Virtex-5 FPGA Configuration User Guide

UG191 (v3.11) October 19, 2012

XILINX®

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
04/14/06	1.0	Initial Xilinx release.
05/12/06	1.1	Minor typographical edits to entire document to improve clarity. Chapter 1: Revised Table 1-1, Table 1-2, Table 1-9, Table 1-11. Chapter 2: Clarified bit-swapping (Byte Swap rule) throughout. Updated “Page Mode Support,” page 71. Chapter 8: Added new section called “MultiBoot Bitstream Spacing,” page 155. Chapter 9: Added “Post_CRC Constraints” section.

Date	Version	Revision
07/31/06	1.2	Chapter 1: • Moved "Configuration Data File Formats" and "Generating PROM Files" from Chapter 2, and "Bitstream Overview"from Chapter 6 into this chapter to consolidate all the data configuration file information. • Updated sentence in"Device Power-Up (Step 1)," page 24 to say "All dedicated input pins operate at V_{CC_CONFIG} LVCMOS level. " • Moved "Packet Types" section from Chapter 1 to Chapter 6. • Moved "Bit Swapping" and "Parallel Bus Bit Order" sections under "Generating PROM Files" section. • Created "Configuration Sequence" header and moved "Setup (Steps 1-3)," "Bitstream Loading (Steps 4-7)," and "Startup (Step 8)" sections under it. Chapter 2: • Added Note 2 to Table 2-1. Revised RCMD in Table 2-8. • Moved "Board Layout for Configuration Clock (CCLK)" to after "Byte Peripheral Interface Parallel Flash Mode." • Replaced Clock Management Technology (CMT) with Digital Clock Managers (DCM) throughout user guide. Chapter 7: • Revised the paragraph above Table 7-9.
09/06/06	2.0	Updated Table 1-2, Table 1-4, and Table 1-13.
10/12/06	2.1	Chapter 1: Added XC5VLX85T information to Table 1-4 and Table 1-13. Updated the Program Latency value in Table 1-10. Updated the Program Latency value in Table 1-10. Chapter 3: Updated the V_{CCINT} value in Figure 3-6. Chapter 6: Updated Figure 6-1 and Table 6-7 to include system monitor pin information. Chapter 9: Added partial reconfiguration application information to Chapter 9, "Readback CRC."
02/01/07	2.2	Updated Table 1-4, page 18 (added LX220T and SXT devices), Table 1-13, page 29 (added LX220T and SXT devices), "ABORT Status Word," page 59, Table 6-5, page 113 (CBC value), "Configuration Memory Read Procedure (SelectMAP)," page 138 (steps 13 and 14), and Table 6-8, page 118 (block type).
07/05/07	2.3	Updated: Figure 2-3, notes relevant to Figure 2-4, "Guidelines and Design Considerations for Serial Daisy Chains," notes related to Figure 2-21 and Figure 2-22, "Configuration Memory Frames" (including Table 6-1), Table 6-10, Figure 6-9, "Configuration Memory Read Procedure (SelectMAP)," Table 7-2, and Table 7-6.
07/31/07	2.4	General: Updated PROG_B to PROGRAM_B throughout user guide. Chapter 1: Updated "Check Device ID (Step 5)" section, including Table 1-13. Chapter 2: Updated Figure 2-13. Chapter 3: Updated Table 3-3, added state descriptions for TAP controller. Chapter 4: Updated "STARTUP_VIRTEX5" section.
10/10/07	2.5	Chapter 1: Updated "Configuration Modes and Pins." Chapter 2: Updated Figure 2-17 and "Serial Daisy Chains" and "SelectMAP Reconfiguration" sections. Chapter 4: Updated Table 4-3. Chapter 5: Updated "Changing the Multiply and Divide Values." Chapter 6: Updated Table 6-7. Chapter 7: Updated Table 7-1 and relevant text. Updated Table 7-2.

Date	Version	Revision
12/11/07	2.6	Updated legal disclaimer and copyright information. Chapter 1: Updated Table 1-3, Table 1-4, Table 1-10, and Table 1-13. Chapter 2: Updated notes that are relevant to Figure 2-22 and Table 2-9. Chapter 4: Updated the "ICAP_VIRTEX5" section. Chapter 6: Updated Table 6-1. Chapter 8: Updated section name from Fallback Reconfiguration to "Fallback MultiBoot."
02/01/08	2.7	Minor text edits throughout user guide. Chapter 1: Updated the M2, M1, and M0 mode pins setting information in "Configuration Modes and Pins." Also updated the "Device Power-Up (Step 1)" section. Chapter 2: Updated Figure 2-23. Chapter 3: Updated the "Boundary-Scan for Virtex-5 Devices Using IEEE Standard 1149.1" section.
03/31/08	3.0	Updated Preface. Added FXT devices. Chapter 1: Updated Table 1-3 and Table 1-13. Chapter 3: Updated the "Boundary-Scan for Virtex-5 Devices Using IEEE Standard 1149.1" and "Instruction Register"sections. Updated Table 3-2 and Table 3-3. Chapter 6: Updated Table 6-1.
04/25/08	3.1	Chapter 1: Updated Table 1-4 and Table 1-13. Chapter 2: Updated "SelectMAP Reconfiguration" and notes pertaining to Figure 2-22, page 67. Chapter 6: Updated Table 6-1 and "Block RAM Contents." Chapter 7: Updated Table 7-3.
07/11/08	3.2	Chapter 1: Updated Table 1-4 notes, "PROM Files for SelectMAP Configuration," and "Power-On Sequence Precautions." Chapter 2: Added the "High-Performance Platform Flash XL SelectMAP Configuration" section, including Figure 2-7 and Figure 2-8, and "Power-On Sequence Precautions." Updated: Figure 2-5 and Figure 2-9. Updated notes related to Figure 2-3, Figure 2-4, Figure 2-9, Figure 2-13, and Figure 2-20. Updated "Byte Peripheral Interface Parallel Flash Mode," including Figure 2-22 and related notes. Chapter 3: Updated Table 3-4. Chapter 4: Updated Table 4-4. Chapter 6: Updated Table 6-1, Table 6-11. Chapter 8: Updated "Fallback Overview."
09/03/08	3.3	Chapter 1: Updated "Loading the Encryption Key," Chapter 3: Updated Table 3-3. Chapter 4: Updated "FRAME_ECC_VIRTEX5." Chapter 5: Updated Table 5-2. Chapter 6: Updated "Command Register (CMD)." Chapter 9: Updated introductory paragraphs in "Readback CRC," and "Post_CRC Constraints."

Date	Version	Revision
09/23/08	3.4	Chapter 1: Added the TXT platform to Table 1-4, Table 1-13. Chapter 2: Updated Figure 2-9, Figure 2-12 and Figure 2-15. Chapter 6: Added the TXT platform to Table 6-1.
10/29/08	3.5	Chapter 2: Updated notes on Figure 2-9. Chapter 4: Updated "STARTUP_VIRTEX5." Chapter 8: Added cross reference to "IPROG Embedded in the Bitstream."
02/11/09	3.6	Chapter 1: Updated Table 1-2 and the "Sample Mode Pins (Step 3)" section. Chapter 2: Updated notes relating toFigure 2-22. Chapter 3: Updated the "Multiple Device Configuration"section. Chapter 6: Updated Table 6-8. Updated ISC_PROGRAM_SECURITY to ISC_PROGRAM_KEY.
06/24/09	3.7	Chapter 1: Updated Table 1-3 to indicate the `BIN` file format is bit swapped. Updated Table 1-5 by changing the bit order of the data words defined in the header for each row. Updated first paragraph, fourth sentence in "Bit Swapping". Chapter 4: Updated definition of O[31:0] pin in Table 4-3. Chapter 6: Removed CRCC command from Table 6-6. Chapter 7: Updated step 9 of "Configuration Memory Read Procedure (SelectMAP)." Added "NOOP Omitted" label to Figure 7-2. Updated Table 7-2 to conform to changes in "Configuration Memory Read Procedure (SelectMAP)."
08/14/09	3.8	Chapter 1: Added new paragraph to "Bus Width Auto Detection" with an example of the Sync word bit order at the FPGA pins. Updated Table 1-5 by changing the bit order of the data words defined in the header for each row. Updated first paragraph in "Sync Word." Chapter 4: Updated the ECCERROR and the SYNDROME[11:0] descriptions in Table 4-4. Chapter 6: Updated Table 6-6 by replacing the Reserved command with the CRCC command. Chapter 7: Updated "Configuration Register Read Procedure (SelectMAP)" step 1 and step 3. Updated "Configuration Memory Read Procedure (SelectMAP)" step 8 and the last paragraph. Updated step 9 in Table 7-2. Updated Note 1 in Table 7-3. Chapter 9: Updated the Readback CRC logic running conditions in the second paragraph under "Readback CRC."
08/03/10	3.9	Updated Table 1-3 to indicate that BIN is not bit swapped with the BitGen tool, but is bit swapped with the PROMGen tool. Corrected the order of rows in Table 1-16 (placed phase 4 row above phase 5 row). Updated Figure 1-12 (timing of GWE and GTS). Changed "the external configuration interfaces" to "any configuration interface" in the first sentence of the fourth paragraph under "Loading Encrypted Bitstreams". Updated Figure 2-12 by adding PROG inputs to the three FPGA devices. Added Note 10 to "Notes relevant to Figure 2-12". Added second sentence about the persist option selection to second paragraph under "SelectMAP Reconfiguration". Added sentence to definitions of DEN and DWE in Table 5-1 indicating that they should only be pulsed for one DCLK cycle. Updated description of START_ADDR in Table 6-12. Updated Table 6-15 (inserted three rows). Updated second paragraph under "Fallback Overview" indicating that indirect BPI programming is supported with iMPACT version 11.3 and later when using the RS pin. Added sentence to first paragraph in Chapter 9, "Readback CRC" indicating that if a CRC error is detected, the device must be reconfigured.
08/20/10	3.9.1	Fixed corrupt file.

Date	Version	Revision
11/18/11	3.10	Chapter 2: After Figure 2-21 and Figure 2-24, updated reference for supported flash devices. Chapter 5: In first row of Table 5-3, updated Function from "N/A" to "Divide by 1." Updated the last sentence in the chapter regarding DRP access. Chapter 7: Updated Figure 7-2. In the paragraph before Table 7-3, updated number of dummy words from 82 to 42.
10/19/12	3.11	Chapter 1: Added note 3 to Table 1-2. Chapter 2: Added note 10 after Figure 2-5.

Table of Contents

Chapter 3: Boundary-Scan and JTAG Configuration

About This Guide

This document describes Virtex®-5 FPGA configuration. Complete and up-to-date documentation of the Virtex-5 family of FPGAs is available on the Xilinx website at http://www.xilinx.com/virtex5.

Guide Contents

This manual contains the following chapters:

Additional Documentation

The following documents are also available for download at http://www.xilinx.com/virtex5.

- Virtex-5 Family Overview

  The features and product selection of the Virtex-5 family are outlined in this overview.
- Virtex-5 FPGA Data Sheet: DC and Switching Characteristics

  This data sheet contains the DC and Switching Characteristic specifications for the Virtex-5 family.
- Virtex-5 FPGA User Guide

  Chapters in this guide cover the following topics:
  - Clocking Resources
  - Clock Management Technology (CMT)
  - Phase-Locked Loops (PLLs)
  - Block RAM
  - Configurable Logic Blocks (CLBs)

  - SelectIO™ Resources
  - SelectIO Logic Resources
  - Advanced SelectIO Logic Resources
- Virtex-5 FPGA RocketIO GTP Transceiver User Guide

  This guide describes the RocketIO™ GTP transceivers available in the Virtex-5 LXT and SXT platforms.
- Virtex-5 FPGA RocketIO GTX Transceiver User Guide

  This guide describes the RocketIO GTX transceivers available in the Virtex-5 FXT and TXT platforms.
- Virtex-5 FPGA Tri-Mode Ethernet Media Access Controller

  This guide describes the dedicated Tri-Mode Ethernet Media Access Controller available in the Virtex-5 LXT, SXT, FXT, and TXT platforms.
- Virtex-5 FPGA Integrated Endpoint Block User Guide for PCI Express Designs

  This guide describes the integrated Endpoint blocks in the Virtex-5 LXT, SXT, FXT, and TXT platforms used for PCI Express® designs.
- XtremeDSP Design Considerations

  This guide describes the XtremeDSP™ slice and includes reference designs for using DSP48E slices.
- Virtex-5 FPGA System Monitor User Guide

  The System Monitor functionality available in all the Virtex-5 devices is outlined in this guide.
- Virtex-5 FPGA Packaging and Pinout Specifications

  This specification includes the tables for device/package combinations and maximum I/Os, pin definitions, pinout tables, pinout diagrams, mechanical drawings, and thermal specifications.
- Virtex-5 FPGA PCB Designer's Guide

  This guide provides information on PCB design for Virtex-5 devices, with a focus on strategies for making design decisions at the PCB and interface level.
- Virtex-5 FPGA Embedded Processor Block for PowerPC® 440 Designs

  This reference guide is a description of the embedded processor block available in the Virtex-5 FXT platform.
- Platform Flash XL High-Density Storage and Configuration Device

  Platform Flash XL is the industry's highest performing configuration and storage device and is specially optimized for high-performance Virtex-5 FPGA configuration and ease-of-use.
- Platform Flash XL User Guide

  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 to indirectly in-system program the device.

Additional Resources

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.

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`**}

Convention	Meaning or Use	Example

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 files.

XILINX®

Chapter 1

Configuration Overview

Configuration Modes and Pins

Virtex®-5 devices are configured by loading application-specific configuration data—the bitstream—into internal memory. Because Xilinx FPGA configuration memory is volatile, it must be configured each time it is powered-up. The bitstream is loaded into the device through special configuration pins. These configuration pins serve as the interface for a number of different configuration modes:

- Master-serial configuration mode
- Slave-serial configuration mode
- Master SelectMAP (parallel) configuration mode (x8 and x16 only)
- Slave SelectMAP (parallel) configuration mode (x8, x16, and x32)
- JTAG/Boundary-Scan configuration mode
- Master Serial Peripheral Interface (SPI) Flash configuration mode
- Master Byte Peripheral Interface Up (BPI-Up) Flash configuration mode (x8 and x16 only)
- Master Byte Peripheral Interface Down (BPI-Down) Flash configuration mode (x8 and x16 only)

The configuration modes are explained in detail in Chapter 2, "Configuration Interfaces." The specific configuration mode is selected by setting the appropriate level on the dedicated Mode input pins M[2:0]. The M2, M1, and M0 mode pins should be set at a constant DC voltage level, either through pull-up or pull-down resistors, or tied directly to ground or V_{CC_CONFIG}. The mode pins should not be toggled during and after configuration. See Table 2-1, page 37 for the mode pin setting options.

The terms *Master* and *Slave* refer to the direction of the configuration clock (CCLK):

- In Master configuration modes, the Virtex-5 device drives CCLK from an internal oscillator. To get the desired frequency, **`BitGen -g ConfigRate`** is used. The "BitGen" section of the *Development System Reference Guide* provides more information. After configuration, the CCLK is turned off unless the **`persist`** option is selected or SEU detection is used. The CCLK pin is 3-stated with a weak pull-up.
- In Slave configuration modes, CCLK is an input.

The JTAG/Boundary-Scan configuration interface is always available, regardless of the Mode pin settings. The JTAG/Boundary-Scan configuration mode disables all other configuration modes to prevent conflicts between configuration interfaces.

Certain pins are dedicated to configuration (Table 1-1), while others are dual-purpose (Table 1-2). Dual-purpose pins serve both as configuration pins and as user I/O after configuration. Dedicated configuration pins retain their function after configuration.

Configuration constraints can be selected when generating the Virtex-5 bitstream. Certain configuration operations can be affected by these constraints. For a description of the available constraints, see the software constraints guide.

Table 1-1: **Virtex-5 FPGA Dedicated Configuration Pins**

Pin Name	Type(1)	Description
M[2:0]	Input	Mode pins that determine configuration mode. Sampled on the rising edge of INIT_B.
CCLK	Input or Output	Configuration clock source for all configuration modes except JTAG. Refer to the "Board Layout for Configuration Clock (CCLK)" section for details.
D_IN	Input	Serial data input for serial configuration modes.
DOUT_BUSY	Output	In Serial configuration mode, pin acts as serial data output for daisy-chain configuration. In SelectMAP mode, pin acts as BUSY output.
DONE	Bidirectional, Open-Drain, or Active	Active High signal indicating configuration is complete. 0 = FPGA not configured 1 = FPGA configured Refer to the "BitGen" section of the *Development System Reference Guide* for software settings.
INIT_B	Bidirectional, Input or Output, Open-Drain	Before the Mode pins are sampled, INIT_B is an input that can be held Low to delay configuration. INIT_B is bidirectional during configuration. After the Mode pins are sampled, INIT_B is an open-drain, active-Low output, indicating whether a CRC error occurred during configuration or a readback CRC error occurred after configuration (when enabled): 0 = CRC or IDCODE Error (DONE Low) or Readback CRC Error (DONE High and Readback CRC enabled) 1 = No CRC error, housecleaning complete.
PROGRAM_B(2)	Input	Active-Low, asynchronous full-chip reset.
HSWAPEN	Input	Active-High input used to disable weak preconfiguration I/O pull-up resistors: 0 = Weak preconfiguration I/O pull-up resistors enabled 1 = Weak preconfiguration I/O pull-up resistors disabled HSWAPEN has a weak pull-up prior to and during configuration. HSWAPEN must be connected to either disable or enable the pull-up resistors. The weak pull-up does not always provide a reliable 1.
TDI	Input	Test Data In. This pin is the serial input to all JTAG instruction and data registers. The state of the TAP controller and the current instruction determine the register that is fed by the TDI pin for a specific operation. TDI has an internal resistive pull-up to provide a logic High to the system if the pin is not driven. TDI is applied into the JTAG registers on the rising edge of TCK.
TDO	Output	Test Data Out. This pin is the serial output for all JTAG instruction and data registers. The state of the TAP controller and the current instruction determine the register (instruction or data) that feeds TDO for a specific operation. TDO changes state on the falling edge of TCK and is only active during the shifting of instructions or data through the device. TDO is an active driver output.
TMS	Input	Test Mode Select. This pin determines the sequence of states through the TAP controller on the rising edge of TCK. TMS has an internal resistive pull-up to provide a logic High if the pin is not driven.
TCK	Input	Test Clock. This pin is the JTAG Test Clock. TCK sequences the TAP controller and the JTAG registers in Virtex-5 devices.

Table 1-1: **Virtex-5 FPGA Dedicated Configuration Pins *(Continued)***

Pin Name	Type(1)	Description
CS_B	Input	Active-Low chip select to enable the SelectMAP data bus (see "SelectMAP Data Loading," page 54): 0 = SelectMAP data bus enabled 1 = SelectMAP data bus disabled
RDWR_B	Input	Determines the direction of the SelectMAP data bus (see "SelectMAP Data Loading," page 54): 0 = inputs (configuration data write) 1 = outputs (configuration data read) RDWR_B input can only be changed while CS_B is deasserted, otherwise an ABORT occurs (see "SelectMAP ABORT," page 58).

Notes:

1. The *Bidirectional* type describes a pin that is bidirectional under all conditions. If the pin is an input for some configuration modes or an output for others, it is listed as an *Input or Output* type.
2. Pulsing PROGRAM_B does not reset the JTAG TAP state machine (this behavior differs from Virtex-4 devices).
3. All JTAG and serial dedicated configuration pins are located in a separate, dedicated bank with a dedicated V_{CC_CONFIG} supply.

Table 1-2: **Virtex-5 FPGA Dual-Purpose Configuration Pins**

Pin Name	BPI-Up/ BPI-Down	SelectMAP	SPI	I/O Bank	Description
ADDR[25:20]	ADDR[25:20]			2	BPI address bus output.
ADDR[19:0]	ADDR[19:0]	D[31:16]	RCMD [7:0]	1	BPI address bus output. SelectMAP data I/O for Slave SelectMAP mode only. SPI read command strapping inputs RCMD[7:0] (muxed on ADDR[7:0]) when FS[2:0] = 001. Sampled on rising edge of INIT_B when used for SPI read command strapping.
RS[1:0](1)	RS[1:0]			2	Revision Select outputs.
FCS_B	FCS_B		FCS_B	2	BPI and SPI Flash chip select output.
FOE_B	FOE_B		MOSI	2	BPI Flash output enable. SPI data output from FPGA.
FWE_B	FWE_B			2	BPI Flash write enable.
CSO_B	CSO_B	CSO_B		2	Parallel daisy-chain chip select output.
D[7:0]	D[7:0]	D[7:0]	FS[2:0]	2	BPI data input, SelectMAP data in/out. SPI Flash variant selection. FS[2:0] are on D[2:0].
D[15:8]	D[15:8]	D[15:8]		4	BPI data input. SelectMAP data in/out.

Notes:

1. RS[1:0] are actively driven to 0 if a configuration error is detected or a watchdog timer event occurs, except in Slave SelectMAP mode. This can interfere with user logic when used as user I/O. When using the RS[1:0] pins, they should be restricted to be output-only in User mode.
2. All dual-purpose pins are located in I/O banks with an associated bank supply.
3. If the D0–D7 pins are used as I/O after configuration with at least one of those pins using DCI inputs or outputs, the FreezeDCI option should not be set to Yes. Otherwise, these I/Os do not function properly. For an output, the I/O is put into a 3-state condition or does not have the correct termination. For an input, the I/O does not have the correct termination.

Configuration Data File Formats

Xilinx design tools can generate configuration data files in a number of different formats, as described in Table 1-3. BitGen converts the post-PAR NCD file into a configuration file or a bitstream. PROMGen, the PROM file generator, converts one or more bitstream files into a PROM file. PROM files can be generated in a number of different file formats and does not need to be used with a PROM. They can be stored anywhere and delivered by any means.

Table 1-3: **Xilinx Configuration File Formats**

File Extension	Bit Swapping(1)	Xilinx Software Tool(2)	Description
BIT	Not Bit Swapped	BitGen (generated by default)	Binary configuration data file containing header information that does not need to be downloaded to the FPGA. Used to program devices from iMPACT with a programming cable.
RBT	Not Bit Swapped	BitGen (generated if **-b** option is set)	ASCII equivalent of the BIT file containing a text header and ASCII 1s and 0s. (Eight bits per configuration bit.)
BIN	• BitGen: Not Bit Swapped • PROMGen: Bit Swapped	BitGen (generated if **-g binary:yes** option is set) or PROMGen	Binary configuration data file with no header information. Similar to BIT file. Can be used for custom configuration solutions (for example, microprocessors), or in some cases to program third-party PROMs.
MCS EXO TEK	Bit Swapped	PROMGen or iMPACT	ASCII PROM file formats containing address and checksum information in addition to configuration data. Used mainly for device programmers and iMPACT.
HEX	Determined by User	PROMGen or iMPACT	ASCII PROM file format containing only configuration data. Used mainly in custom configuration solutions.

Notes:

1. Bit swapping is discussed in the "Bit Swapping" section.
2. For complete BitGen and PROMGen syntax, refer to the *Development System Reference Guide*.

Bitstream Overview

The Virtex-5 bitstream contains commands to the FPGA configuration logic as well as configuration data. Table 1-4 gives a typical bitstream length for each of the Virtex-5 devices.

