Platform Flash XL Configuration and Storage Device

User Guide

UG438 (v3.0) August 5, 2015

XILINX®

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
03/28/2008	1.0	Initial Xilinx release.
05/14/2008	1.1	Added Figure 2-1, page 16 and Figure 3-1, page 32 footnotes. Added “Master SelectMAP Configuration Mode” in Chapter 3. Changed Chapter 3 title to “Alternate Configuration Modes.” Added updates highlighting recommended configuration mode (Slave SelectMAP).
12/10/2008	1.2	Chapter 1: Added cross reference to *Virtex-5 FPGA Configuration User Guide*. Removed NC and DU signals from Table 1-2. Updated description of I/O power supply to include support of 2.5V. Chapter 2: Updated chapter title and “Slave SelectMAP Configuration Mode.” Updated Figure 2-1, including note 7, and added notes 10 and 11. Added Table 2-1 and Figure 2-2. Chapter 3: Updated “Master BPI-Up Configuration Mode.” Added notes 9 and 10 to and updated Figure 3-1. Updated table headings and added $\overline{L}$ (latch enable) signal to Table 3-1. Updated “FPGA BPI-Up Configuration from Platform Flash XL.” Updated Figure 3-3 title. Chapter 5: Deleted note from “Preparing a Programming File.” Updated iMPACT software version. Added “Step 3: Select the XCF128X Device.” Chapter 6: Updated Figure 6-1. Updated note in “Minimum Requirements for Indirect In-System Programming.” Updated iMPACT software version. Interchanged “Step 7: Select iMPACT Programming Properties” and “Step 6: Invoke the iMPACT Program Operation.” Updated “Expectations.” Added Chapter 4, “Calculating Configuration Time”and Chapter 8, “FPGA User Design Recommendations.”

Date	Version	Revision
12/14/2009	2.0	Added support for Virtex-6 FPGAs. Updated iMPACT software version. Updated "Overview." Added Table 1-1, which contains configuration flash information for Virtex-5 and Virtex-6 devices. Updated "Family Description," including Figure 1-1 and Table 1-2. Updated "Slave SelectMAP Configuration Mode" and "Slave SelectMAP Configuration from Platform Flash XL." Updated Figure 2-1 and its associated notes. Added Figure 2-2 and its associated notes. Updated Table 2-1. Updated Figure 2-3 and its associated notes. Updated "Master BPI-Up Configuration Mode," Figure 3-1 and its associated notes, Table 3-1, "FPGA BPI-Up Configuration from Platform Flash XL," Figure 3-2 and its associated notes, "Master SelectMAP Configuration Mode," and notes relating to Figure 3-3. Updated Chapter 4, "Calculating Configuration Time." Updated "Using the iMPACT Graphical Software." Updated Figure 6-1. Deleted Frequency column from Table 6-1. Updated "Minimum Requirements for Indirect In-System Programming." Updated "Xilinx Cable Connections" and "Effect of Indirect Programming on the Rest of the System." Removed Table 6-2; information is available in DS593, *Platform Cable USB II Data Sheet* Updated steps and GUI screen shots in "iMPACT Programming Flow." Added Chapter 7, "System Considerations." Added "Re-Using Configuration Pins for Other Purposes." Updated "FPGA Designs Not Accessing Platform Flash XL after Configuration" and "FPGA Designs Accessing Platform Flash XL after Configuration." Updated Table 8-2.
08/05/2015	3.0	This product is obsolete/discontinued per XCN15008. Updated Notice of Disclaimer.

Table of Contents

XILINX®

Preface

About This Guide

Summary

This guide describes the Platform Flash XL feature set, demonstrates the common configuration mode setups supported, and provides the software flows necessary to generate the programming files and indirectly program the device in-system. More information on Platform Flash XL is available online in DS617, *Platform Flash XL High-Density Configuration and Storage Device*.

Guide Contents

This manual contains the following chapters:

- Chapter 1, "Overview," provides a brief description of key features of Platform Flash XL.
- Chapter 2, "High-Speed Configuration," describes the most popular setup for high-performance applications.
- Chapter 3, "Alternate Configuration Modes," describes the BPI-Up configuration mode setup and the Master SelectMAP configuration mode considerations.
- Chapter 4, "Calculating Configuration Time," describes the considerations and parameters used to determine the minimum configuration time.
- Chapter 5, "Platform Flash XL File Generation," demonstrates the steps required to prepare programming files in popular PROM formats.
- Chapter 6, "Programming Platform Flash XL," describes the flow used to program the device indirectly using iMPACT software.
- Chapter 7, "System Considerations," provides V_{DDQ} power budget estimates from a real example test board.
- Chapter 8, "FPGA User Design Recommendations," describes recommendations for FPGA designs that use Platform Flash XL for storage after configuration.

Additional Resources

General

To find additional documentation, see the Xilinx® website at:

http://www.xilinx.com/literature.

To search the Answer Database of silicon, software, and IP questions and answers, or to create a technical support WebCase, see the Xilinx website at:

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

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
- 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 software manuals for supported Xilinx cables.

Additional Documentation

- DS152, *Virtex-6 FPGA Data Sheet: DC and Switching Characteristics*

  This data sheet contains the DC and switching characteristic specifications for the Virtex-6 family.
- UG360, *Virtex-6 FPGA Configuration User Guide*

  This all-encompassing configuration guide includes chapters on configuration interfaces (serial and SelectMAP), bitstream encryption, Boundary-Scan and JTAG configuration, reconfiguration techniques, and readback through the SelectMAP and JTAG interfaces.
- UG191, *Virtex-5 FPGA Configuration User Guide*

  This all-encompassing configuration guide includes chapters on configuration interfaces (serial and SelectMAP), bitstream encryption, Boundary-Scan and JTAG configuration, reconfiguration techniques, and readback through the SelectMAP and JTAG interfaces.
- DS202, *Virtex-5 FPGA Data Sheet: DC and Switching Characteristics*

  This data sheet contains the DC and switching characteristic specifications for the Virtex-5 family.
- DS617, *Platform Flash XL High-Density Configuration and Storage Device*

  This data sheet describes the Platform Flash XL configuration and storage device optimized for the Virtex-5 FPGA.

Conventions

This document uses the following conventions. An example illustrates each convention.

Typographical

The following typographical conventions are used in this document:

Convention	Meaning or Use	Example
`Courier font`	Messages, prompts, and program files that the system displays	`speed grade: - 100`
`Courier bold`	Literal commands that you enter in a syntactical statement	**`ngdbuild`** *design_name*
Helvetica bold	Commands that you select from a menu	**File → Open**
	Keyboard shortcuts	**Ctrl+C**
Italic font	Variables in a syntax statement for which you must supply values	**`ngdbuild`** *design_name*
	References to other manuals	See the *Development System Reference Guide* for more information.
	Emphasis in text	If a wire is drawn so that it overlaps the pin of a symbol, the two nets are *not* connected.
Square brackets [ ]	An optional entry or parameter. However, in bus specifications, such as **`bus[7:0]`**, they are required.	**`ngdbuild`** [*option_name*] *design_name*
Braces { }	A list of items from which you must choose one or more	**`lowpwr =`**{**`on`**|**`off`**}
Vertical bar |	Separates items in a list of choices	**`lowpwr =`**{**`on`**|**`off`**}

Vertical ellipsis
.
.
.
	Repetitive material that has been omitted	`IOB #1: Name = QOUT'`
		`IOB #2: Name = CLKIN'`
		.
		.
		.

Horizontal ellipsis ...	Repetitive material that has been omitted	**`allow block`** *block_name loc1 loc2 ... locn;*

Online Document

The following conventions are used in this document:

Convention	Meaning or Use	Example
Blue text	Cross-reference link to a location in the current document	See the section "Additional Resources" for details. Refer to "Title Formats" in Chapter 1 for details.
Red text	Cross-reference link to a location in another document	See Figure 2-5 in the *Virtex-5 FPGA User Guide.*
Blue, underlined text	Hyperlink to a website (URL)	Go to http://www.xilinx.com for the latest speed specifications.

XILINX®

Chapter 1

Overview

This guide supplements DS617, *Platform Flash XL High-Density Storage and Configuration Device*, giving specific user information on device features and usage modes.

The Platform Flash XL (XCF128X) configuration and storage device is optimized for use with Virtex®-5 and Virtex-6 FPGAs (see Table 1-1). The Platform Flash XL does not support older Virtex families, Spartan® families, or AES encrypted bitstreams. See UG161, *Platform Flash PROM User Guide,* for alternative Platform Flash PROMs.

For bitstream length (bits), see UG360, *Virtex-6 FPGA Configuration User Guide* or UG191, *Virtex-5 FPGA Configuration User Guide*.

Single-chip Platform Flash XL provides a high-density non-volatile storage with the industry's highest performing configuration, ease-of-use and flexibility within a small footprint package.

With in-system configuration speeds up to 800 Mb/s, Platform Flash XL is ideally suited for high-performance PCI Express® technology applications used in optical and enterprise networking, WiMAX digital front-end and baseband processing, video broadcast equipment, and medical ultrasound and scanners.

Table 1-1: **Configuration Flash for Virtex-5 and Virtex-6 FPGAs**

FPGA	Configuration Flash
Virtex-6 FPGAs	
XC6VLX75T	XCF128X
XC6VLX130T	XCF128X
XC6VLX195T	XCF128X
XC6VLX240T	XCF128X
XC6VHX250T	XCF128X
XC6VHX255T	XCF128X
XC6VHX380T	XCF128X
XC6VHX565T	See UG360, *Virtex-6 FPGA Configuration User Guide* for BPI flash
XC6VSX315T	XCF128X
XC6VLX365T	XCF128X
XC6VSX475T	See UG360, *Virtex-6 FPGA Configuration User Guide* for BPI flash
XC6VLX550T	
XC6VLX760	

Table 1-1: **Configuration Flash for Virtex-5 and Virtex-6 FPGAs *(Cont'd)***

FPGA	Configuration Flash
Virtex-5 FPGAs	
XC5VLX30	XCF128X
XC5VLX50	XCF128X
XC5VLX85	XCF128X
XC5VLX110	XCF128X
XC5VLX155	XCF128X
XC5VLX220	XCF128X
XC5VLX330	XCF128X
XC5VLX20T	XCF128X
XC5VLX30T	XCF128X
XC5VLX50T	XCF128X
XC5VLX85T	XCF128X
XC5VLX110T	XCF128X
XC5VLX155T	XCF128X
XC5VLX220T	XCF128X
XC5VLX330T	XCF128X
XC5VSX35T	XCF128X
XC5VSX50T	XCF128X
XC5VSX95T	XCF128X
XC5VSX240T	XCF128X
XC5VTX150T	XCF128X
XC5VTX240T	XCF128X
XC5VFX30T	XCF128X
XC5VFX70T	XCF128X
XC5VFX100T	XCF128X
XC5VFX130T	XCF128X
XC5VFX200T	XCF128X

Family Description

To achieve high-performance configuration, Platform Flash XL supports the SelectMAP configuration port with power-on synchronization and an immediate bitstream burst capability. A wide, 16-bit data bus delivers the bitstream synchronous to the FPGA configuration clock (CCLK) at up to 50 MHz in the Slave SelectMAP configuration mode without wait states. The device provides a READY_WAIT signal that synchronizes the start of the FPGA configuration process, both improving system reliability and simplifying

board design. When the device signals a ready status after power-on, the device can immediately burst the FPGA design bitstream (`.bit`) file to the FPGA. The configuration performance of Platform Flash XL is ideal for PCI Express end-points and other high-performance applications (see Chapter 2, "High-Speed Configuration"). Platform Flash XL is optimized for the FPGA Slave SelectMAP configuration mode, but it also supports Master SelectMAP and Master BPI-Up modes (see Chapter 3, "Alternate Configuration Modes"). When the highest-performance configuration is required, the Platform Flash XL must be used with the FPGA in Slave SelectMAP mode. The Master SelectMAP or Master BPI can be considered in cases where less configuration performance is needed and the FPGA Fallback feature is needed.

In addition to high-performance configuration, Platform Flash XL provides a 128 Mb (8M x 16 bits) nonvolatile single-chip flash memory configuration solution with a small footprint (FT64) and advanced system-level capabilities. A standard NOR flash interface (Figure 1-1) and support for common flash interface (CFI) queries provide industry-standard access to the device memory space. Platform Flash XL's 128 Mb capacity can hold one or more FPGA bitstreams, with any unused memory available for general-purpose data or embedded processor code storage. For bitstream requirements, see UG191, *Virtex-5 FPGA Configuration User Guide* or UG360, *Virtex-6 FPGA Configuration User Guide*.

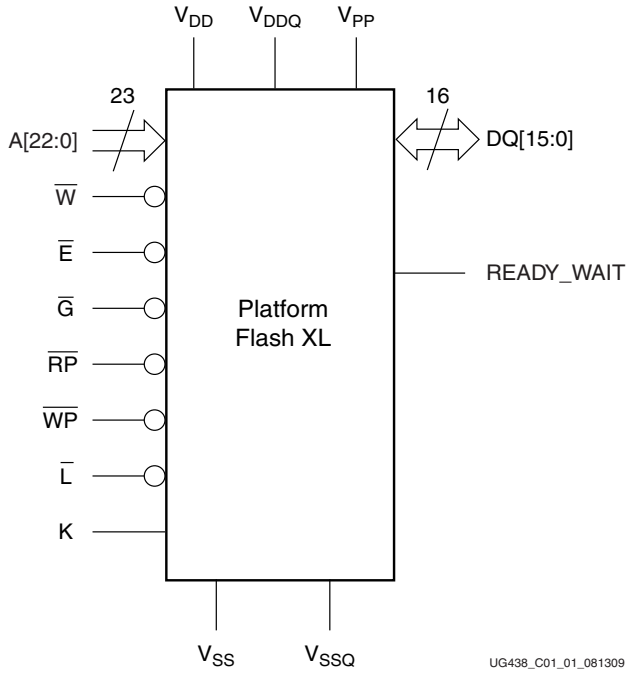


Figure 1-1: **Platform Flash XL Logic Diagram**

The Platform Flash XL signals and functions are described in Table 1-2. See Chapter 2, "High-Speed Configuration" and Chapter 3, "Alternate Configuration Modes" for recommended connections to the FPGA.

Table 1-2: **Platform Flash XL Signal Descriptions**

Signal Name	Function	Direction
A[22:0]	Address inputs	Inputs
DQ[15:0]	Data input/outputs, command inputs	I/O
$\overline{E}$	Chip enable	Input

Table 1-2: **Platform Flash XL Signal Descriptions *(Cont'd)***

Signal Name	Function	Direction
$\overline{G}$	Output enable	Input
$\overline{W}$	Write enable	Input
$\overline{RP}$	Reset	Input
$\overline{WP}$	Write protect	Input
K	Clock	Input
$\overline{L}$	Latch enable	Input
READY_WAIT	Ready wait	I/O
V_{DD}	Supply voltage	–
V_{DDQ}	Supply voltage for input/output buffers	–
V_{PP}	Optional supply voltage for fast program and erase	–
V_{SS}	Ground	–
V_{SSQ}	Ground input/output supply	–

For FPGA designs that require data storage, the Platform Flash XL has a multiple-bank architecture—an array of 131 individually erasable blocks that are divided into sixteen 8 Mb banks. Fifteen main banks contain uniform blocks of 64 Kwords, and one parameter bank contains seven main blocks of 64 Kwords, plus four parameter blocks of 16 Kwords (the parameter blocks are located at the top of the memory address space in Platform Flash XL). The device has a 23-bit address bus providing random read access to each 16-bit word. See DS617, *Platform Flash XL High-Density Configuration and Storage Device* for memory map information and application programming instructions.

The device is in-system programmable with a 1.8V core (V_{DD}) power supply (electronically erasable at the block level and programmable on a word-by-word basis). A separate I/O (V_{DDQ}) power supply enables I/O operation at 2.5V or 3.3V.

The ISE® iMPACT software supports indirect, in-system programming of Platform Flash XL via the IEEE Standard Test Access Port and Boundary-Scan Architecture (IEEE Std 1149.1) port (i.e., the JTAG port) on the FPGA (see Chapter 6, "Programming Platform Flash XL").

For FPGA configuration, the Platform Flash XL powers-on directly into synchronous read mode. After latching a starting read address, the synchronous read mode enables the flash to output its array data as a continuous stream of 16-bit words based on an internally incrementing address counter. In each clock cycle, a 16-bit word is delivered. The Platform Flash XL's interface can latch the starting address at the beginning of the FPGA's SelectMAP or BPI mode configuration sequence. The Platform Flash XL's synchronous read function is directly compatible with the FPGA's SelectMAP or BPI mode configuration interface.

XILINX®

Chapter 2

High-Speed Configuration

Slave SelectMAP Configuration Mode

Applications including PCI Express® systems require fast FPGA configuration. Platform Flash XL provides the highest performance and power sequencing immunity in the Slave SelectMAP mode. Platform Flash XL achieves maximum configuration performance when a precise external clock source drives the FPGA Slave SelectMAP configuration mode up to the maximum 50 MHz burst read frequency limit (CCLK) of the device.

This chapter describes the configuration of an FPGA from Platform Flash XL in Slave SelectMAP configuration mode (M[2:0] = `110`). The first section of this chapter describes the required setup for Slave SelectMAP configuration and indirect in-system Platform Flash XL programming (see Figure 2-1 and Figure 2-2). Next, the FPGA configuration flow for this mode is demonstrated (see Figure 2-3). Refer to the SelectMAP Configuration Interface section in UG191, *Virtex-5 FPGA Configuration User Guide* or UG360, *Virtex-6 FPGA Configuration User Guide*, for additional information on the Slave SelectMAP mode.

The Platform Flash XL has these system-level requirements for configuring an FPGA in the FPGA's SelectMAP mode:

- An external clock source drives the synchronous bitstream transfer resulting in a precise FPGA configuration time.
- On board pull-up or pull-down resistors set the configuration device control pins for output read mode.
- On board pull-up or pull-down resistors set the configuration device burst read start address.
- The FPGA does not support the Fallback feature in the Slave SelectMAP mode. However, an external configuration manager (for example XAPP693, *A CPLD-Based Configuration and Revision Manager for Xilinx Platform Flash PROMs and FPGAs*) can be implemented to achieve the equivalent fallback function.

See Figure 2-1, page 16 or Figure 2-2, page 18 for example Slave SelectMAP configuration mode connections with the Virtex-5 FPGA or Virtex-6 FPGA, respectively. Refer to "Slave SelectMAP Configuration from Platform Flash XL," page 27 for details on the Slave SelectMAP configuration sequence.

