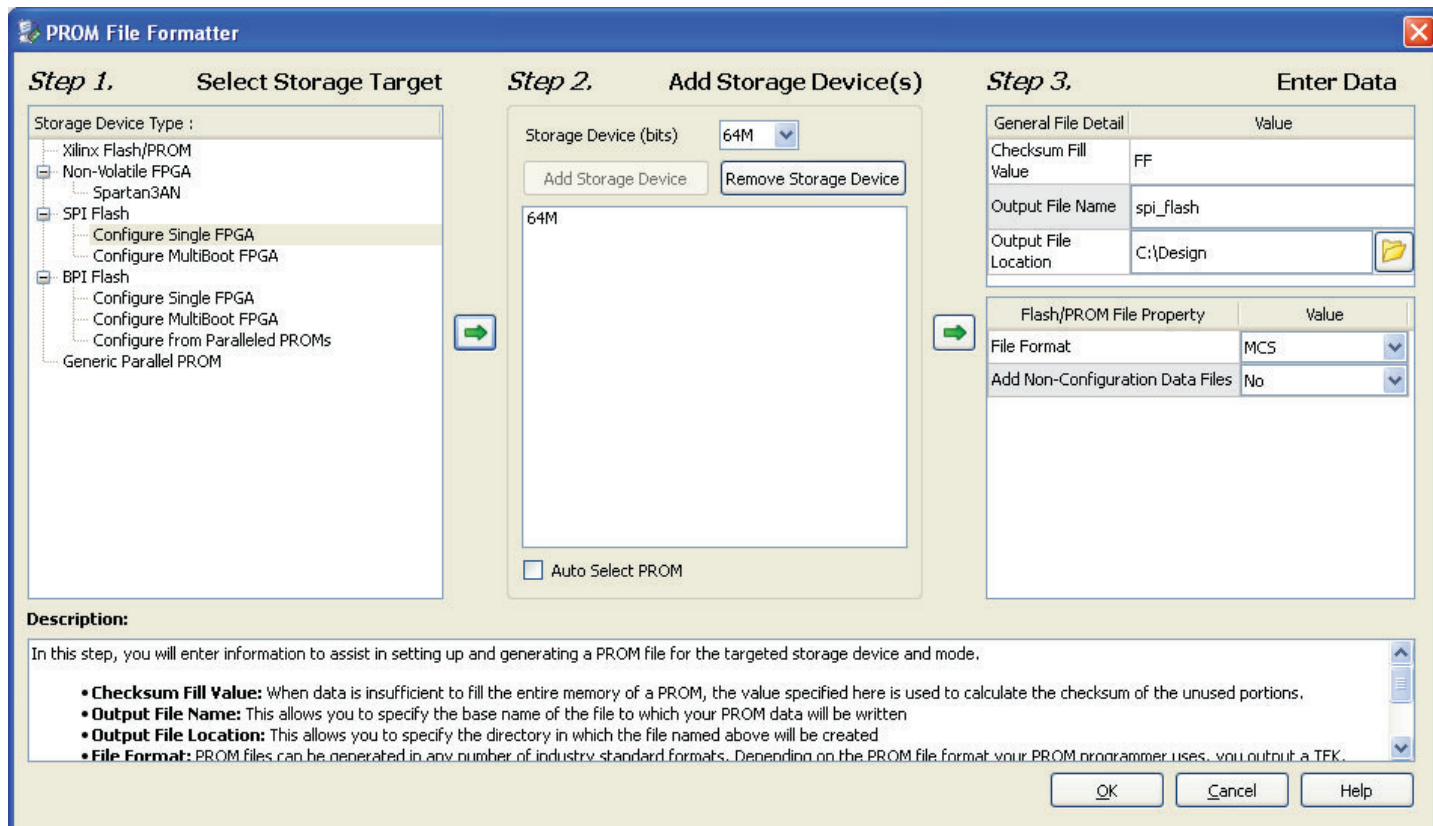


X951_08_072410

Figure 8: Specify an SPI PROM Density

Step 5: Specify Device Format Properties

The fifth step of the process specifies the SPI device file format properties. Click **Output File Name** and **Output File Location** to select a desired name and location where the new PROM file will be generated. The other options remain as default for the example shown in Figure 9. Click **OK** to proceed to step 6.