Table 1-4: **Virtex-5 FPGA Bitstream Length**

Device	Total Number of Configuration Bits(1)
XC5VLX30	8,374,016
XC5VLX50	12,556,672
XC5VLX85	21,845,632
XC5VLX110	29,124,608
XC5VLX155	41,048,064
XC5VLX220	53,139,456
XC5VLX330	79,704,832

Table 1-4: **Virtex-5 FPGA Bitstream Length *(Continued)***

Device	Total Number of Configuration Bits(1)
XC5VLX20T	6,251,200
XC5VLX30T	9,371,136
XC5VLX50T	14,052,352
XC5VLX85T	23,341,312
XC5VLX110T	31,118,848
XC5VLX155T	43,042,304
XC5VLX220T	55,133,696
XC5VLX330T	82,696,192
XC5VSX35T	13,349,120
XC5VSX50T	20,019,328
XC5VSX95T	35,716,096
XC5VSX240T	79,610,368
XC5VFX30T	13,517,056
XC5VFX70T	27,025,408
XC5VFX100T	39,389,696
XC5VFX130T	49,234,944
XC5VFX200T	70,856,704
XC5VTX150T	43,278,464
XC5VTX240T	65,755,648

Notes:

1. All of the Virtex-5 devices can be configured by the 128 Mb Platform Flash XL.
2. The bitstream length represents the typical cases. Certain BitGen options can vary the bitstream length, such as **`Compress`**.

A Virtex-5 bitstream consists of three sections:

- "Bus Width Auto Detection"
- "Sync Word"
- FPGA configuration (see Chapter 6, "Configuration Details")

Bus Width Auto Detection

Bus width auto detection pattern is inserted at the beginning of every bitstream. It is used in parallel configuration modes to automatically detect configuration bus width. Because it appears before the Sync word, serial configuration modes ignore it.

For parallel configuration modes, the bus width is auto-detected by the configuration logic. A bus width detection pattern is put in the front of the bitstream. The configuration logic only checks the low eight bits of the parallel bus. Depending on the byte sequence received, the configuration logic can automatically switch to the appropriate external bus

width. Table 1-5 shows an example bitstream with an inserted bus width detection pattern. When observing the pattern on the FPGA data pin, the bits are bit swapped, as described in "Parallel Bus Bit Order."

The bitstream data in Table 1-5 shows the 32-bit configuration word for an unswapped bitstream. For a swapped bitstream format, the LSB and MSB order for the individual bytes are swapped. For example, the Sync word at the FPGA pins in the D[31:00] bit order would be: 0x5599AA66. For swapped and unswapped formats see "Configuration Data File Formats.".

Table 1-5: **Bus Width Detection Pattern**

D[24:31]	D[16:23]	D[8:15]	D[0:7]	Comments
0xFF	0xFF	0xFF	0xFF	
0x00	0x00	0x00	0xBB	Bus Width Pattern
0x11	0x22	0x00	0x44	Bus Width Pattern
0xFF	0xFF	0xFF	0xFF	
0xFF	0xFF	0xFF	0xFF	
0xAA	0x99	0x55	0x66	Sync Word
…	…	…	…	…

Bus width auto detection is transparent to most users, because all configuration bitstreams (BIT or RBT files) generated by the Xilinx ISE® Bitstream Generator (BitGen) software include the Bus Width Auto Detection pattern. These patterns are ignored by the configuration logic if the Mode pins are set to Master Serial, Slave Serial, JTAG, or SPI mode.

For the x8 bus, the configuration bus width detection logic first finds 0xBB on the D[0:7] pins, followed by 0x11. For the x16 bus, the configuration bus width detection logic first finds 0xBB on D[0:7] followed by 0x22. For the x32 bus, the configuration bus width detection logic first finds 0xBB, on D[0:7], followed by 0x44.

If the immediate byte after 0xBB is not 0x11, 0x22, or 0x44, the bus width state machine is reset to search for the next 0xBB until a valid sequence is found. Then it switches to the appropriate external bus width and starts looking for the Sync word.

Sync Word

A special Sync word is used to allow configuration logic to align at a 32-bit word boundary. No packet processed by the FPGA until the Sync word is found. The bus width must be detected successfully for parallel configuration modes before the Sync word can be detected. Table 1-6 shows the Sync word in an unswapped bitstream format.

Table 1-6: **Sync Word**

31:24	23:16	15:8	7:0
0xAA	0x99	0x55	0x66

Generating PROM Files

PROM files are generated from bitstream files with the PROMGen utility. Users can access PROMGen directly from the command line or indirectly through the iMPACT File Generation Mode. For PROMGen syntax, refer to the *Development System Reference Guide*. For information on iMPACT, refer to the ISE® Software Documentation). PROM files serve to reformat bitstream files for PROM programming and combine bitstream files for serial daisy chains (see "PROM Files for Serial Daisy Chains").

PROM Files for Serial Daisy Chains

Configuration data for serial daisy chains requires special formatting because separate BIT files cannot simply be concatenated together to program the daisy chain. The special formatting is performed by PROMGen (or iMPACT) when generating a PROM file from multiple bitstreams. To generate the PROM file, specify multiple bitstreams using the PROMGen **-n**, **-u**, and **-d** options or the iMPACT File Generation Wizard. Refer to software documentation for details.

PROMGen reformats the configuration bitstreams by nesting downstream configuration data into configuration packets for upstream devices. Attempting to program the chain by sending multiple bitstreams to the first device causes the first device to configure and then ignore the subsequent data.

PROM Files for SelectMAP Configuration

The `MCS` file format is most commonly used to program Xilinx configuration PROMs that in turn program a single FPGA in SelectMAP mode. For custom configuration solutions, the `BIN` and `HEX` files are the easiest PROM file formats to use due to their *raw* data format. In some cases, additional formatting is required; refer to XAPP502, *Using a Microprocessor to Configure Xilinx FPGAs via Slave Serial or SelectMAP Mode* for details.

If multiple configuration bitstreams for a SelectMAP configuration reside on a single memory device, the bitstreams must not be combined into a serial daisy chain PROM file. Instead, the target memory device should be programmed with multiple `BIN` or `HEX` files. If a single PROM file with multiple, separate data streams is needed, one can be generated in iMPACT by targeting a *Parallel PROM*, then selecting the appropriate number of data streams. This can also be accomplished through the PROMGen command line. Refer to PROMGen software documentation for details.

For Platform Flash XL-based SelectMAP configuration, use the iMPACT software to generate an MCS PROM file. Select the Xilinx XCF128X device as the target PROM device type for creating the file. See UG438, *Platform Flash XL User Guide*, for PROM file generation instructions.

PROM Files for SPI/BPI Configuration

The **-d** and **-u** options in PROMGen or the iMPACT File Generation Wizard are used to create PROM files for third-party Flash devices. The output format supported by your third-party programmer should be chosen. Some BPI devices require endian-swapping to be enabled when programming the PROM file. Refer to the Flash vendor's documentation.

Bit Swapping

Bit swapping is the swapping of the bits within a byte. The `MCS`, `EXO`, and `TEK` PROM file formats are always bit swapped. The `HEX` file format can be bit swapped or not bit

swapped, depending on user options. The bitstream files generated by the BitGen (`BIT`, `RBT`, `BIN`) are never bit swapped.

The `HEX` file format contains only configuration data. The other PROM file formats include address and checksum information that should not be sent to the FPGA. The address and checksum information is used by some third-party device programmers, but is not programmed into the PROM.

Figure 1-1 shows how two bytes of data (`0xABCD`) are bit swapped.

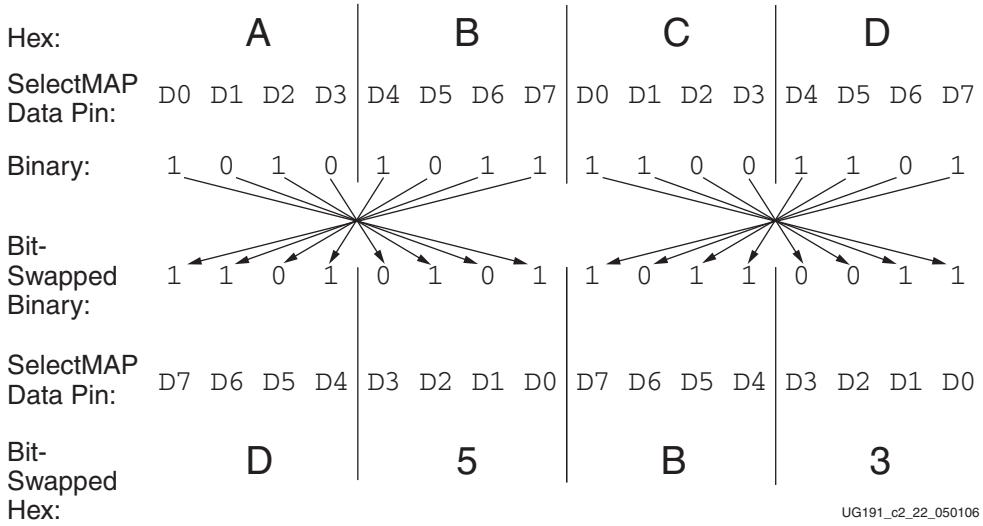


Figure 1-1: **Bit Swapping Example**

The MSB of each byte goes to the D0 pin regardless of the orientation of the data:

- In the bit-swapped version of the data, the bit that goes to D0 is the right-most bit
- In the non-bit-swapped data, the bit that goes to D0 is the left-most bit.

Whether or not data must be bit swapped is entirely application-dependent. Bit swapping is applicable for Master Serial, Master SelectMAP, or BPI PROM files.

Parallel Bus Bit Order

Traditionally, in SelectMAP x8 mode, configuration data is loaded one byte per CCLK, with the most-significant bit (MSB) of each byte presented to the D0 pin. Although this convention (D0 = MSB, D7 = LSB) differs from many other devices, it is consistent across all Xilinx FPGAs. The bit-swap rule also applies to Virtex-5 BPI-Up and BPI-Down x8 modes (see "Bit Swapping," page 21).

In Virtex-5 devices, the bit-swap rule is extended to x16 and x32 bus widths. That is, the data is bit swapped within each byte. Virtex-4 SelectMAP x32 mode does not bit swap.

Table 1-7 and Table 1-8 show examples of a sync word inside a bitstream. These examples illustrate what is expected at the FPGA data pins when using parallel configuration modes, such as Slave SelectMAP, Master SelectMAP, BPI-Up, and BPI-Down modes.

Table 1-7: **Sync Word Bit Swap Example**

Sync Word	[31:24](1)	[23:16]	[15:8]	[7:0]
Bitstream Format	0xAA	0x99	0x55	0x66
Bit Swapped	0x55	0x99	0xAA	0x66

Notes:

1. [31:24] changes from 0xAA to 0x55 after bit swapping.

Table 1-8: **Sync Word Data Sequence Example for x8, x16, and x32 Modes**

CCLK Cycle	1	2	3	4
D[7:0] pins for x8	0x55	0x99	0xAA	0x66
D[15:0] pins for x16	0x5599	0xAA66		
D[31:0] pins for x32	0x5599AA66			

Configuration Sequence

While each of the configuration interfaces is different, the basic steps for configuring a Virtex-5 device are the same for all modes. Figure 1-2 shows the Virtex-5 configuration process. The following subsections describe each step in detail, where the current step is highlighted in gray at the beginning of each subsection.

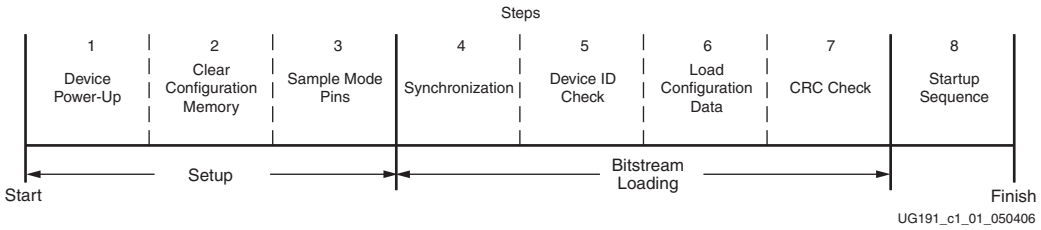


Figure 1-2: **Virtex-5 Device Configuration Process**

The Virtex-5 device is initialized and the configuration mode is determined by sampling the mode pins in three setup steps.

Setup (Steps 1-3)

The setup process is similar for all configuration modes (see Figure 1-3).

The setup steps are critical for proper device configuration. The steps include Device Power-Up, Clear Configuration Memory, and Sample Mode Pins.

Device Power-Up (Step 1)

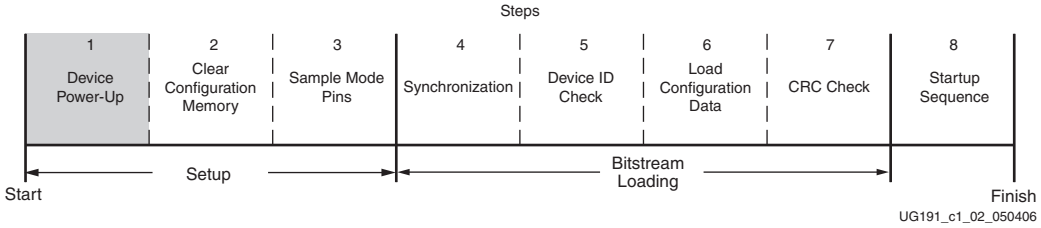


Figure 1-3: **Device Power-Up (Step 1)**

For configuration, Virtex-5 devices require power on the V_{CC_CONFIG} (V_{CCO_0}), V_{CCAUX}, and V_{CCINT} pins. There are no power-supply sequencing requirements.

All JTAG and serial configuration pins are located in a separate, dedicated bank with a dedicated V_{CC_CONFIG} supply (V_{CC_CONFIG} = V_{CCO_0}). The dual-mode pins are located in Banks 1, 2, and 4. All dedicated input pins operate at V_{CC_CONFIG} LVCMOS level. All active dedicated output pins operate at the V_{CC_CONFIG} voltage level with the output standard set to LVCMOS_12F.

For all modes that use dual-mode I/O, the associated V_{CCO_X} must be connected to the appropriate voltage to match the I/O standard of the configuration device. The pins are also LVCMOS_12F during configuration.

For power-up, the V_{CCINT} power pins must be supplied with 1.0V sources. None of the I/O voltage supplies (V_{CCO}) needs to be powered for Virtex-5 configuration in JTAG or serial modes when RS[1:0] is not used. Otherwise, V_{CCO_2} must be supplied. Table 1-9 shows the power supplies required for configuration. Table 1-10 shows the timing for power-up.

Table 1-9: **Power Supplies Required for Configuration**

Pin Name	Value	Units	Description
V_{CCINT}	1.0	Volts	Internal core voltage
V_{BATT} (1)	1.0 - 3.6	Volts	Encryption Key battery supply
V_{CC_CONFIG}	1.5, 1.8, 2.5, 3.3	Volts	Configuration bank supply voltage (V_{CCO_0})
V_{CCAUX}	2.5	Volts	Auxiliary power input for configuration logic and other FPGA functions
V_{CCO_1} V_{CCO_2} V_{CCO_4}		Volts	Dual-mode configuration pin output supply voltage

Notes:

1. V_{BATT} is required only when using bitstream encryption.

Table 1-10: **Power-Up Timing**

Description	Symbol
Program Latency	T_{PL}
Power-on Reset (POR)	T_{POR}
CCLK output delay	T_{ICCK}

Table 1-10: **Power-Up Timing**

Description	Symbol
Program Pulse Width	$T_{PROGRAM}$

Notes:

1. See *Configuration Switching Characteristics* in DS202, *Virtex-5 Data Sheet*: DC and Switching Characteristics for power-up timing characteristics.

Figure 1-4 shows the power-up waveforms.

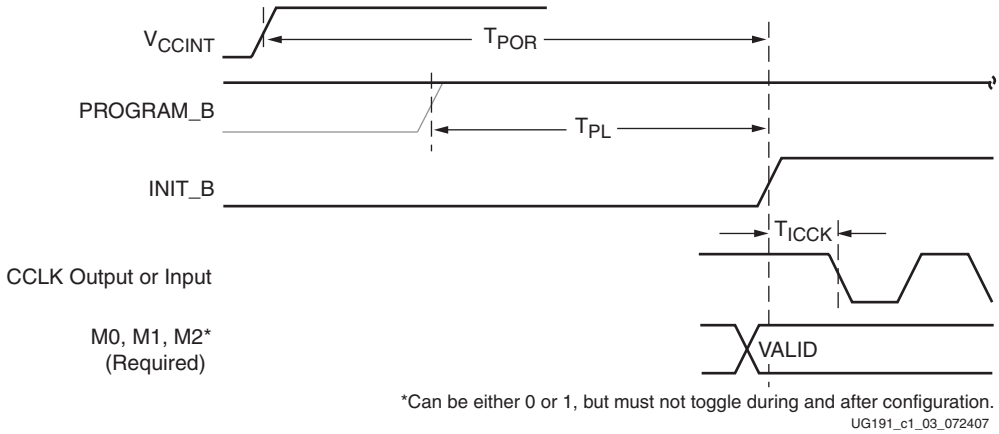


Figure 1-4: **Device Power-Up Timing**

V_{CCINT} should rise monotonically within the specified ramp rate. If this is not possible, delay configuration by holding the INIT_B pin or the PROGRAM_B pin Low (see "Delaying Configuration") while the system power reaches V_{POR}.

The configuration logic power input (V_{CC_CONFIG}) and the auxiliary voltage input (V_{CCAUX}) are used as a logic input to the Power-On-Reset (POR) circuitry. If either of these voltage planes dips below the specified level, POR can trigger again.

Clear Configuration Memory (Step 2, Initialization)

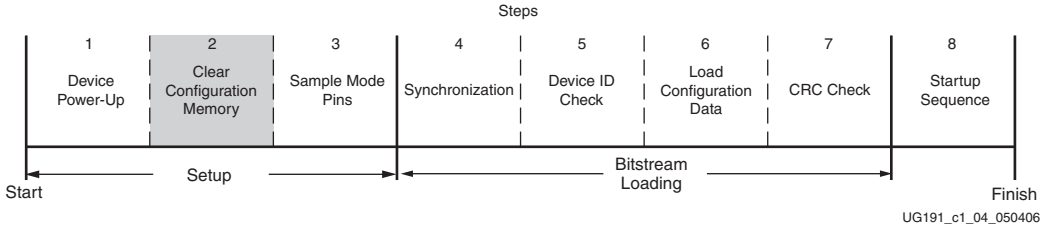


Figure 1-5: **Initialization (Step 2)**

Configuration memory is cleared sequentially any time the device is powered up, after the PROGRAM_B pin is pulsed Low, after the JTAG JPROGRAM instruction or the IPROG command is used, or during a fallback retry configuration sequence. During this time, I/Os are placed in a High-Z state except for the dedicated Configuration and JTAG pins. INIT_B is internally driven Low during initialization, then released after T_{POR} (Figure 1-4) for the power-up case, and T_{PL} for other cases. If the INIT_B pin is held Low externally, the device waits at this point in the initialization process until the pin is released.

The minimum Low pulse time for PROGRAM_B is defined by the $T_{PROGRAM}$ timing parameter. The PROGRAM_B pin can be held active (Low) for as long as necessary, and the device clears the configuration memory twice after PROGRAM_B is released.

Sample Mode Pins (Step 3)

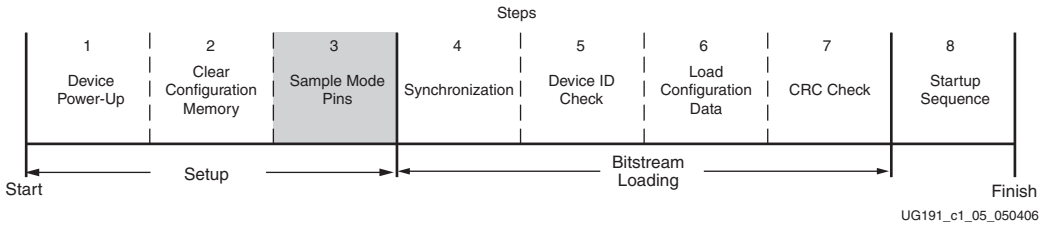


Figure 1-6: **Sample Mode Pins (Step 3)**

When the INIT_B pin transitions to High, the device samples the M[2:0], FS[2:0], and RCMD[7:0] pins and begins driving CCLK if in the Master modes. Mode pins have internal weak pullups. These pullups should not be relied upon to provide valid configuration. FS[2:0] and RCMD[7:0] are only used in SPI mode (see Table 1-2). At this point, the device begins sampling the configuration data input pins on the rising edge of the configuration clock.

Delaying Configuration

There are two ways to delay configuration for Virtex-5 devices:

- The first is to hold the INIT_B pin Low during initialization (Figure 1-4). When INIT_B has gone High, configuration cannot be delayed by subsequently pulling INIT_B Low.
- The second is to hold the PROGRAM_B pin Low. The signals relating to initialization and delaying configuration are defined in Table 1-11.

Table 1-11: **Signals Relating to Initialization and Delaying Configuration**

Signal Name	Type	Access(1)	Description
PROGRAM_B	Input	Externally accessible via the PROGRAM_B pin	Global asynchronous chip reset. Can be held Low to delay configuration.
INIT_B	Input, Output, or Open Drain	Externally accessible via the INIT_B pin	Before the Mode pins are sampled, INIT_B is an input that can be held Low to delay configuration. After the Mode pins are sampled, INIT_B is an open-drain, active-Low output indicating whether a CRC error occurred during configuration or a readback CRC error occurred after configuration (when enabled): 0 = CRC error 1 = No CRC error (needs an external pull-up)
INIT_COMPLETE	Status(2)	Internal signal, accessible through the Virtex-5 status register	Indicates whether INIT_B signal is internally released.
MODE_STATUS[2:0]	Status	Internal signals, accessible through the Virtex-5 status register	Reflects the values sampled on the Mode pins when the status is read.

Table 1-11: **Signals Relating to Initialization and Delaying Configuration *(Continued)***

Signal Name	Type	Access(1)	Description
FS_STATUS[2:0]	Status	Internal signals, accessible through the Virtex-5 status register	Reflects the values sampled on the FS[2:0] pins when INIT_B is asserted High.

Notes:

1. Information on the Virtex-5 status register is available in Table 6-9, page 119. Information on accessing the device status register via JTAG is available in Table 7-5, page 142. Information on accessing the device status register via SelectMAP is available in Table 7-1.
2. The Status type is an internal status signal without a corresponding pin.

Bitstream Loading (Steps 4-7)

The bitstream loading process is similar for all configuration modes; the primary difference between modes is the interface to the configuration logic. Details on the different configuration interfaces are provided in Chapter 2, "Configuration Interfaces."

The most important steps in the bitstream loading process are: bus width auto detection (for SelectMAP and BPI modes, refer to "Bus Width Auto Detection"), synchronization, device ID check, loading configuration data, and the CRC check. Each of these steps involves distinct parts of the configuration bitstream.