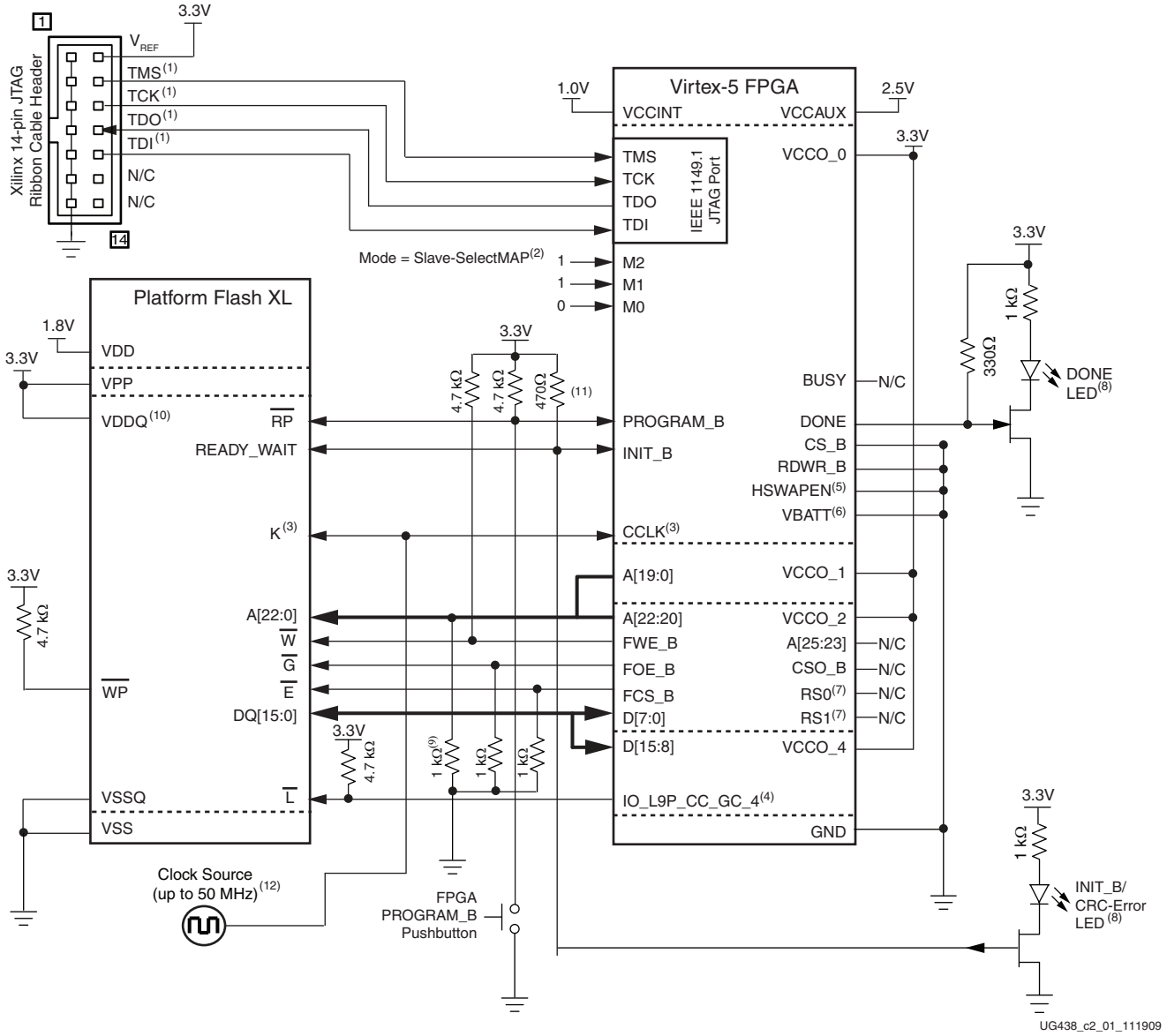


Figure 2-1: **Virtex-5 FPGA Slave SelectMAP Configuration Mode from Platform Flash XL**

Notes relating to Figure 2-1:

1. The JTAG connections are shown for a simple, single-device JTAG scan chain. When multiple devices are on the JTAG scan chain, use the proper IEEE Std 1149.1 daisy-chain technique to connect the JTAG signals. The TCK signal integrity is critical for JTAG operation. Route, terminate, and if necessary, buffer the TCK signal appropriately to ensure signal integrity for the devices in the JTAG scan chain.
2. The FPGA mode (M[2:0]) pins are shown set to Slave SelectMAP mode (`110`). The implementation of a board-level option that enables the user to change the FPGA mode pins to JTAG mode (`101`) is recommended to enable JTAG-based debug capability for the FPGA during design prototyping without interference from the Slave SelectMAP configuration activities.

3. CCLK signal integrity is critical. Route and terminate the CCLK signal appropriately to ensure good signal integrity at the XCF128X $\overline{K}$ pin and at the FPGA CCLK pin.
4. The iMPACT software requires the Virtex-5 FPGA's IO_L9P_CC_GC_4 connection to the device's $\overline{L}$ pin to support JTAG-based indirect programming.
5. The FPGA HSWAPEN pin is tied to ground in this sample schematic. HSWAPEN can alternatively be tied High. Review the FPGA configuration user guide for the effect of the alternate HSWAPEN setting.
6. The Virtex-5 FPGA does not support AES decryption in the 16-bit-wide configuration mode shown in this sample schematic. Thus, the V_{BATT} decryptor key battery power supply is unused and is tied to GND.
7. The FPGA RS[1:0] pins are not connected in this basic configuration schematic. See XAPP1100, *MultiBoot with Virtex-5 FPGAs and Platform Flash XL*, for an example XCF128X implementation with MultiBoot support. Fallback is not supported in Slave SelectMAP mode.
8. DONE LED lights when DONE is High. INIT_B/CRC-Error LED lights when INIT_B is Low. Adjust the LED circuits and pull-up values for desired lighting results.
9. Each Platform Flash XL address pin requires a separate pull-down resistor to GND to ensure the XCF128X flash latches the zero address at the start of configuration.
10. V_{DDQ}, V_{CCO} supplies, pull-up resistors, and V_{REF}, can alternately be connected to a 2.5V supply for I/O operation at 2.5V.
11. The required READY_WAIT (INIT_B) pull-up value is board dependent, but it must be strong enough to meet the READY_WAIT rise time (T_{RWRT}) requirement.
12. For the maximum clock frequency, see "Determining the Maximum Configuration Clock Frequency," page 45.

Figure 2-2 shows a Virtex-6 FPGA connected to Platform Flash XL for Slave SelectMAP configuration mode with indirect programming support.

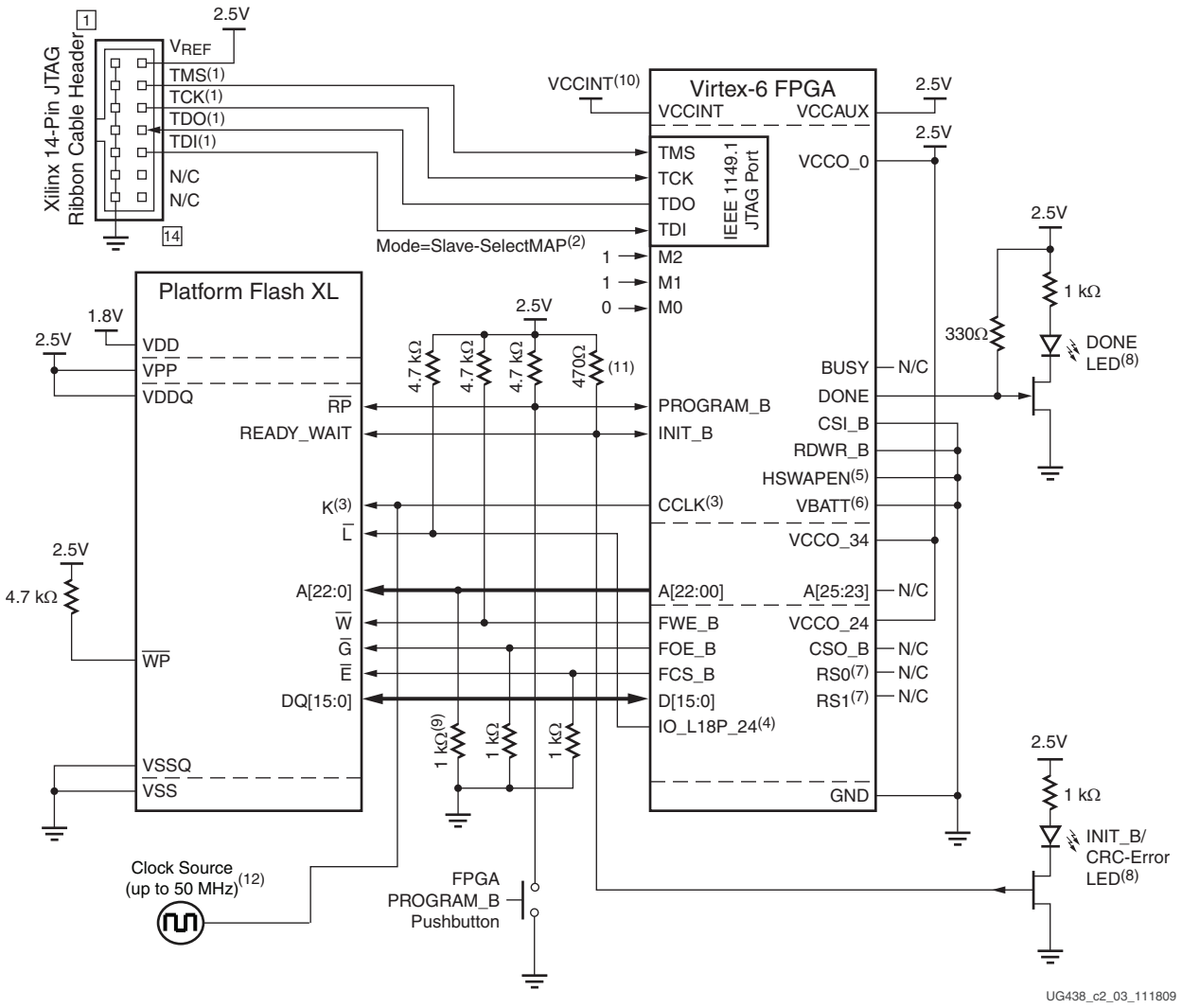


Figure 2-2: **Virtex-6 FPGA Slave SelectMAP Configuration Mode from Platform Flash XL**

Notes relating to Figure 2-2:

1. The JTAG connections are shown for a simple, single-device JTAG scan chain. When multiple devices are on the JTAG scan chain, use the proper IEEE Std 1149.1 daisy-chain technique to connect the JTAG signals. The TCK signal integrity is critical for JTAG operation. Route, terminate, and if necessary, buffer the TCK signal appropriately to ensure signal integrity for the devices in the JTAG scan chain.
2. The FPGA mode (M[2:0]) pins are shown set to Slave SelectMAP mode (110). The implementation of a board-level option that enables the user to change the FPGA mode pins to JTAG mode (101) is recommended to enable JTAG-based debug capability for the FPGA during design prototyping without interference from the Slave SelectMAP configuration activities.
3. CCLK signal integrity is critical. Route and terminate the CCLK signal appropriately to ensure good signal integrity at the XCF128X K pin and at the FPGA CCLK pin.

4. The iMPACT software requires the Virtex-6 FPGA's IO_L18P_24 connection to the device's L pin to support JTAG-based indirect programming.
5. The FPGA HSWAPEN pin is tied to ground in this sample schematic. HSWAPEN can alternatively be tied High. Review the FPGA data sheet for the effect of the alternate HSWAPEN setting.
6. The Virtex-6 FPGA does not support AES decryption in the 16-bit-wide configuration mode shown in this sample schematic. Thus, the V_{BATT} decryptor key battery power supply is unused and is tied to GND.
7. The FPGA RS[1:0] pins are not connected in this basic configuration schematic. See XAPP1100, *MultiBoot with Virtex-5 FPGAs and Platform Flash XL*, for an example XCF128X implementation with MultiBoot support. Fallback is not supported in Slave SelectMAP mode.
8. DONE LED lights when DONE is High. INIT_B/CRC-Error LED lights when INIT_B is Low. Adjust the LED circuits and pull-up values for desired lighting results.
9. Each Platform Flash XL address pin requires a separate pull-down resistor to GND to ensure the XCF128X flash latches the zero address at the start of configuration.
10. See DS152, *Virtex-6 FPGA Data Sheet*, for the V_{CCINT} supply voltage.
11. The required READY_WAIT (INIT_B) pull-up value is board dependent, but it must be strong enough to meet the READY_WAIT rise time (T_{RWRT}) requirement.
12. For the maximum clock frequency, see "Determining the Maximum Configuration Clock Frequency," page 45.

The Slave SelectMAP configuration mode interface signals that influence the successful start and stop of data transfer are listed in Table 2-1.

Table 2-1: **Platform Flash XL and FPGA SelectMAP Configuration Signals and Connections**

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
N/A	TMS	JTAG test mode select. Connect from cable header to TMS pin in all devices in the JTAG chain. Buffer as necessary.	JTAG functionality is always available. Do not perform JTAG operations during SelectMAP configuration. JTAG operations during configuration can interrupt the configuration sequence.	JTAG functionality is always available.	JTAG is the required access path to the FPGA for indirect programming of the XCF128X. An FPGA design (i.e., indirect programming core) is downloaded through JTAG to the FPGA. This bridges the FPGA's JTAG port with the FPGA's pin interface to the XCF128X. Commands and data are sent and received through JTAG to the FPGA indirect programming core to program and perform other operations on the XCF128X.
N/A	TCK	JTAG test clock. Connect from cable header to TCK pin in all devices in JTAG chain. Signal integrity is critical. Buffer as necessary.			
N/A	TDO	JTAG test data output. Daisy-chain to TDI pin of next device in JTAG chain. Last device returns TDO to the cable header.			
N/A	TDI	JTAG test data input. Daisy-chain from TDO of prior device in JTAG chain. First device receives TDI from cable header.			

Table 2-1: **Platform Flash XL and FPGA SelectMAP Configuration Signals and Connections *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
N/A	M[2:0]	These are the FPGA configuration mode inputs. Set M[2:0]=110 for Slave SelectMAP mode. Implementing mode switches to enable JTAG mode (M[2:0]=101) allows JTAG debug without interference from the Slave SelectMAP configuration activities.	On the rising edge of INIT_B, the FPGA samples the M[2:0] pins to determine the configuration mode. See the FPGA configuration user guide for the full list of modes.	Dedicated for configuration.	These pins are dedicated for configuration.
RP#	PROGRAM_B	These are the active-Low FPGA configuration and XCF128X reset inputs. Connect XCF128X RP# to FPGA PROGRAM_B. Connect the signal to an external pull-up. Connect the signal to a pushbutton to ground to allow manual initiation of the reconfiguration process.	Driving PROGRAM_B Low clears the FPGA configuration and initiates an FPGA reconfiguration sequence. The signal must remain High during the configuration process.	PROGRAM_B must remain High after configuration. Driving PROGRAM_B Low clears the FPGA configuration and initiates an FPGA reconfiguration process.	PROGRAM_B must remain High during indirect programming. If PROGRAM_B drives Low during indirect programming, the programming process will be interrupted.
READY_WAIT	INIT_B	This is the configuration sequence initialization handshake signal. Connect XCF128X READY_WAIT to FPGA INIT_B. Connect the signal to an external pull-up resistor capable of transitioning the signal from Low to High in T_{RWRT} time as specified in DS617, *Platform Flash XL High-Density Configuration and Storage Device*. The FPGA and XCF128X drive this signal Low during their respective power-on reset (POR) processes. When each device completes its POR process, it releases its pin to high-impedance. The external pull-up pulls the signal High. The rising edge causes the XCF128X to latch the starting read address to its internal address counter and causes the FPGA to start its configuration process.	At the start of configuration, the FPGA drives INIT_B Low during its initialization process. At the end of initialization, the FPGA releases INIT_B to high-impedance. The external resistor pulls INIT_B High. The rising edge of INIT_B causes the FPGA to sample the configuration mode from the M[2:0] pins and start a corresponding configuration sequence. If a CRC error is detected during the configuration process, the FPGA drives INIT_B Low.	This signal is normally high-impedance. It can drive Low to indicate CRC error from configuration process or from a readback CRC function.	This signal can drive Low or can be in high-impedance. It indicates configuration initialization or CRC error for the FPGA design download during the indirect programming process.

Table 2-1: **Platform Flash XL and FPGA SelectMAP Configuration Signals and Connections *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
K	CCLK	This is the configuration clock. The FPGA Slave SelectMAP mode requires an external source to drive both the XCF128X K and FPGA CCLK. Configuration clock signal integrity is critical. Apply necessary termination to ensure signal integrity.	This clock drives the SelectMAP configuration sequence. For each clock cycle, the XCF128X's internal address counter is incremented, the corresponding 16 bits of array data is output to the data bus, and the FPGA registers the data on the next rising edge of CCLK.	This is the dedicated CCLK pin. The external clock source continues to drive the CCLK input. Typically, CCLK is not used after configuration. However, the CCLK pin can be accessed through the STARTUP primitive when applicable. See the FPGA libraries guide for a description of the STARTUP primitive.	This clock is unused.
A[22:0]	A[22:00] A[25:23]	This is the XCF128X address bus. Connect each XCF128X address pin (A[22:0]) to the corresponding FPGA address pin (A[22:00]). Connect each address line (A[22:0]) to an external pull-down resistor. The pull-down resistors provide an address value of `0x000000` when the XCF128X latches its starting read address at the beginning of the configuration sequence. Do not connect the FPGA A[25:23] pins. A[25:23] pins can drive High or Low during indirect programming operations.	During SelectMAP configuration, the A[25:00] pins are not used and are high-impedance.	These pins are used for user I/O. For FPGA designs that do not use the XCF128X, the FPGA A[25:00] pins are ignored by the XCF128X A[22:0] pins when E# is disabled. For FPGA designs that access the XCF128X, connect the flash controller address bus to the FPGA A[22:00] pins.	The indirect programming core drives a 26-bit address to the FPGA A[25:00] pins. The lower 23 address bits (A[22:00]) are applied to the XCF128X. Thus, the FPGA A[25:23] are actively driven during indirect programming, but unused by the XCF128X.

Table 2-1: **Platform Flash XL and FPGA SelectMAP Configuration Signals and Connections *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
DQ[15:0]	D[15:0]	This is the SelectMAP (x16) data bus. Connect each XCF128X data pin (DQ[15:0]) to the corresponding FPGA SelectMAP data pin (D[15:0]). During the configuration sequence, the XCF128X outputs a 16-bit data word from its array to the data bus after the rising-edge in each clock cycle The FPGA SelectMAP data bus also includes pins D[31:16]. However, the FPGA detects the 16-bit bus width from the initial bitstream data pattern on D[7:0]. Thus, the FPGA ignores activity on D[31:16].	The FPGA registers the data from the SelectMAP (x16) data bus on the rising edge of CCLK.	These pins can be used for user I/O(1). For FPGA designs that do not use the XCF128X, the FPGA D[15:0] pins are ignored by the XCF128X DQ[15:0] pins when E# is disabled. For FPGA designs that access the XCF128X, connect the flash controller data bus to the FPGA D[15:0] pins.	The indirect programming core uses the FPGA D[15:0] pins to drive/receive data to/from the XCF128X.
W#	FWE_B	This is the XCF128X active-Low, write enable control input. Connect the XCF128X W# pin to the FPGA FWE_B pin. Connect the signal to an external pull-up resistor to keep W# High throughout the configuration process.	During SelectMAP configuration, FWE_B is not used and is high-impedance. The external resistor pulls the W# pin High to disable the XCF128X write function.	This pin can be used for user I/O(1). For FPGA designs that do not use the XCF128X, the FPGA FWE_B state is ignored by the XCF128X W# pin when E# is disabled. For FPGA designs that access the XCF128X, connect the flash controller write-enable to the FWE_B pin.	The indirect programming core drives FWE_B Low or High as necessary for programming the XCF128X.
G#	FOE_B	This is the XCF128X active-Low, output enable control input. Connect the XCF128X G# pin to the FPGA FOE_B pin. Connect this input to an external pull-down resistor to keep G# Low throughout the configuration process.	During SelectMAP configuration, FOE_B is not used and is high-impedance. The external resistor pulls the G# pin Low to enable the XCF128X output.	This pin can be used for user I/O(1). For FPGA designs that do not use the XCF128X, the FPGA FOE_B state is ignored by the XCF128X G# pin when E# is disabled. For FPGA designs that access the XCF128X, connect the flash controller output-enable to the FOE_B pin.	The indirect programming core drives FOE_B Low or High as necessary for programming the XCF128X.