X951_09_072410

Figure 9: Specify Device Format Properties

Step 6: Automated Notification to Add a Device File to the SPI PROM File

After the iMPACT project wizard is finished, the iMPACT SPI PROM generation project is set to generate a specific PROM file with specific parameters. At this stage, the PROM file memory image is empty. The sixth step is to add an FPGA bitstream to the PROM file memory image. This step begins immediately after completion of the iMPACT project wizard with an automatic notification that the next step is to add a device file to the SPI PROM memory image. Click **OK** in the Add Device notification dialog box (Figure 10) to proceed to step 7.



X951_10_072410

Figure 10: Add Device Notification Dialog Box

Step 7: Select the FPGA Bitstream File to Add to the SPI PROM Memory Image

After the Add Device notification, iMPACT automatically opens a file browser to select the FPGA bitstream (.bit) file to add to the SPI PROM memory image (Figure 11). Select the FPGA bitstream file to be written to the SPI PROM. Click **Open** in the browser to add the selected FPGA bitstream to the SPI PROM memory image. For this example, click **NO** when asked to add another device file to Revision 0, and then click **OK** to continue.

Note: This demonstration flow is for a single FPGA target. When targeting FPGA daisy-chains, the user would continue to add device files for each FPGA target in the chain.

This action completes the automated iMPACT process for preparing an SPI PROM file to be generated. Proceed to step 8 to generate the SPI PROM file.