Synchronization (Step 4)

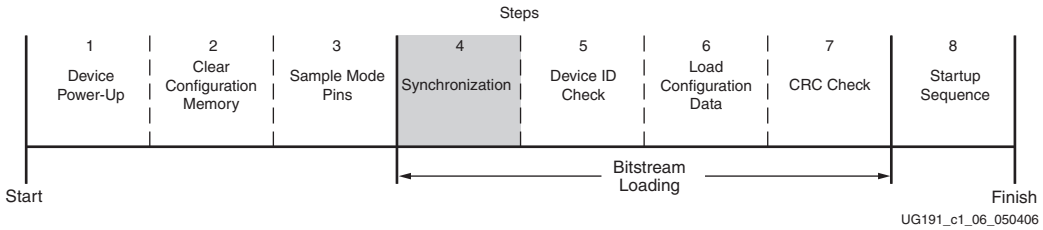


Figure 1-7: **Synchronization (Step 4)**

For BPI-Up, BPI-Down, Slave SelectMAP, and Master SelectMAP modes, the bus width must be first detected (refer to "Bus Width Auto Detection"). The bus width detection pattern is ignored by Slave Serial, Master Serial, SPI, and JTAG modes. Then a special 32-bit synchronization word (`0xAA995566`) must be sent to the configuration logic. The synchronization word alerts the device to upcoming configuration data and aligns the configuration data with the internal configuration logic. Any data on the configuration input pins prior to synchronization is ignored, except the "Bus Width Auto Detection" sequence.

Synchronization is transparent to most users because all configuration bitstreams (BIT files) generated by the BitGen software include both the bus width detection pattern and the synchronization word. Table 1-12 shows signals relating to synchronization.

Table 1-12: **Signals Relating to Synchronization**

Signal Name	Type	Access	Description
DALIGN	Status	Only available through the SelectMAP interface during an ABORT sequence. (See "Configuration Abort Sequence Description," page 58.)	Indicates whether the device is synchronized.

IWIDTH	Status	Internal signal. Accessed only through the Virtex-5 Status register.(1)	Indicates the detected bus width:
			`00 = x1`
			`01 = x8`
			`10 = x16`
			`11 = x32`
			If ICAP is enabled, this signal reflects the ICAP width after configuration is done.

Notes:

1. Information on the Virtex-5 status register is available in Table 6-9. Information on accessing the device status register via JTAG is available in Table 7-5. Information on accessing the device status register via SelectMAP is available in Table 7-1.

Check Device ID (Step 5)

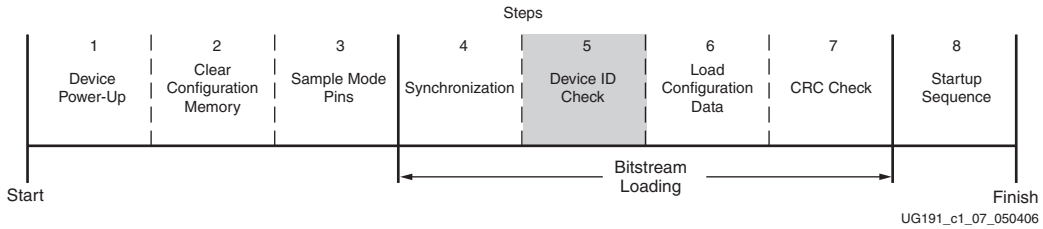


Figure 1-8: **Check Device ID (Step 5)**

After the device is synchronized, a device ID check must pass before the configuration data frames can be loaded. This prevents a configuration with a bitstream that is formatted for a different device. For example, the device ID check should prevent an XC5VLX30 from being configured with an XC5VLX50 bitstream.

If an ID error occurs during configuration, the device attempts to do a fallback reconfiguration (see "Fallback MultiBoot," page 153).

The device ID check is built into the bitstream, making this step transparent to most designers. Table 1-13 shows the Virtex-5 device ID codes, and Table 1-14 shows the signals relating to the device ID check. The device ID check is performed through commands in the bitstream to the configuration logic, not through the JTAG IDCODE register in this case.

The Virtex-5 JTAG ID Code register has the following format:

`vvvv:fffffff:aaaaaaaaa:ccccccccc11`

where

v = revision
f = 7-bit family code (`0010101` = XC5VLXT, `0010100` = XC5VLX, `0010111` = XC5VSXT, and `0011001` =XC5VFXT)
a = number of array rows plus array columns
c = company code

Table 1-13: **Virtex-5 Device ID Codes**

Device	ID Code (Hex)	
	Revision Code	Family, Array, and Company Code
XC5VLX30	See Note 1	286E093
XC5VLX50	See Note 1	2896093
XC5VLX85	See Note 1	28AE093
XC5VLX110	See Note 1	28D6093
XC5VLX155	See Note 1	28EC093
XC5VLX220	See Note 1	290C093
XC5VLX330	See Note 1	295C093
XC5VLX20T	See Note 1	2A56093
XC5VLX30T	See Note 1	2A6E093
XC5VLX50T	See Note 1	2A96093
XC5VLX85T	See Note 1	2AAE093
XC5VLX110T	See Note 1	2AD6093
XC5VLX155T	See Note 1	2AEC093
XC5VLX220T	See Note 1	2B0C093
XC5VLX330T	See Note 1	2B5C093
XC5VSX35T	See Note 1	2E72093
XC5VSX50T	See Note 1	2E9A093
XC5VSX95T	See Note 1	2ECE093
XC5VSX240T	See Note 1	2F3E093
XC5VFX30T	See Note 1	3276093
XC5VFX70T	See Note 1	32C6093
XC5VFX100T	See Note 1	32D8093
XC5VFX130T	See Note 1	3300093
XC5VFX200T	See Note 1	3334093
XC5VTX150T	See Note 1	4502093
XC5VTX240T	See Note 1	453E093

Notes:

1. The value of the Version code can be 0x0 to 0xF.

Table 1-14: **Signals Relating to the Device ID Check**

Signal Name	Type	Access(1)	Description
ID_Err	Status	Internal signal. Accessed only through the Virtex-5 Status register.	Indicates a mismatch between the device ID specified in the bitstream and the actual device ID.

Notes:

1. Information on the Virtex-5 status register is available in Table 6-9. Information on accessing the device status register via JTAG is available in Table 7-5. Information on accessing the device status register via SelectMAP is available in Table 7-1.

Load Configuration Data Frames (Step 6)

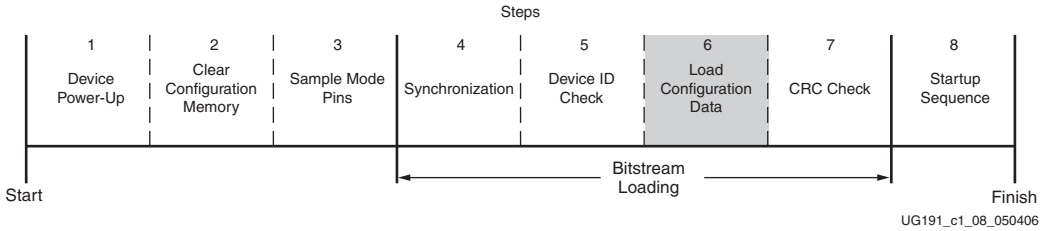


Figure 1-9: **Load Configuration Data Frames (Step 6)**

After the synchronization word is loaded and the device ID has been checked, the configuration data frames are loaded. This process is transparent to most users. For details, refer to Chapter 6, "Configuration Details."

Cyclic Redundancy Check (Step 7)

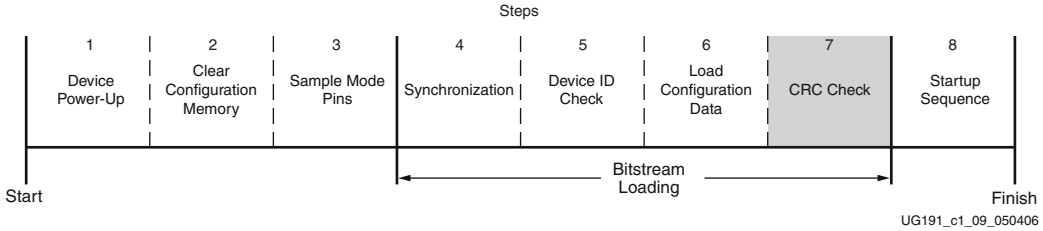


Figure 1-10: **Cyclic Redundancy Check (Step 7)**

As the configuration data frames are loaded, the device calculates a Cyclic Redundancy Check (CRC) value from the configuration data packets. After the configuration data frames are loaded, the configuration bitstream can issue a *Check CRC* instruction to the device, followed by an expected CRC value. If the CRC value calculated by the device does not match the expected CRC value in the bitstream, the device pulls INIT_B Low and aborts configuration. The CRC check is included in the configuration bitstream by default, although the designer can disable it if desired. (Refer to the "BitGen" section of the *Development System Reference Guide*.) If the CRC check is disabled, there is a risk of loading incorrect configuration data frames, causing incorrect design behavior or damage to the device.

If a CRC error occurs during configuration from a mode where the FPGA is the configuration master, the device can attempt to do a fallback reconfiguration (see "Fallback MultiBoot," page 153). In BPI-Up, BPI-Down, and SPI modes, if fallback reconfiguration fails again, the BPI/SPI interface can only be resynchronized by pulsing the PROGRAM_B

pin and restarting the configuration process from the beginning. The JTAG interface is still responsive and the device is still alive, only the BPI/SPI interface is inoperable. In SelectMAP modes, either the PROGRAM_B pin can be pulsed Low or an ABORT sequence can be initiated (see "SelectMAP Configuration Interface" in Chapter 2).

Virtex-5 devices use a 32-bit CRC check. The CRC check is designed to catch errors in transmitting the configuration bitstream. There is a scenario where errors in transmitting the configuration bitstream can be missed by the CRC check: certain clocking errors, such as double-clocking, can cause loss of synchronization between the 32-bit bitstream packets and the configuration logic. Once synchronization is lost, any subsequent commands are not understood, including the command to check the CRC. In this situation, configuration fails with DONE Low and INIT_B High because the CRC was ignored. In BPI Modes, the address counter eventually overflows or underflows to cause wraparound, which triggers fallback reconfiguration (see "Fallback MultiBoot," page 153).

Startup (Step 8)

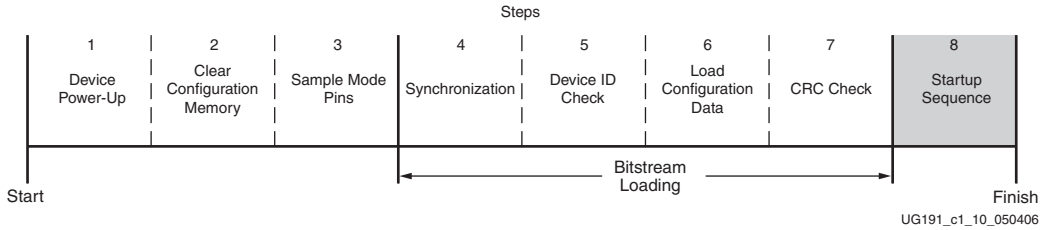


Figure 1-11: **Startup Sequence (Step 8)**

After the configuration frames are loaded, the bitstream instructs the device to enter the startup sequence. The startup sequence is controlled by an 8-phase (phases 0–7) sequential state machine. The startup sequencer performs the tasks outlined in Table 1-15.

Table 1-15: **User-Selectable Cycle of Startup Events**

Phase	Event
1–6	Wait for DCMs to Lock (optional)
1–6	Wait for DCI to Match (optional)
1–6	Assert Global Write Enable (GWE), allowing RAMs and flip-flops to change state
1–6	Negate Global 3-State (GTS), activating I/O
1–6	Release DONE pin
7	Assert End Of Startup (EOS)

The specific order of startup events (except for EOS assertion) is user-programmable through BitGen options (refer to the *Development System Reference Guide*). Table 1-15 shows the general sequence of events, although the specific phase for each of these startup events is user-programmable (EOS is always asserted in the last phase). Refer to Chapter 2, "Configuration Interfaces" for important startup option guidelines. By default, startup events occur as shown in Table 1-16.

Table 1-16: **Default BitGen Sequence of Startup Events**

Phase	Event
4	Release DONE pin
5	Negate GTS, activating I/O
6	Assert GWE, allowing RAMs and flip-flops to change state
7	Assert EOS

The startup sequence can be forced to wait for the DCMs to lock or for DCI to match with the appropriate BitGen options. These options are typically set to prevent DONE, GTS, and GWE from being asserted (preventing device operation) before the DCMs have locked and/or DCI has matched.

The DONE signal is released by the startup sequencer on the cycle indicated by the user, but the startup sequencer does not proceed until the DONE pin actually sees a logic High. The DONE pin is an open-drain bidirectional signal by default. By releasing the DONE pin, the device simply stops driving a logic Low and the pin goes into a High-Z state. An external pull-up resistor is needed for the DONE pin to reach a logic High in this case. Table 1-17 shows signals relating to the startup sequencer. Figure 1-12 shows the waveforms relating to the startup sequencer.

Table 1-17: **Signals Relating to Startup Sequencer**

Signal Name	Type	Access(1)	Description
DONE	Bidirectional(2)	DONE pin or Virtex-5 Status Register	Indicates configuration is complete. Can be held Low externally to synchronize startup with other FPGAs.
Release_DONE	Status	Virtex-5 Status Register	Indicates whether the device has stopped driving the DONE pin Low. If the pin is held Low externally, Release_DONE can differ from the actual value on the DONE pin.
GWE			Global Write Enable (GWE). When deasserted, GWE disables the CLB and the IOB flip-flops as well as other synchronous elements on the FPGA.
GTS			Global 3-State (GTS). When asserted, GTS disables all the I/O drivers except for the configuration pins.
EOS			End of Startup (EOS). EOS indicates the absolute end of the configuration and startup process.
DCI_MATCH			DCI_MATCH indicates when all the Digitally Controlled Impedance (DCI) controllers have matched their internal resistor to the external reference resistor.
DCM_LOCK			DCM_LOCK indicates when all the Digital Clock Managers (DCMs) have locked. This signal is asserted by default. It is active if the LOCK_WAIT option is used on a DCM and the LockCycle option is used when the bitstream is generated.

Notes:

1. Information on the Virtex-5 status register is available in Table 6-9. Information on accessing the device status register via JTAG is available in Table 7-5. Information on accessing the device status register via SelectMAP is available in Table 7-1.
2. Open-drain output by default; optional driver enabled using the BitGen **`drivedone`** option.

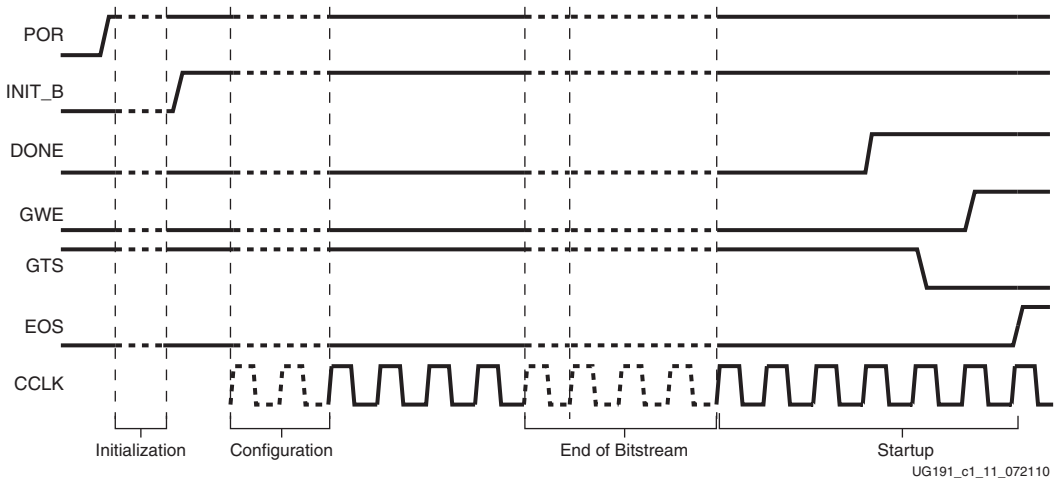


Figure 1-12: **Configuration Signal Sequencing (Default Startup Settings)**

Bitstream Encryption

Virtex-5 devices have on-chip Advanced Encryption Standard (AES) decryption logic to provide a high degree of design security. Without knowledge of the encryption key, potential pirates cannot analyze an externally intercepted bitstream to understand or clone the design. Encrypted Virtex-5 designs cannot be copied or reverse-engineered.

The Virtex-5 AES system consists of software-based bitstream encryption and on-chip bitstream decryption with dedicated memory for storing the encryption key. Using the Xilinx ISE software, the user generates the encryption key and the encrypted bitstream. Virtex-5 devices store the encryption key internally in dedicated RAM, backed up by a small externally connected battery. The encryption key can only be programmed onto the device through the JTAG interface; once programmed, it is not possible to read the encryption key out of the device through JTAG or any other means.

During configuration, the Virtex-5 device performs the reverse operation, decrypting the incoming bitstream. The Virtex-5 AES encryption logic uses a 256-bit encryption key.

The on-chip AES decryption logic cannot be used for any purpose other than bitstream decryption; i.e., the AES decryption logic is not available to the user design and cannot be used to decrypt any data other than the configuration bitstream.

AES Overview

The Virtex-5 encryption system uses the Advanced Encryption Standard (AES) encryption algorithm. AES is an official standard supported by the National Institute of Standards and Technology (NIST) and the U.S. Department of Commerce (http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf).

The Virtex-5 AES encryption system uses a 256-bit encryption key (the alternate key lengths of 128 and 192 bits described by NIST are not implemented) to encrypt or decrypt blocks of 128 bits of data at a time. According to NIST, there are 1.1×10^{77} possible key combinations for a 256-bit key.

Symmetric encryption algorithms such as AES use the same key for encryption and decryption. The security of the data is therefore dependent on the secrecy of the key.

Creating an Encrypted Bitstream

BitGen, provided with the Xilinx ISE software, can generate encrypted as well as non-encrypted bitstreams. For AES bitstream encryption, the user specifies a 256-bit key as an input to BitGen. BitGen in turn generates an encrypted bitstream file (BIT) and an encryption key file (NKY).

For specific BitGen commands and syntax, refer to the *Development System Reference Guide.*

Loading the Encryption Key

The encryption key can only be loaded onto a Virtex-5 device through the JTAG interface. The iMPACT tool, provided with the Xilinx ISE software, can accept the NKY file as an input and program the device with the key through JTAG, using a supported Xilinx programming cable.

To program the key, the device enters a special *key-access mode* using the ISC_PROGRAM_KEY instruction. In this mode, all FPGA memory, including the encryption key and configuration memory, is cleared. After the key is programmed and the key-access mode is exited, the key cannot be read out of the device by any means, and it cannot be reprogrammed without clearing the entire device. The key-access mode is transparent to most users.

Loading Encrypted Bitstreams

Once the device has been programmed with the correct encryption key, the device can be configured with an encrypted bitstream. After configuration with an encrypted bitstream, it is not possible to read the configuration memory through JTAG or SelectMAP readback, regardless of the BitGen security setting.

While the device holds an encryption key, a non-encrypted bitstream can be used to configure the device; in this case the key is ignored. After configuring with a non-encrypted bitstream, readback is possible (if allowed by the BitGen security setting). The encryption key still cannot be read out of the device, preventing the use of *Trojan Horse* bitstreams to defeat the Virtex-5 encryption scheme.

The method of configuration is not affected by encryption. The configuration bitstream can be delivered in any mode (Serial, JTAG, or any x8 parallel modes) from any configuration solution (PROM, System ACE™ controller, etc.). The x16 and x32 bus widths are not supported for encrypted bitstreams. Configuration timing and signaling are also unaffected by encryption.

The encrypted bitstream must configure the entire device because partial reconfiguration through any configuration interface is not permitted for encrypted bitstreams. After configuration, the device cannot be reconfigured without toggling the PROGRAM_B pin, cycling power, or issuing the JPROGRAM instruction. Fallback reconfiguration and IPROG reconfiguration (see "Fallback MultiBoot," page 153) are disabled after encryption is turned on. Readback is available through the ICAP primitive (see "Bitstream Encryption and Internal Configuration Access Port (ICAP)"). None of these events resets the key if V_{BATT} or V_{CCAUX} is maintained.

A mismatch between the key in the encrypted bitstream and the key stored in the device causes configuration to fail with the INIT_B pin going Low and the DONE pin remaining Low.

Bitstream Encryption and Internal Configuration Access Port (ICAP)

The Internal Configuration Access Port (ICAP) primitive provides the user logic with access to the Virtex-5 configuration interface. The ICAP interface is similar to the SelectMAP interface, although the restrictions on readback for the SelectMAP interface do not apply to the ICAP interface after configuration. Users can perform readback through the ICAP interface even if bitstream encryption is used. Unless the designer wires the ICAP interface to user I/O, this interface does not offer attackers a method for defeating the Virtex-5 AES encryption scheme.

Users concerned about the security of their design should *not*:

- Wire the ICAP interface to user I/O

-or-

- Instantiate the ICAP primitive.

Like the other configuration interfaces, the ICAP interface does not provide access to the key register.

V_{BATT}

The encryption key memory cells are volatile and must receive continuous power to retain their contents. During normal operation, these memory cells are powered by the auxiliary voltage input (V_{CCAUX}), although a separate V_{BATT} power input is provided for retaining the key when V_{CCAUX} is removed. Because V_{BATT} draws very little current (on the order of nanoamperes), a small watch battery is suitable for this supply. (To estimate the battery life, refer to V_{BATT} DC Characteristics in DS202, *Virtex-5 Data Sheet*: *DC and Switching Characteristics* and the battery specifications.) At less than a 100 nA load, the endurance of the battery should be limited only by its shelf life.

V_{BATT} does not draw any current and can be removed while V_{CCAUX} is applied. V_{BATT} cannot be used for any purpose other than retaining the encryption keys when V_{CCAUX} is removed.

XILINX®

Chapter 2

Configuration Interfaces

Virtex®-5 devices have six configuration interfaces. Each configuration interface corresponds to one or more configuration modes and bus width, shown in Table 2-1. For detailed interface timing information, see DS202, *Virtex-5 FPGA Data Sheet: DC and Switching Characteristic*.

Table 2-1: **Virtex-5 Device Configuration Modes**

Configuration Mode	M[2:0]	Bus Width	CCLK Direction
Master Serial(2)	000	1	Output
Master SPI(2)	001	1	Output
Master BPI-Up(2)	010	8, 16	Output
Master BPI-Down(2)	011	8, 16	Output
Master SelectMAP(2)	100	8, 16	Output
JTAG	101	1	Input (TCK)
Slave SelectMAP	110	8, 16, 32	Input
Slave Serial	111	1	Input

Notes:

1. Parallel configuration mode bus is auto-detected by the configuration logic.
2. In Master configuration mode, the CCLK pin is the clock source for the Virtex-5 internal configuration logic. The Virtex-5 CCLK output pin must be free from reflections to avoid double-clocking the internal configuration logic. Refer to the "Board Layout for Configuration Clock (CCLK)" section for more details.

Serial Configuration Interface

In serial configuration modes, the FPGA is configured by loading one configuration bit per CCLK cycle:

- In Master Serial mode, CCLK is an output.
- In Slave Serial mode, CCLK is an input.