Table 2-1: **Platform Flash XL and FPGA SelectMAP Configuration Signals and Connections *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
E#	FCS_B	This is the XCF128X active-Low, chip select control input. Connect the XCF128X E# pin to the FPGA FCS_B pin. Connect this input to an external pull-down resistor to keep E# Low throughout the configuration process.	During SelectMAP configuration, FCS_B is not used and is high-impedance. The external resistor pulls the E# pin Low to enable the XCF128X.	For FPGA designs that do not use the XCF128X, the FPGA design must drive FCS_B High to disable the XCF128X and put it into a low-power, quiescent state. Otherwise, the XCF128X can continue to drive array data to the FPGA data pins. For FPGA designs that access to the XCF128X, connect the flash controller chip-select to the FCS_B pin.	The indirect programming core drives FCS_B Low or High as necessary for programming the XCF128X.
L#	IO_L9P_CC_GC_4 (Virtex-5 FPGA) or IO_L18P_24 (Virtex-6 FPGA)	This is the XCF128X active-Low, address latch enable control input. Connect this input to the named FPGA pin. (See FPGA Pin Name column.) An external pull-up resistor must be connected to L# to keep L# High throughout the configuration process. A starting read address is latched into the XCF128X's internal address counter at the beginning of the configuration sequence. Disabling L# prevents other addresses from corrupting the internal address counter during the configuration sequence.	During configuration, the named FPGA I/O is unused and is high-impedance. The external resistor pulls the L# pin High.	This pin can be used for user I/O(1). For FPGA designs that do not use the XCF128X, the activity on the named FPGA pin is ignored by the XCF128X L# pin when the E# is disabled. For FPGA designs that access the XCF128X, drive L# to a constant Low to enable the XCF128X address latch in asynchronous read/write operations.	The indirect programming core drives the XCF128X L# control input Low through the named FPGA pin to enable the XCF128X address latch for asynchronous read/write operations.
N/A	BUSY	This is the FPGA BUSY output pin. Do not connect this pin. Use it only for SelectMAP readback, which is not supported in this setup.	This pin drives Low during configuration.	This pin is dedicated for configuration.	This pin is dedicated for configuration.

Table 2-1: **Platform Flash XL and FPGA SelectMAP Configuration Signals and Connections *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
N/A	DONE	This is the FPGA open-drain DONE status indicator. The open-drain DONE must be connected to a strong, external pull-up to ensure a fast rising-edge at the end of the configuration process. The sample schematic also connects DONE to an LED driver for visible indication of FPGA configuration DONE status.	The FPGA drives DONE Low during configuration. At the end of a successful configuration sequence, DONE switches to high-impedance, allowing an external pull-up to pull DONE High.	DONE maintains its state from the end of the configuration sequence.	Because the indirect programming solution downloads a design to the FPGA, a strong external pull-up is required to pull DONE High and enable the FPGA indirect programming design to start.
N/A	CS_B (Virtex-5 FPGA) or CSI_B (Virtex-6 FPGA)	This is the FPGA SelectMAP chip select control input. For a single FPGA, and for the first FPGA of a multi-FPGA daisy-chain, connect CS_B/CSI_B to ground to select the FPGA for configuration. For other FPGAs in a multi-FPGA parallel configuration daisy-chain, see the CSO_B signal description for CS_B/CSI_B connections. The Virtex-5 FPGA chip select pin is named CS_B. The Virtex-6 FPGA select pin is named CSI_B.	The CS_B/CSI_B input must be held Low throughout configuration to keep the FPGA selected.	The sample schematic grounds CS_B/CSI_B, making it unusable.	This input is unused. CS_B/CSI_B is grounded in the sample schematic.
N/A	RDWR_B	FPGA SelectMAP read/write control input. Connect this input to ground to put the SelectMAP interface into write mode for configuration.	The RDWR_B input must be held Low throughout configuration to keep SelectMAP bus in write mode.	The sample schematic grounds RDWR_B, making it unusable.	Unused. RDWR_B is grounded in the sample schematic.

Table 2-1: **Platform Flash XL and FPGA SelectMAP Configuration Signals and Connections *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
N/A	HSWAPEN	This signal enables or disables internal pull-ups on FPGA I/O pins during configuration. The sample schematic connects HSWAPEN to ground in order to enable internal pull-ups. Optionally, HSWAPEN can be pulled High to disable the internal pull-ups.	The HSWAPEN input enables or disables internal pull-ups on non-dedicated FPGA I/O pins during configuration. • HSWAPEN=0 enables the internal pull-ups. • HSWAPEN=1 disables the internal pull-ups.	N/A	Because the indirect programming sequence has periods in which the FPGA is unconfigured or configured, the HSWAPEN input state can cause internal pull-ups to switch on or off during the indirect programming sequence. The sample schematic and indirect programming FPGA core is designed to maintain internal pull-ups for most pins.
N/A	CSO_B	When configuring a single FPGA, CSO_B is unused. For multiple-FPGA, parallel configuration daisy-chains, connect the CSO_B pin of each FPGA to the CS_B/CSI_B pin of the next FPGA in the parallel configuration daisy-chain. Connect a strong external pull-up to each CSO_B signal used in the daisy-chain. See Parallel Daisy Chain in the FPGA configuration user guide for more information.	CSO_B is not used for single FPGA configuration. CSO_B is normally high-impedance during configuration. For a multiple-FPGA parallel configuration daisy-chain, CSO_B can drive Low to enable the next FPGA in the daisy-chain to receive a bitstream from the data bus.	This pin can be used for user I/O(1).	This pin is high-impedance with an internal pull-up.

Table 2-1: **Platform Flash XL and FPGA SelectMAP Configuration Signals and Connections *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
N/A	RS[1:0]	These are the FPGA revision select pins. For the basic FPGA configuration shown in the sample schematic, do not connect the RS[1:0] pins (See XAPP1100, *MultiBoot with Virtex-5 FPGAs and Platform Flash XL*, for alternate implementations that support MultiBoot).	These pins are high-impedance. They can drive High or Low due to MultiBoot command during configuration.	These pins can be used for user I/O(1).	When RS[1:0] pins are deemed unused, the indirect programming core sets the RS[1:0] pins to high-impedance with an internal pull-up. When using the MultiBoot scheme from the *MultiBoot with Virtex-5 FPGAs and Platform Flash XL* application note, specify the XCF128X address bit assignments for the RS[1:0] pins in the iMPACT (11.3 or later) programming flow.
WP#	N/A	This is the XCF128X active-Low, write-protect pin. Connect WP# to an external pull-up resistor to enable in-system programming.	N/A	The external resistor pulls the WP# pin High.	WP# must be High to enable XCF128X programming.
N/A	V_{BATT}	This is the battery-backed AES key supply. Connect this to ground. It is not used because XCF128X is not compatible with the AES 8-bit parallel data bus (or serial) requirement.	N/A	N/A	N/A
N/A	V_{CCINT}	This is the FPGA V_{CCINT} power supply.	This is the FPGA V_{CCINT} power supply.	This is the FPGA V_{CCINT} power supply.	This is the FPGA V_{CCINT} power supply.
N/A	V_{CCAUX}	This is the 2.5V FPGA V_{CCAUX} power supply.	This is the FPGA V_{CCAUX} power supply.	This is the FPGA V_{CCAUX} power supply.	This is the FPGA V_{CCAUX} power supply.
V_{DD}	N/A	1.8V XCF128X core power supply	XCF128X core power supply.	XCF128X core power supply.	XCF128X core power supply.
V_{DDQ}/V_{PP}	VCCO_0 VCCO_1 VCCO_2 VCCO_4 (Virtex-5 FPGA) or VCCO_0 VCCO_24 VCCO_34 (Virtex-6 FPGA)	This is the I/O (bank) power supply: • Virtex-5 FPGA V_{CCO} = 3.3V (or 2.5V). • Virtex-6 FPGA V_{CCO} = 2.5V.	This is the I/O power supply.	This is the I/O power supply.	This is the I/O power supply.

Table 2-1: **Platform Flash XL and FPGA SelectMAP Configuration Signals and Connections *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
V_{SS}/V_{SSQ}	GND	This is a ground.	This is a ground.	This is a ground.	This is a ground.
N/A	User I/O	This is FPGA user (non-dedicated) I/O.	This I/O is high-impedance. Internal pull-up is dictated by HSWAPEN control input (see HSWAPEN above).	This is user-defined I/O.	All user I/O that are not included in the set of FPGA BPI configuration pins are high-impedance with an internal pull-up. This matches unconfigured I/O behavior with HSWAPEN=0.

Notes:

1. After configuration, the default FPGA design configuration of an unused User I/O is high-impedance with an internal pull-down.

Slave SelectMAP Configuration from Platform Flash XL

The FPGA Slave SelectMAP mode configuration process from a Platform Flash XL is described in Figure 2-3 and its notes.

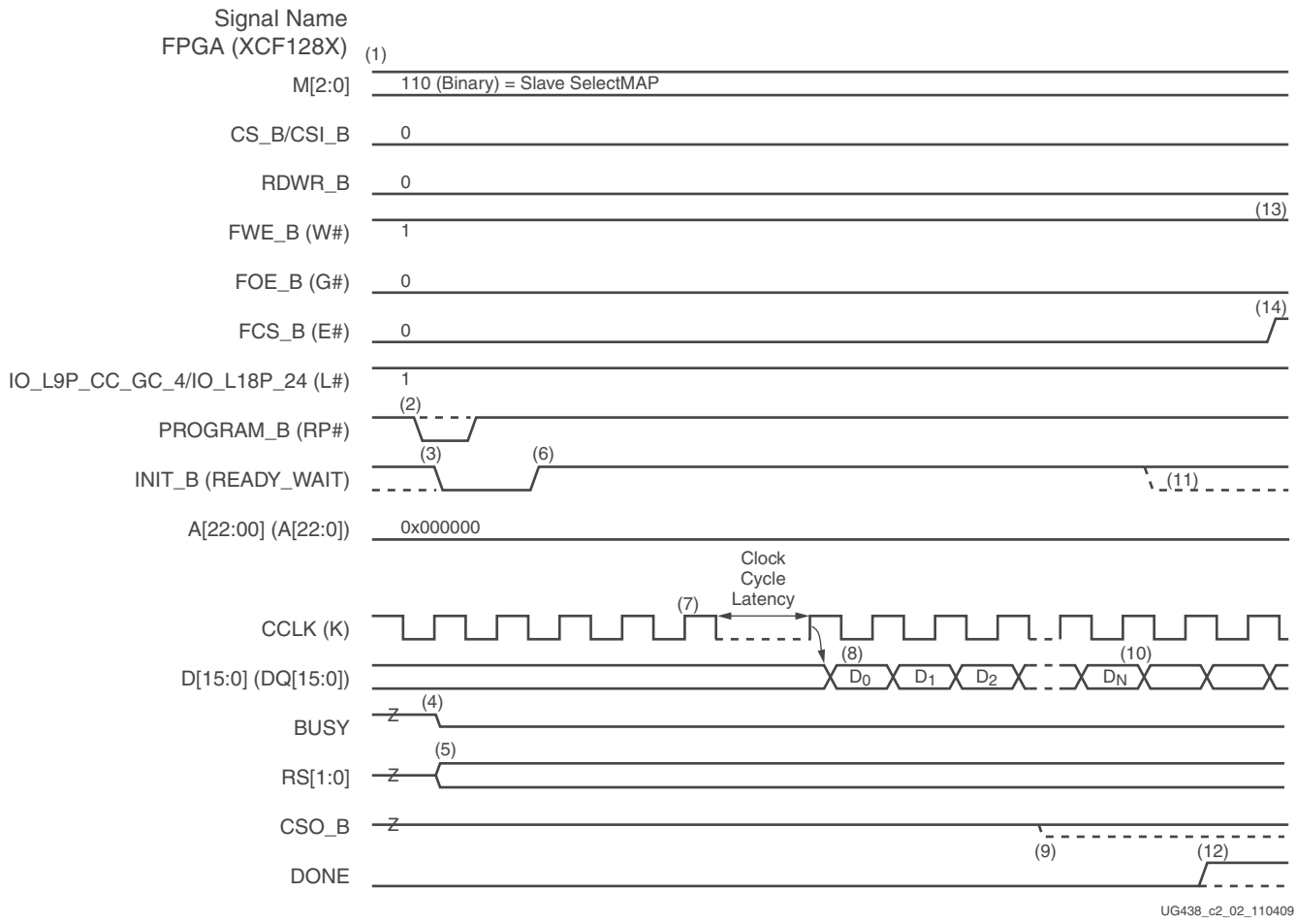


Figure 2-3: **FPGA Slave SelectMAP Configuration Flow**

Notes:

1. External resistors hold many of the XCF128X and FPGA control/input signals High or Low to enable XCF128X array output and FPGA configuration. These settings include:
   - M[2:0], CS_B/CSI_B=0
   - RDWR_B=0
   - W#=1
   - G#=0
   - E#=0
   - L#=1
   - A[22:0]=0x000000
   - A free-running clock drives the XCF128X K and FPGA CCLK.
2. Configuration starts with power-up (PROGRAM_B is pulled and kept High) or with a High-Low-High pulse to PROGRAM_B.
3. For power-up configuration, INIT_B starts Low. For configuration initiated using PROGRAM_B, INIT_B drives Low when PROGRAM_B is pulsed Low.
4. When CS_B/CSI_B is held Low, BUSY is driven Low at the start of configuration.

5. RS[1:0] are typically high-impedance. However, a MultiBoot (or Fallback event in Master Select MAP or Master BPI mode) can cause RS[1:0] to drive High or Low.
6. INIT_B releases at the end of the FPGA's internal initialization process. An external resistor pulls INIT_B High. On the rising-edge of INIT_B, the FPGA samples its M[2:0] pins to determine the configuration mode.
7. Three K clock cycles after READY_WAIT goes High, the XCF128X latches the starting read address from A[22:0] to its internal address counter for the ensuing synchronous array read operation.
8. After a few clock cycles of latency following the address latching event, the XCF128X outputs the first 16-bit data word from the starting address in its array. On every clock cycle thereafter, the XCF128X outputs the next 16-bit data word from the array. The FPGA registers the 16-bit data word on the rising-edge of CCLK.
9. For a multi-FPGA parallel configuration daisy-chain, CSO_B can drive Low to select the next FPGA in the daisy-chain for bitstream loading from the data bus.
10. Near the last 16-bit word of the bitstream, the FPGA begins its startup sequence.
11. If the FPGA detects a CRC error during bitstream delivery, the FPGA drives its INIT_B pin Low. DONE stays Low.
12. If the FPGA successfully receives the bitstream, the FPGA releases its DONE pin during the startup sequence, and an external resistor pulls DONE High.
13. During the startup sequence, the multi-purpose pins are activated with their configurations from the user's FPGA design.
14. The FPGA design must drive the FPGA FCS_B pin High to disable the XCF128X (i.e., stop data output from the XCF128X).

XILINX®

Chapter 3

Alternate Configuration Modes

Master SelectMAP Configuration Mode

Platform Flash XL is optimized for Slave SelectMAP configuration mode, but Master SelectMAP configuration mode can also be used. Platform Flash XL connectivity for Master SelectMAP mode is similar to the Slave SelectMAP mode, shown in Figure 2-1, page 16, except for the differences listed below.

The user should be aware of the differences between Master SelectMAP mode and Slave SelectMAP configuration mode:

- FPGA Mode pins (M[2:0]) must be properly set for Master SelectMAP configuration mode.
- The FPGA drives the configuration clock (CCLK) — no external clock source is needed.
- BitGen **-g ConfigRate** sets the nominal CCLK configuration frequency. For appropriate settings, see "Determining the Maximum Configuration Clock Frequency," page 45.
- The FPGA supports the Fallback feature in Master SelectMAP mode. When a configuration error is detected, the Fallback feature causes the FPGA to drive its RS[1:0] pins Low and re-initiate configuration. If the FPGA detects a configuration error on the Fallback attempt, the FPGA drives INIT_B Low and ends the configuration process.

Master BPI-Up Configuration Mode

Platform Flash XL is optimized for Slave SelectMAP configuration mode, but the standard flash interface is also compatible with the Master BPI-Up configuration mode with these key requirements:

- Mode pins (M[2:0]) must be set for Master BPI-Up configuration mode.
- Special control of the power supply sequence or delay of the FPGA configuration process can be required to ensure power-on readiness of Platform Flash XL before the FPGA BPI address sequence. For details, refer to the "Power-On Sequence Precautions" section in UG191, *Virtex-5 FPGA Configuration User Guide* or UG360, *Virtex-6 FPGA Configuration User Guide.*
- The FPGA drives the configuration clock (CCLK) — no external clock source is needed.
- BitGen **-g ConfigRate** sets the nominal CCLK configuration frequency. For appropriate settings, see "Determining the Maximum Configuration Clock Frequency," page 45.

- The FPGA drives the main flash control pins ($\overline{E}$, $\overline{G}$, $\overline{W}$). The external resistor requirements for these pins in Master BPI mode are different than the requirements for Slave SelectMAP. External pull-up resistors are recommended for the XCF128X $\overline{E}$, $\overline{G}$, and $\overline{W}$ pins.
- The FPGA sets the flash burst read start address. No external pull-down resistors are required on the address lines in Master BPI mode.
- The FPGA CS_B/CSI_B and RDWR_B pins are not used in Master BPI mode and do not require a connection to ground as shown in the Slave SelectMAP setup.

The FPGA Master BPI-Up configuration mode enables MultiBoot and Fallback. See the FPGA configuration user guide for details on the Master BPI-Up mode and the ISE® software manuals for details on the BitGen software utility. See Figure 3-1 or Figure 3-2 for example Master BPI-Up configuration mode connections with the Virtex-5 FPGA or Virtex-6 FPGA, respectively. Refer to "FPGA BPI-Up Configuration Signals," page 35 for details on the Master BPI-Up configuration sequence.

Caution! Platform Flash XL does not support the FPGA Master BPI-Down mode in which the bitstream is stored in a descending fashion from a higher to a lower address.

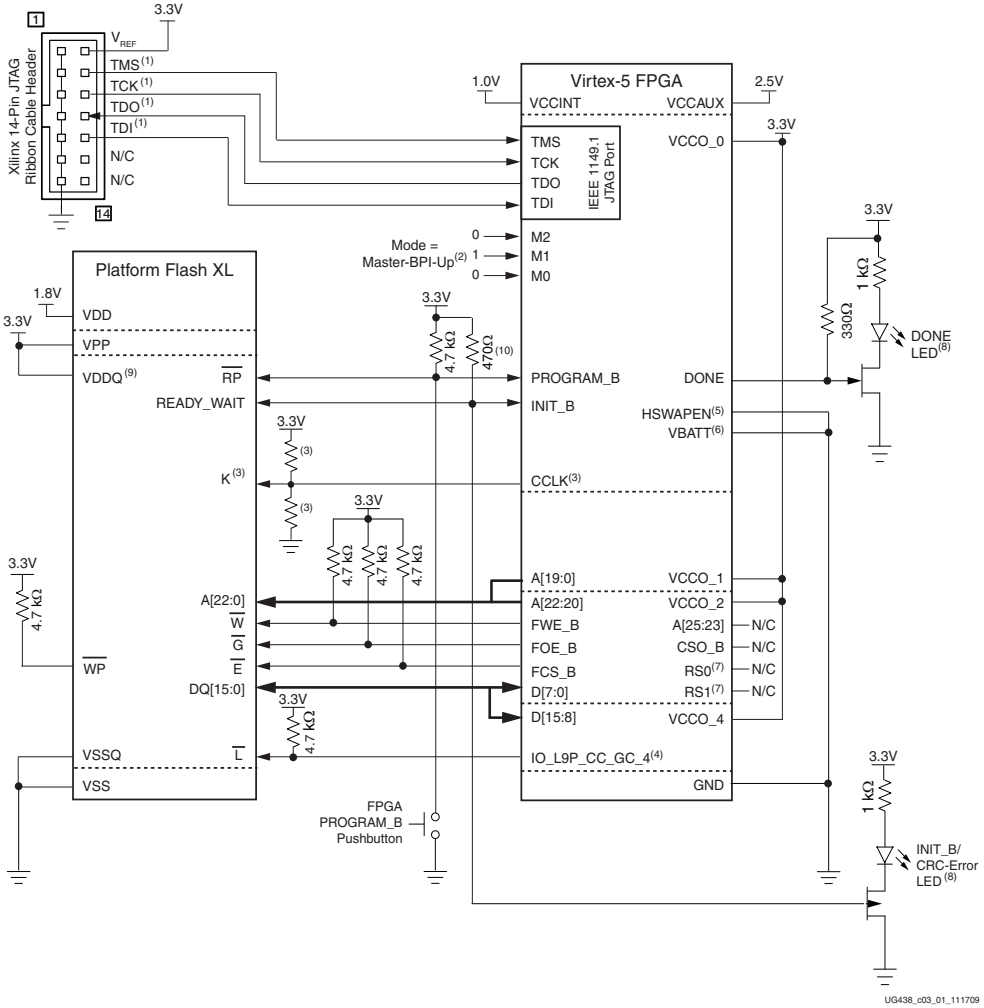


Figure 3-1: **Virtex-5 FPGA Master BPI-Up Configuration Mode from Platform Flash XL with Indirect Programming Support**

Notes relating to Figure 3-1:

1. The JTAG connections are shown for a simple, single-device JTAG scan chain. When multiple devices are on the JTAG scan chain, use the proper IEEE Std. 1149.1 daisy-chain technique to connect the JTAG signals. The TCK signal integrity is critical for JTAG operation. Route, terminate, and if necessary, buffer the TCK signal appropriately to ensure signal integrity for the devices in the JTAG scan chain.
2. The FPGA mode (M[2:0]) pins are shown set to Master BPI-Up mode (`010`). The implementation of a board-level option that enables the user to change the FPGA mode pins to JTAG mode (`101`) is recommended to enable JTAG-based debug capability for the FPGA during design prototyping—without interference from the Master BPI configuration activities.
3. CCLK signal integrity is critical. Route and terminate the CCLK signal appropriately to ensure good signal integrity at the device K pin and at the FPGA CCLK pin.
4. The iMPACT software requires the Virtex-5 FPGA's IO_L9P_CC_GC_4 connection to the XCF128X Flash device L pin to support JTAG based indirect programming.
5. The FPGA HSWAPEN pin is tied to ground in this sample schematic. HSWAPEN can alternatively be tied High. Review the data sheet for the effect of the alternate HSWAPEN setting.
6. The Virtex-5 FPGA does not support AES decryption in the 16-bit-wide configuration mode shown in this sample schematic. Thus, the V_{BATT} decryptor key battery power supply is unused and is tied to GND.
7. This sample schematic supports single bitstream configuration. Thus, the FPGA RS[1:0] pins are not connected in this sample schematic. See the FPGA configuration user guide for use of the RS[1:0] pins to enable Fallback feature support. iMPACT 11.2 (or earlier) does not support indirect programming with addresses routed through the RS[1:0] pins.
8. DONE LED lights when DONE is High. INIT_B/CRC-Error LED lights when INIT_B is Low. Adjust the LED circuits and pull-up values for desired lighting results.
9. V_{DDQ}, V_{CCO} supplies, pull-up resistors, and V_{REF}, can alternately be connected to a 2.5V supply for I/O operation at 2.5V.
10. The required READY_WAIT (INIT_B) pull-up value is board dependent, but it must be strong enough to meet the READY_WAIT rise time (T_{RWRT}) requirement.