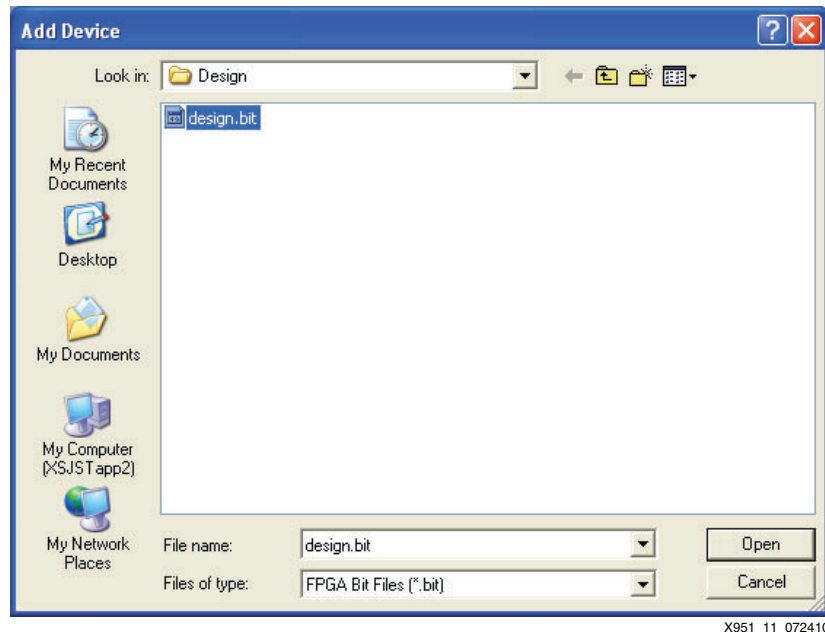


Figure 11: Add Device File Browser

Step 8: iMPACT Generate File Operation

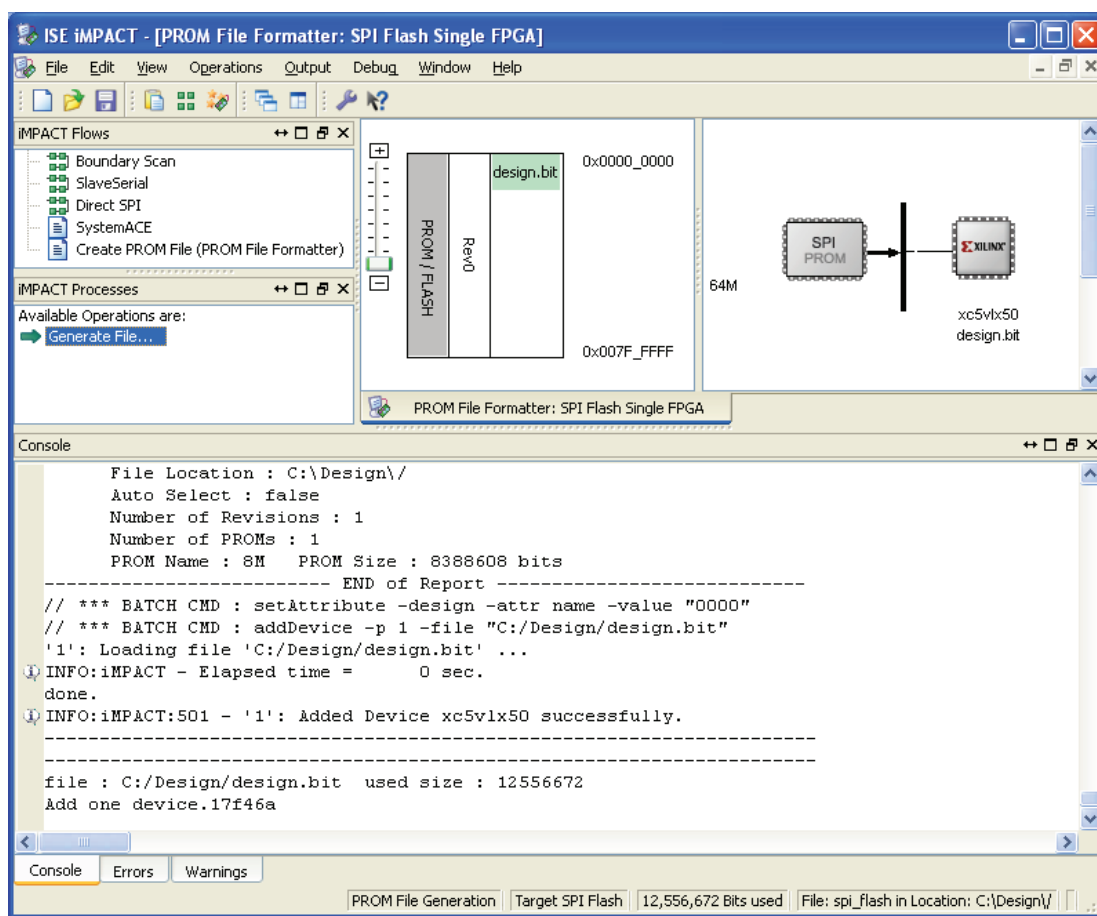
The eighth and final step is to generate the PROM file. In the iMPACT Processes window, invoke **Generate File** (Figure 12, page 19). Once invoked, the Generate File operation causes iMPACT to generate the specified SPI PROM file.

iMPACT reports a “PROM File Generation Succeeded” message after successful generation of the SPI PROM file.

After the Generate File operation has completed, the generated `spi_flash.mcs` file is available in the specified location. The `spi_flash.mcs` file can be used in any of the supported programming solutions to program the SPI PROM.

Save the iMPACT SPI PROM generation project for quick regenerating of the SPI PROM file whenever the FPGA bitstream design is revised. To regenerate an SPI PROM file, reopen the saved iMPACT project, and invoke the Generate File operation. iMPACT generates a revised SPI PROM file from the new version of the FPGA bitstream file, assuming the revised bitstream file is located in the same location as the original bitstream file.

If a project is not loaded when using the iMPACT GUI interface, a user is guided through the wizard steps each time to create a new SPI-formatted PROM file.



X951_12_072410

Figure 12: Generate File Menu

Using the ISE iMPACT Software to In-System Program SPI PROMs

In prototype applications, the ISE iMPACT 8.2i (or later) software can be used to in-system program select SPI serial flash devices with a memory image from a given SPI PROM file (see [Preparing an SPI PROM File, page 12](#) for instructions on the generation of an SPI PROM file).

[Table 7](#) lists the recommended SPI serial flash that can be programmed directly by iMPACT.