Figure 2-1 shows the basic Virtex-5 serial configuration interface.

There are four methods of configuring an FPGA in serial mode:

- Master serial configuration
- Slave serial configuration
- Serial daisy-chain configuration

- Ganged serial configuration

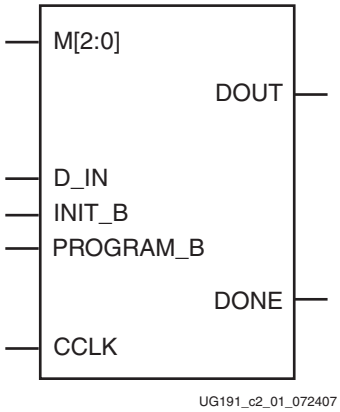


Figure 2-1: **Virtex-5 FPGA Serial Configuration Interface**

Table 2-2 describes the Serial Configuration Interface.

Table 2-2: **Virtex-5 FPGA Serial Configuration Interface Pins**

Pin Name	Type	Dedicated or Dual-Purpose	Description
M[2:0]	Input	Dedicated	Mode Pins – determine configuration mode
CCLK	Input or Output	Dedicated	Configuration clock source for all configuration modes except JTAG
D_IN	Input	Dedicated	Serial configuration data input, synchronous to rising CCLK edge
DOUT_BUSY	Output	Dedicated	Serial data output for downstream daisy-chained devices
DONE	Bidirectional, Open-Drain, or Active	Dedicated	Active High signal indicating configuration is complete: 0 = FPGA not configured 1 = FPGA configured Refer to the BitGen section of the *Development System Reference Guide* for software settings.
INIT_B	Input or Output, Open-Drain	Dedicated	Before the Mode pins are sampled, INIT_B is an input that can be held Low to delay configuration. After the Mode pins are sampled, INIT_B is an open-drain active Low output indicating whether a CRC error occurred during configuration: 0 = CRC error 1 = No CRC error
PROGRAM_B	Input	Dedicated	Active-Low asynchronous full-chip reset

Clocking Serial Configuration Data

Figure 2-2 shows how configuration data is clocked into Virtex-5 devices in Slave Serial and Master Serial modes.

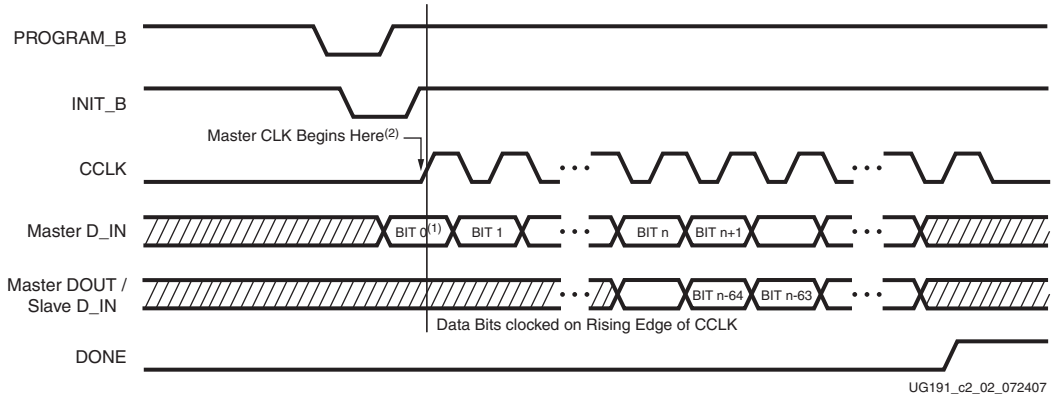


Figure 2-2: **Serial Configuration Clocking Sequence**

Notes relevant to Figure 2-2:

1. Bit 0 represents the MSB of the first byte. For example, if the first byte is `0xAA` (`1010_1010`), bit 0 = `1`, bit 1 = `0`, bit 2 = `1`, etc.
2. For Master configuration mode, CCLK does not transition until after the Mode pins are sampled, as indicated by the arrow.
3. CCLK can be free-running in Slave serial mode.

Master Serial Configuration

The Master Serial mode is designed so that the FPGA can be configured from a Xilinx configuration PROM, as shown in Figure 2-3.

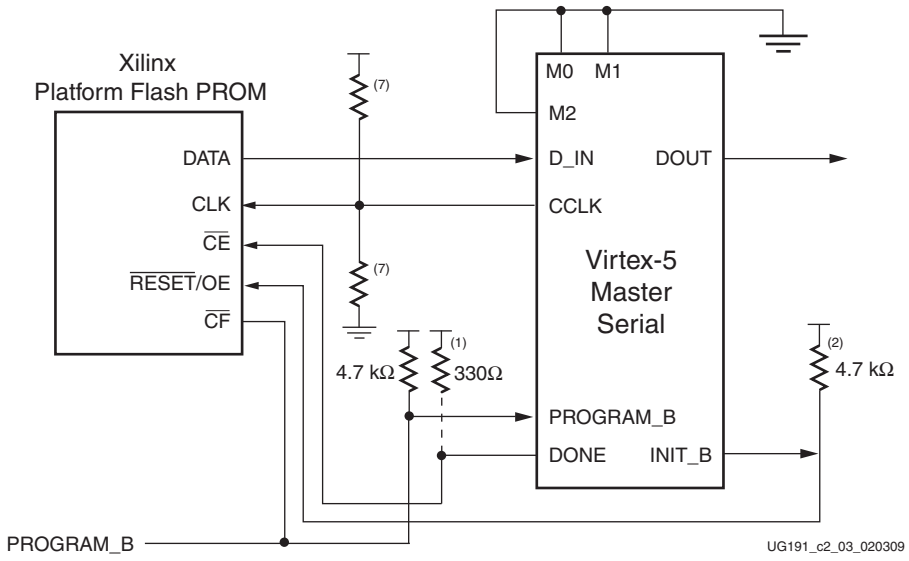


Figure 2-3: **Master Serial Mode Configuration**

Notes relevant to Figure 2-3:

1. The DONE pin is by default an open-drain output requiring an external pull-up resistor. The DONE pin has a programmable active driver. To enable it, enable the **`DriveDONE`** option in BitGen.
2. The INIT_B pin is a bidirectional, open-drain pin. An external pull-up resistor is required.
3. The BitGen startup clock setting must be set for CCLK for serial configuration.
4. The PROM in this diagram represents one or more Xilinx PROMs. Multiple Xilinx PROMs can be cascaded to increase the overall configuration storage capacity.
5. The `BIT` file must be reformatted into a PROM file before it can be stored on the Xilinx PROM. Refer to the "Generating PROM Files" section.
6. On some Xilinx PROMs, the reset polarity is programmable. $\overline{\text{RESET}}$ should be configured as active Low when using this setup.
7. The CCLK net requires Thevenin parallel termination. See "Board Layout for Configuration Clock (CCLK)," page 73.
8. Master serial mode configuration is specific to the Platform Flash XCFS and XCFP PROM only.

Slave Serial Configuration

Slave serial configuration is typically used for devices in a serial daisy chain or when configuring a single device from an external microprocessor or CPLD. Design considerations are similar to Master serial configuration except for the direction of CCLK. CCLK must be driven from an external clock source.

Serial Daisy Chains

Multiple Virtex-5 devices can be configured from a single configuration source by arranging the devices in a serial daisy chain. In a serial daisy chain, devices receive their configuration data through their D_IN pin, passing configuration data along to downstream devices through their DOUT pin. The device closest to the configuration data source is considered the most *upstream* device, while the device furthest from the configuration data source is considered the most *downstream* device.

In a serial daisy chain, the configuration clock is typically provided by the most upstream device in Master serial mode. All other devices are set for Slave serial mode. Figure 2-4 illustrates this configuration.

Another alternative is to use SPI mode for the first device. The daisy chain data is still sent out through DOUT in SPI mode.

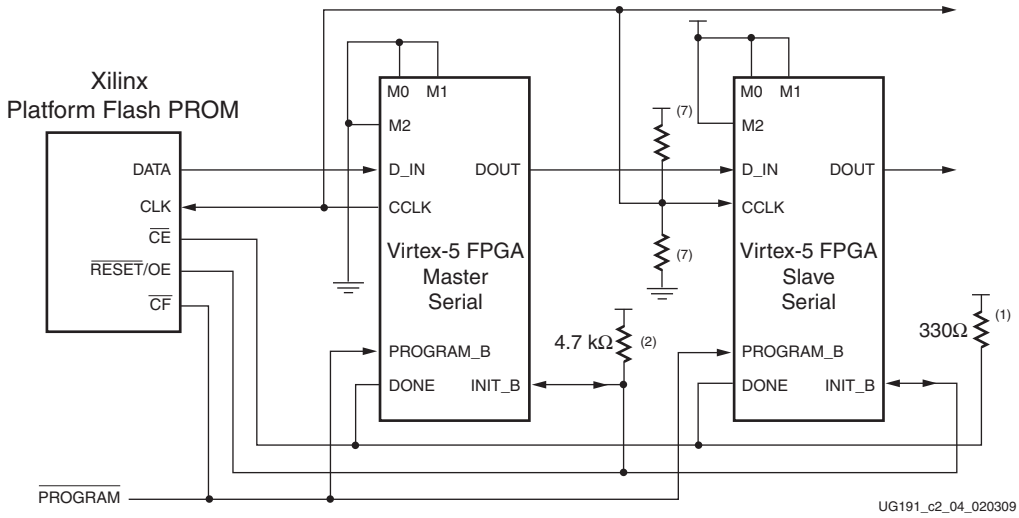


Figure 2-4: **Master/Slave Serial Mode Daisy Chain Configuration**

Notes relevant to Figure 2-4:

1. The DONE pin is by default an open-drain output requiring an external pull-up resistor. For all devices except the first, the active driver on DONE must be disabled. For the first device in the chain, the active driver on DONE can be enabled. See "Guidelines and Design Considerations for Serial Daisy Chains."
2. The INIT_B pin is a bidirectional, open-drain pin. An external pull-up resistor is required.
3. The BitGen startup clock setting must be set for CCLK for serial configuration.
4. The PROM in this diagram represents one or more Xilinx PROMs. Multiple Xilinx PROMs can be cascaded to increase the overall configuration storage capacity.
5. The `BIT` file must be reformatted into a PROM file before it can be stored on the Xilinx PROM. Refer to the "Generating PROM Files" section.
6. On some Xilinx PROMs, the reset polarity is programmable. $\overline{\text{RESET}}$ should be configured as active Low when using this setup.
7. The CCLK net requires Thevenin parallel termination. See "Board Layout for Configuration Clock (CCLK)," page 73.
8. Serial daisy chains are specific to the Platform Flash XCFS and XCFP PROM only.

The first device in a serial daisy chain is the last to be configured. CRC checks only include the data for the current device, not for any others in the chain. (See "Cyclic Redundancy Check (Step 7)" in Chapter 1.)

After the last device in the chain finishes configuration and passes its CRC check, it enters the Startup sequence. At the *Release DONE pin* phase in the Startup sequence, the device places its DONE pin in a High-Z state while the next to the last device in the chain is configured. After all devices release their DONE pins, the common DONE signal is either pulled High externally or driven High by the first device in the chain. On the next rising CCLK edge, all devices move out of the *Release DONE pin* phase and complete their startup sequences.

It is important that all DONE pins in a Slave serial daisy chain be connected. Only the first device in the serial daisy chain should have the DONE active pull-up driver enabled. Enabling the DONE driver on downstream devices causes contention on the DONE signal.

Mixed Serial Daisy Chains

Virtex-5 devices can be daisy-chained with the Virtex, Spartan™-II, Virtex-E, Spartan-IIE, Virtex-II, Virtex-II Pro, Spartan-3, and Virtex-4 families. There are three important design considerations when designing a mixed serial daisy chain:

- Many older FPGA devices cannot accept as fast a CCLK frequency as a Virtex-5 device can generate. Select a configuration CCLK speed supported by all devices in the chain.
- Virtex-5 devices should always be at the beginning of the serial daisy chain, with older family devices located at the end of the chain.
- All Virtex device families have similar BitGen options. The guidelines provided for Virtex-5 BitGen options should be applied to all Virtex devices in a serial daisy chain.
- The number of configuration bits that a device can pass through its DOUT pin is limited. This limit varies for different families (Table 2-3). The sum of the bitstream lengths for all downstream devices must not exceed the number in Table 2-3 for each family.

Table 2-3: **Maximum Number of Configuration Bits, Various Device Families**

Architecture	Maximum DOUT Bits
Virtex-5, Virtex-4, Virtex-II Pro, and Virtex-II Devices	32 x (2^{27} – 1) = 4,294,967,264
Spartan-3 Devices	32 x (2^{27} – 1) = 4,294,967,264
Virtex, Virtex-E, Spartan-II, and Spartan-IIE Devices	32 x (2^{20} – 1) = 33,554,216

Guidelines and Design Considerations for Serial Daisy Chains

There are a number of important considerations for serial daisy chains:

Startup Sequencing (GTS)

GTS should be released before DONE or during the same cycle as DONE to ensure the Virtex-5 device is operational when all DONE pins have been released.

Active DONE Driver

All devices except the first should disable the driver on the DONE pin (refer to the BitGen section of the *Development System Reference Guide* for software settings). The first device in a chain is programmed last:

- DriveDone is disabled (all devices except the first)
- DriveDone is enabled (first device)

Alternatively, the driver can be disabled for all DONE pins and an external pull-up resistor can be added to pull the signal High after all devices have released it.

Connect All DONE Pins

It is important to connect the DONE pins for all devices in a serial daisy chain. Failing to connect the DONE pins can cause configuration to fail. For debugging purposes, it is often helpful to have a way of disconnecting individual DONE pins from the common DONE signal, so that devices can be individually configured through the serial or JTAG interface.

DONE Pin Rise Time

After all DONE pins are released, the DONE pin should rise from logic 0 to logic 1 in one CCLK cycle. External pull-up resistors are required. If additional time is required for the DONE signal to rise, the BitGen **donepipe** option can be set for all devices in the serial daisy chain. (Refer to the BitGen section of the *Development System Reference Guide* for software settings.)

Ganged Serial Configuration

More than one device can be configured simultaneously from the same bitstream using a *ganged* serial configuration setup (Figure 2-5). In this arrangement, the serial configuration pins are tied together such that each device sees the same signal transitions. One device is typically set for Master serial mode (to drive CCLK) while the others are set for Slave serial mode. For ganged serial configuration, all devices must be identical. Configuration can be driven from a configuration PROM or from an external configuration controller.

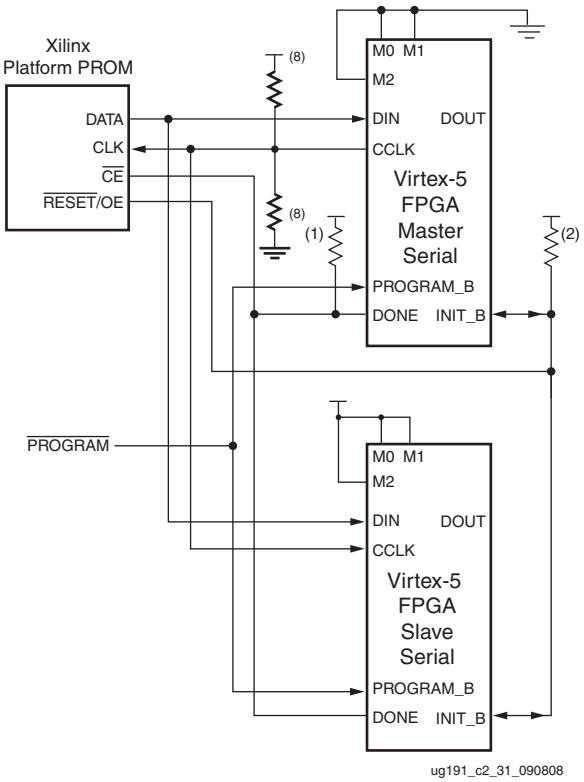


Figure 2-5: **Ganged Serial Configuration**

Notes relevant to Figure 2-5:

1. For ganged serial configuration, the optional DONE driver must be disabled for all devices if one device is set for Master mode because each device might not start up on exactly the same CCLK cycle. An external pull-up resistor is required in this case.
2. The INIT_B pin is a bidirectional, open-drain pin. An external pull-up resistor is required.
3. The BitGen startup clock setting must be set for CCLK for serial configuration.

4. The PROM in this diagram represents one or more Xilinx PROMs. Multiple PROMs can be cascaded to increase the overall configuration storage capacity.
5. The `BIT` file must be reformatted into a PROM file before it can be stored on the PROM. Refer to the "Generating PROM Files" section.
6. On some Xilinx PROMs, the reset polarity is programmable. $\overline{\text{RESET}}$ should be configured as active Low when using this setup.
7. For ganged serial configuration, all devices must be identical (same IDCODE) and must be configured with the same bitstream.
8. The CCLK net requires Thevenin parallel termination. See "Board Layout for Configuration Clock (CCLK)," page 73.
9. Ganged serial configuration is specific to the Platform Flash XCFS and XCFP PROM only.
10. Fallback and Multiboot are not supported in ganged serial configuration.

There are a number of important considerations for ganged serial configuration:

- Startup Sequencing (GTS)

  GTS should be released before DONE or during the same cycle as DONE to ensure all devices are operational when all DONE pins have been released.

- Disable the Active DONE Driver for All Devices

  For ganged serial configuration, the active DONE driver must be disabled for all devices if the DONE pins are tied together, because there can be variations in the startup sequencing of each device. A pull-up resistor is therefore required on the common DONE signal.

  `-g DriveDone:no` (BitGen option, all devices)

- Connect all DONE pins if using a Master Device

  It is important to connect the DONE pins for all devices in ganged serial configuration if one FPGA is used as the Master device. Failing to connect the DONE pins can cause configuration to fail for individual devices in this case. If all devices are set for Slave serial mode, the DONE pins can be disconnected (if the external CCLK source continues toggling until all DONE pins go High).

  For debugging purposes, it is often helpful to have a way of disconnecting individual DONE pins from the common DONE signal.

- DONE Pin Rise Time

  After all DONE pins are released, the DONE pin should rise from logic 0 to logic 1 in one CCLK cycle. If additional time is required for the DONE signal to rise, the BitGen **`donepipe`** option can be set for all devices in the serial daisy chain.

- Configuration Clock (CCLK) as Clock Signal for Board Layout

  The CCLK signal is relatively slow, but the edge rates on the Virtex-5 input buffers are very fast. Even minor signal integrity problems on the CCLK signal can cause the configuration to fail. (Typical failure mode: DONE Low and INIT_B High.) Therefore, design practices that focus on signal integrity, including signal integrity simulation with IBIS, are recommended.

- Signal Fanout

  Designers must focus on good signal integrity when using ganged serial configuration. Signal integrity simulation is recommended.

- PROM Files for Ganged Serial Configuration

PROM files for ganged serial configuration are identical to the PROM files used to configure single devices. There are no special PROM file considerations.

SelectMAP Configuration Interface

The SelectMAP configuration interface (Figure 2-6) provides an 8-bit, 16-bit, or 32-bit bidirectional data bus interface to the Virtex-5 configuration logic that can be used for both configuration and readback. (For details, refer to Chapter 7, "Readback and Configuration Verification.") The bus width of SelectMAP is automatically detected (see "Bus Width Auto Detection").

CCLK is an output in Master SelectMAP mode; in Slave SelectMAP, CCLK is an input. One or more Virtex-5 devices can be configured through the SelectMAP bus.

There are four methods of configuring an FPGA in SelectMAP mode:

- Single device Master SelectMAP
- Single device Slave SelectMAP
- Multiple device SelectMAP bus
- Multiple device ganged SelectMAP

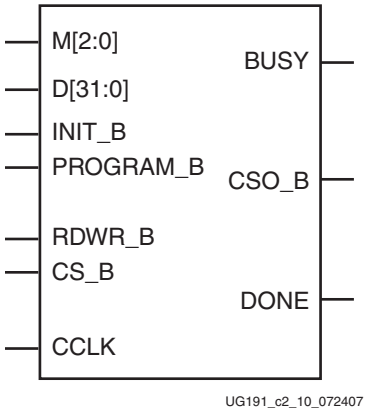


Figure 2-6: **Virtex-5 Device SelectMAP Configuration Interface**

Table 2-4 describes the SelectMAP configuration interface.

Table 2-4: **Virtex-5 Device SelectMAP Configuration Interface Pins**

Pin Name	Type	Dedicated or Dual-Purpose	Description
M[2:0]	Input	Dedicated	Mode pins - determine configuration mode
CCLK	Input and Output	Dedicated	Configuration clock source for all configuration modes except JTAG
D[31:0]	Three-State Bidirectional	Dual-Purpose	Configuration and readback data bus, clocked on the rising edge of CCLK. See "Parallel Bus Bit Order" and Table 1-2.

Table 2-4: **Virtex-5 Device SelectMAP Configuration Interface Pins *(Continued)***

Pin Name	Type	Dedicated or Dual-Purpose	Description
BUSY	Three-State Output	Dedicated	Indicates that the device is not ready to send readback data. For Virtex-5 devices, the BUSY signal is only needed for readback; it is not needed for configuration (see "SelectMAP Data Loading").
DONE	Bidirectional, Open-Drain or active	Dedicated	Active-High signal indicating configuration is complete: 0 = FPGA not configured 1 = FPGA configured
INIT_B	Input or Output, Open-Drain	Dedicated	Before the Mode pins are sampled, INIT_B is an input that can be held Low to delay configuration. After the Mode pins are sampled, INIT_B is an open-drain, active-Low output indicating whether a CRC error occurred during configuration: 0 = CRC error 1 = No CRC error When the SEU detection function is enabled, INIT_B is optionally driven Low when a read back CRC error is detected.
PROGRAM_B	Input	Dedicated	Active-Low asynchronous full-chip reset.
CS_B	Input	Dedicated	Active-Low chip select to enable the SelectMAP data bus (see "SelectMAP Data Loading"): 0 = SelectMAP data bus enabled 1 = SelectMAP data bus disabled
RDWR_B	Input	Dedicated	Determines the direction of the D[x:0] data bus (see "SelectMAP Data Loading"): 0 = inputs 1 = outputs RDWR_B input can only be changed while CS_B is deasserted, otherwise an ABORT occurs (see "SelectMAP ABORT").
CSO_B	Output	Dual-Purpose	Parallel daisy chain active-Low chip select output. Not used in single FPGA applications.

Single Device SelectMAP Configuration

High-Performance Platform Flash XL SelectMAP Configuration

The Platform Flash XL is specially optimized for high-performance Virtex-5 FPGA configuration and ease-of-use. Platform Flash XL integrates 128 Mb of in-system programmable flash storage and performance features for configuration within a small-footprint FT64 package. Power-on burst read mode and dedicated I/O power supply enable Platform Flash XL to mate seamlessly with the Virtex-5 FPGA SelectMAP configuration interface. A wide, 16-bit data bus delivers the FPGA configuration bitstream at speeds to 800 Mb/s without wait states. A simplified model of the Platform Flash XL configuration solution for a Virtex-5 FPGA is shown in Figure 2-7.