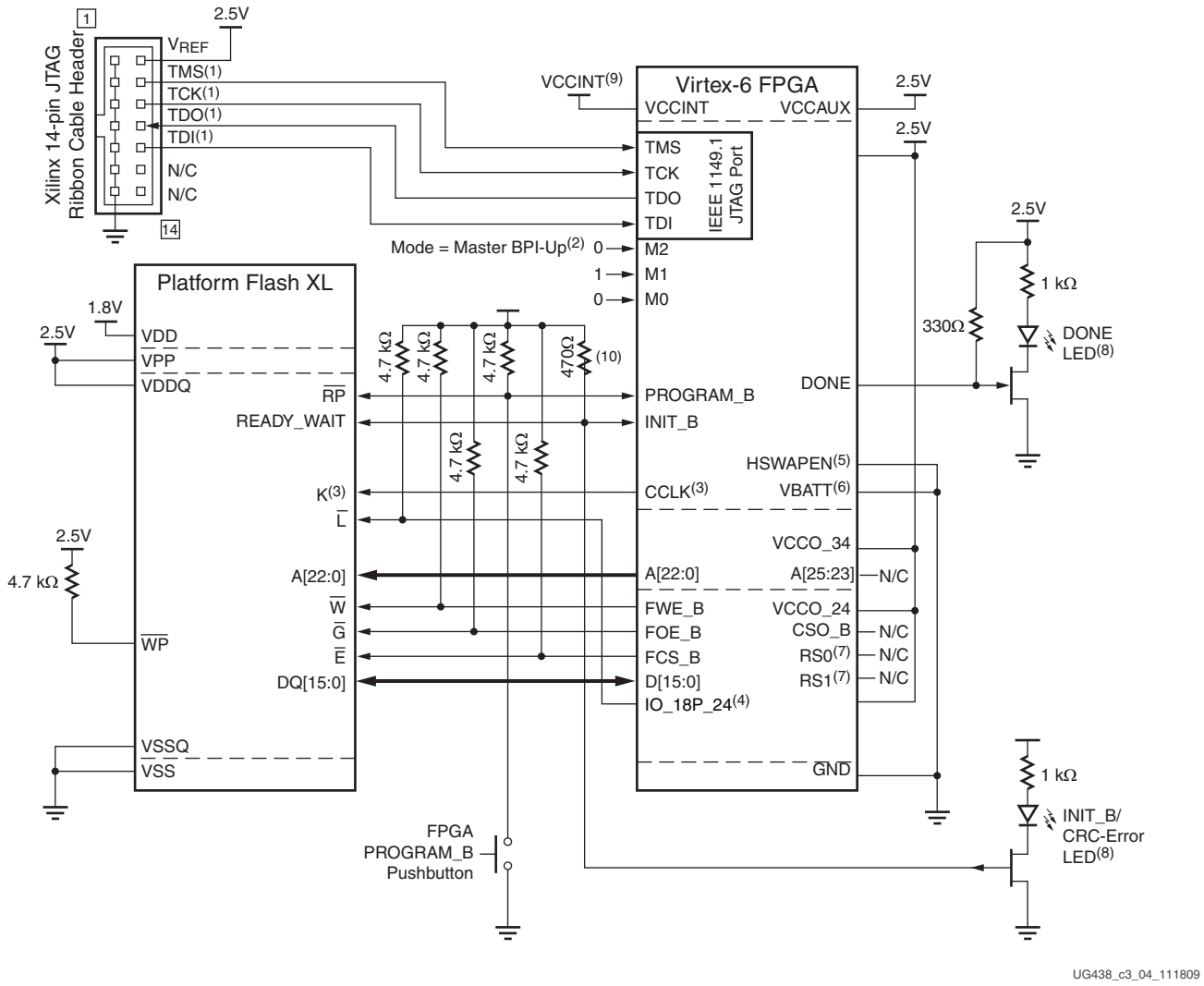


Figure 3-2: **Virtex-6 FPGA Master BPI-Up Configuration Mode from Platform Flash XL with Indirect Programming Support**

Notes relating to Figure 3-2:

1. The JTAG connections are shown for a simple, single-device JTAG scan chain. When multiple devices are on the JTAG scan chain, use the proper IEEE Std 1149.1 daisy-chain technique to connect the JTAG signals. The TCK signal integrity is critical for JTAG operation. Route, terminate, and if necessary, buffer the TCK signal appropriately to ensure signal integrity for the devices in the JTAG scan chain.
2. The FPGA mode (M[2:0]) pins are shown set to Master BPI-Up mode (`010`). The implementation of a board-level option that enables the user to change the FPGA mode pins to JTAG mode (`101`) is recommended to enable JTAG-based debug capability for the FPGA during design prototyping—without interference from the Master BPI-Up configuration activities.
3. CCLK signal integrity is critical. Route and terminate the CCLK signal appropriately to ensure good signal integrity at the XCF128X K pin and at the FPGA CCLK pin.
4. The iMPACT software requires the Virtex-6 FPGA's IO_L18P_24 connection to the device's L pin to support JTAG-based indirect programming.
5. The FPGA HSWAPEN pin is tied to ground in this sample schematic. HSWAPEN can alternatively be tied High. Review the FPGA data sheet for the effect of the alternate HSWAPEN setting.
6. The Virtex-6 FPGA does not support AES decryption in the 16-bit wide configuration mode shown in this sample schematic. Thus, the V_{BATT} decryptor key battery power supply is unused and is tied to GND.
7. This sample schematic supports single bitstream configuration. Thus, FPGA RS[1:0] pins are not connected in this sample schematic. See the FPGA configuration user guide for use of the RS[1:0] pins. Use iMPACT 11.3 (or later) to indirectly program the XCF128X when a few address bits are routed through the RS[1:0] pins to the XCF128X.
8. DONE LED lights when DONE is High. INIT_B/CRC-Error LED lights when INIT_B is Low. Adjust the LED circuits and pull-up values for desired lighting results.
9. See DS152, *Virtex-6 FPGA Data Sheet: DC and Switching Characteristics*, for the V_{CCINT} supply voltage.
10. The required READY_WAIT (INIT_B) pull-up value is board dependent, but it must be strong enough to meet the READY_WAIT rise time (T_{RWRT}) requirement.

FPGA BPI-Up Configuration Signals

The BPI-Up configuration mode interface signals that influence the successful start and stop of data transfer are listed in Table 3-1. Details on the Virtex-5 FPGA configuration sequence and power-up considerations are discussed in Chapter 3, "FPGA BPI-Up Configuration from Platform Flash XL."

Table 3-1: **FPGA BPI-Up Configuration Mode Signals and Descriptions**

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
N/A	TMS	JTAG test mode select. Connect from cable header to TMS pin in all devices in the JTAG chain. Buffer as necessary.	JTAG functionality is always available. Do not perform JTAG operations during BPI configuration. JTAG operations during configuration can interrupt the configuration sequence.	JTAG functionality is always available.	JTAG is the required access path to the FPGA for indirect programming of the XCF128X. An FPGA design (i.e., indirect programming core) is downloaded through JTAG to the FPGA. This bridges the FPGA's JTAG port with the FPGA's pin interface to the XCF128X. Commands and data are sent/received through JTAG to the FPGA indirect programming core to program and perform other operations on the XCF128X.
N/A	TCK	JTAG test clock. Connect from cable header to TCK pin in all devices in JTAG chain. Signal integrity is critical. Buffer as necessary.			
N/A	TDO	JTAG test data output. Daisy-chain to TDI pin of next device in JTAG chain. Last device returns TDO to the cable header.			
N/A	TDI	JTAG test data input. Daisy-chain from TDO of prior device in JTAG chain. First device receives TDI from cable header.			
N/A	M[2:0]	These are the FPGA configuration mode inputs. Set M[2:0]=010 for Master BPI-Up mode. Implementing mode switches to enable JTAG mode (M[2:0]=101) allows JTAG debug without interference from the Master BPI configuration activities.	On the rising edge of INIT_B, the FPGA samples the M[2:0] pins to determine the configuration mode. See the FPGA configuration user guide for the full list of modes.	Dedicated for configuration.	Dedicated for configuration.
RP#	PROGRAM_B	These are the active-Low FPGA configuration and XCF128X reset inputs. Connect XCF128X RP# to FPGA PROGRAM_B. Connect the signal to an external pull-up. Connect the signal to a pushbutton to ground to allow manual initiation of the reconfiguration process.	Driving PROGRAM_B Low clears the FPGA configuration and initiates an FPGA reconfiguration sequence. The signal must remain High during the configuration process.	PROGRAM_B must remain High after configuration. Driving PROGRAM_B Low clears the FPGA configuration and initiates an FPGA reconfiguration process.	These inputs are unused. PROGRAM_B must remain High during indirect programming. If PROGRAM_B drives Low during indirect programming, the programming process will be interrupted.

Table 3-1: **FPGA BPI-Up Configuration Mode Signals and Descriptions *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
READY_WAIT	INIT_B	This is the configuration sequence initialization handshake signal. Connect XCF128X READY_WAIT to FPGA INIT_B. Connect this signal to an external pull-up resistor capable of transitioning the signal from Low to High in T_{RWRT} time as specified in DS617, *Platform Flash XL High-Density Configuration and Storage Device*. The FPGA and XCF128X drive this signal Low during their respective power-on-reset (POR) processes. When each device completes its POR process, it releases its pin to high-impedance. The external pull-up pulls the signal High. The rising edge causes the XCF128X to latch the starting read address to its internal address counter and causes the FPGA to start its configuration process.	At the start of configuration, the FPGA drives INIT_B Low during its initialization process. At the end of initialization, the FPGA releases INIT_B to high-impedance. The external resistor pulls INIT_B High. The rising edge of INIT_B causes the FPGA to sample the configuration mode from the M[2:0] pins and start a corresponding configuration sequence. If a CRC error is detected during the configuration process, the FPGA drives INIT_B Low.	This signal is normally high-impedance. This signal can drive Low to indicate CRC error from configuration process or from a Readback CRC function.	This signal must be allowed to be pulled High by the external pull-up resistor. The FPGA can drive INIT_B Low or can put INIT_B in high-impedance. The FPGA drives INIT_B Low during the indirect programming process to indicate configuration initialization or CRC error for the FPGA design download.
K	CCLK	This is the configuration clock. Connect the XCF128X K pin to the FPGA CCLK pin. The FPGA drives the CCLK pin with a clock signal in Master BPI-Up mode. Configuration clock signal integrity is critical. Apply necessary termination to ensure signal integrity at both the XCF128X K pin and FPGA CCLK pin.	After INIT_B goes High and the FPGA detects the Master BPI-Up mode setting, the FPGA drives a clock signal out of the CCLK pin. The CCLK signal is also used to drive the FPGA's internal configuration logic. For each clock cycle, the XCF128X's internal address counter is incremented, the corresponding 16 bits of array data is output to the data bus, and the FPGA registers the data on the next rising edge of CCLK.	This is a dedicated CCLK pin. After successful configuration, the FPGA puts the CCLK pin in a high-impedance state. If the CCLK termination scheme (e.g., parallel Thevenin termination) maintains a high-impedance CCLK at an intermediate voltage level, then drive the CCLK to a valid High or Low state through the STARTUP primitive. See the FPGA libraries guide.	Unused.

Table 3-1: **FPGA BPI-Up Configuration Mode Signals and Descriptions *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
A[22:0]	A[22:00] A[25:23]	This is the XCF128X address bus. Connect each XCF128X address pin (A[22:0]) to the corresponding FPGA address pin (A[22:00]). In Master BPI-Up mode, the FPGA drives addresses onto the address bus (A[25:00]). Do not connect the FPGA A[25:23] pins.	In Master BPI-Up mode, the FPGA drives addresses onto the address bus. After the rising edge of each CCLK cycle, the FPGA drives the next address onto the address bus. The XCF128X latches the initial BPI address from the FPGA as the starting read address for the XCF128X's internal address counter. After the XCF128X latches the initial BPI address, the XCF128X uses its internal address counter for reading array data and ignores the address from the FPGA on the address bus.	This bus can be used for user I/O(1). For FPGA designs that do not use the XCF128X, the FPGA A[25:00] pins are ignored by the XCF128X A[22:0] pins when E# is disabled. For FPGA designs that access the XCF128X, connect the flash controller address bus to the FPGA A[22:00] pins.	The indirect programming core drives a 26-bit address to the FPGA A[25:00] pins. The lower 23 address bits (A[22:00]) are applied to the XCF128X. Thus, the A[25:23] FPGA pins are actively driven during indirect programming, but unused by the XCF128X.
DQ[15:0]	D[15:0]	This is the BPIx16 data bus. Connect each XCF128X data pin (DQ[15:0]) to the corresponding FPGA BPI data pin (D[15:0]). During the configuration sequence, the XCF128X outputs a 16-bit data word from its array to the data bus after the rising-edge in each clock cycle.	The FPGA registers the data from the BPIx16 data bus on the rising edge of CCLK.	This bus can be used for user I/O(1). For FPGA designs that do not use the XCF128X, the FPGA D[15:0] pins are ignored by the XCF128X DQ[15:0] pins when E# is disabled. For FPGA designs that access the XCF128X, connect the flash controller data bus to the FPGA D[15:0] pins.	The indirect programming core uses the FPGA D[15:0] pins to drive and receive data to and from the XCF128X.
W#	FWE_B	This is the XCF128X active-Low, write enable control input. Connect the XCF128X W# pin to the FPGA FWE_B pin. Connect this signal to an external pull-up resistor to keep W# High when the XCF128X is not used.	During Master BPI-Up configuration, the FPGA drives the FWE_B pin High.	This pin can be used for user I/O(1). For FPGA designs that do not use the XCF128X, the FPGA FWE_B state is ignored by the XCF128X W# pin when E# is disabled. For FPGA designs that access the XCF128X, connect the flash controller write-enable to the FWE_B pin.	The indirect programming core drives FWE_B Low or High as necessary for programming the XCF128X.

Table 3-1: **FPGA BPI-Up Configuration Mode Signals and Descriptions *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
G#	FOE_B	This is the XCF128X active-Low, output enable control input. Connect the XCF128X G# pin to the FPGA FOE_B pin. Connect this input to an external pull-up resistor to keep G# High when not used.	During Master BPI-Up configuration, the FPGA drives the FOE_B pin Low.	This pin can be used for user I/O(1). For FPGA designs that do not use the XCF128X, the FPGA FOE_B state is ignored by the XCF128X G# pin when E# is disabled. For FPGA designs that access the XCF128X, connect the flash controller output-enable to the FOE_B pin.	The indirect programming core drives FOE_B Low or High as necessary for programming the XCF128X.
E#	FCS_B	This is the XCF128X active-Low, chip select control input. Connect the XCF128X E# pin to the FPGA FCS_B pin. Connect this input to an external pull-up resistor to keep E# High when not used.	During Master BPI-Up configuration, the FPGA drives the FCS_B pin Low.	For FPGA designs that do not use the XCF128X, the FPGA design must drive FCS_B High to disable the XCF128X and put it into a low-power, quiescent state. For FPGA designs that access to the XCF128X, connect the flash controller chip-select to the FCS_B pin.	The indirect programming core drives FCS_B Low or High as necessary for programming the XCF128X.
L#	IO_L9P_CC_GC_4 (Virtex-5 FPGA) or IO_L18P_24 (Virtex-6 FPGA)	This is the XCF128X active-Low, address latch enable control input. Connect this input to the named FPGA pin (see FPGA pin name column). An external pull-up resistor must be connected to L# to keep L# High throughout the configuration process. A starting read address is latched into the XCF128X's internal address counter at the beginning of the configuration sequence. Disabling L# prevents other addresses from corrupting the internal address counter during the configuration sequence.	During configuration, the named I/O is unused and is high-impedance.	This pin can be used for user I/O(1). For FPGA designs that do not use the XCF128X, the activity on the named FPGA pin is ignored by the XCF128X L# pin when E# is disabled. For FPGA designs that access the XCF128X, drive L# to a constant Low to enable the XCF128X address latch in asynchronous read/write operations.	The indirect programming core drives the XCF128X L# control input Low through the named FPGA pin to enable the XCF128X address latch for asynchronous read/write operations.

Table 3-1: **FPGA BPI-Up Configuration Mode Signals and Descriptions *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
N/A	DONE	This is the FPGA open-drain DONE status indicator. The open-drain DONE must be connected to a strong, external pull-up to ensure a fast rising-edge at the end of the configuration process. The sample schematic also connects DONE to an LED driver for visible indication of FPGA configuration DONE status.	The FPGA drives DONE Low during configuration. At the end of a successful configuration sequence, DONE switches to high-impedance, allowing an external pull-up to pull DONE High.	DONE maintains its state from the end of the configuration sequence.	Because the indirect programming solution downloads a design to the FPGA, a strong external pull-up is required to pull DONE High and enable the FPGA indirect programming design to start.
N/A	HSWAPEN	This input enables or disables internal pull-ups on FPGA I/O pins during configuration. The sample schematic connects HSWAPEN to ground in order to enable internal pull-ups. Optionally, HSWAPEN can be pulled High to disable the internal pull-ups.	The HSWAPEN input enables or disables internal pull-ups on FPGA non-dedicated I/O pins during configuration. • HSWAPEN=0 enables the internal pull-ups. • HSWAPEN=1 disables the internal pull-ups.	N/A	Because the indirect programming sequence has periods in which the FPGA is unconfigured or configured, the HSWAPEN input state can cause internal pull-ups to switch on or off during the indirect programming sequence. The sample schematic and indirect programming FPGA core is designed to maintain internal pull-ups for most pins.
N/A	CSO_B	For configuring a single FPGA, CSO_B is unused. For multiple-FPGA, parallel configuration daisy-chains, connect the CSO_B pin of each FPGA to the CS_B/CSI_B pin of the next FPGA in the parallel configuration daisy-chain. Connect a strong external pull-up to each CSO_B signal used in the daisy-chain. See the Parallel Daisy-Chain section in the FPGA configuration user guide for more information.	CSO_B is not used for configuring a single FPGA. CSO_B is normally high-impedance during configuration. For a multiple-FPGA parallel configuration daisy-chain, CSO_B can drive Low to enable the next FPGA in the daisy-chain to receive a bitstream from the data bus.	This pin can be used for user I/O(1).	This pin is high-impedance with an internal pull-up.