Table 7: SPI Serial Flash Programming Capability with iMPACT

SPI Serial Flash Vendor	Family ⁽¹⁾
Numonyx	M25P, M25PE, M45PE
Atmel	AT45DB

Notes:

1. Refer to the software version iMPACT Help Manual for complete details on the SPI serial flash supported for each release.

The iMPACT software can program select SPI PROM using a simple process. A prescribed sequence of dialog boxes (or wizard) acts as a guide through most of the iMPACT programming process.

The following section demonstrates the iMPACT 11.4 software process for in-system programming a M25P64 (64 Mb) Numonyx SPI PROM. The demonstrated process takes the `spi_flash.mcs` SPI PROM file (generated in section [Preparing an SPI PROM File, page 12](#)) as input, erases the SPI PROM, programs the PROM file contents into the SPI serial flash device, and verifies the SPI PROM contents against the given SPI PROM file contents.

Step 1: Create a New Project for Direct In-System Programming

After launching the iMPACT software, the iMPACT Project dialog box is displayed (Figure 5, page 14). Choose the “create a new project (.ipf)” option. Optionally, specify a project location via the **Browse...** button. Then, click **OK** button to continue to step 2 in the process.

Step 2: Configure Devices Using the Direct SPI Configuration Mode

The second step begins with the iMPACT project wizard. The first dialog box of the wizard displays the available kinds of projects that can be created (Figure 13). Choose the **Configure devices** option. Then, select the **using Direct SPI Configuration mode** item from the associated drop-down list box. Click **OK** to complete the new project setup process. At the completion of this process, iMPACT is set into a mode for in-system programming SPI serial flash memories using a direct cable connection to the SPI bus.

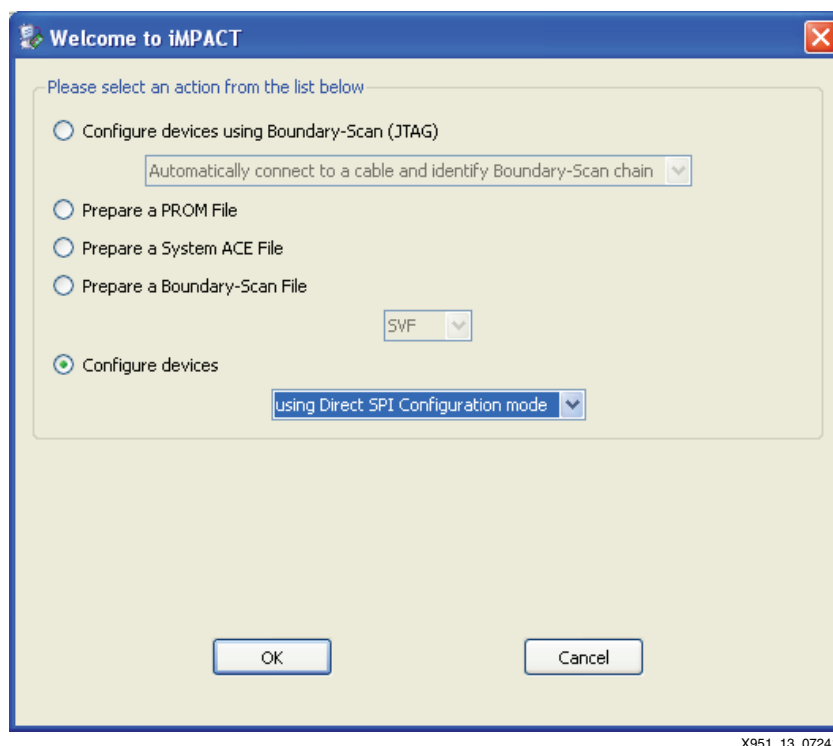
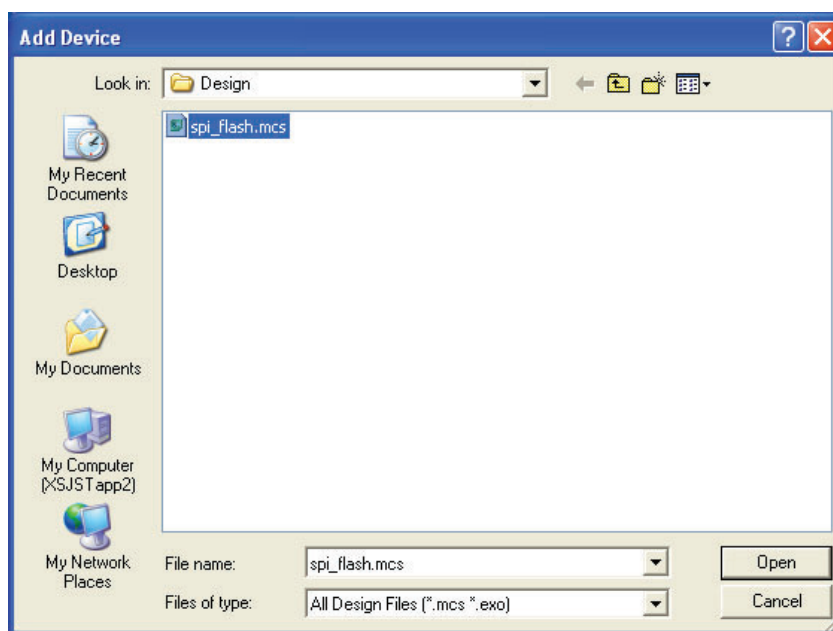