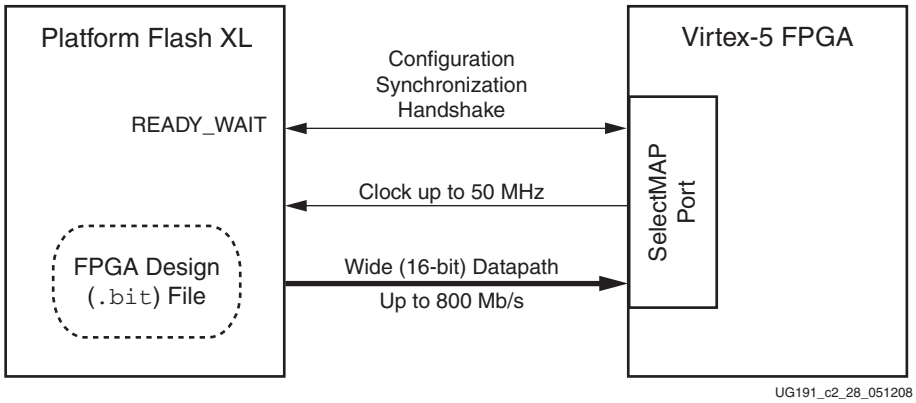


Figure 2-7: **Platform Flash XL High-Performance FPGA Configuration**

For Virtex-5 FPGA configuration, the Platform Flash XL takes advantage of the 16-bit SelectMAP feature to accomplish high-speed configuration. Minimal configuration time is achieved with an external, free-running oscillator driving the configuration clock in Slave SelectMAP mode. For Platform Flash XL details, see DS617, *Platform Flash XL High-Density Configuration and Storage Device* data sheet.

After configuration, the Virtex-5 FPGA can access any remaining memory space, beyond the bitstream, in the Platform Flash XL. The Platform Flash XL has a standard BPI NOR Flash interface. In addition to the 16-bit data bus, which is dually used for SelectMAP configuration, the Platform Flash XL has a standard address bus and read/write control pins for random access reads and for sending CFI-compliant commands.

For prototype designs, the ISE® iMPACT software provides an indirect Platform Flash XL programming solution through the IEEE Standard 1149.1 (JTAG) port of the Virtex-5 FPGA. The iMPACT software downloads a pre-built design bitstream into the Virtex-5 FPGA that bridges the FPGA JTAG port to the FPGA BPI Flash configuration interface. The FPGA BPI Flash configuration interface is a superset of the FPGA SelectMAP interface. When the Platform Flash XL's standard address and control pins are connected to the corresponding FPGA BPI Flash interface pins, the iMPACT indirect programming solution can program the Platform Flash XL with the prototype design bitstream, as shown in Figure 2-8.

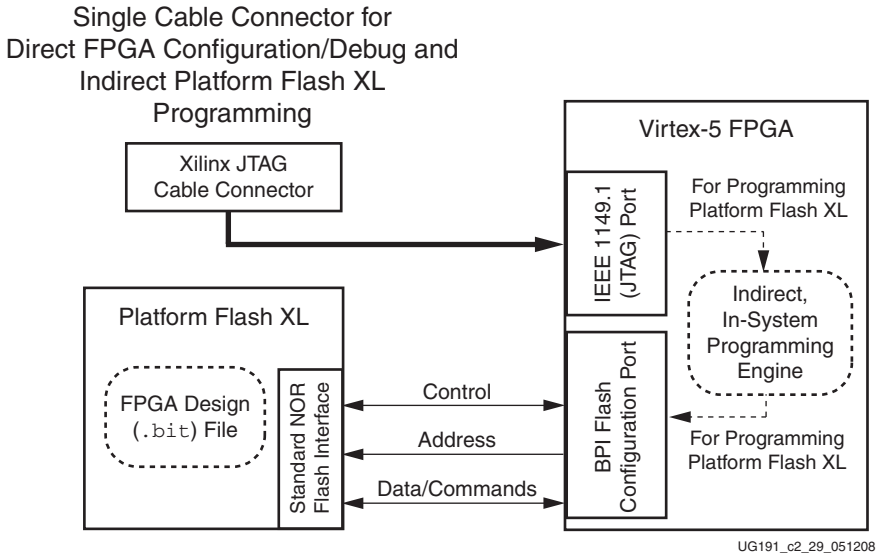


Figure 2-8: **Indirect Programming Solution for Platform Flash XL**

For details regarding the Virtex-5 FPGA SelectMAP configuration setup or the iMPACT indirect programming solution for the Platform Flash XL, see UG438, *Platform Flash XL User Guide*.

Platform Flash PROM SelectMap Configuration

The simplest way to configure a single device in SelectMAP mode is to connect it directly to a configuration PROM, as shown in Figure 2-9. In this arrangement, the device is set for Master SelectMAP mode, and the RDWR_B and CS_B pins are tied to ground for continuous data loading (see "SelectMAP Data Loading").

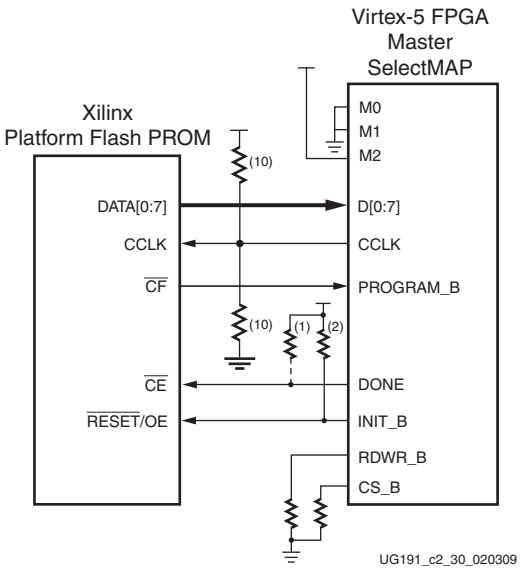


Figure 2-9: **Single Device Master SelectMAP Configuration**

Notes relevant to Figure 2-9:

1. The DONE pin is by default an open-drain output requiring an external pull-up resistor. In this arrangement, the active DONE driver can be enabled, eliminating the need for an external pull-up resistor.

The INIT_B pin is a bidirectional, open-drain pin. An external pull-up resistor is required. For a recommended value with the Platform Flash XL device, see DS617, *Platform Flash XL High-Density Configuration and Storage Device* data sheet.

2. The BitGen startup clock setting must be set for CCLK for SelectMAP configuration.
3. The PROM in this diagram represents one or more Xilinx PROMs. Multiple PROMs can be cascaded to increase the overall configuration storage capacity.
4. The `BIT` file must be reformatted into a PROM file before it can be stored on the PROM. Refer to the "Generating PROM Files" section.
5. On some Xilinx PROMs, the reset polarity is programmable. $\overline{\text{RESET}}$ should be configured as active Low when using this setup.
6. The Xilinx PROM must be set for parallel mode. This mode is not available for all devices.
7. When configuring a Virtex-5 device in SelectMAP mode from a Xilinx configuration PROM, the RDWR_B and CS_B signals can be tied Low (see "SelectMAP Data Loading").
8. The BUSY signal does not need to be monitored for this setup and can be left unconnected (see "SelectMAP Data Loading").
9. The CCLK net requires Thevenin parallel termination. See "Board Layout for Configuration Clock (CCLK)," page 73.
10. The D bus can be x8 or x16 for Master SelectMAP configuration.
11. Platform Flash PROM SelectMap configuration is specific to the Platform Flash XCFS and XCFP PROM only.

Microprocessor-driven SelectMAP Configuration

For custom applications where a microprocessor or CPLD is used to configure a single Virtex-5 device, either Master SelectMAP mode (use CCLK from the FPGA) or Slave SelectMAP mode can be used (Figure 2-10). Slave SelectMAP mode is preferred. See XAPP502, *Using a Microprocessor to Configure Xilinx FPGAs via Slave Serial or SelectMAP Mode*, for information on configuring Virtex devices using a microprocessor). Refer to the "Data Loading" section in XAPP502 for details on handling the CS_B, RDWR_B, and BUSY signals.

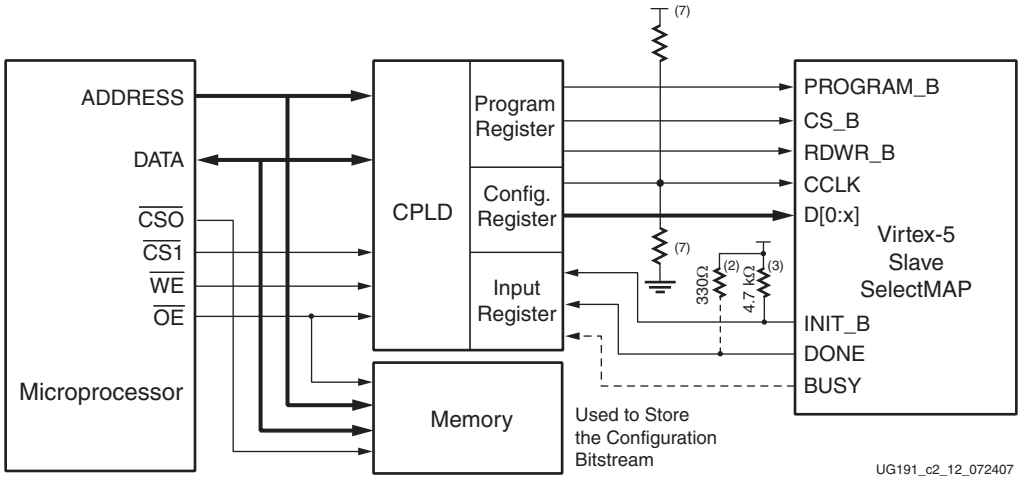


Figure 2-10: **Single Slave Device SelectMAP Configuration from Microprocessor and CPLD**

Notes relevant to Figure 2-10:

1. This schematic is from XAPP502. It is one of many possible implementations.
2. The DONE pin is by default an open-drain output requiring an external pull-up resistor. In this arrangement, the active DONE driver can be enabled, eliminating the need for an external pull-up resistor.
3. The INIT_B pin is a bidirectional, open-drain pin. An external pull-up resistor is required.
4. The BitGen startup clock setting must be set for CCLK for SelectMAP configuration.
5. The BUSY signal can be left unconnected if readback is not needed.
6. The CS_B and RDWR_B signals can be tied to ground if only one FPGA is going to be configured and readback is not needed.
7. The CCLK net requires Thevenin parallel termination. See "Board Layout for Configuration Clock (CCLK)," page 73.
8. The D bus can be x8, x16, or x32 for Slave SelectMAP configuration.

Multiple Device SelectMAP Configuration

Multiple Virtex-5 devices in Slave SelectMAP mode can be connected on a common SelectMAP bus (Figure 2-11). In a SelectMAP bus, the DATA, CCLK, RDWR_B, BUSY, PROGRAM_B, DONE, and INIT_B pins share a common connection between all of the devices. To allow each device to be accessed individually, the CS_B (Chip Select) inputs must not be tied together. External control of the CS_B signal is required and is usually provided by a microprocessor or CPLD.

If Readback is going to be performed on the device after configuration, the RDWR_B and BUSY signals must be handled appropriately. (For details, refer to Chapter 7, "Readback and Configuration Verification.")

Otherwise, RDWR_B can be tied Low and BUSY can be ignored. Unlike earlier Virtex devices, the BUSY signal never needs to be monitored when configuring Virtex-5 devices. Refer to "Bitstream Loading (Steps 4-7)" in Chapter 1 and to Chapter 7, "Readback and Configuration Verification."

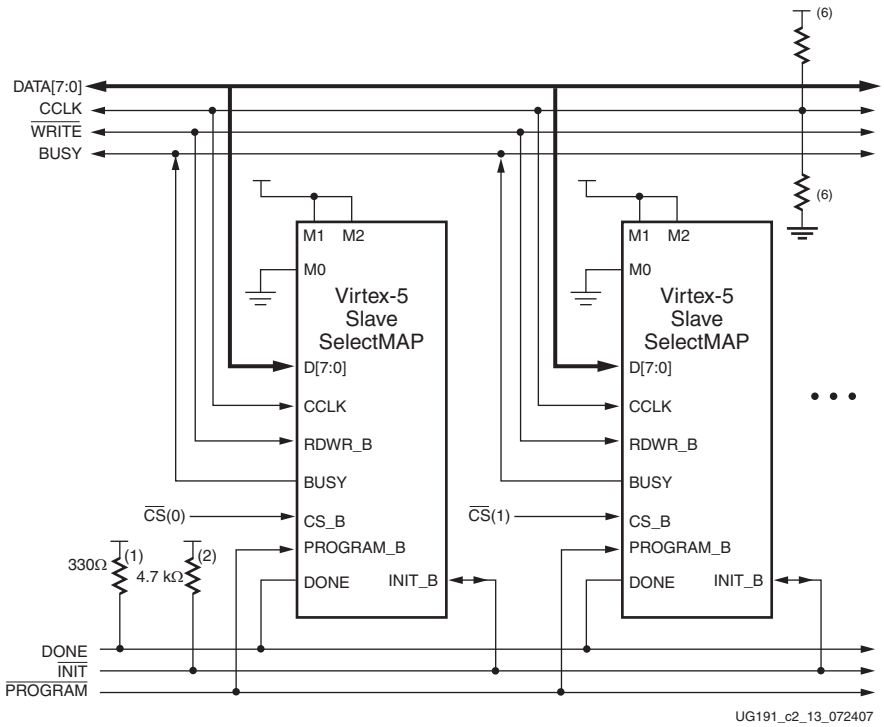


Figure 2-11: **Multiple Slave Device Configuration on an 8-Bit SelectMAP Bus**

Notes relevant to Figure 2-11:

1. The DONE pin is by default an open-drain output requiring an external pull-up resistor. In this arrangement, the active DONE driver must be disabled.
2. The INIT_B pin is a bidirectional, open-drain pin. An external pull-up resistor is required.
3. The BitGen startup clock setting must be set for CCLK for SelectMAP configuration.
4. The BUSY signals can be left unconnected if readback is not needed.
5. An external controller such as a microprocessor or CPLD is needed to control configuration.
6. The CCLK net requires Thevenin parallel termination. See "Board Layout for Configuration Clock (CCLK)," page 73.
7. The data bus can be x8, x16, or x32.

Parallel Daisy Chain

Virtex-5 FPGA configuration supports parallel daisy-chain. Figure 2-12 shows an example schematic of the leading device in BPI mode. The leading device can also be in Master or Slave SelectMAP modes. The D[15:0], CCLK, RDWR_B, PROGRAM_B, DONE, and INIT_B pins share a common connection between all of the devices. The CS_B pins are daisy chained.

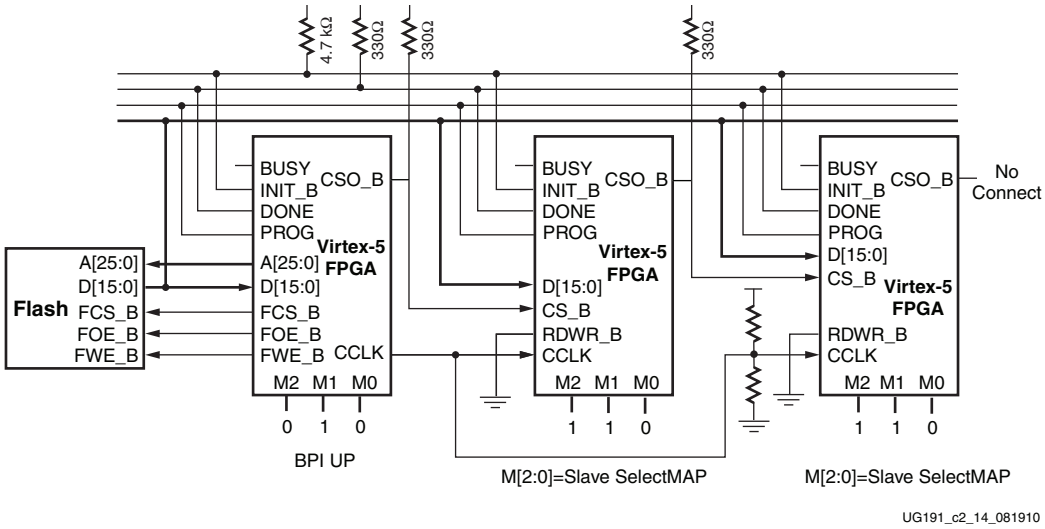


Figure 2-12: **Parallel Daisy Chain**

Notes relevant to Figure 2-12:

1. The DONE pin is by default an open-drain output requiring an external pull-up resistor. In this arrangement, the active DONE driver must be disabled.
2. The INIT_B pin is a bidirectional, open-drain pin. An external pull-up is required.
3. The BitGen startup clock setting must be set for CCLK for SelectMAP configuration.
4. The BUSY signals can be left unconnected if readback is not needed.
5. The CCLK net requires Thevenin parallel termination. See "Board Layout for Configuration Clock (CCLK)."
6. The FCS_B, FWE_B, FOE_B, CSO_B weak pull-up resistors should be enabled, otherwise external pull-up resistors are required for each pin. By default, all dual-mode I/Os have weak pull-downs after configuration.
7. The first device in the chain can be Master SelectMAP, Slave SelectMAP, BPI-Up, or BPI-Down.
8. Readback in the parallel daisy chain scheme is currently not supported.
9. AES decryption is not available in x16 or x32 mode, only in x8 mode.
10. Fallback is not supported in parallel daisy-chain.

Ganged SelectMAP

It is also possible to configure simultaneously multiple devices with the same configuration bitstream by using a ganged SelectMAP configuration. In a ganged SelectMAP arrangement, the CS_B pins of two or more devices are connected together (or tied to ground), causing all devices to recognize data presented on the D pins.

All devices can be set for Slave SelectMAP mode if an external oscillator is available, or one device can be designated as the Master device, as illustrated in Figure 2-13.

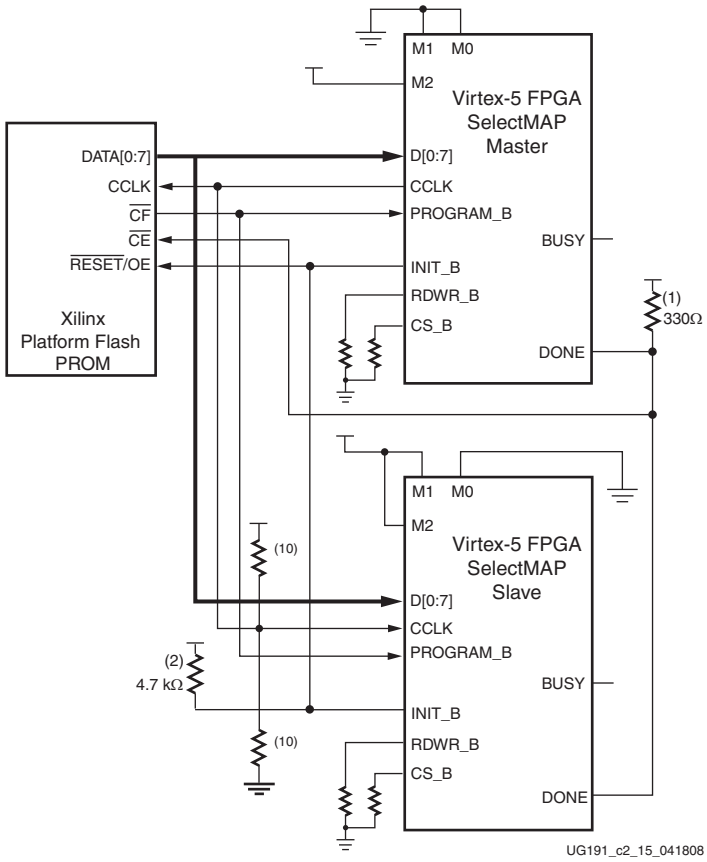


Figure 2-13: **Ganged x8 SelectMAP Configuration**

Notes relevant to Figure 2-13:

1. The DONE pin is by default an open-drain output requiring an external pull-up resistor. In this arrangement, the active DONE driver must be disabled for both devices.
2. The INIT_B pin is a bidirectional, open-drain pin. An external pull-up resistor is required.
3. The BitGen startup clock setting must be set for CCLK for SelectMAP configuration.
4. The BUSY signal is not used for ganged SelectMAP configuration.
5. The PROM in this diagram represents one or more Xilinx PROMs. Multiple Xilinx PROMs can be cascaded to increase the overall configurations storage capacity.
6. The `BIT` file must be reformatted into a PROM file before it can be stored on the Xilinx PROM. Refer to the "Generating PROM Files" section.

7. On some Xilinx PROMs, the reset polarity is programmable. Reset should be configured as active Low when using this setup.
8. The Xilinx PROM must be set for parallel mode. This mode is **not** available for all devices.
9. When configuring a Virtex-5 device in SelectMAP mode from a Xilinx configuration PROM, the RDWR_B and CS_B signals can be tied Low (see "SelectMAP Data Loading").
10. The CCLK net requires Thevenin parallel termination. See "Board Layout for Configuration Clock (CCLK)," page 73.
11. Ganged SelectMap configuration is specific to the Platform Flash XCFS and XCFP PROM only.

If one device is designated as the Master, the DONE pins of all devices must be connected with the active DONE drivers disabled. An external pull-up resistor is required on the common DONE signal. Designers must carefully focus on signal integrity due to the increased fanout of the outputs from the PROM. Signal integrity simulation is recommended.

Readback is not possible if the CS_B signals are tied together, because all devices simultaneously attempt to drive the D signals.

SelectMAP Data Loading

The SelectMAP interface allows for either continuous or non-continuous data loading. Data loading is controlled by the CS_B, RDWR_B, CCLK, and BUSY signals.

CS_B

The Chip Select input (CS_B) enables the SelectMAP bus. When CS_B is High, the Virtex-5 device ignores the SelectMAP interface, neither registering any inputs nor driving any outputs. D and BUSY are placed in a High-Z state, and RDWR_B is ignored.

- If CS_B = 0, the device's SelectMAP interface is enabled.
- If CS_B = 1, the device's SelectMAP interface is disabled.

For a multiple device SelectMAP configuration, refer to Figure 2-12.

If only one device is being configure through the SelectMAP and readback is not required, or if ganged SelectMAP configuration is used, the CS_B signal can be tied to ground, as illustrated in Figure 2-9 and Figure 2-13.

RDWR_B

RDWR_B is an input to the Virtex-5 device that controls whether the data pins are inputs or outputs:

- If RDWR_B = 0, the data pins are inputs (writing to the FPGA).
- If RDWR_B = 1, the data pins are outputs (reading from the FPGA).

For configuration, RDWR_B must be set for write control (RDWR_B = 0). For readback, RDWR_B must be set for read control (RDWR_B = 1) while CS_B is deasserted. (For details, refer to Chapter 7, "Readback and Configuration Verification.")

Changing the value of RDWR_B while CS_B is asserted triggers an ABORT if the device gets a rising CCLK edge (see "SelectMAP ABORT"). If readback is not needed, RDWR_B can be tied to ground or used for debugging with SelectMAP ABORT.

The RDWR_B signal is ignored while CS_B is deasserted. Read/write control of the 3-stating of the data pins is asynchronous. The FPGA actively drives SelectMAP data without regard to CCLK if RDWR_B is set for read control (RDWR_B = 1, Readback) while CS_B is asserted.