Table 3-1: **FPGA BPI-Up Configuration Mode Signals and Descriptions *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
N/A	RS[1:0]	These are the FPGA revision select pins. For the basic FPGA configuration shown in the sample schematic, do not connect the RS[1:0] pins (see the FPGA configuration user guide for details on using the RS[1:0] pins to enable the FPGA Fallback feature).	These pins are high-impedance. These pins can drive High or Low due to the MultiBoot command during configuration. In Master BPI mode, the configuration fallback feature can drive RS[1:0] pins Low.	This pin can be used for user I/O(1).	When RS[1:0] pins are deemed unused, the indirect programming core sets the RS[1:0] pins to high impedance with an internal pull-up. If using a modified design that uses the RS[1:0] pins to control a few XCF128X address pins, then use iMPACT 11.3 (or later) to indirectly program the XCF128X with addressing support through the RS[1:0] pins.
WP#	N/A	This is the XCF128X active-Low, write-protect. Connect WP# to an external pull-up resistor to enable in-system programming.	N/A	The external resistor pulls the WP# pin High.	WP# must be High to enable XCF128X programming.
N/A	V_{BATT}	This is the battery-backed AES key supply. Connect this to ground. This supply is not used because XCF128X is not compatible with the AES 8-bit parallel data bus (or serial) requirement.	N/A	N/A	N/A
N/A	V_{CCINT}	This is the FPGA V_{CCINT} power supply.	This is the FPGA V_{CCINT} power supply.	This is the FPGA V_{CCINT} power supply.	This is the FPGA V_{CCINT} power supply.
N/A	V_{CCAUX}	This is the 2.5V FPGA V_{CCAUX} power supply.	This is the FPGA V_{CCAUX} power supply.	This is the FPGA V_{CCAUX} power supply.	This is the FPGA V_{CCAUX} power supply.
V_{DD}	N/A	This is the 1.8V XCF128X core power supply	This is the XCF128X core power supply.	This is the XCF128X core power supply.	This is the XCF128X core power supply.
V_{DDQ}/V_{PP}	VCCO_0 VCCO_1 VCCO_2 VCCO_4 (Virtex-5 FPGA) or VCCO_0 VCCO_24 VCCO_34 (Virtex-6 FPGA)	This is the I/O (bank) power supply: • Virtex-5 FPGA: V_{CCO} = 3.3V (or 2.5V) • Virtex-6 FPGA: V_{CCO} = 2.5V	This is the I/O power supply.	This is the I/O power supply.	This is the I/O power supply.
V_{SS}/V_{SSQ}	GND	Ground	Ground	Ground	Ground

Table 3-1: **FPGA BPI-Up Configuration Mode Signals and Descriptions *(Cont'd)***

Platform Flash XL Pin Name	FPGA Pin Name	Signal/Connection Description	Function during FPGA Configuration	Function after FPGA Configuration	Function during Indirect XCF128X Programming
N/A	User I/O	This is FPGA user (non-dedicated) I/O.	This I/O is high-impedance. Internal pull-up is dictated by HSWAPEN control input (see HSWAPEN above).	This is user-defined I/O.	All user I/O that are not included in the set of FPGA BPI configuration pins are high-impedance with an internal pull-up. This matches the behavior of unconfigured I/O with HSWAPEN=0.

Notes:
1. After configuration, the default FPGA design configuration of an unused User I/O is high-impedance with an internal pull-down.

FPGA BPI-Up Configuration from Platform Flash XL

The FPGA Master BPI-Up mode configuration process from a Platform Flash XL is described in Figure 3-3 and its notes.

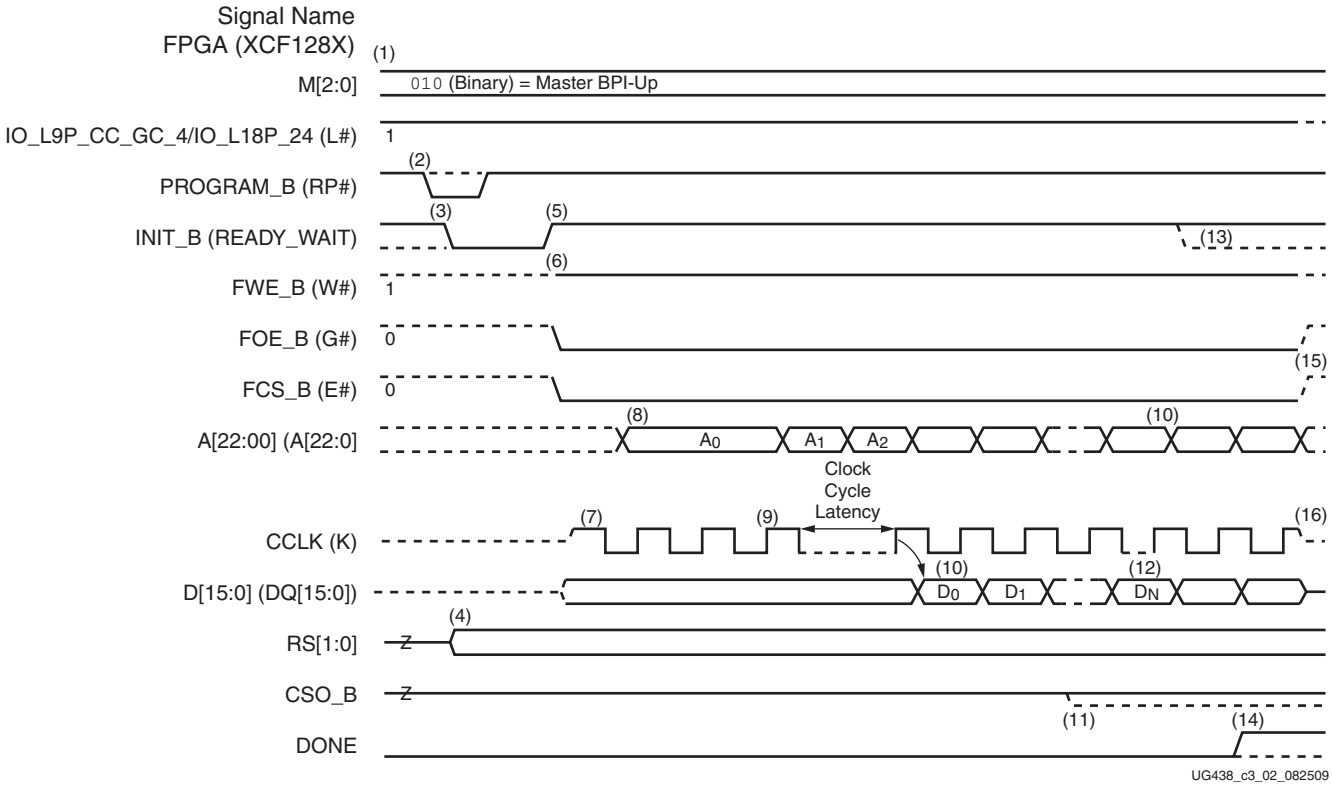


Figure 3-3: **FPGA BPI-Up Configuration Flow**

Notes relating to Figure 3-3:

1. External resistors hold a few of the XCF128X and FPGA control/input signals High or Low including:
   - M[2:0]=101
   - L#=1.
2. Configuration starts with power-up (PROGRAM_B is pulled and kept High) or with a High-Low-High pulse to PROGRAM_B.
3. For power-up configuration, INIT_B starts Low. For PROGRAM_B-initiated configuration, INIT_B drives Low when PROGRAM_B is pulsed Low.
4. RS[1:0] are typically high-impedance. However, a MultiBoot (or Fallback event in Master SelectMAP or Master BPI mode) can cause RS[1:0] to drive High or Low.
5. INIT_B releases at the end of the FPGA's internal initialization process. An external resistor pulls INIT_B High. On the rising-edge of INIT_B, the FPGA samples its M[2:0] pins to determine the configuration mode.
6. Upon determining the Master BPI-Up mode from the M[2:0] pins, the FPGA drives FWE_B High, FOE_B Low, and FCS_B Low.
7. For the Master mode, the FPGA drives CCLK T_{ICCK} time after the rising-edge of INIT_B.
8. The FPGA drives the initial address (A0) through its A[25:0] pins and holds the initial address at $T_{INITADDR}$ CCLK cycles. For a power-on configuration, the initial address is 0x0000000. For a MultiBoot-triggered configuration, the address can be different.
9. Three K clock cycles after READY_WAIT goes High, the XCF128X latches the starting read address from A[22:0] to its internal address counter for the ensuing synchronous array read operation. Although the FPGA drives new address values to the address bus, the XCF128X ignores the addresses from the FPGA after latching the initial address and instead uses its own internal address counter.
10. After a few clock cycles of latency following the address latching event, the XCF128X outputs the first 16-bit data word from the starting address in its array. On every clock cycle thereafter, the XCF128X outputs the next 16-bit data word from the array. The FPGA registers the 16-bit data word on the rising-edge of CCLK.
11. For a multi-FPGA parallel configuration daisy-chain, CSO_B can drive Low to select the next FPGA in the daisy-chain for bitstream loading from the data bus.
12. Near the last 16-bit word of the bitstream, the FPGA begins its startup sequence.
13. If the FPGA detects a CRC error during bitstream delivery, the FPGA drives its INIT_B pin Low. DONE stays Low.
14. If the FPGA successfully receives the bitstream, then the FPGA releases its DONE pin during the startup sequence, and an external resistor pulls DONE High.
15. During the startup sequence, the multi-purpose pins are activated with their configurations from the user's FPGA design. If not used in the FPGA design, the FCS_B pin is high-impedance and is pulled High by the external resistor to disable the XCF128X.
16. At the end of configuration, the master CCLK is disabled into a high-impedance state by default.

Master Mode Considerations

When the FPGA is set to the Master SelectMAP or Master BPI-Up mode, the FPGA supplies the configuration clock by driving an internal oscillator to its CCLK pin. As with any configuration mode, the configuration clock signal integrity is critical. It is recommended that the user perform IBIS simulations of the configuration clock board trace and the application of termination to ensure good signal integrity. Many references for termination strategies exist on the web. After a successful FPGA configuration, the FPGA puts the CCLK pin into a high-impedance state, by default.

To ensure good signal integrity for the master configuration clock, the structure of the master FPGA CCLK is important. Signal integrity must be ensured both at the XCF128X K (clock) pin and the FPGA CCLK pin. See Figure 3-4 for a description of the FPGA master CCLK structure.

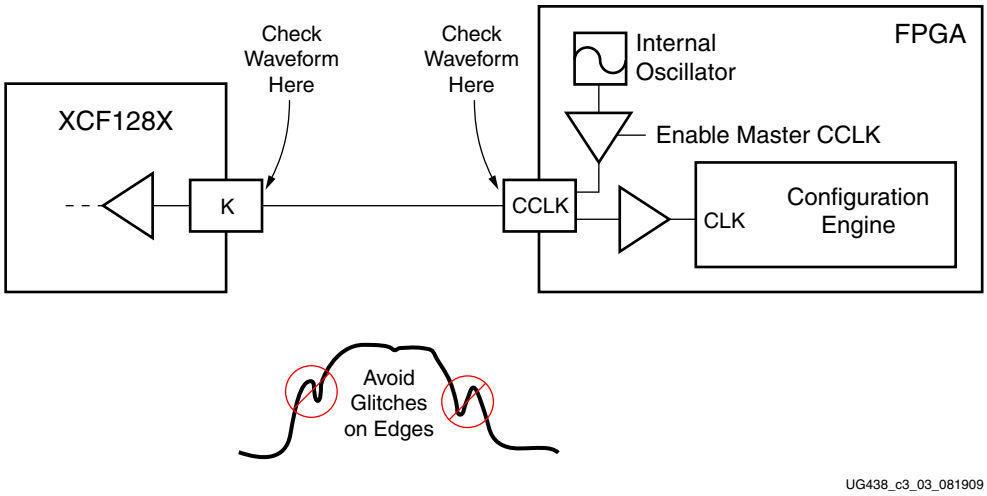


Figure 3-4: **FPGA Master CCLK Structure**

XILINX®

Chapter 4

Calculating Configuration Time

Determining the Maximum Configuration Clock Frequency

The maximum clock frequency for configuring an FPGA from XCF128X configuration flash depends on the data transfer timing within a given configuration clock cycle (see Figure 4-1). After the initial rising-edge of a configuration clock cycle, the XCF128X's synchronous read operation outputs data to the data bus. The XCF128X flash has a data output valid time. The FPGA registers the data from the data bus on the following rising-edge of the configuration clock cycle. The FPGA requires a data setup time. Printed circuit board (PCB) propagation delays can impose addition time requirements.

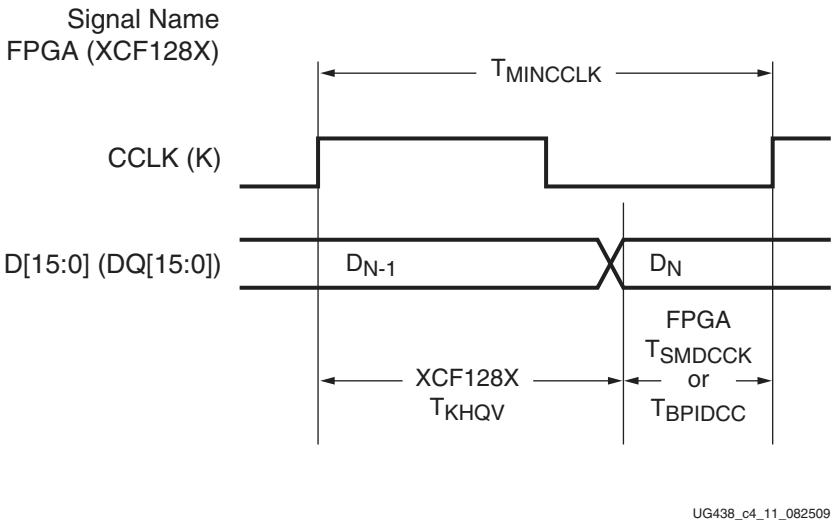


Figure 4-1: **Data Transfer Timing within a Configuration Clock Cycle**

Note relating to Figure 4-1:

1. PCB propagation delay is not shown.

The minimum CCLK period is shown in Figure 4-1 and calculated by Equation 4-1.

$$T_{MINCCLK} = (T_{KHQV} + T_{SETUP} + T_{PCBPROP}) \quad \text{Equation 4-1}$$

where,

T_{KHQV} = XCF128X clock high to output valid time (see DS617, *Platform Flash XL High-Density Configuration and Storage Device*).

T_{SETUP} = SelectMAP data setup time (T_{SMDCCK}) or BPI data setup time (T_{BPIDCC}) (see the associated FPGA data sheet).

$T_{PCBPROP}$ = PCB propagation delays.

The maximum configuration clock frequency can be calculated from the minimum clock period (see Equation 4-2). Equation 4-3 substitutes Equation 4-1 into Equation 4-2.

$$F_{MAXCCLK} = \frac{1}{T_{MINCCLK}} \quad \text{Equation 4-2}$$

$$F_{MAXCCLK} = \frac{1}{(T_{KHQV} + T_{SETUP} + T_{PCBPROP})} \quad \text{Equation 4-3}$$

When $T_{PCBPROP}$ is negligible, Equation 4-3 can be simplified to Equation 4-4.

$$F_{MAXCCLK} = \frac{1}{(T_{KHQV} + T_{SETUP})} \quad \text{Equation 4-4}$$

Slave SelectMAP Maximum Configuration Clock Frequency

For Slave SelectMAP, the maximum frequency for the external configuration clock source is the maximum configuration clock frequency calculated in Equation 4-3 (or Equation 4-4 when PCB propagation delays are negligible).

Master SelectMAP/BPI Maximum Configuration Clock Setting

In Master SelectMAP and Master BPI-Up modes, the FPGA delivers the configuration clock. The FPGA's master configuration clock frequency is set through the BitGen **-g ConfigRate** option. The BitGen **-g ConfigRate** option sets the nominal configuration clock frequency. The FPGA's master configuration clock has a tolerance of $T_{MCCKTOL}$.

Due to the master configuration clock tolerance ($T_{MCCKTOL}$), the BitGen **-g ConfigRate** must be selected so that the high end of the tolerance range does not exceed the maximum configuration clock frequency ($T_{MAXFREQ}$). The BitGen **-g ConfigRate** is set as an integer in MHz. BitGen offers a limited set of valid ConfigRate settings. The maximum BitGen **-g ConfigRate** setting can be calculated using Equation 4-5.

$$MAXCONFIGRATE \leq \left(\frac{\frac{F_{MAXCCLK}}{1\ MHz}}{1 + T_{MAX\_MCCKTOL}} \right) \quad \text{Equation 4-5}$$

where,

$T_{MAX\_MCCKTOL}$ = The maximum percent master configuration clock tolerance ($F_{MCCKTOL}$) (see the associated FPGA data sheet).

$F_{MAXCCLK}$ = The system maximum configuration frequency as calculated in Equation 4-3.

Due to the master configuration clock tolerance ($T_{MCCKTOL}$), the master CCLK can run at frequencies slower than the nominal BitGen **-g ConfigRate** setting. The worst-case master

configuration clock frequency can be calculated for a given BitGen **-g ConfigRate** setting using Equation 4-6.

$$F_{WORSTMASTERCCLK} = CONFIGRATE \times 1\ MHz \times T_{MIN\_MCCKTOL} \quad \text{Equation 4-6}$$

where,

CONFIGRATE = The BitGen **-g ConfigRate** setting.

$T_{MIN\_MCCKTOL}$ = The minimum percent master configuration clock tolerance ($T_{MCCKTOL}$) (see the associated FPGA data sheet).

The worst-case master configuration clock frequency is useful for determining the maximum FPGA configuration time.

Determining Configuration Time

The basic FPGA configuration sequence when using Platform Flash XL is similar for all three modes (Slave SelectMAP, Master SelectMAP, and Master BPI-Up). This configuration sequence begins with a configuration reset that is initiated from a power-on-reset (POR) or other method of configuration reset. Within the configuration reset period, the FPGA clears its configuration memory. Then, the FPGA enables the selected configuration interface, loads the design data (bitstream), and starts running the loaded design. Refer to the FPGA configuration user guide for configuration sequence details.

Although the configuration sequence consists of several logical steps, many of the steps consume negligible time. Two factors overwhelmingly dominate the FPGA configuration time:

- T_{RDY}: The delay required before the FPGA is ready for configuration. This can be due to:
  - The FPGA configuration memory clearing time from an FPGA configuration reset event.
  - The FPGA POR time. This includes FPGA configuration memory clearing time and other POR activities.
- T_{BIT}: The FPGA configuration data load time. This is based on the size of the FPGA bitstream, the configuration bus width, and the clock rate.

The FPGA configuration time formula is given by Equation 4-7:

$$Configuration\ Time = T_{RDY} + T_{BIT} \quad \text{Equation 4-7}$$

For power-on configuration, T_{RDY} is specified in the FPGA data sheet as T_{POR}.

For configuration initiated via a configuration reset event (pulse to PROGRAM_B, issuance of a JPROGRAM JTAG instruction, or issuance of a IPROG ICAP/SelectMAP command), the T_{RDY} time is specified as T_{PL} in the FPGA data sheet.

The FPGA indicates the end of the FPGA POR or configuration reset event by releasing its INIT_B pin. Typically, an external resistor pulls INIT_B High upon release. When INIT_B goes High, the FPGA is ready to begin bitstream loading.