Figure 13: Configure Devices Using the Direct SPI Configuration Mode

Step 3: Add an SPI PROM File

After finishing the new project wizard, right-click in the Direct SPI window and select Add SPI Device. iMPACT displays a file browser window to select an SPI PROM file for programming into the SPI serial flash device (Figure 14). Choose the `spi_flash.mcs` file, and click **Open**.

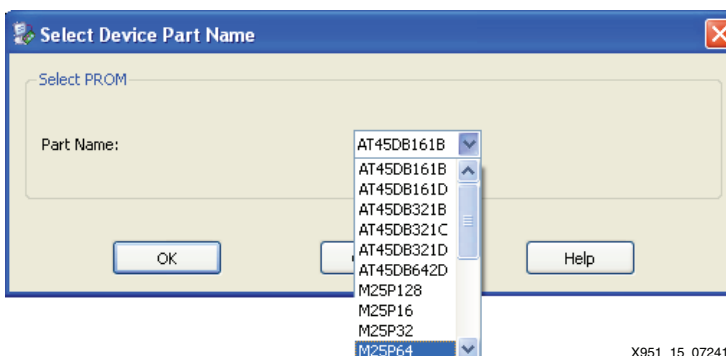


X951_14_072410

Figure 14: Add an SPI PROM File

Step 4: Select Numonyx M25P64 Device Part Number

After selecting the SPI PROM file to load, iMPACT displays the Select Device Part Name dialog box (Figure 15). The fourth step of the process requires the target type of SPI PROM to be specified in this dialog box. Select the Numonyx M25P64 part number for the target SPI PROM type used in this demonstration. Click OK to complete the SPI PROM programming setup.



X951_15_072410

Figure 15: Select Device Part Name Dialog Box

Step 5: (iMPACT 10.1 or later) Set Device Programming Properties

iMPACT presents the Device Programming Properties dialog box (Figure 16) which sets the additional functions to be performed each time the **Operation** → **Program** menu item is invoked. It is recommended that both the **Verify** and **Erase Before Programming** options be checked in the Device Programming Properties dialog box. Click **OK** to save the programming properties.