CCLK

All activity on the SelectMAP data bus is synchronous to CCLK. When RDWR_B is set for write control (RDWR_B = 0, Configuration), the FPGA samples the SelectMAP data pins on rising CCLK edges. When RDWR_B is set for read control (RDWR_B = 1, Readback), the FPGA updates the SelectMAP data pins on rising CCLK edges.

In Slave SelectMAP mode, configuration can be paused by stopping CCLK (see "Non-Continuous SelectMAP Data Loading").

BUSY

BUSY is an output from the FPGA indicating when the device is ready to drive readback data. Unlike earlier Virtex devices, Virtex-5 FPGAs never drive the BUSY signal during configuration, even at the maximum configuration frequency with an encrypted bitstream. The Virtex-5 device only drives BUSY during readback. (For details, refer to Chapter 7, "Readback and Configuration Verification.")

- If BUSY = 0 during readback, the SelectMAP data pins are driving valid readback data.
- If BUSY = 1 during readback, the SelectMAP data pins are not driving valid readback data.

When CS_B is deasserted (CS_B = 1), the BUSY pin is placed in a High-Z state.

BUSY remains in a High-Z state until CS_B is asserted. If CS_B is asserted before power-up (that is, if the pin is tied to ground), BUSY initially is in a High-Z state, then driven Low after POR finishes, usually a few milliseconds (T_{BUSY}), after V_{CCINT} reaches V_{POR} but before INIT_B goes High.

Unless readback is used, the BUSY pin can be left unconnected.

Continuous SelectMAP Data Loading

Continuous data loading is used in applications where the configuration controller can provide an uninterrupted stream of configuration data. After power-up, the configuration controller sets the RDWR_B signal for write control (RDWR_B = 0) and asserts the CS_B signal (CS_B = 0), causing the device to drive BUSY Low (this transition is asynchronous). RDWR_B must be driven Low before CS_B is asserted, otherwise an ABORT occurs (see "SelectMAP ABORT").

On the next rising CCLK edge, the device begins sampling the data pins. Only D[0:7] are sampled by Configuration until the bus width is determined. See "Bus Width Auto Detection" for details. After bus width is determined, the proper width of the data bus is sampled for the Synchronization word search. Configuration begins after the synchronization word is clocked into the device.

After the configuration bitstream is loaded, the device enters the startup sequence. The device asserts its DONE signal High in the phase of the startup sequence that is specified by the bitstream (see "Startup (Step 8)" in Chapter 1). The configuration controller should continue sending CCLK pulses until after the startup sequence has finished. (This can require several CCLK pulses after DONE goes High. See "Startup (Step 8)" in Chapter 1 for

details).

After configuration, the CS_B and RDWR_B signals can be deasserted, or they can remain asserted. Because the SelectMAP port is inactive, toggling RDWR_B at this time does not cause an abort. Figure 2-14 summarizes the timing of SelectMAP configuration with continuous data loading.

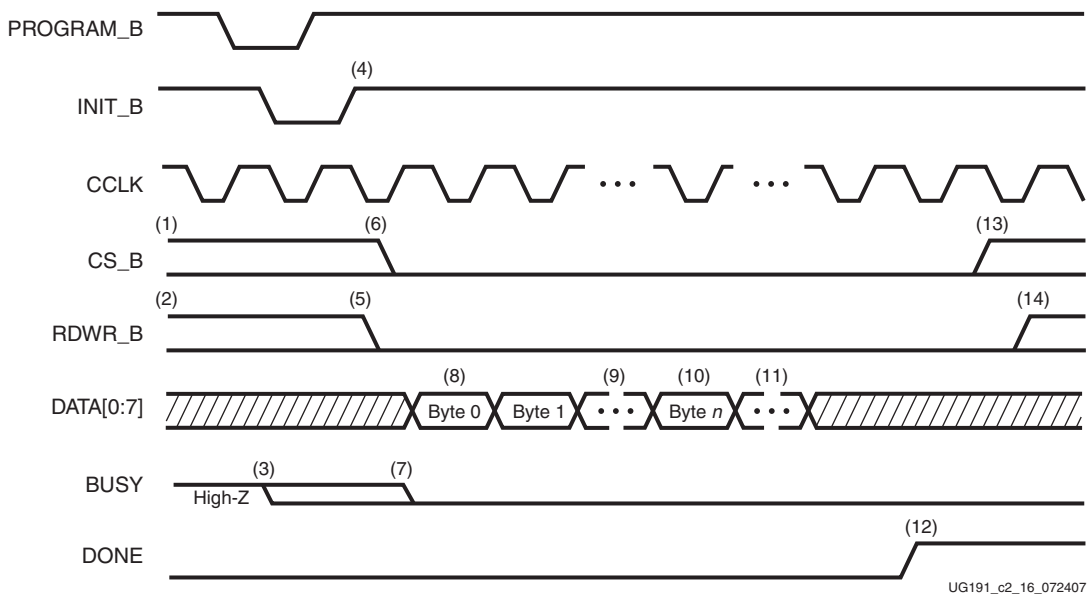


Figure 2-14: **Continuous x8 SelectMAP Data Loading**

Notes relevant to Figure 2-14:

1. CS_B signal can be tied Low if there is only one device on the SelectMAP bus. If CS_B is not tied Low, it can be asserted at any time.
2. RDWR_B can be tied Low if readback is not needed. RDWR_B should not be toggled after CS_B has been asserted because this triggers an ABORT. (See "SelectMAP ABORT").
3. If CS_B is tied Low, BUSY is driven Low before INIT_B toggles High.
4. The Mode pins are sampled when INIT_B goes High.
5. RDWR_B should be asserted before CS_B to avoid causing an abort.
6. CS_B is asserted, enabling the SelectMAP interface.
7. BUSY remains in High-Z state until CS_B is asserted.
8. The first byte is loaded on the first rising CCLK edge after CS_B is asserted.
9. The configuration bitstream is loaded one byte per rising CCLK edge.
10. After the last byte is loaded, the device enters the startup sequence.
11. The startup sequence lasts a minimum of eight CCLK cycles. (See "Startup (Step 8)" in Chapter 1.)
12. The DONE pin goes High during the startup sequence. Additional CCLKs can be required to complete the startup sequence. (See "Startup (Step 8)" in Chapter 1.)
13. After configuration has finished, the CS_B signal can be deasserted.
14. After the CS_B signal is deasserted, RDWR_B can be deasserted.
15. The data bus can be x8, x16, or x32.

Non-Continuous SelectMAP Data Loading

Non-continuous data loading is used in applications where the configuration controller cannot provide an uninterrupted stream of configuration data—for example, if the controller pauses configuration while it fetches additional data.

Configuration can be paused in two ways: by deasserting the CS_B signal (Free-Running CCLK method, Figure 2-15) or by halting CCLK (Controlled CCLK method, Figure 2-16).

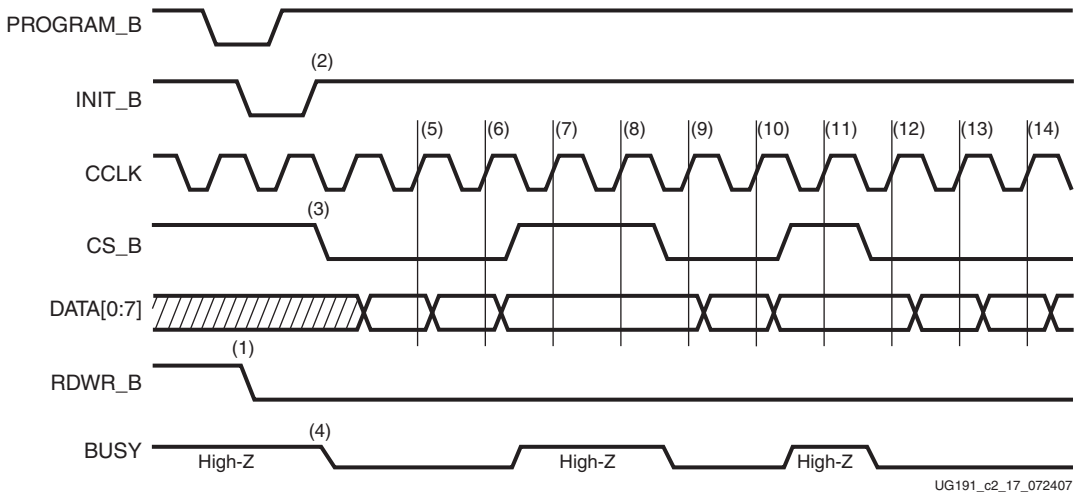


Figure 2-15: **Non-Continuous SelectMAP Data Loading with Free-Running CCLK**

Notes relevant to Figure 2-15:

1. RDWR_B is driven Low by the user, setting the D[0:7] pins as inputs for configuration. RDWR_B can be tied Low if readback is not needed. RDWR_B should not be toggled after CS_B has been asserted because this triggers an ABORT. (See "SelectMAP ABORT").
2. The device is ready for configuration after INIT_B toggles High.
3. The user asserts CS_B Low, enabling the SelectMAP data bus. CS_B signal can be tied Low if there is only one device on the SelectMAP bus. If CS_B is not tied Low, it can be asserted at any time.
4. BUSY goes Low shortly after CS_B is asserted. If CS_B is tied Low, BUSY is driven Low before INIT_B toggles High.
5. A byte is loaded on the rising CCLK edge. The data bus can be x8, x16, or x32 wide.
6. A byte is loaded on the rising CCLK edge.
7. The user deasserts CS_B, and the byte is ignored.
8. The user deasserts CS_B, and the byte is ignored.
9. A byte is loaded on the rising CCLK edge.
10. A byte is loaded on the rising CCLK edge.
11. The user deasserts CS_B, and the byte is ignored.
12. A byte is loaded on the rising CCLK edge.
13. A byte is loaded on the rising CCLK edge.
14. A byte is loaded on the rising CCLK edge.

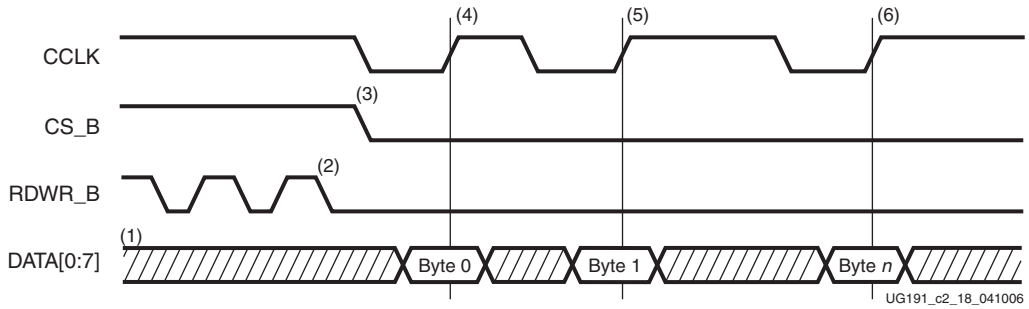


Figure 2-16: **Non-Continuous SelectMAP Data Loading with Controlled CCLK**

Notes relevant to Figure 2-16:

1. The Data pins are in the High-Z state while CS_B is deasserted. The data bus can be x8, x16, or x32.
2. RDWR_B has no effect on the device while CS_B is deasserted.
3. CS_B is asserted by the user. The device begins loading configuration data on rising CCLK edges.
4. A byte is loaded on the rising CCLK edge.
5. A byte is loaded on the rising CCLK edge.
6. A byte is loaded on the rising CCLK edge.

SelectMAP ABORT

An ABORT is an interruption in the SelectMAP configuration or readback sequence occurring when the state of RDWR_B changes while CS_B is asserted. During a configuration ABORT, internal status is driven onto the D[7:4] pins over the next four CCLK cycles. The other D pins are always High. After the ABORT sequence finishes, the user can resynchronize the configuration logic and resume configuration. For applications that must deassert RDWR_B between bytes, see Controlled CCLK method, Figure 2-16.

Configuration Abort Sequence Description

An ABORT is signaled during configuration as follows:

1. The configuration sequence begins normally.
2. The user pulls the RDWR_B pin High while the device is selected (CS_B asserted Low).
3. BUSY goes High if CS_B remains asserted (Low). The FPGA drives the status word onto the data pins if RDWR_B remains set for read control (logic High).
4. The ABORT lasts for four clock cycles, and Status is updated.

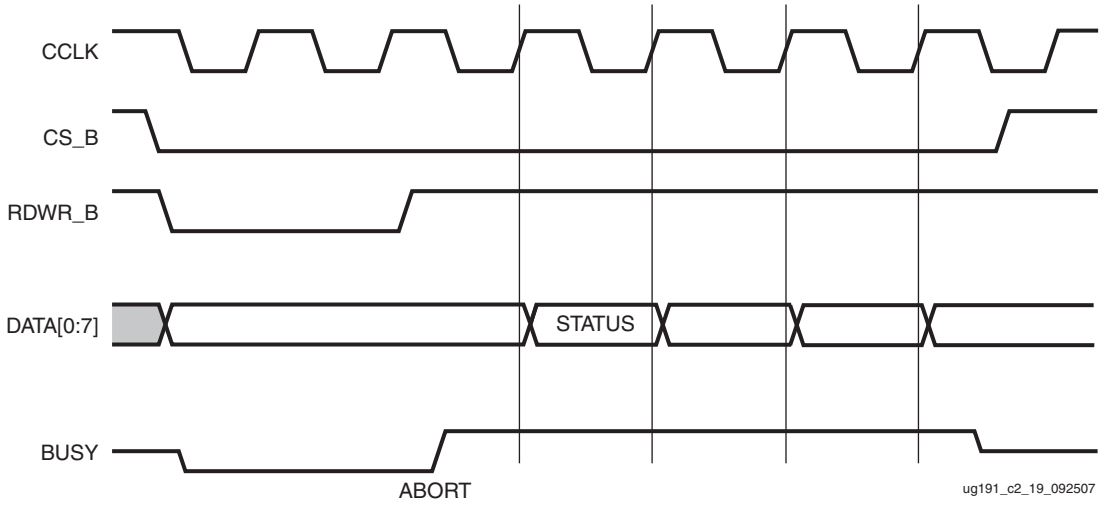


Figure 2-17: **Configuration Abort Sequence for SelectMAP Modes**

Readback Abort Sequence Description

An ABORT is signaled during readback as follows:

1. The readback sequence begins normally.
2. The user pulls the RDWR_B pin Low while the device is selected (CS_B asserted Low).
3. BUSY goes High if CS_B remains asserted (Low).
4. The ABORT ends when CS_B is deasserted.

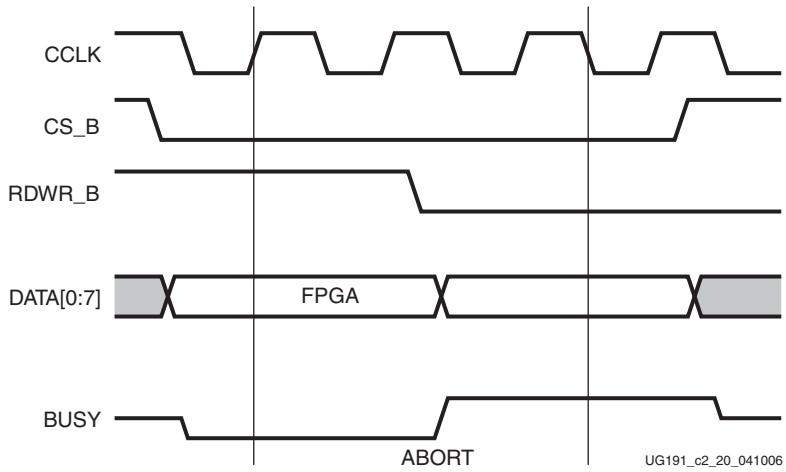


Figure 2-18: **Readback Abort Sequence**

ABORTs during readback are *not* followed by a status word because the RDWR_B signal is set for write control (FPGA D[x:0] pins are inputs).

ABORT Status Word

During the configuration ABORT sequence, the device drives a status word onto the D[7:0] pins. The status bits do not bit-swap. The other data pins are always High. The key for that status word is given in Table 2-5.

Table 2-5: **ABORT Status Word**

Bit Number	Status Bit Name	Meaning
D7	CFGERR_B	Configuration error (active Low) 0 = A configuration error has occurred. 1 = No configuration error.
D6	DALIGN	Sync word received (active High) 0 = No sync word received. 1 = Sync word received by interface logic.
D5	RIP	Readback in progress (active High) 0 = No readback in progress. 1 = A readback is in progress.
D4	IN_ABORT_B	ABORT in progress (active Low) 0 = Abort is in progress. 1 = No abort in progress.
D3-D0	1111	Fixed to ones.

The ABORT sequence lasts four CCLK cycles. During those cycles, the status word changes to reflect data alignment and ABORT status. A typical sequence might be:

```
11011111  => DALIGN = 1,  IN_ABORT_B = 1
11001111  => DALIGN = 1,  IN_ABORT_B = 0
10001111  => DALIGN = 0,  IN_ABORT_B = 0
10011111  => DALIGN = 0,  IN_ABORT_B = 1
```

After the last cycle, the synchronization word can be reloaded to establish data alignment.

Resuming Configuration or Readback After an Abort

There are two ways to resume configuration or readback after an ABORT:

- The device can be resynchronized after the ABORT completes.
- The device can be reset by pulsing PROGRAM_B Low at any time.

To resynchronize the device, CS_B must be deasserted then reasserted. Configuration or readback can be resumed by sending the last configuration or readback packet that was in progress when the ABORT occurred. Alternatively, configuration or readback can be restarted from the beginning.

SelectMAP Reconfiguration

The term *reconfiguration* refers to reprogramming an FPGA after its DONE pin has gone High. Reconfiguration can be initiated by pulsing the PROGRAM_B pin (this method is identical to configuration) or by resynchronizing the device and sending configuration data.

To reconfigure a device in SelectMAP mode without pulsing PROGRAM_B, the BitGen **`persist`** option must be set—otherwise, the DATA pins become user I/O after configuration. The persist option must also be selected for the new bitstream reconfiguring the device. RS[1:0], CSO_B, and A[19:16] pins are not available for User mode when **`persist`** is on. Reconfiguration must be enabled in BitGen. By default, the SelectMAP 8 interface (D0–D7) is preserved unless another SelectMAP width has been selected with the CONFIG_MODE constraint.

Reconfiguration begins when the synchronization word is clocked into the SelectMAP port. The remainder of the operation is identical to configuration as described above.

SelectMAP Data Ordering

In many cases, SelectMAP configuration is driven by a user application residing on a microprocessor, CPLD, or in some cases another FPGA. In these applications, it is important to understand how the data ordering in the configuration data file corresponds to the data ordering expected by the FPGA.

In SelectMAP x8 mode, configuration data is loaded at one byte per CCLK, with the MSB of each byte presented to the D0 pin. This convention (D0 = MSB, D7 = LSB) *differs* from many other devices. For x16 and x32 modes, see "Parallel Bus Bit Order." This convention can be a source of confusion when designing custom configuration solutions. Table 2-6 shows how to load the hexadecimal value `0xABCD` into the SelectMAP data bus.

Table 2-6: **Bit Ordering for SelectMAP 8-Bit Mode**

CCLK Cycle	Hex Equivalent	D0	D1	D2	D3	D4	D5	D6	D7
1	0xAB	1	0	1	0	1	0	1	1
2	0xCD	1	1	0	0	1	1	0	1

Notes:

1. D[0:7] represent the SelectMAP DATA pins.

Some applications can accommodate the non-conventional data ordering without difficulty. For other applications, it can be more convenient for the source configuration data file to be *bit swapped*, meaning that the bits in each byte of the data stream are reversed. For these applications, the Xilinx PROM file generation software can generate bit-swapped PROM files (see "Configuration Data File Formats").

Figure 2-19 shows the bit ordering for x8, x16, and x32 modes. It also shows the bit ordering for Virtex-4 FPGA x32 mode.

Virtex-5 Mode	Pin																													
	31	**30**	**29**	**28**	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
x32	24	25	26	27	28	29	30	31	16	17	18	19	20	21	22	23	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7
x16																	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7
x8																									0	1	2	3	4	5	6	7
Virtex-4 x32 Mode	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

Figure 2-19: **Bit Ordering**

SPI Configuration Interface

In SPI serial Flash mode, M[2:0]=001. The Virtex-5 FPGA configures itself from an attached industry-standard SPI serial Flash PROM. Although SPI is a standard four-wire interface, various available SPI Flash memories use different read commands and protocol. Besides M[2:0], FS[2:0] pins are sampled by the INIT_B rising edge to determine the type of read commands used by SPI Flash (see Table 2-8). For Virtex-5 FPGA configurations, the default address always starts from 0. Figure 2-20 shows the SPI related configuration pins, and the standard connection between Virtex-5 devices and SPI Flash.

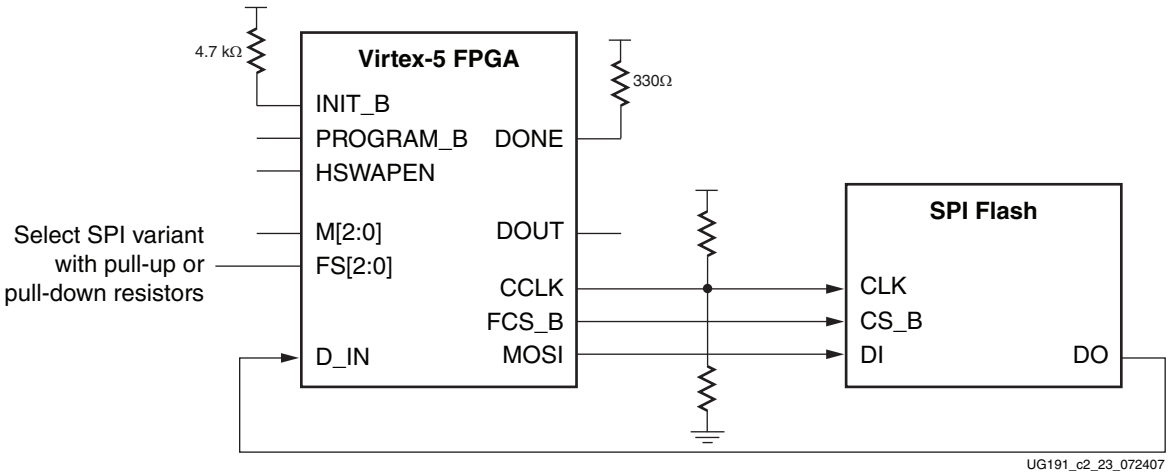


Figure 2-20: **Virtex-5 Device SPI Configuration Interface**

Notes related to Figure 2-20:

- FCS_B and MOSI are clocked by the CCLK falling edge.
- D_IN is clocked on the rising edge of the CCLK.
- CCLK and D_IN are dedicated Configuration I/Os.
- FCS_B is a dual-mode I/O. MOSI is a dual-mode I/O, muxed with FOE_B. FS[2:0] are dual-mode I/Os sampled on the INIT_B rising edge, muxed with D[2:0].
- The internal I/O pull-up resistors should be enabled for FCS_B, MOSI, and D_IN.
- There are additional pins on the SPI Flash side, such as Write Protect and Hold. These pins are not used in FPGA configuration (read only). But they should be tied off appropriately according to the SPI vendor's specification.
- If HSWAPEN is left unconnected or tied High, a pull-up resistor is required for FCS_B and MOSI.
- If HSWAPEN is tied Low, the FCS_B and MOSI pins have internal weak pull-up resistors during configuration. After configuration, FCS_B and MOSI can be either controlled by I/O in user mode or by enabling a weak pull-up resistor through constraints.
- HSWAPEN must be connected to either disable or enable the pull-up resistors.
- CCLK always has a weak internal pull-up resistor. The CCLK frequency can be adjusted using the **`ConfigRate`** BitGen option.
- To enable the active driver on DONE, the **`DriveDONE`** option in BitGen must be enabled.