T_{BIT} is the time required to load the FPGA bitstream. Ignoring the brief delay for the FPGA master CCLK to start (T_{ICCK}) and a few cycles for the XCF128X to latch its first read address, the bitstream loading time can be modeled as shown in Equation 4-8.

$$T_{BIT} = \frac{\text{bitstream length in bits}}{\text{16 bits per read}} \times \frac{1}{\text{configuration frequency}} \quad \text{Equation 4-8}$$

See "Determining the Maximum Configuration Clock Frequency," page 45 for the effective clock frequency. For Slave SelectMAP mode, the effective frequency is typically the specified frequency of the external clock source within a few parts-per-million. For master modes, the worst-case CCLK frequency is typically the effective clock frequency for configuration time calculations.

Near the end of the bitstream loading process, the FPGA enters a logical start-up sequence to start the design. The start-up sequence consists of eight phases and can be optionally configured to perform various tasks. The few cycles required to complete the start-up sequence (with or without various options) are insignificant in the configuration time calculation because of the relative dominance of the T_{RDY} and T_{BIT} times.

In summary, many of the considerations for configuration time are trivial compared to the dominating T_{RDY} and T_{BIT} times. Ignoring trivial considerations, the time for configuration at power-up can be calculated using Equation 4-9.

$$T_{POWER\_ON\_CONFIG} = T_{POR} + \frac{\text{bitstream length in bits}}{\dfrac{((16\text{ bits})/\text{cycle})}{\text{configuration frequency}}} \qquad \textit{Equation 4-9}$$

These are some further considerations for calculating configuration times:

- T_{POR} can be found in the FPGA data sheet.
- The effective configuration frequency can be found in "Determining the Maximum Configuration Clock Frequency," page 45 and "Slave SelectMAP Maximum Configuration Clock Frequency," page 46.
- Re-configuration time from an FPGA reset event can be calculated using Equation 4-9 with T_{PL} substituted for T_{POR}.

XILINX®

Chapter 5

Platform Flash XL File Generation

Preparing a Programming File

The FPGA design bitstream (`.bit`) file must be converted to a standard format PROM (`.mcs`) file for programming the Platform Flash XL. The iMPACT programming flow and third-party device programmers require the standard format PROM file as input. The `.bit` file is not a valid data input to the iMPACT programming flow or to the third-party device programmers.

This section details the software flow for creating standard format PROM files for programming the Platform Flash XL. The ISE® software tools, PROMGen (command line) or iMPACT (GUI), generate programming files from the FPGA bitstream.

Before converting an FPGA bitstream to a PROM file format, the designer must verify that the bitstream was generated with the proper start-up clock. The **bitgen -g StartupClk:Cclk** option ensures proper FPGA configuration and functionality by synchronizing the start-up sequence to the FPGA configuration clock (CCLK).

Using the PROMGen Command-Line Software

The PROMGen software takes an FPGA bitstream (`.bit`) file as input and, with the appropriate options, generates a memory image file to program the data array of Platform Flash XL. The output memory image file format is Intel Hex (`.mcs`).

The ISE PROMGen software utility is easily executed from a command-line. A PROMGen software command-line example to generate an MCS-formatted file for a 16 MB (or 128 Mb) Platform Flash XL is shown below:

```
promgen -p mcs -o design.mcs -x xcf128x -data_width 16 -u 0
bitfile.bit
```

The **`-p mcs`** option specifies Intel Hex (`.mcs`) output file format. The **`-o design.mcs`** specifies the output file name to `design.mcs`. The **`-x xcf128x`** specifies the file is to be created for a 128 Mb Platform Flash XL. The **`-u 0`** option specifies the data to start at address zero and fill the data array in the up direction. The `bitfile.bit` is the input FPGA bitstream file used to create the PROM-formatted file.

Using the iMPACT Graphical Software

The iMPACT software integrates PROM file formatting and in-system programming features behind an intuitive graphical user interface. 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. After selecting all of the parameters using the wizard, the final step creates the Platform Flash XL programming file.

The following section demonstrates the iMPACT 11.3 (or later) software process for generating an MCS-formatted file for a 128 Mb Platform Flash XL. This process takes the `bitfile.bit` FPGA bitstream file as input and generates a PROM file named `design.mcs`.

When iMPACT is launched, the iMPACT project dialog box is displayed (Figure 5-1). Choose the **create a new project (.ipf)** option and specify a project location using the **Browse...** button. Then, click **OK** to continue the PROM file generation process with "Step 1: Prepare a PROM File."

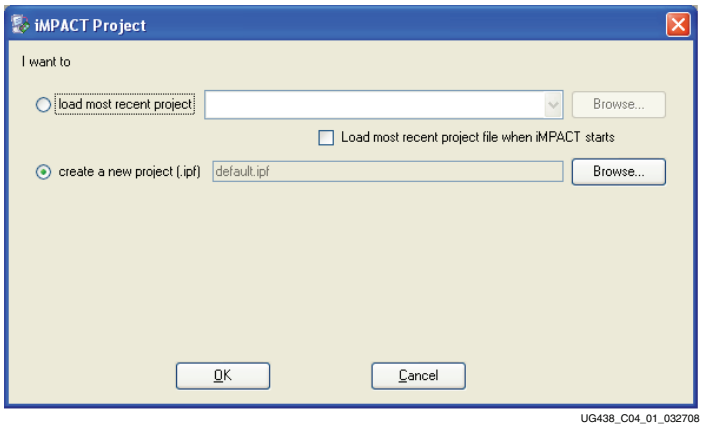


Figure 5-1: **Project Creation in iMPACT**

Step 1: Prepare a PROM File

The first dialog box of the wizard displays the available actions that can be performed (Figure 5-2).

Select **Prepare a PROM File**, and click **OK** to proceed to "Step 2: Specify the BPI Flash Storage for the FPGA Configuration Type" of the process.

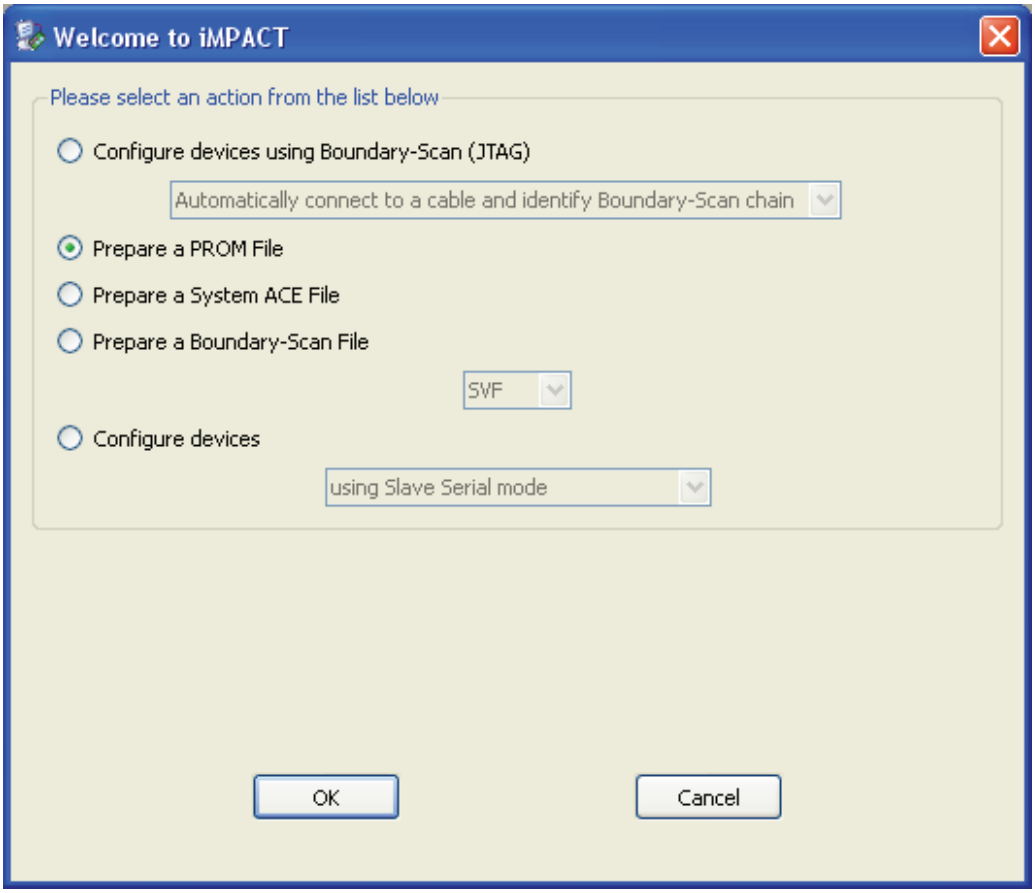
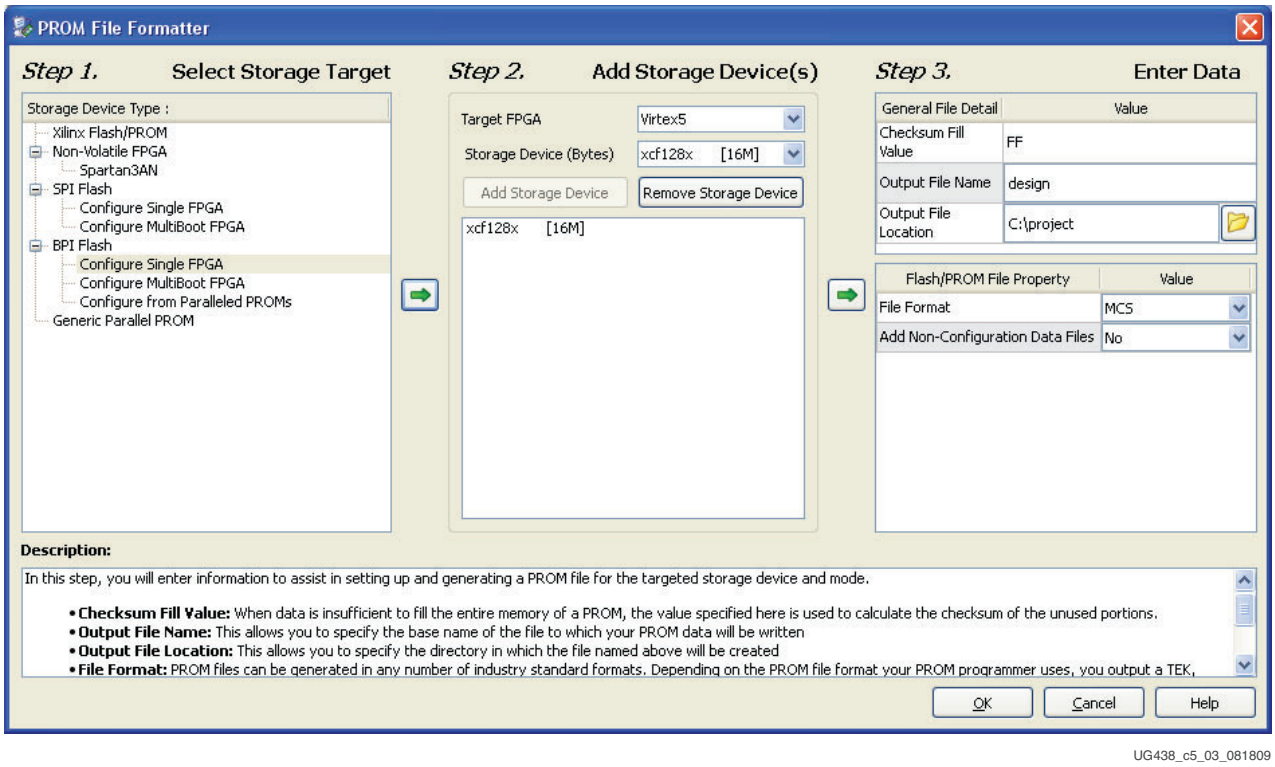


Figure 5-2: **iMPACT Prepares a PROM File Flow**

Step 2: Specify the BPI Flash Storage for the FPGA Configuration Type

For the second step of the process, iMPACT displays the PROM File Formatter dialog box (Figure 5-3). In the Storage Device Type section of the PROM File Formatter dialog box, select the **Configure Single FPGA** item under the **BPI Flash** branch. Press the green arrow button between Step 1 and Step 2 within the dialog box to proceed to "Step 3: Select the XCF128X Device."



UG438_c5_03_081809

Figure 5-3: **Specify PROM Type and Output PROM File Properties**

Step 3: Select the XCF128X Device

In the Add Storage Device(s) section of the PROM File Formatter dialog box, select the **Virtex-5** FPGA as the Target FPGA. Then, select the **xcf128x** as the Storage Device. Press the **Add Storage Device** button to add the XCF128X to the storage device list. Press the green arrow button between Step 2 and Step 3 within the dialog box to proceed to "Step 4: Specify Output PROM File Name and Location."

Step 4: Specify Output PROM File Name and Location

In the Enter Data section of the PROM File Formatter dialog box, specify the output **PROM File Name** and select the output **File Location**. Keep the default File Format of MCS. When the PROM File Formatter entries are satisfactory, press the **OK** button to confirm the PROM type and output file name and location, and proceed to "Step 5: Notification to Add a Device to the PROM File."

Step 5: Notification to Add a Device to the PROM File

After pressing **OK** in the PROM File Formatter dialog box, the iMPACT PROM generation project is set to generate a specific PROM file with the specified parameters.

At this stage in the process, the PROM file memory image is empty. The fifth step in the process is to add an FPGA bitstream to the PROM file memory image. This step begins immediately after completion of the iMPACT PROM File Formatter with an automatic notification that the next step is to add a device file to the Xilinx PROM memory image. Click **OK** in the Add Device notification dialog box (Figure 5-4) to proceed to "Step 6:

Select the FPGA Bitstream File to Add."

Figure 5-4: **Add Device Notification Dialog Box**

Step 6: Select the FPGA Bitstream File to Add

After the Add Device notification, iMPACT opens a file browser to select the FPGA bitstream (`.bit`) file to add to the Platform Flash XL memory image (Figure 5-5).

Select the FPGA bitstream file to be written to the Platform Flash XL image. Click **Open** in the browser to add the target FPGA bitstream. If a single bitstream is to be stored in the device, click **NO** when asked if another design file is to be added (Figure 5-6). Next, click **OK** to complete the automated iMPACT process for preparing XCF128X file to be generated. Upon completion of the process, iMPACT displays a tabular summary of the XCF128X memory map in the MultiBoot BPI Flash Data Stream and Data File Assignment dialog box (Figure 5-7). Press **OK** to accept the default memory map for the given FPGA bitstream. Proceed to "Step 7: Generate File Operation" to generate the Platform Flash XL programming file.

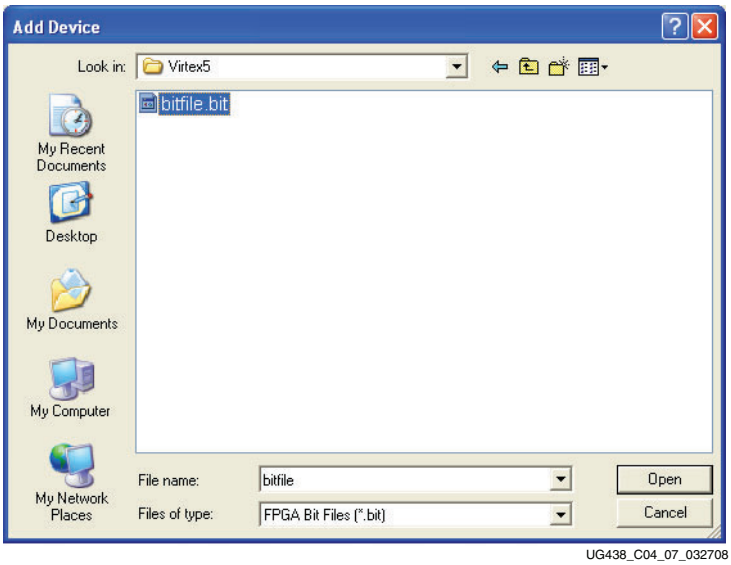


Figure 5-5: **Add Device File Browser**

Figure 5-6: **Add Another Device File to Data Stream**

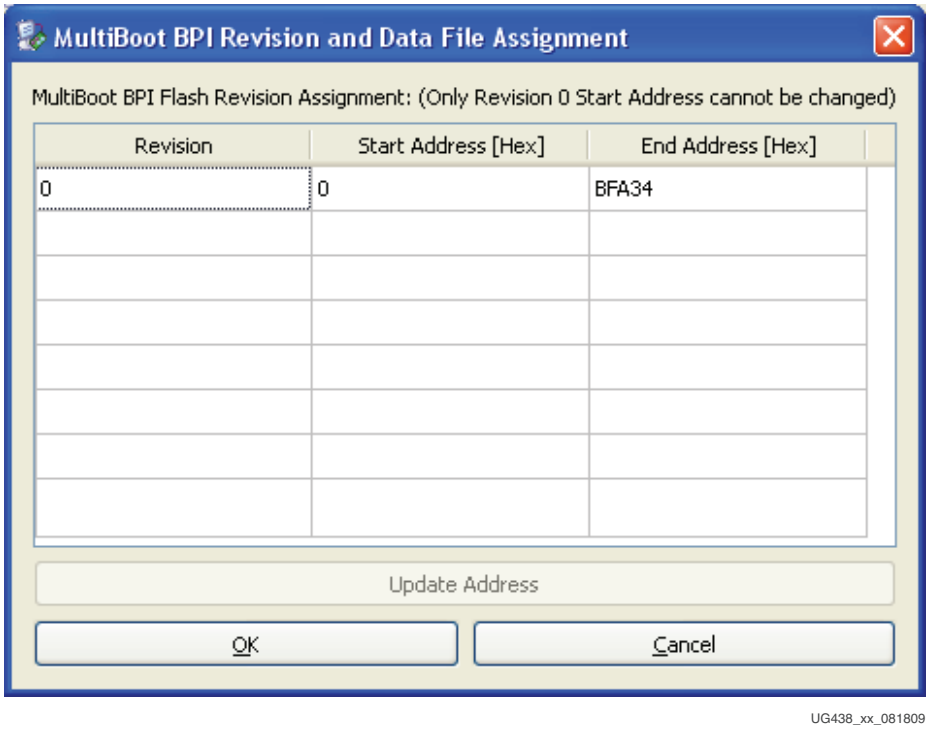


Figure 5-7: **XCF128X Memory Map Summary in Tabular Format**

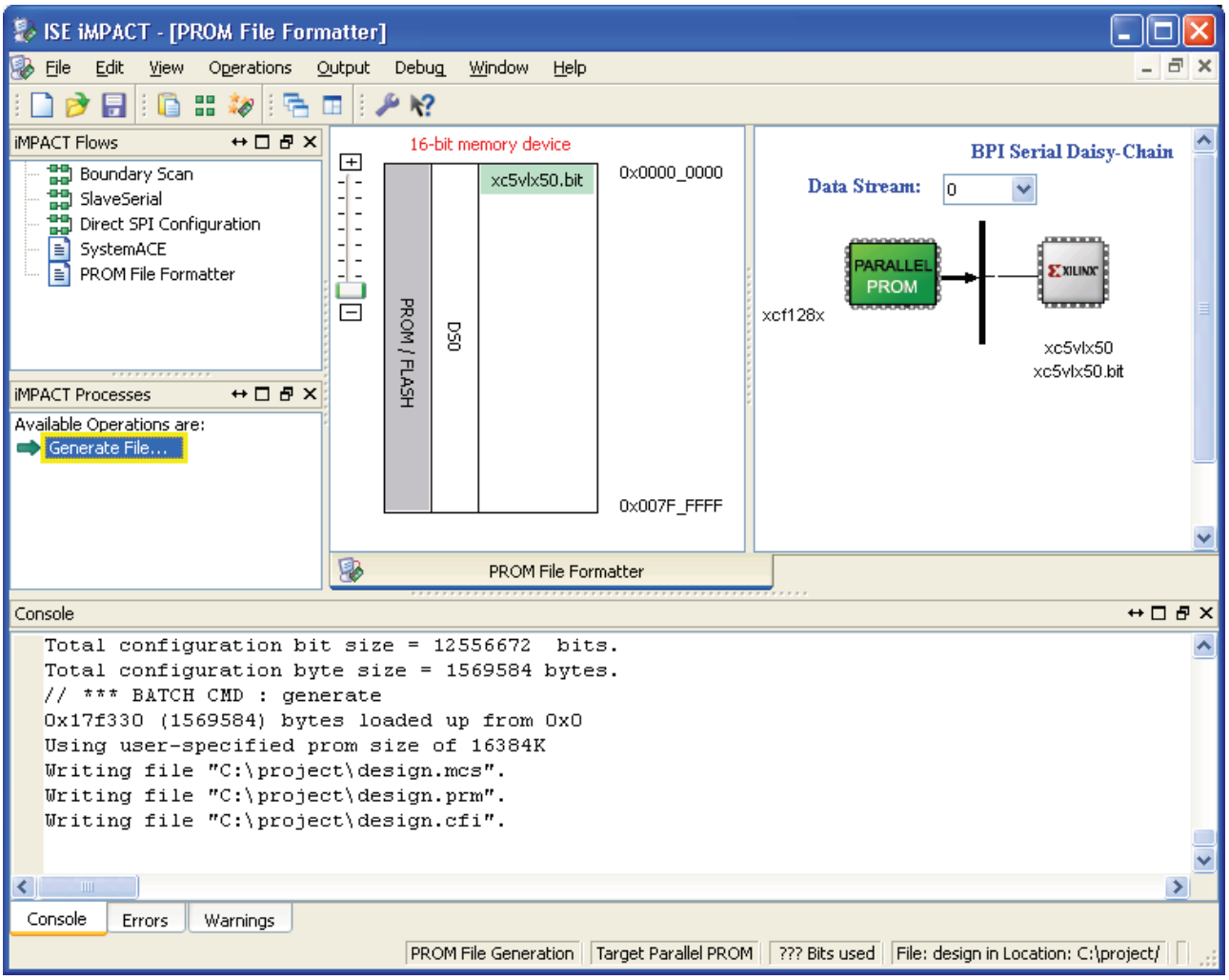
Step 7: Generate File Operation

The seventh and final step in the process is to generate the PROM file.