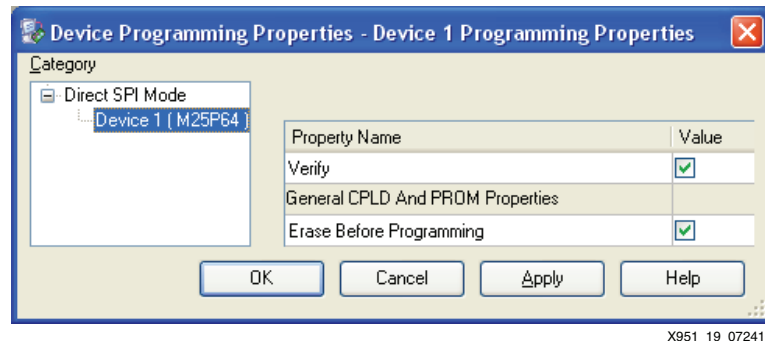


Figure 16: (iMPACT 10.1 or Later) Specify Device Programming Properties

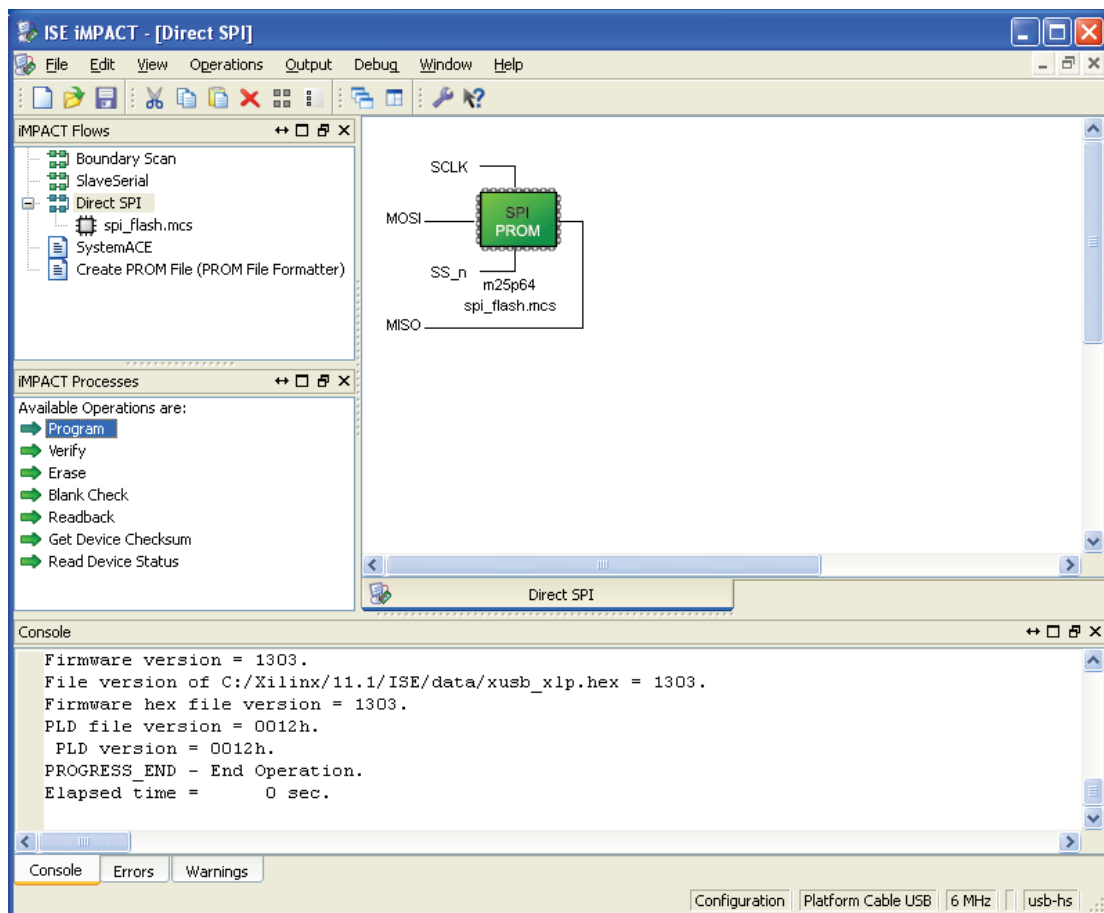
Step 6: Specify the Hardware Requirements for In-System Programming

The sixth step of the programming process specifies the hardware requirements for in-system programming of the SPI PROM:

- **Proper Xilinx cable connection:** The Xilinx cable must be properly connected to the computer and to the SPI bus of the target SPI PROM (see [Figure 2, page 5](#) for hardware connections from the Xilinx cable to the SPI bus of the target SPI PROM).
- **Cable power:** If using the Xilinx Parallel Cable IV cable, then power must be applied to the cable.
- **Target system Power:** Power must also be supplied to the target system containing the SPI PROM.
- **Isolate target FPGA signals during the SPI programming process:** The target FPGA (and any other potential SPI PROM master device on the SPI bus) must be put into a mode to keep its SPI pins in a high-impedance state. Methods to put the FPGA SPI pins in a high-impedance state are described in [Hardware and Connections for SPI Programming, page 11](#). One common method requires the FPGA PROG_B pin to be held Low. The high-impedance condition of the FPGA SPI pins must be maintained throughout the remainder of the SPI PROM programming process in order to avoid contention with the in-system programming cable.