- When DCI match wait or DCM lock wait is enabled before the DONE release cycle during startup, the FPGA continues to clock in data until the startup wait condition is met and DONE is released. See "MultiBoot Bitstream Spacing" in Chapter 8 for considerations specific to MultiBoot Configuration.

Table 2-7 describes the SPI configuration interface pins.

Table 2-7: **Virtex-5 Device SPI Configuration Interface Pins**

Pin Name	Type	Dedicated or Dual-Purpose	Description
M[2:0]	Input	Dedicated	Mode pins – `001` for SPI
HSWAPEN	Input	Dedicated	Controls I/O (except bank0 dedicated I/Os) Pull-up during configuration. A weak pull-up resistor is built into this pin. `0` = Pull-up during configuration `1` = 3-stated during configuration
DOUT	Three-State Output	Dedicated	Used for serial daisy chain configuration.
DONE	Bidirectional, Open-Drain, or Active	Dedicated	Active-High signal indicating configuration is complete: `0` = FPGA not configured `1` = FPGA configured
INIT_B	Input or Output, Open-Drain	Dedicated	Before the Mode pins are sampled, INIT_B is an input that can be held Low to delay configuration. After the Mode pins are sampled, INIT_B is an open-drain active Low output indicating whether a CRC error occurred during configuration: `0` = CRC error `1` = No CRC error When the SEU detection function is enabled, INIT_B is optionally driven Low when read back CRC error is detected.
PROGRAM_B	Input	Dedicated	Active-Low asynchronous full-chip reset
FS[2:0]	Input	Dual-Purpose	SPI Variant Select pins, sampled by the INIT_B rising edge. They are multiplexed with the DATA[2:0] pins.
CCLK	Output	Dedicated	Configuration clock output (to SPI).

Table 2-7: **Virtex-5 Device SPI Configuration Interface Pins *(Continued)***

Pin Name	Type	Dedicated or Dual-Purpose	Description
FCS_B	Output	Dual-Purpose	Active-Low chip select output, clocked by the CCLK falling edge.
MOSI	Output	Dual-Purpose	FPGA serial data output, clocked by the CCLK falling edge. This pin is multiplexed with the FOE_B pin.
D_IN	Input	Dedicated	FPGA serial data input (from SPI), sampled by the CCLK rising edge.
RCMD[7:0]	Input	Dual-Purpose	SPI read command strapping inputs RCMD[7:0] (muxed on ADDR[7:0]) when FS[2:0] = 001. Sampled on the rising edge of INIT_B when used for SPI read command strapping.

Table 2-8 defines the SPI read command based on the FS[2:0] settings.

Table 2-8: **Virtex-5 Device SPI Read Command Variant Select Table**

FS[2:0]	SPI Read Command	Comments
000	0xFF	
001	RCMD[7:0]	RCMD[7:0] on ADDR[7:0] are sampled by the INIT_B rising edge, along with M[2:0] and FS[2:0]. RCMD[7:0] can be used to support any SPI read commands not supported here. The timing requirements for FS[2:0] and RCMD[7:0] are the same as for M[2:0].
010	0x52	
011	Reserved	
100	0x55	
101	0x03	
110	0xE8	
111	0x0B	

The Virtex-5 SPI flash timing diagram is shown in Figure 2-21.

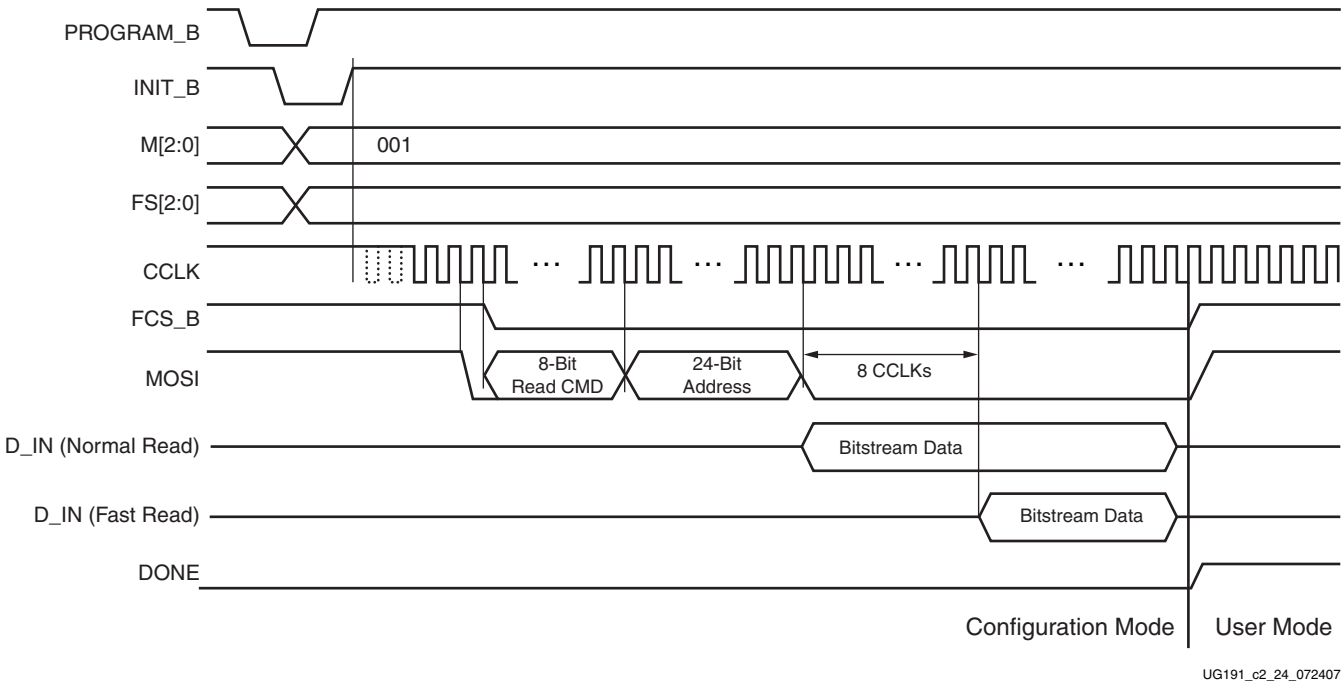


Figure 2-21: **Virtex-5 Device SPI Flash Timing Diagram**

Notes related to Figure 2-21:

- The Virtex-5 FPGA stops loading the bitstream after the DONE pin goes High. FCS_B and MOSI can be used as user I/Os.
- FCS_B is either controlled by user logic after configuration or pulled up by a pull-up resistor enabled through constraints.

For supported flash devices, consult the iMPACT help menu indirect programming tables for Virtex-5 FPGA support. Other flash devices not listed in the help menu can be compatible with Virtex-5 devices.

Power-On Sequence Precautions

At power-on, the FPGA automatically starts its configuration procedure. When the FPGA is in Master SPI configuration mode, the FPGA asserts FCS_B Low to select the SPI Flash and drives a read command to the SPI Flash. The SPI Flash must be awake and ready to receive commands before the FPGA drives FCS_B Low and sends the read command.

Because different power rails can supply the FPGA and SPI Flash or because the FPGA and SPI flash can respond at different times along the ramp of a shared power supply, special attention to the FPGA and SPI Flash power-on sequence or power-on ramps is essential. The power-on sequence or power supply ramps can cause the FPGA to awake before the SPI Flash or vice versa. In addition, some SPI Flash devices specify a minimum time period, which can be several milliseconds from power-on, during which the device must not be selected. For many systems with near-simultaneous power supply ramps, the FPGA power-on reset time (TPOR) can sufficiently delay the start of the FPGA configuration procedure such that the SPI Flash becomes ready before the start of the FPGA configuration procedure. In general, the system design must consider the affect of the power sequence, the power ramps, FPGA power-on reset timing, and SPI Flash power-up timing on the timing relation between the start of FPGA configuration and the readiness of the SPI Flash. Check DS202, *Virtex-5 FPGA Data Sheet: DC and Switching Characteristics* for

Virtex-5 FPGA power supply requirements and timing. Check the SPI Flash data sheet for the SPI Flash power-up timing requirements.

One of the following system design approaches can ensure that the SPI Flash is ready to receive commands before the FPGA starts its configuration procedure:

- Control the sequence of the power supplies such that the SPI Flash is certain to be powered and ready for asynchronous reads before the FPGA begins its configuration procedure.
- Hold the FPGA PROGRAM_B pin Low from power-up to delay the start of the FPGA configuration procedure and release the PROGRAM_B pin to High after the SPI flash is fully powered and is able to receive commands.
- Hold the FPGA INIT_B pin Low from power-up to delay the start of the FPGA configuration procedure and release the INIT_B pin to High after the SPI flash becomes ready to receive commands.

For more information on how to configure FPGAs with SPI Flash and how to use iMPACT software perform in-system SPI programming, see XAPP951, *Configuring Xilinx FPGAs with SPI Serial Flash*.

SPI Serial Daisy Chain

In a serial daisy chain application, the leading device can be in SPI mode and all downstream devices in Slave Serial mode. In this case, all configuration bitstreams can be stored inside one SPI device. The bitstream format for master and slave serial daisy chains is exactly the same. See "Serial Daisy Chains" for details.

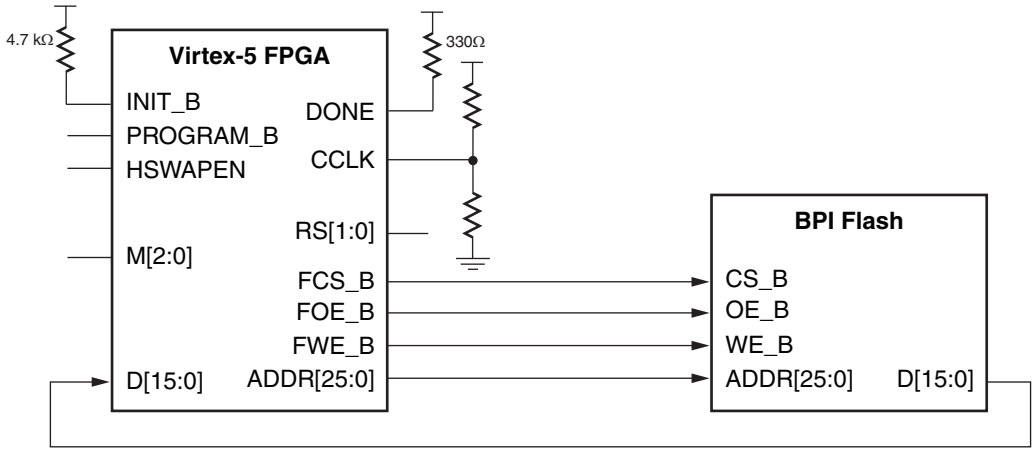
Byte Peripheral Interface Parallel Flash Mode

In BPI-Up (M[2:0]=010) or BPI-Down (M[2:0] = 011) mode, the Virtex-5 FPGA configures itself from an industry-standard parallel NOR Flash PROM, as illustrated in Figure 2-22. The FPGA drives up to 26 address lines to access the attached parallel Flash. For configuration, only async read mode is used, where the FPGA drives the address bus, and the Flash PROM drives back the bitstream data. Bus widths of x8 and x16 are supported. Bus widths are auto detected, as described in "Bus Width Auto Detection." Refer to DS617, *Platform Flash XL High-Density Configuration and Storage Device* data sheet for the BPI-compatible Flash device from Xilinx.

In BPI modes, the CCLK output is not connected to the BPI Flash device. However, Flash data is still sampled on the rising edge of CCLK. The CCLK output is driven during the BPI modes and therefore must receive the same parallel termination as in the other Master modes. See "Board Layout for Configuration Clock (CCLK)," page 73. The timing parameters related to BPI use CCLK as a reference. Virtex-5 BPI modes also support asynchronous page-mode reads to allow an increase in the CCLK frequency. See "Page Mode Support," page 71 for details.

In the BPI-Up mode, the address starts at 0 and increments by 1 until the DONE pin is asserted. If the address reaches the maximum value (26'h3FFFFF) and configuration is not done (DONE is not asserted), an error flag is raised in the status register, and fallback reconfiguration starts. See "Fallback MultiBoot," page 153.

In the BPI-Down mode, the address start at 26'h3FFFFF and decrements by 1 until the DONE pin is asserted. If the address reaches the bottom (26'h0), and configuration is still not done (DONE is not asserted), an error flag is raised in the status register and fallback reconfiguration starts. See "Fallback MultiBoot," page 153.



Note: The BPI Flash vendor data sheet should be referred to for details on the specific Flash signal connectivity.
To prevent address misalignment, close attention should be paid to the Flash family address LSB for the byte/word mode used.
Not all Flash families use the A0 as the address LSB.

UG191_c2_25_061108

Figure 2-22: **Virtex-5 BPI Configuration Interface**

Additional notes related to Figure 2-22:

- M[2:0] = 010 for BPI-Up mode and M[2:0]=011 for BPI-Down mode.
- Figure 2-22 shows the x16 BPI interface. For x8 BPI interfaces, only D[7:0] are used. See "Bus Width Auto Detection."

- Sending a bitstream to the data pin follows the same bit-swapping rule as in SelectMAP mode. See "Parallel Bus Bit Order."
- If Flash programming is not required, FCS_B, FOE_B, and FWE_B can be tied off; that is, DONE is connected to FCS_B, FOE_B is tied Low, and FWE_B is tied High.
- The CCLK outputs are not used to connect to Flash but are used to sample Flash read data during configuration. All timings are referenced to CCLK. The CCLK pin must *not* be driven or tied High or Low.
- The RS[1:0] pins are not connected as shown in Figure 2-22. These output pins are only required for MultiBoot configuration. See Chapter 8, "Reconfiguration and MultiBoot."
- HSWAPEN must be connected to either disable or enable the pull-up resistors.
- If HSWAPEN is left unconnected or tied High, a pull-up resistor is required for FCS_B.
- If HSWAPEN is tied Low, the FCB_B, FOE_B, FWE_B, and the address pins have internal weak pull-up resistors during configuration. After configuration, FCS_B can be either controlled by I/O in user mode or by enabling a weak pull-up resistor through constraints.
- To enable the active driver on DONE, the **DriveDONE** option in BitGen must be enabled.
- "MultiBoot Bitstream Spacing," page 155 provides information on when DCI or DCM lock wait is turned on.
- For daisy chaining FPGAs in BPI mode, see Figure 2-12, page 52.
- The BPI Flash vendor data sheet should be referred to for details on the specific Flash signal connectivity. To prevent address misalignment, close attention should be paid to the Flash family address LSB for the byte/word mode used. Not all Flash families use the A0 as the address LSB.

Table 2-9 defines the BPI configuration interface pins.

If the FPGA is subject to reprogramming or fallback during configuration from the BPI flash, then the INIT pin can be connected to the BPI reset to set the BPI into a known state.

Table 2-9: **Virtex-5 Device BPI Configuration Interface Pins**

Pin Name	Type	Dedicated or Dual-Purpose	Description
M[2:0]	Input	Dedicated	The Mode pins determine the BPI mode: 010 = BPI-Up mode 011 = BPI-Down mode
HSWAPEN	Input	Dedicated	Controls I/O (except Bank 0 dedicated I/Os) pull-up resistors during configuration. This pin has a built-in weak pull-up resistor. 0 = Pull-up during configuration 1 = 3-state during configuration
DONE	Bidirectional, Open-Drain, or Active	Dedicated	Active-High signal indicating configuration is complete: 0 = FPGA not configured 1 = FPGA configured

Table 2-9: **Virtex-5 Device BPI Configuration Interface Pins *(Continued)***

Pin Name	Type	Dedicated or Dual-Purpose	Description
INIT_B	Input or Output, Open-Drain	Dedicated	Before the Mode pins are sampled, INIT_B is an input that can be held Low to delay configuration. After the Mode pins are sampled, INIT_B is an open-drain, active-Low output indicating whether a CRC error occurred during configuration: 0 = CRC error 1 = No CRC error When the SEU detection function is enabled, INIT_B is optionally driven Low when a read back CRC error is detected.
PROGRAM_B	Input	Dedicated	Active-Low asynchronous full-chip reset
CCLK	Output	Dedicated	Configuration clock output. CCLK does not directly connect to BPI Flash but is used internally to generate the address and sample read data.
FCS_B	Output	Dual	Active-Low Flash chip select output. This output is actively driven Low during configuration and 3-stated after configuration. It has a weak pull-up resistor during configuration. By default, this signal has a weak pull-down resistor after configuration.
FOE_B	Output	Dual	Active-Low Flash output enable. This output is actively driven Low during configuration and 3-stated after configuration. It has a weak pull-up resistor during configuration. By default, this signal has a weak pull-down resistor after configuration.
FWE_B	Output	Dual	Active-Low Flash write enable. This output is actively driven High during configuration and 3-stated after configuration. It has a weak pull-up resistor during configuration. By default, this signal has a weak pull-down resistor after configuration.
ADDR[25:0]	Output	Dual	Address output. For I/O bank locations, see Table 1-2, page 17.
D[15:0]	Input	Dual	Data input, sampled by the rising edge of the FPGA CCLK. For I/O bank location, see Table 1-2, page 17.

Table 2-9: **Virtex-5 Device BPI Configuration Interface Pins *(Continued)***

Pin Name	Type	Dedicated or Dual-Purpose	Description
RS[1:0]	Output	Dual	Revision Select pins. Not used for typical single bitstream applications. RS[1:0] are 3-stated and pulled up with weak resistors during the initial configuration if the HSWAP pin enables the pull ups. If pull ups are disabled, then a weak external pull up is required (after power-up or assertion of PROGRAM_B). RS[1:0] are actively driven Low to load the fallback bitstream when a configuration error is detected. RS[1:0] can also be controlled by the user through the bitstream or ICAP. See "Fallback MultiBoot," page 153.
CSO_B	Output	Dual	Parallel daisy chain active-Low chip select output. Not used in single FPGA applications.

Figure 2-23 shows the BPI-Up configuration waveforms.

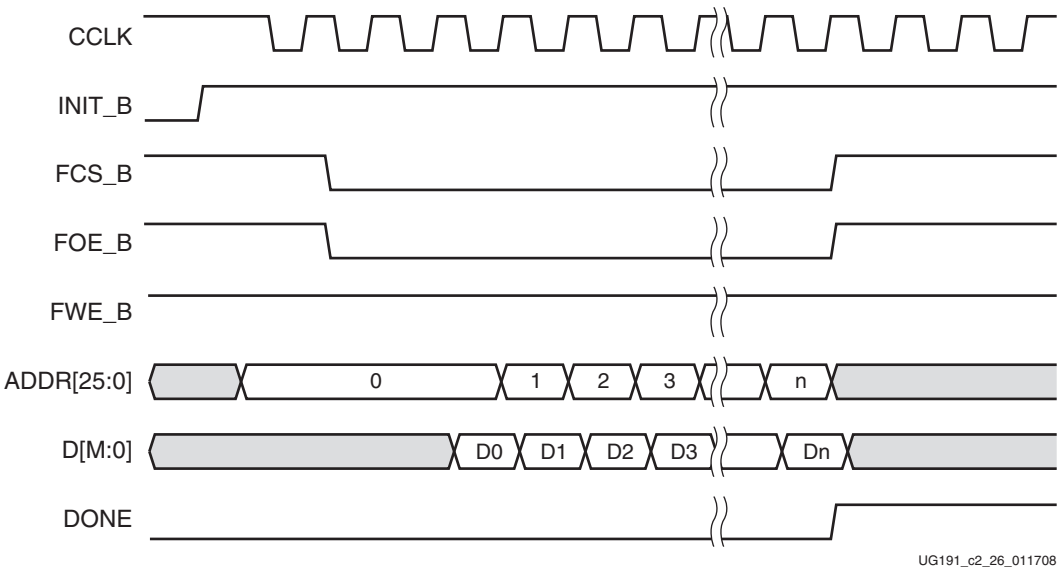


Figure 2-23: **Virtex-5 Device BPI-Up Configuration Waveforms**

Notes related to Figure 2-23:

- CCLK is output in BPI modes. The BPI Flash does not require CCLK, but the Virtex-5 FPGA uses the rising edge of CCLK to sample D[max:0] pins.
- The Virtex-5 FPGA stops loading the bitstream after the DONE pin goes High.
- Dual-mode configuration I/O switches to User mode after the GTS_cycle. By default, this is one cycle after DONE goes High.
- M can be 7 or 15.
- FCS_B, FOE_B, and FWE_B should have weak pull-ups after configuration through either I/O constraints or external pull-up resistors.
- The first address 0 for BPI-Up is extended for multiple cycles due to the initial latency.

Power-On Sequence Precautions

At power-on, the FPGA automatically starts its configuration procedure. When the FPGA is in a Master-BPI configuration mode, the FPGA asserts FCS_B Low and drives a sequence of addresses to read the bitstream from a BPI Flash. The BPI Flash must be ready for asynchronous reads before the FPGA drives FCS_B Low and outputs the first address to ensure the BPI Flash can output the stored bitstream.

Because different power rails can supply the FPGA and BPI Flash or because the FPGA and BPI flash can respond at different times along the ramp of a shared power supply, special attention to the FPGA and BPI Flash power-on sequence or power-on ramps is essential. The power-on sequence or power supply ramps can cause the FPGA to awake before the BPI Flash or vice versa. For many systems with near-simultaneous power supply ramps, the FPGA power-on reset time (TPOR) can sufficiently delay the start of the FPGA configuration procedure such that the BPI Flash becomes ready before the start of the FPGA configuration procedure. In general, the system design must consider the effect of the power sequence, the power ramps, FPGA power-on reset time, and BPI Flash power-on reset time on the timing relation between the start of FPGA configuration and the readiness of the BPI Flash for asynchronous reads. Check DS202, *Virtex-5 FPGA Data Sheet: DC and Switching Characteristics* data sheet for Virtex-5 FPGA power supply requirements and timing. Check DS617, *Platform Flash XL High-Density Configuration and Storage Device* data sheet for the BPI Flash power supply requirements and timing.

One of the following system design approaches can ensure that the BPI Flash is ready for asynchronous reads before the FPGA starts its configuration procedure:

- Control the sequence of the power supplies such that the BPI Flash is certain to be powered and ready for asynchronous reads before the FPGA begins its configuration procedure.
- Hold the FPGA PROGRAM_B pin Low from power-up to delay the start of the FPGA configuration procedure and release the PROGRAM_B pin to High after the BPI flash is fully powered and is able to perform asynchronous reads.
- Hold the FPGA INIT_B pin Low from power-up to delay the start of the FPGA configuration procedure and release the INIT_B pin to High after the BPI flash becomes ready for asynchronous reads.

See the *Power-On Precautions if 3.3V Supply is Last in Sequence* subsection of the Master BPI Mode section in UG332, *Spartan-3 Generation Configuration User Guide*, for reference.