Double-click on the **Generate File** item in the iMPACT Processes window (Figure 5-8) to generate the specified Platform Flash XL programming file. iMPACT reports a "PROM File Generation Succeeded" message after successful generation of the PROM file. After the Generate File operation completes, the generated `design.mcs` file is available in the user-specified location. The `design.mcs` file can be used in any of the supported programming solutions to program Platform Flash XL with the specified FPGA bitstream contained within the PROM file.

Save the iMPACT PROM generation project for quick regeneration of the PROM file whenever the FPGA bitstream design is revised. To regenerate a Platform Flash XL programming file, re-open the saved iMPACT project, and invoke the **Generate File** operation. iMPACT generates a revised 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 PROM file. The designer is prompted to name the project and select the option **Prepare a PROM File**, following the steps 1-7 outlined above to generate a new PROM File.



UG438_c5_10_081809

Figure 5-8: **Double-Click on Generate File in iMPACT Processes**

XILINX®

Chapter 6

Programming Platform Flash XL

Programming solutions satisfying the requirements for each product phase are available for Platform Flash XL. ISE® software provides integrated programming support for the FPGA design engineer in the prototyping environment. EDK Flashwriter does not support Platform Flash XL programming. Third-party programmer support is also available for the demands of manufacturing environments.

Programming Platform Flash XL During Prototyping

ISE software has integral support for in-system programming, enabling rapid develop-program-and-test cycles for prototyping FPGA designs. The software compiles an FPGA design into a configuration bitstream, converts the bitstream into a PROM file, and then programs the PROM file into Platform Flash XL in-system using a Xilinx® download cable. During the programming process, the iMPACT software downloads a pre-built bitstream containing an in-system programming engine into the FPGA via the JTAG port. Platform Flash XL can then be indirectly programmed by the connected FPGA.

The basic hardware setup required for the iMPACT 11.3 (or later) indirect Platform Flash XL programming method is shown in Figure 6-1.

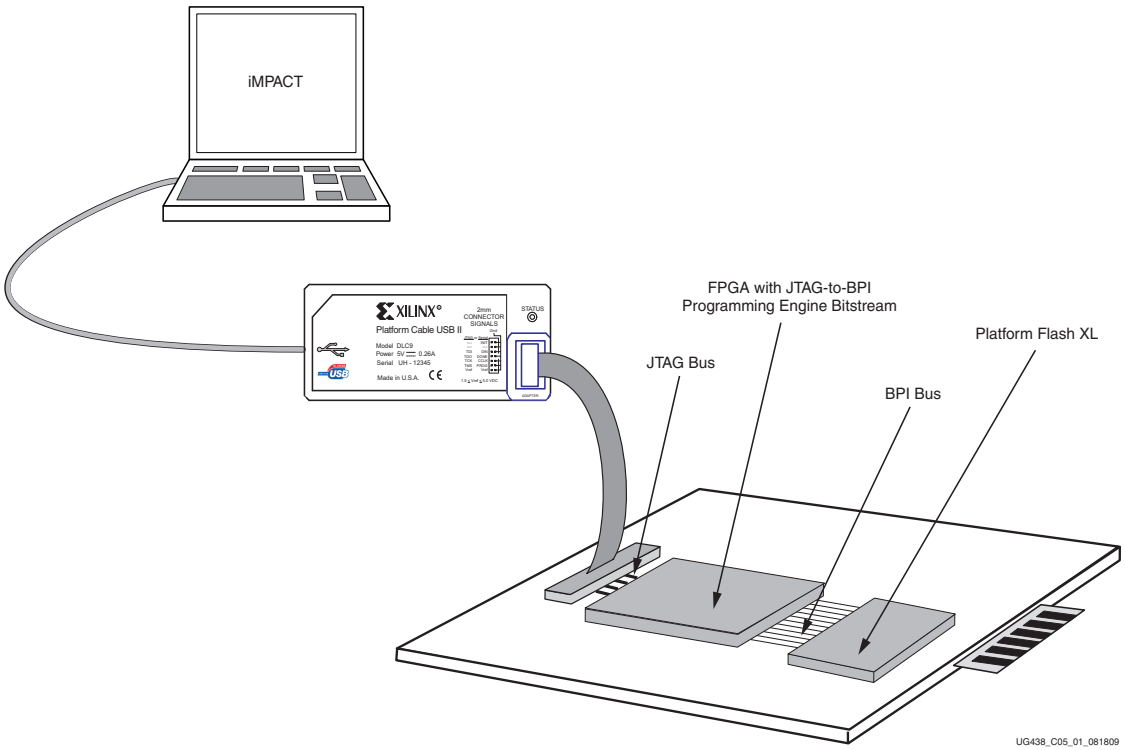


Figure 6-1: **iMPACT Indirect Platform Flash XL Programming with a Virtex-5 FPGA**

Minimum Requirements for Indirect In-System Programming

- Virtex®-5 or Virtex-6 FPGA with JTAG connector for the programming cable
- Platform Flash XL (XCF128X)
- A Xilinx cable
- ISE iMPACT software (ISE 11.3, or later)

The FPGA must be connected to Platform Flash XL following the recommendations in this user guide. Refer to Chapter 2, "High-Speed Configuration" or Chapter 3, "Alternate Configuration Modes" for recommended connections.

Xilinx Cable Connections

Xilinx cables are used with iMPACT to indirectly program Platform Flash XL through the traditional 1149.1 JTAG interface available on the FPGAs. Table 6-1 lists the Xilinx cables that can be used for indirect Platform Flash XL programming using iMPACT.

Table 6-1: **Xilinx Cables Supporting Indirect Platform Flash XL Programming(1)**

Xilinx Cables	Interface
Platform Cable USB, Platform Cable USB-II	USB
Parallel Cable IV	Parallel

Notes:

1. Refer to the specific Xilinx Cable data sheet for additional information.

Before starting the software sequence to program the XCF128X, there are few key hardware checks to perform:

- Proper Xilinx cable connection:

  The Xilinx cable must be properly connected to the computer and to the JTAG bus of the FPGA connected to the target Platform Flash XL (see Figure 6-1 for hardware connections from the Xilinx cable to the JTAG bus of the FPGA).

- Cable power:

  If using the Xilinx Parallel Cable IV, then power must be applied to the cable.

- Target system power:

  Power must also be supplied to the target system containing the FPGA and Platform Flash XL.

- Isolate BPI bus signals from other devices other than the FPGA during the programming process:

  The target FPGA must be allowed to program Platform Flash XL without contention from other devices that might access the memory device (in other words, any other potential master device on the address or data bus or control signals should be isolated).

iMPACT Programming Flow

In prototyping applications, the iMPACT software can be used to program Platform Flash XL in-system with a memory image from a given PROM file (see Chapter 5, "Platform Flash XL File Generation" for instructions on the generation of the PROM file).

The following section demonstrates the iMPACT 11.3 (or later) software process for indirectly in-system programming Platform Flash XL through a Virtex-5 FPGA. The iMPACT software process for a Platform Flash XL with a Virtex-6 FPGA is similar. The process takes the `design.mcs` file (generated in Chapter 5, "Platform Flash XL File Generation") as input, erases the device, programs the file contents into Platform Flash XL, and verifies the device contents against the given PROM file contents.

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

When launching the iMPACT software, the iMPACT Project dialog box is displayed (Figure 6-2). Select the **create a new project (.ipf)** option and (optionally) specify a project location using the **Browse...** button. Then, click **OK** to continue to "Step 2: Configure Devices Using the JTAG-to-BPI Method" in the process.

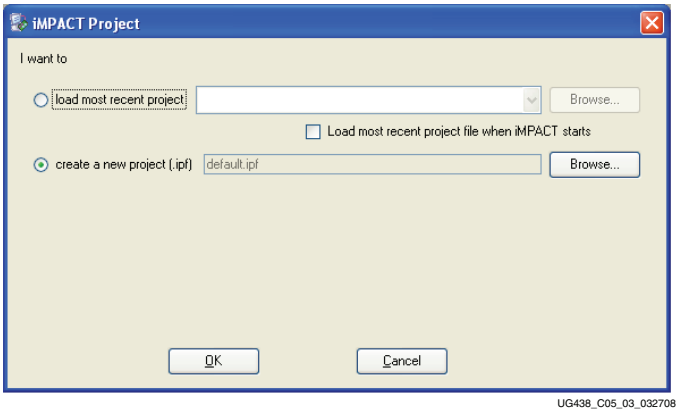


Figure 6-2: **Create a New Project**

Step 2: Configure Devices Using the JTAG-to-BPI Method

The second step of the process begins with the iMPACT project wizard. The first dialog box of the wizard displays the available kinds of projects that can be created (Figure 6-3). Select **Configure devices using Boundary-Scan (JTAG)** and the **Automatically connect to a cable and identify Boundary Scan chain** from the associated drop-down list. Click **Finish** to complete the new project setup process. At the completion of this process, iMPACT is set into a mode for in-system programming using a direct cable connection to the FPGA JTAG bus.

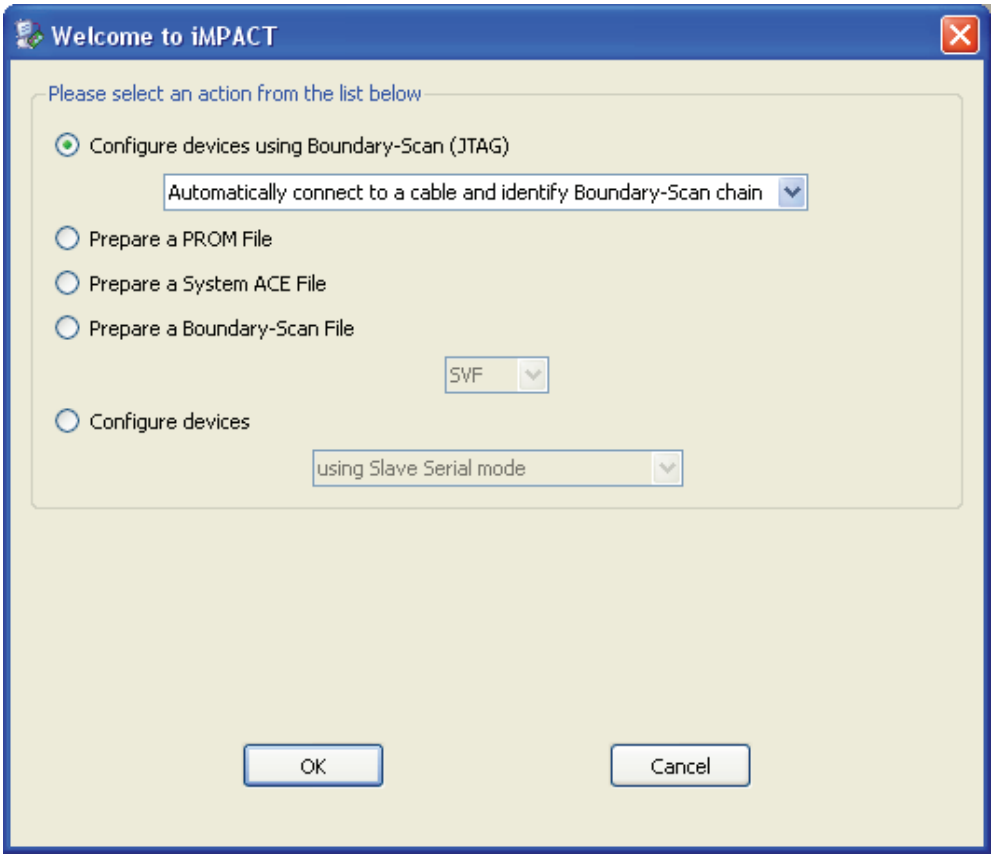
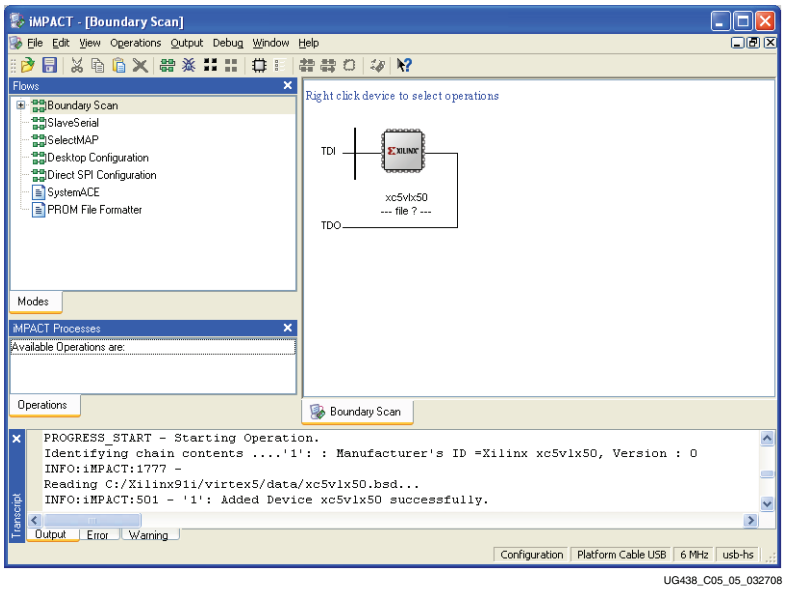


Figure 6-3: **Configure Devices Using Boundary Scan**

After clicking **Finish**, the JTAG chain appears in the iMPACT GUI (Figure 6-4). For this demonstration a single Virtex-5 FPGA exists in the JTAG chain. This Virtex-5 FPGA device is connected to Platform Flash XL via the BPI configuration interface.

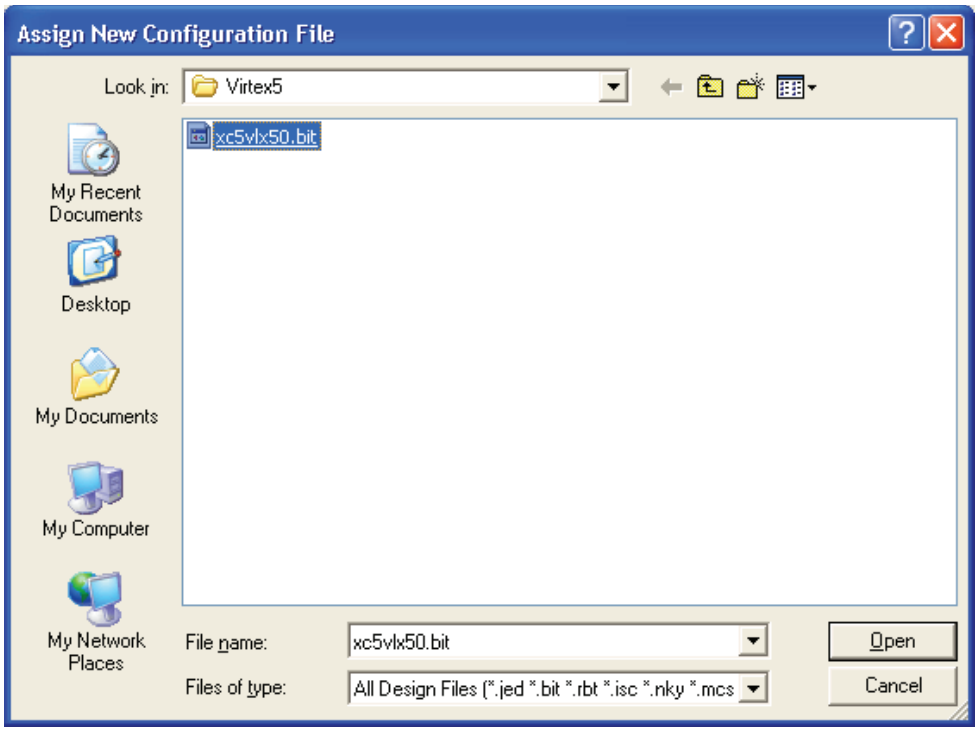


UG438_C05_05_032708

Figure 6-4: **Chain Initialization**

Step 3: Assign the FPGA Configuration File

Select the FPGA bitstream and press the **Open** button (Figure 6-5).



UG438_c6_06_081809

Figure 6-5: **Add a New Configuration File**

Step 4: Add a PROM File for Indirect Programming

After assigning the Virtex-5 FPGA bitstream, the iMPACT Boundary-Scan window shows a Xilinx device icon. The icon represents the Virtex-5 FPGA with an "SPI/BPI" bubble above the icon. Right-click on the **"SPI/BPI"** bubble and select the **Add SPI/BPI Flash** menu item to specify a PROM file. This assigns an XCF128X that is attached to the Virtex-5 FPGA. Browse and select the PROM file for programming into the XCF128X device (Figure 6-6). Choose `design.mcs`, and click **Open**.

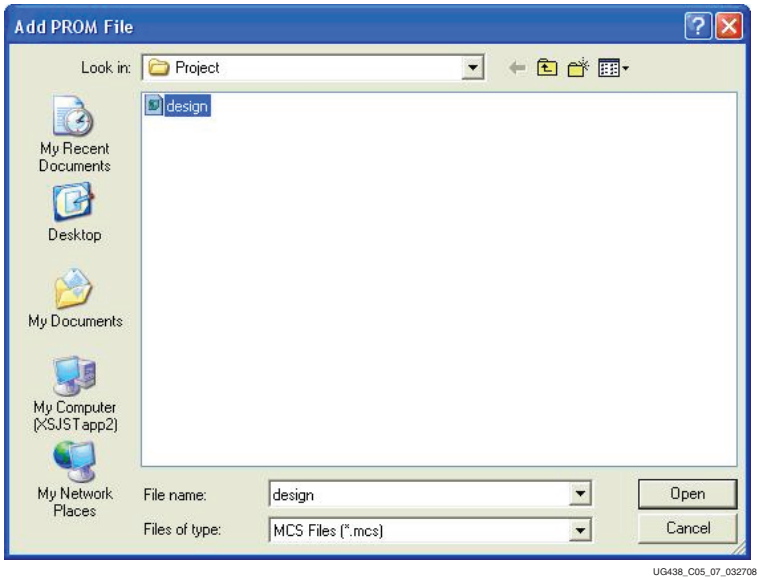


Figure 6-6: **Add a PROM File**

Step 5: Select Xilinx XCF128X Device Part Number

After selecting the PROM file to load, iMPACT displays the Select Attached SPI/BPI dialog box (Figure 6-7). The fifth step of the process requires the target PROM type to be specified in this dialog box. Select **BPI PROM and XCF128X** for the target PROM type used in this demonstration. Click **OK** to complete the PROM programming setup.