Step 7: Invoke the iMPACT Program Operation

The sixth step of the process programs the target SPI PROM with the selected SPI PROM file contents. Ensure the SPI PROM icon in the iMPACT window is selected by left-clicking on the SPI PROM icon (the SPI PROM icon is highlighted in green when selected). Double-click the **Program** operation from the iMPACT Processes window to begin programming (Figure 17).

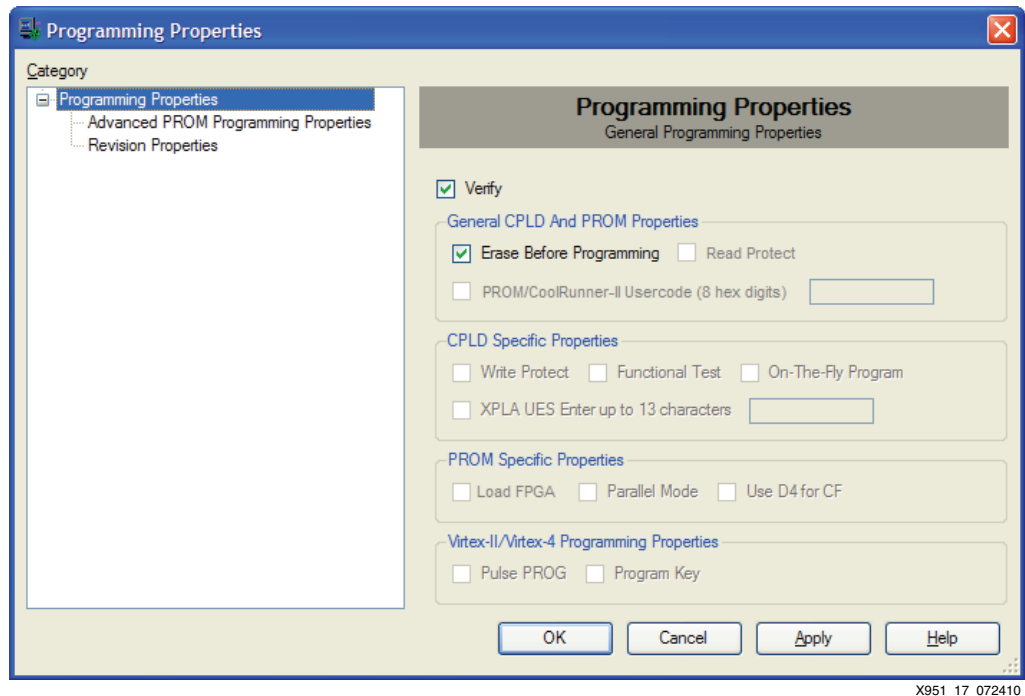


X951_16_072410

Figure 17: Program Operation

Step 8: (iMPACT 9.2.04i or earlier) Select iMPACT Programming Properties

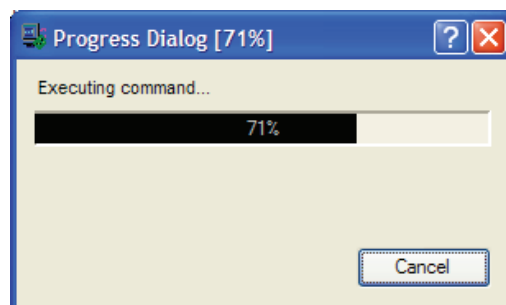
In response to the invocation of the **Program** operation, iMPACT presents the Programming Properties dialog box (Figure 18). The seventh step of the process ensures the selection of proper programming properties. Ensure that both the **Verify** and the **Erase Before Programming** options are checked, ensuring proper programming of the SPI PROM. Click **OK** to begin the erase, program, and verify operations.



X951_17_072410

Figure 18: Programming Properties Dialog Box

At the start of the programming operation, iMPACT automatically connects to the cable attached to the computer. Then, iMPACT displays a Progress Dialog box as it progresses through the in-system erase, program, and verify operations (Figure 19). Depending on the size of the SPI PROM, size of the SPI PROM file image, and speed of the cable configuration, the programming operation can take anywhere from a few seconds to a few minutes to complete.



X951_18_072410

Figure 19: Progress Dialog Box

At the end of a successful Program operation, iMPACT reports a “Program Succeeded” message.

Save the iMPACT Direct SPI Configuration Mode project for quickly reprogramming the SPI PROM whenever the SPI PROM file is revised. To reprogram the SPI PROM, reopen the saved

iMPACT project, and invoke the **Program** operation, ensure the selection of the **Erase** and **Verify Programming Properties**, and click **OK**. iMPACT reprograms the SPI PROM, assuming the revised SPI PROM file is located in the same location as the original SPI PROM file.

Conclusion

The addition of the SPI interface in new Xilinx FPGA families allows designers to use multi-vendor small footprint SPI PROM for configuration. Systems with an existing onboard SPI PROM can leverage the single memory source for storing configuration data in addition to the existing user data.

References

Device

Xilinx documents

1. [Virtex-5 Documents](#).
2. [DS202](#), *Virtex-5 FPGA Family Data Sheet*.
3. [UG191](#), *Virtex-5 Configuration User Guide*.
4. [DS312](#), *The Spartan-3E FPGA Family Data Sheet*.
5. [UG332](#), *Spartan-3 Generation Configuration User Guide*.

Software

The Xilinx PROMGen and iMPACT software are available with the main Xilinx ISE Foundation software or with the downloadable Xilinx ISE WebPACK™ software packages.

ISE Foundation software:

http://www.xilinx.com/ise/logic_design_prod/foundation.htm

ISE WebPACK software:

http://www.xilinx.com/ise/logic_design_prod/webpack.htm

The Xilinx ISE software manuals are available at:

http://www.xilinx.com/support/software_manuals.htm

Hardware

Information regarding the Xilinx cables are found on the Xilinx Configuration Solutions website:

http://www.xilinx.com/products/design_resources/config_sol/

See the ISE iMPACT 8.2i (or later) software manuals for supported Xilinx cables.

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
12/13/06	1.0	Initial Xilinx release.
10/03/07	1.1	- Updated document template. - Added a cautionary note regarding CCLK layout to Figure 2 and Figure 3.
11/20/07	1.1.1	Updated URLs.
01/29/09	1.2	- Software flow updated for iMPACT 10.1. - STMicrosystems updated to Numonyx. Table 3 updated for DONE pull-up value reference.
09/23/10	1.3	- Software flow updated for iMPACT 11.4 (last supported release for direct in-system SPI programming). Replaced all changed screens with updated versions (Figure 5 through Figure 19). Table 2: Updated with all current Virtex-5 devices. Figure 2 and Figure 3: DONE pull-up changed from 300Ω to 330Ω Table 3: Specific pull-up voltages removed, replaced by reference to device data sheets. Changed reference to reclaiming configuration pins as user I/O to state that D_IN and CCLK are dedicated configuration pins and cannot be reclaimed. - Throughout, changed filenames <code>spi_prom.mcs</code> and <code>bitfile.bit</code> to <code>spi_flash.mcs</code> and <code>design.bit</code> respectively.

Notice of Disclaimer

Xilinx is disclosing this Application Note to you “AS-IS” with no warranty of any kind. This Application Note is one possible implementation of this feature, application, or standard, and is subject to change without further notice from Xilinx. You are responsible for obtaining any rights you may require in connection with your use or implementation of this Application Note. XILINX MAKES NO REPRESENTATIONS OR WARRANTIES, WHETHER EXPRESS OR IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHANTABILITY, NONINFRINGEMENT, OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL XILINX BE LIABLE FOR ANY LOSS OF DATA, LOST PROFITS, OR FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, OR INDIRECT DAMAGES ARISING FROM YOUR USE OF THIS APPLICATION NOTE.