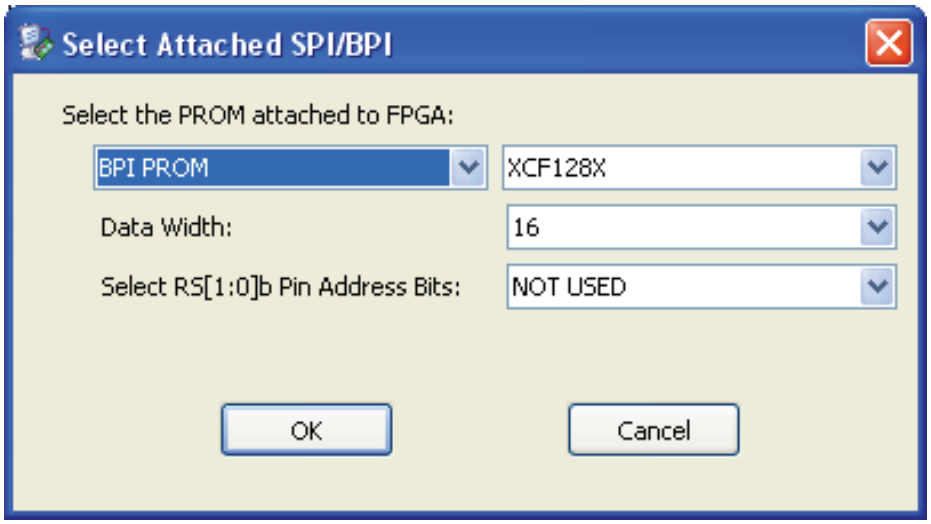


Figure 6-7: **Select a Device Part Name**

Step 6: Invoke the iMPACT Program Operation

The sixth step of the process programs the target device with the selected PROM file contents. Ensure the **PROM icon** in the **iMPACT** window is selected by left-clicking on the **PROM icon** (the PROM icon is highlighted in green when selected). **Select Operations** → **Program** to begin programming (Figure 6-8).

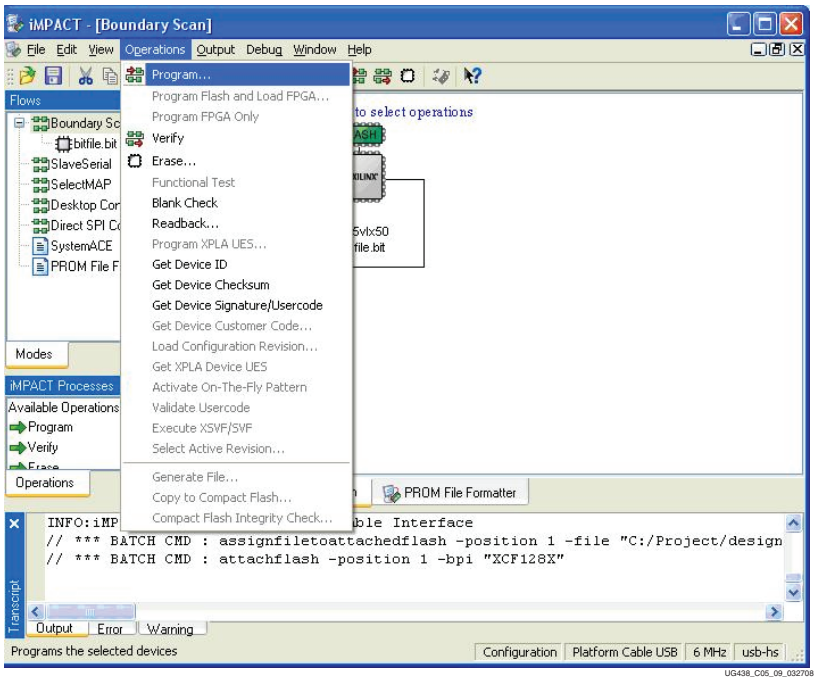


Figure 6-8: **Operations Menu**

Step 7: Select iMPACT Programming Properties

In response to the invocation of the Program operation, iMPACT presents the Programming Properties dialog box (Figure 6-9). The seventh step of the process ensures the selection of proper programming properties. Ensure that **Erase Before Programming** value is checked and the **Verify** value is checked. Click **OK** to begin the erase, program, and verify operations.

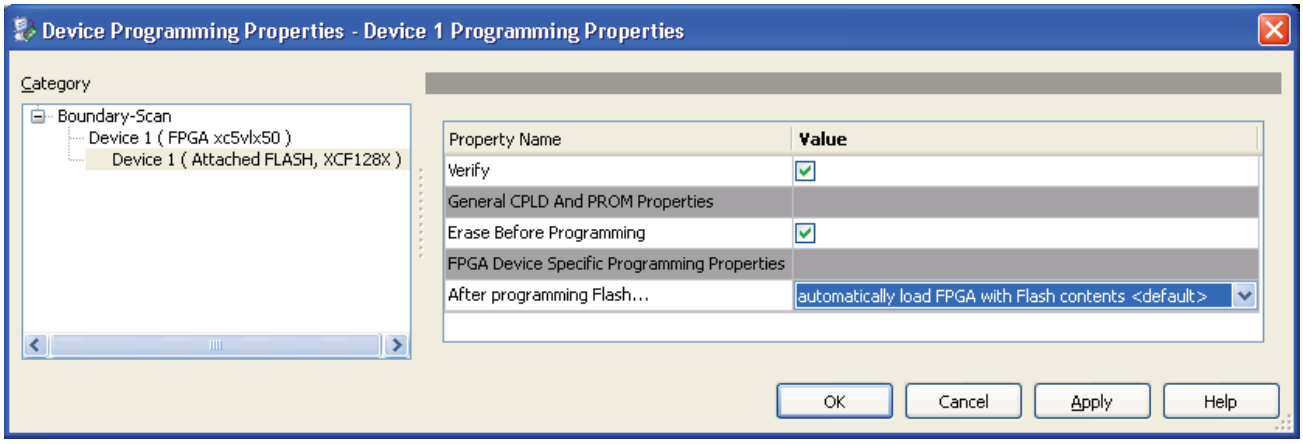


Figure 6-9: **PROM Programming Properties Dialog Box**

Save the iMPACT project for quick reprogramming of the device whenever the PROM file is revised. To reprogram the device, reopen the saved iMPACT project, and invoke the **Program** operation, ensure the selection of the **Erase Programming Property**, and click **OK**. iMPACT reprograms the device, assuming the revised PROM file is located in the same location as the original PROM file.

Expectations

iMPACT Operations and Programming Times

Since Platform Flash XL can combine user data and configuration storage, iMPACT only performs operations on the area bounded by the target device file. This restriction is to prevent other user data from being modified unintentionally. The erase operation for example does not erase the entire device contents, but rather erases only the user area required to store the selected PROM file. In addition, program, verify, and blankcheck all target only the area addressed by the PROM file.

With iMPACT 11.3 (or later) and the Xilinx Platform Cable USB at the default 6 MHz, the user can expect a programming time of approximately 30 seconds per 8 Mb of memory, and a verify time of approximately 15 seconds per 8 Mb of memory. These times are only guidelines because Platform Flash XL operations vary slightly from device to device. In addition, the cable and cable TCK speed selection can be changed in iMPACT, increasing or decreasing this time slightly.

Effect of Indirect Programming on the Rest of the System

When using the JTAG interface to program Platform Flash XL through a Virtex-5 FPGA, the user must understand the behavior of the FPGA during this process and how it can affect other devices in the system. To access Platform Flash XL through the JTAG interface, a Xilinx proprietary JTAG-to-BPI bitstream must be loaded into the FPGA. Loading the JTAG-to-BPI bitstream requires the FPGA to be configured and results in existing logic being lost.

The programming engine core is automatically selected and loaded by iMPACT when an operation is performed on the device. The core processes are preformed in the background and are transparent to the user. However, the DONE status signal is activated whenever the core programming design is loaded for a device operation.

Caution!

- This user guide demonstrates a single FPGA-to-BPI-PROM use case. For daisy-chained FPGA applications, a special procedure can be necessary to enable the JTAG-to-BPI programming engine bitstream in the lead FPGA. In a multiple-FPGA parallel daisy-chain, the DONE pins are tied together across FPGAs. If unconfigured downstream FPGAs are holding DONE Low, first use iMPACT to configure all FPGAs in the daisy-chain with configuration bitstreams via JTAG. After the FPGAs are configured, the downstream FPGAs are not holding DONE Low. iMPACT can then, during the programming process, download and start the indirect JTAG-to-BPI programming engine in the lead FPGA.
- If the user application utilizes the DONE signal status as a flag, the signal is released both when the core design is loaded and then again when the application design is configured into the FPGA from the programmed device.
- For indirect programming iMPACT requires all Platform Flash XL address lines (A[22:0]) to be accessible. If the SelectMAP configuration modes are used then the additional address lines must still be connected to the FPGA although they are unused for configuration. Connecting all address lines allows iMPACT to indirectly program Platform Flash XL via the BPI-UP bus.
- For the Virtex-5 FPGA, the IO_L9P_CC_GC_4 pin must be tied to the Platform Flash XL signal, $\overline{L}$, for indirect programming.
- For the Virtex-6 FPGA, the IO_L18P_24 pin must be tied to the Platform Flash XL's L# pin for indirect programming.
- iMPACT versions prior to 11.3 do not support indirect programming with XCF128X address control through the FPGA RS[1:0] pins.

Pull-Up and Pull-Down Consideration

The designer should ensure that the board's device control signals, such as reset or enable, are tied appropriately on the board and do not rely on the FPGA's internal I/O pull-up or pull-down settings. As well as being good design practice, it is also important because the JTAG-to-BPI programming core's I/O settings can differ from the board's target application I/O requirements. When using the indirect programming method, the FPGA is configured with a JTAG-to-BPI programming core with all unused I/Os set to PULLUP. (All non-dedicated pins that are not included in the set of BPI configuration pins are unused I/Os in the JTAG-to-BPI programming core design.) This I/O setting activates the internal pull-up on all I/Os while the core is loaded. If dictated by system requirements, the user can pull down any I/O, using a 1.1 kΩ resistor. In addition, before the FPGA is configured, FPGA I/Os can be controlled by the HSWAPEN pin. When this pin is held

Low, internal pull-ups on all the I/Os are active. Designers must ensure that the correct HSWAPEN settings are used if any of the FPGA pins are connected to a control signal of any other device.

Caution! Software releases prior to 10.1.01 have the JTAG-to-BPI programming unused I/Os set to PULLDOWN.

Production Programming Solutions

For the requirements of manufacturing environments, solutions exist for programming Platform Flash XL. Check with the third-party vendor for the availability of Platform Flash XL programmer support.

Device Programmers

Device programmers can gang program a high volume of Platform Flash XL devices in a minimal amount of time. Third-party device programmer vendors, such as BPM Microsystems, support programming of Platform Flash XL. See http://www.xilinx.com/support/programr/dev_sup.htm for a sample list of third-party programmer vendors that support Platform Flash XL.

A third-party programmer requires the PROM data in the form of a standard formatted PROM file, such as an MCS data file. See Chapter 5, "Platform Flash XL File Generation," for instructions on generating a standard PROM file that is properly formatted for a third-party programmer.

XILINX®

Chapter 7

System Considerations

Platform Flash XL VDDQ Power Budget

The Platform Flash XL data sheet specifies maximum current draw for the V_{DD} core voltage supply and V_{PP} programming voltage supply under various internal operating conditions. The data sheet does not specify maximum current draw for the V_{DDQ} I/O voltage supply because the current draw for the I/O voltage supply is dependent on factors external to the Platform Flash XL. The I/O voltage supply is strongly dependent on the I/O switching frequency and on the output (DQ[15:0]) loads. The output loads are unique to each board due to board design, board construction, and schematic connectivity.

Table 7-1 provides V_{DDQ} power budget estimates from a real example test board. The test board contains one Platform Flash XL connected to one Virtex-5 FPGA as shown in the Slave SelectMAP (x16) schematic in Figure 2-1, page 16. Current measurements are taken from an isolated supply to the Platform Flash XL VDDQ at room temperature. To provide margin for the accuracy of the current measurement, part variation, and temperature variation, the entries in Table 7-1 are at least twice the actual measurements.

Table 7-1: **Platform Flash XL Power Supply Budget Estimates for VDDQ**

	VDDQ Power Supply Budge for Different K (Clock) Frequencies							
VDDQ (V)	**5 MHz (mA)**	**10 MHz (mA)**	**20 MHz (mA)**	**30 MHz (mA)**	**33 MHz (mA)**	**40 MHz (mA)**	**50 MHz (mA)**	**Simple Recommendation for Each VDDQ Voltage Level (mA)**
3.3V	19	41	80	123	137	160	200	200
2.5V	9	26	51	81	88	102	137	140

XILINX®

Chapter 8

FPGA User Design Recommendations

On a typical board design, most of the Platform Flash XL pins are connected to dual-purpose or standard user I/O pins of the FPGA. During FPGA configuration, the FPGA can drive many of the dual-purpose pins to read the configuration bitstream. After FPGA configuration, the dual-purpose pins become user-configured I/Os. The FPGA design determines the configuration of the dual-purpose or standard user I/Os that are connected to Platform Flash XL. The user's design configuration of these FPGA pins can affect the behavior of Platform Flash XL.

Appropriate FPGA design constraints are required to ensure a well-behaved Platform Flash XL after the FPGA is configured. The appropriate constraints depend on the FPGA design's usage of Platform Flash XL.

FPGA Designs Not Accessing Platform Flash XL after Configuration

When the FPGA design does not need to access Platform Flash XL for reading or writing, FPGA design constraints are recommended that keep Platform Flash XL in a disabled state. Disabling Platform Flash XL puts it in a standby mode that minimizes power consumption, avoids unexpected signal activity and corresponding noise, and allows for potential reuse of a few dual-purpose pins.

The recommended design constraints for FPGA pins connected to Platform Flash XL pins are shown in Table 8-1. Driving the FPGA FCS_B pin High to disable the Platform Flash XL $\overline{E}$ (chip enable) pin is critical. The remaining constraints are recommendations for disabling secondary Platform Flash XL control pins or for ensuring valid logic levels in different internal and external resistor scenarios. For example, FWE_B is typically tied to an external pull-up resistor, and the external pull-up resistor can compete with the FPGA's internal pull-down resistor. If FWE_B is not defined in the FPGA design, FWE_B is an unused I/O. BitGen, by default, enables the FPGA pin's internal pull-down resistor on unused I/Os.

Table 8-1: **FPGA Pin Constraints When the FPGA Design Does Not Access Platform Flash XL**

Platform Flash XL Pin	FPGA Pin	FPGA Pin Constraints	Description
$\overline{E}$	FCS_B	Drive High (Required)	Active-Low chip enable. Disables Platform Flash XL.
$\overline{G}$	FOE_B	Drive High (Recommended)	Active-Low output enable. Disables the Platform Flash XL data output.

Table 8-1: **FPGA Pin Constraints When the FPGA Design Does Not Access Platform Flash XL *(Cont'd)***

Platform Flash XL Pin	FPGA Pin	FPGA Pin Constraints	Description
$\overline{W}$	FWE_B	Drive High (Recommended)	Active-Low write enable. Disables the Platform Flash XL write function.
$\overline{L}$	IO_L9P_CC_GC_4 (Virtex-5 FPGA) or IO_L18P_24 (Virtex-6 FPGA)	Drive High (Recommended)	Active-Low address latch enable. Disables the Platform Flash XL address latch.

Review the pin descriptions in Table 2-1, page 19 or Table 3-1, page 36 for additional FPGA design considerations that can apply to the specific use of the FPGA with the Platform Flash XL.

Re-Using Configuration Pins for Other Purposes

When the FPGA design does not access the Platform Flash XL, some of the configuration pins can be reused to communicate with other devices on the board. For example, FPGA pins D[15:0] or A[25:0] might be available for multi-purpose. To avoid conflicts when reusing the pins, check the reference schematic figures and pin tables in Chapter 2, "High-Speed Configuration" or Chapter 3, "Alternate Configuration Modes" to determine the behavior of the FPGA pins during configuration and indirect programming. Also see Table 8-1 for pin requirements of the FPGA design.

FPGA Designs Accessing Platform Flash XL after Configuration

When the FPGA design needs read or write access to Platform Flash XL, FPGA design considerations are recommended that enable Platform Flash XL for asynchronous read and write access.

Platform Flash XL operates in one of two possible read modes: synchronous and asynchronous. At power-on or upon a reset through the RP pin, Platform Flash XL is put into synchronous read mode. This mode enables optimal performance for downloading a bitstream and configuring the FPGA. For FPGA configuration, Platform Flash XL is expected to be in synchronous read mode. User-level read and write access to Platform Flash XL is typically achieved through asynchronous read mode.

FPGA designs needing user-level read or write access to Platform Flash XL typically must consider the implementation of special functions and pin constraints for managing the read mode of Platform Flash XL.

FPGA Design Preparation for Asynchronous Read Mode Access

Before accessing Platform Flash XL for user-level reads or writes, the FPGA design must perform a one-time setup that involves:

- Setting up a few FPGA pins to be compatible with Platform Flash XL's asynchronous read mode interface (see Table 8-2)
- Explicitly setting the Platform Flash XL configuration register for asynchronous read mode operation

Table 8-2: **FPGA Pin Constraints When the FPGA Design Accesses Platform Flash XL**

Platform Flash XL Pin	FPGA Pin	FPGA Pin Constraints	Description
$\overline{E}$	FCS_B	Drive Low as needed by the application (Required)	Active-Low chip enable. Enables Platform Flash XL.
$\overline{L}$	IO_L9P_CC_GC_4 (Virtex-5 FPGA) or IO_L18P_24 (Virtex-6 FPGA)	Drive Low (Required)	Active-Low address latch enable. Enables the Platform Flash XL address latch for asynchronous read or write access.

Review the pin descriptions in Table 2-1, page 19 or Table 3-1, page 36 for additional FPGA design considerations that can apply to the specific use of the FPGA with the Platform Flash XL.

To set Platform Flash XL to asynchronous read mode, a Set Configuration Register command must be applied from the FPGA design to Platform Flash XL with the Configuration Register bit 15 (CR15) set to `1`. The Set Configuration Register command is a two write-cycle operation. CR15 must be set to `1` for asynchronous read mode and all remaining Configuration Register bits are recommended to be set to their default values. See DS617, *Platform Flash XL High-Density Configuration and Storage Device* for the Set Configuration Register command, Configuration Register bit values, write AC waveforms, and timing.

FPGA Design Preparation for FPGA Reconfiguration

The SelectMAP and Master BPI-Up configuration mode setups described in Chapter 2, "High-Speed Configuration" and Chapter 3, "Alternate Configuration Modes" expect Platform Flash XL to be in synchronous read mode.

If the FPGA design can trigger a reconfiguration of the FPGA from Platform Flash XL (for example, using the FPGA MultiBoot feature), the FPGA design must return the Platform Flash XL configuration to synchronous read mode before triggering the FPGA reconfiguration process.

To set Platform Flash XL to synchronous read mode, a Set Configuration Register command must be applied from the FPGA design to Platform Flash XL with CR15 set to `0`. The Set Configuration Register command is a two write-cycle operation. All Configuration Register bits are recommended to be set to their default values, including the default value of `0` for CR15. See *Platform Flash XL High-Density Configuration and Storage Device* for the Set Configuration Register command, Configuration Register bit values, write AC waveforms, and timing.

Alternatively, external configuration control logic can simply apply a reset pulse to simultaneously initiate FPGA reconfiguration and reset Platform Flash XL to synchronous read mode. The FPGA configuration setups shown in Figure 2-1 and Figure 3-1 connect the Platform Flash XL RP (reset) pin to the FPGA PROGRAM_B pin. Thus, a reset pulse to these connected pins simultaneously initiates FPGA reconfiguration and resets Platform Flash XL to its default synchronous read mode.