Page Mode Support

Many NOR Flash devices support asynchronous page reads. The first access to a page usually takes the longest time (~100 ns), subsequent accesses to the same page take less time (~25 ns). The following parameters are bitstream programmable in Virtex-5 devices to take advantage of page reads and maximize the CCLK frequency:

- Page sizes of 1 (default), 4, or 8.

  If the actual Flash page size is larger then 8, the value of 8 should be used to maximize the efficiency.
- First access CCLK cycles of 1 (default), 2, 3, or 4. CCLK cycles must be 1 if the page size is 1.
- CCLK frequency

The sequence of page-mode operation is controlled by the Virtex-5 bitstream (see Table 6-15). After an FPGA reset, the default page size is 1, the first access CCLK is 1, and

the master CCLK is running at slowest default frequency. The COR0 register contains master CCLK frequency control bits (see "Configuration Options Register 0 (COR0)," page 120). The COR1 register contains BPI flash page mode control bits (see "Configuration Options Register 1 (COR1)," page 122). After the COR1 register is programmed, the BPI address timing switches at the page boundary as shown in Figure 2-24. When the SWITCH command is received, the master CCLK switches to a user-desired frequency, using it to load the rest of the configuration.

Refer to the "BitGen" section of the *Development System Reference Guide* for details on BitGen options.

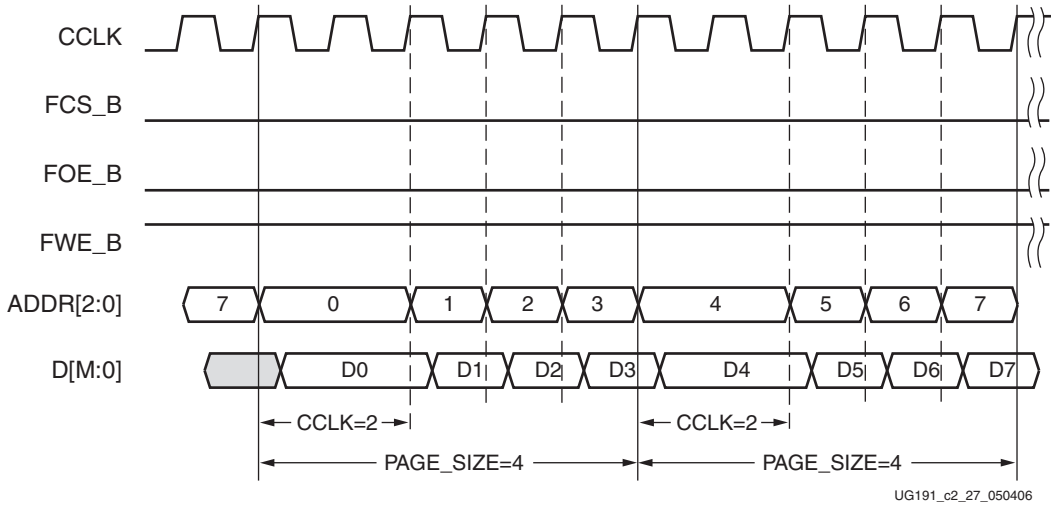


Figure 2-24: **BPI-Up Waveforms (Page Size = 4 and First Access CCLK = 2)**

Notes related to Figure 2-24:

- Figure 2-24 shows BPI-Up mode, a page size of 4, and a first access CCLK of 2.
- M can be 7 or 15.
- For BPI-Down mode, the ADDR[25:0] bus is extended for the desired CCLK cycles when ADDR[1:0]= 2'b11 for a page size of 4.

For supported flash devices, consult the iMPACT help menu indirect programming tables for Virtex-5 FPGA support. Other flash devices not listed in the help menu can be compatible with Virtex-5 devices.

Board Layout for Configuration Clock (CCLK)

The Virtex-5 FPGA configuration I/Os use the LVCMOS fast slew rate 12 mA standard. This I/O standard has faster edge rates to support higher configuration frequencies. This requires more attention to PCB trace routing and termination for proper signal integrity.

Note: The CCLK pin is the clock source for the Virtex-5 configuration logic during Master modes. the CCLK output must be free from reflections to avoid double-clocking.

These basic guidelines must be followed:

- Route the CCLK net as a 50Ω controlled impedance transmission line.
- Always route the CCLK net without any branching; do *not* use a *star* topology (Figure 2-28).
- Stubs, if necessary, must be shorter than 8 mm (0.3 inches).
- Terminate the end of the CCLK transmission line with a parallel termination of 100Ω to V_{CCO} and 100Ω to GND (the Thevenin equivalent of $V_{CCO}/2$, and assuming a trace characteristic impedance of 50Ω).

Xilinx recommends simulating the CCLK distribution with an IBIS simulator (such as HyperLynx) to check for glitches on each CLK input and output, including the CCLK of the master FPGA.

Figure 2-25 through Figure 2-27 show the recommended topologies for CCLK distribution.

Figure 2-25 shows the basic point-to-point topology for one CCLK driver (FPGA master) and one CCLK receiver (PROM or FPGA slave).

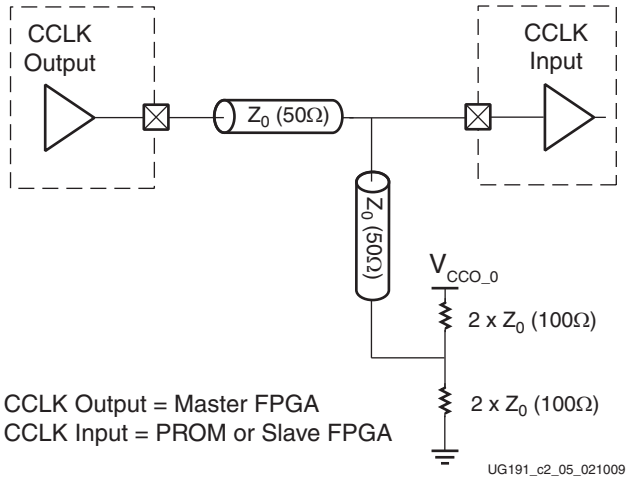


Figure 2-25: **Point-to-Point: One CCLK Output, One CCLK Input**

Figure 2-26 shows the basic multi-drop *flyby* topology for one CCLK driver and two CCLK receivers. The stub at CCLK input *1* has a length constraint.

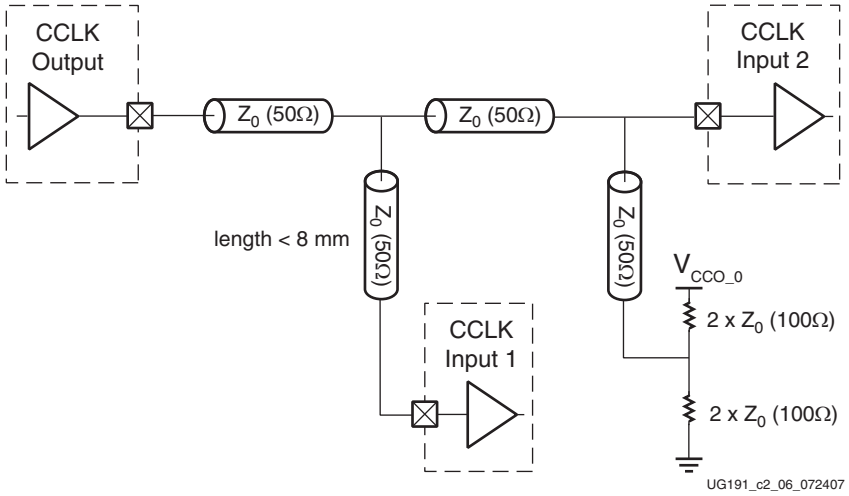


Figure 2-26: **Multi-Drop: One CCLK Output, Two CCLK Inputs**

Figure 2-27 shows the multi-drop *flyby* topology for one CCLK driver and more than two CCLK receivers (four in this example). All CCLK inputs except input *4* have length constraints.

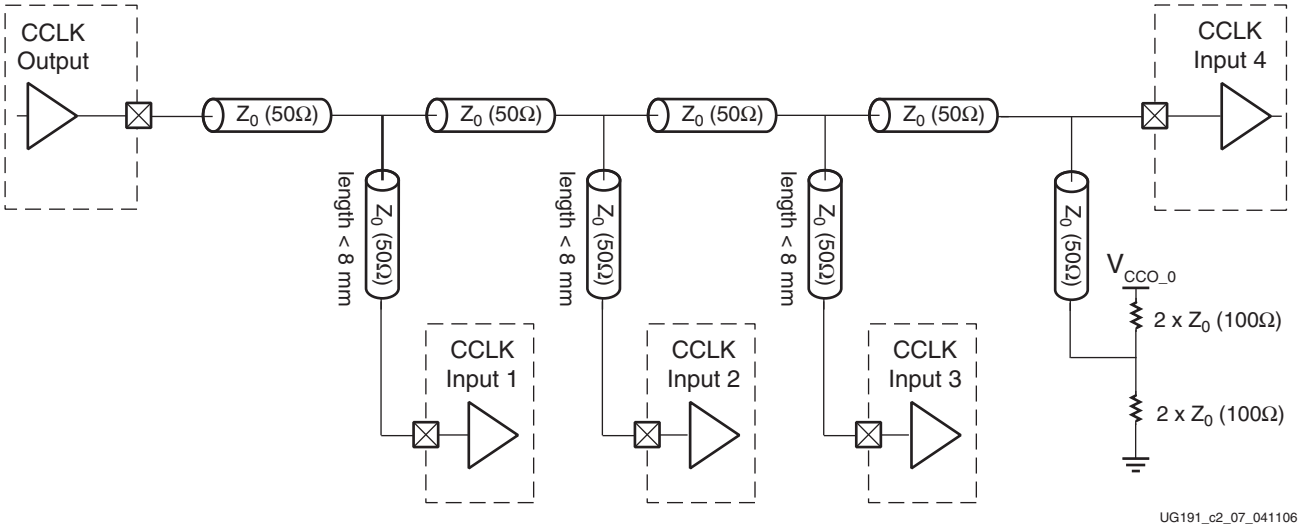


Figure 2-27: **Multi-Drop: One CCLK Output, More Than Two CCLK Inputs**

Figure 2-28 shows a *star* topology where the transmission line branches to the multiple CCLK inputs. The branch point creates a significant impedance discontinuity. This arrangement is **Not Recommended**.

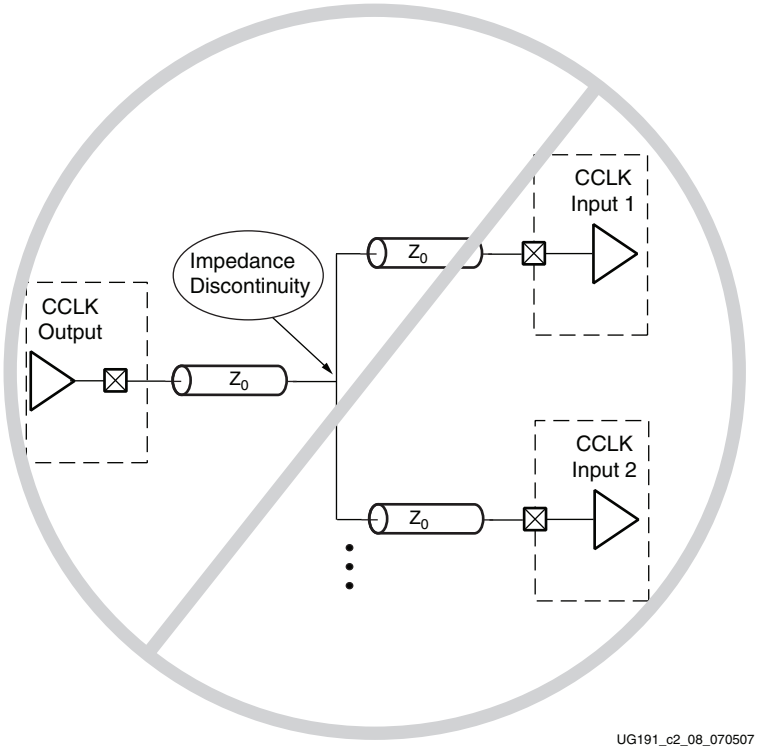


Figure 2-28: **Not Recommended**
Star Topology: One CCLK Output, Two CCLK Inputs

XILINX®

Chapter 3

Boundary-Scan and JTAG Configuration

Introduction

Virtex®-5 devices support IEEE standards 1149.1 and 1532. IEEE 1532 is a standard for In-System Configuration (ISC), based on the IEEE 1149.1 standard. JTAG is an acronym for the Joint Test Action Group, the technical subcommittee initially responsible for developing the standard. This standard provides a means to ensure the board-level integrity of individual components and the interconnections between them. The IEEE 1149.1 Test Access Port and Boundary-Scan Architecture is commonly referred to as JTAG. With multi-layer PC boards becoming increasingly dense and more sophisticated surface mounting techniques in use, Boundary-Scan testing is becoming widely used as an important debugging tool.

Devices containing Boundary-Scan logic can send data out on I/O pins in order to test connections between devices at the board level. The circuitry can also be used to send signals internally to test the device-specific behavior. These tests are commonly used to detect opens and shorts at both the board and device level.

In addition to testing, Boundary-Scan offers the flexibility for a device to have its own set of user-defined instructions. The added common vendor-specific instructions, such as configure and verify, have increased the popularity of Boundary-Scan testing and functionality.

JTAG Configuration/Readback

Full Initial Configuration or Reconfiguration

1. Load the JPROGRAM instruction into the JTAG Instruction Register (IR).
2. Loop on an Instruction Register load/capture with the CFG_IN instruction and wait for the captured value of INIT_COMPLETE (bit 4 of IR capture) to be 1.
3. Go to Shift-DR and load the new bitstream.
4. Go to the Test-Logic-Reset state (TLR).
5. Load the JSTART instruction into the JTAG IR.
6. Go to Run-Test-Idle (RTI).
7. Clock TCK for 12 cycles.
8. Load the CFG_IN instruction into the JTAG IR.

9. Go to Shift-DR and load the following bitstream fragment to read the Configuration STATUS register:

```
1111 1111 1111 1111 1111 1111 1111 1111 // Dummy word
1010 1010 1001 1001 0101 0101 0110 0110 // SYNCHWORD
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0010 1000 0000 0000 1110 0000 0000 0001 // Type 1 header: Read 1 word
                                        // from STAT
0000 0000 0000 0000 0000 0000 0000 0000 // flush pipeline
```

10. Load the CFG_OUT instruction into the JTAG IR.
11. Go to Shift-DR and shift out the STAT register data. Check that the *crc_error* (bit 0) is 0 and that the *release_done* (bit 13) is 1.
12. Go to TLR.

Partial Reconfiguration

1. Load the CFG_IN instruction into the JTAG IR.
2. Go to Shift-DR and load the following bitstream fragment to clear the CRC_ERROR signal:

```
1111 1111 1111 1111 1111 1111 1111 1111 // Dummy word
1010 1010 1001 1001 0101 0101 0110 0110 // SYNCHWORD
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0011 0000 0000 0000 1000 0000 0000 0001 // Write 1 word to CMD reg
0000 0000 0000 0000 0000 0000 0000 0111 // RCRC command
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0000 0000 0000 0000 0000 0000 0000 0000 // flush pipeline
```

3. Load the JSHUTDWN instruction into JTAG IR.
4. Go to Run-Test-Idle (RTI).
5. Clock TCK for 12 cycles to clock shutdown sequence (asserts GTS_CFG and deasserts GWE and DONE).
6. Load the CFG_IN instruction.
7. Go to Shift-DR and load the following bitstream fragment to assert the GHIGH_B signal:

```
1111 1111 1111 1111 1111 1111 1111 1111 // Dummy word
1010 1010 1001 1001 0101 0101 0110 0110 // SYNCHWORD
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0011 0000 0000 0000 1000 0000 0000 0001 // Write 1 word to CMD reg
0000 0000 0000 0000 0000 0000 0000 1000 // AGHIGH command
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
```

8. Load the reconfiguration bitstream.
9. Go to TLR.
10. Load the JSTART instruction into the JTAG IR.
11. Go to Run-Test-Idle (RTI).
12. Clock TCK for 12 cycles.
13. Go to Test-Logic-Reset (TLR).

Readback - Type 1: No Block RAM Frames

1. Load the CFG_IN instruction into the JTAG IR.

2. Go to Shift-DR and load the following bitstream fragment to write the RCFG command to the CMD register:

```
1111 1111 1111 1111 1111 1111 1111 1111 // Dummy word
1010 1010 1001 1001 0101 0101 0110 0110 // SYNCHWORD
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0011 0000 0000 0000 1000 0000 0000 0001 // Write 1 word to CMD reg
0000 0000 0000 0000 0000 0000 0000 0100 // RCFG command
0011 0000 0000 0000 0010 0000 0000 0001 // Write 1 word to FAR
0000 0000 0000 0000 0000 0000 0000 0000 // Frame address: Top row 0/CLB
                                        // Block Type/Column 0/Frame 0
0010 1000 0000 0000 0110 0000 0000 0000 // Type 1 header: Read FDRO
0100 1bbb bbbb bbbb bbbb bbbb bbbb bbbb // Type 2 header: Readback
                                        // wordcount (27 bits) - CLB
                                        // frames only
0000 0000 0000 0000 0000 0000 0000 0000 // Flush pipeline
```

3. Load the CFG_OUT instruction into the JTAG IR.
4. Go to Shift-DR and shift out the readback data.
5. Go to Test-Logic-Reset (TLR).

Readback - Type 2: Including Block RAM Frames

1. Load the CFG_IN instruction into the JTAG IR.
2. Go to Shift-DR and load the following bitstream fragment to clear the CRC_ERROR signal:

```
1111 1111 1111 1111 1111 1111 1111 1111 // Dummy word
1010 1010 1001 1001 0101 0101 0110 0110 // SYNCHWORD
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0011 0000 0000 0000 1000 0000 0000 0001 // Write 1 word to CMD reg
0000 0000 0000 0000 0000 0000 0000 0111 // RCRC command
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0000 0000 0000 0000 0000 0000 0000 0000 // flush pipeline
```

3. Load the JSHUTDWN instruction into the JTAG IR.
4. Go to Run-Test-Idle (RTI).
5. Clock TCK for 12 cycles to clock the shutdown sequence (asserts GTS_CFG and deasserts GWE and DONE).
6. Load the CFG_IN instruction into the JTAG IR.
7. Go to Shift-DR and load the following bitstream fragment to write the RCFG command to the CMD register:

```
1111 1111 1111 1111 1111 1111 1111 1111 // Dummy word
1010 1010 1001 1001 0101 0101 0110 0110 // SYNCHWORD
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0011 0000 0000 0000 1000 0000 0000 0001 // Write 1 word to CMD reg
0000 0000 0000 0000 0000 0000 0000 0100 // RCFG command
0011 0000 0000 0000 0010 0000 0000 0001 // Write 1 word to FAR
0000 0000 0000 0000 0000 0000 0000 0000 // Frame address: Top row 0/CLB
                                        // Block Type/Column 0/Frame 0
0010 1000 0000 0000 0110 0000 0000 0000 // Type 1 header: Read FDRO
0100 1bbb bbbb bbbb bbbb bbbb bbbb bbbb // Type 2 header: Readback
                                        // wordcount (27 bits) - CLB
                                        // and Block RAM frames
0000 0000 0000 0000 0000 0000 0000 0000 // Flush pipeline
```

8. Load the CFG_OUT instruction into the JTAG IR.

9. Go to Shift-DR and shift out the readback data.
10. Go to Test-Logic-Reset (TLR).
11. Load the CFG_IN instruction into the JTAG IR.
12. Go to Shift-DR and load the following bitstream fragment to clear the CRC_ERROR signal:

```
1111 1111 1111 1111 1111 1111 1111 1111 // Dummy word
1010 1010 1001 1001 0101 0101 0110 0110 // SYNCHWORD
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0011 0000 0000 0000 1000 0000 0000 0001 // Write 1 word to CMD reg
0000 0000 0000 0000 0000 0000 0000 0111 // RCRC command
0010 0000 0000 0000 0000 0000 0000 0000 // NO-OP
0000 0000 0000 0000 0000 0000 0000 0000 // flush pipeline
```

13. Load the JSTART instruction into the JTAG IR.
14. Go to Run-Test-Idle (RTI).
15. Clock TCK for 12 cycles to clock the startup sequence (deasserts GTS_CFG and asserts GWE and DONE).
16. Go to Test-Logic-Reset (TLR).

Boundary-Scan for Virtex-5 Devices Using IEEE Standard 1149.1

The Virtex-5 family is fully compliant with the IEEE Standard 1149.1 Test Access Port and Boundary-Scan Architecture. The architecture includes all mandatory elements defined in the IEEE 1149.1 Standard. These elements include the Test Access Port (TAP), the TAP controller, the Instruction register, the instruction decoder, the Boundary-Scan register, and the BYPASS register. The Virtex-5 family also supports a 32-bit Identification register and a Configuration register in full compliance with the standard. Outlined in the following sections are the details of the JTAG architecture for Virtex-5 devices.

If Boundary-Scan is used as part of the product verification in the LXT or SXT, the analog supply voltage pin MGTAVCC of all GTP_DUAL tiles must be powered. The analog supply voltage pin MGTAVCC of all unused GTP_DUAL tiles must be connected to the same supply that supplies V_{CCINT}, which is the power supply pin for the internal core logic.

Test Access Port (TAP)

The Virtex-5 TAP contains four mandatory dedicated pins as specified by the protocol given in Table 3-1 and illustrated in Figure 3-1, a typical JTAG architecture. Three input pins and one output pin control the 1149.1 Boundary-Scan TAP controller. Optional control pins, such as TRST (Test Reset) and enable pins might be found on devices from other manufacturers. It is important to be aware of these optional signals when interfacing Xilinx devices with parts from different vendors because they might need to be driven.

The TAP controller is a state machine (16 states) shown in Figure 3-2. The four mandatory TAP pins are outlined in Table 3-1.

Table 3-1: **Virtex-5 FPGA TAP Controller Pins**

Pin	Description
TDI	**Test Data In**. This pin is the serial input to all JTAG instruction and data registers. The state of the TAP controller and the current instruction determine the register that is fed by the TDI pin for a specific operation. TDI has an internal resistive pull-up to provide a logic High to the system if the pin is not driven. TDI is applied into the JTAG registers on the rising edge of TCK.
TDO	**Test Data Out**. This pin is the serial output for all JTAG instruction and data registers. The state of the TAP controller and the current instruction determine the register (instruction or data) that feeds TDO for a specific operation. TDO changes state on the falling edge of TCK and is only active during the shifting of instructions or data through the device. TDO is an active driver output.
TMS	**Test Mode Select**. This pin determines the sequence of states through the TAP controller on the rising edge of TCK. TMS has an internal resistive pull-up to provide a logic High if the pin is not driven.
TCK	**Test Clock**. This pin is the JTAG Test Clock. TCK sequences the TAP controller and the JTAG registers in the Virtex-5 devices.

Notes:

1. As specified by the IEEE Standard, the TMS and TDI pins both have internal pull-up resistors. These internal pull-up resistors of 50-150 kΩ are active, regardless of the mode selected.

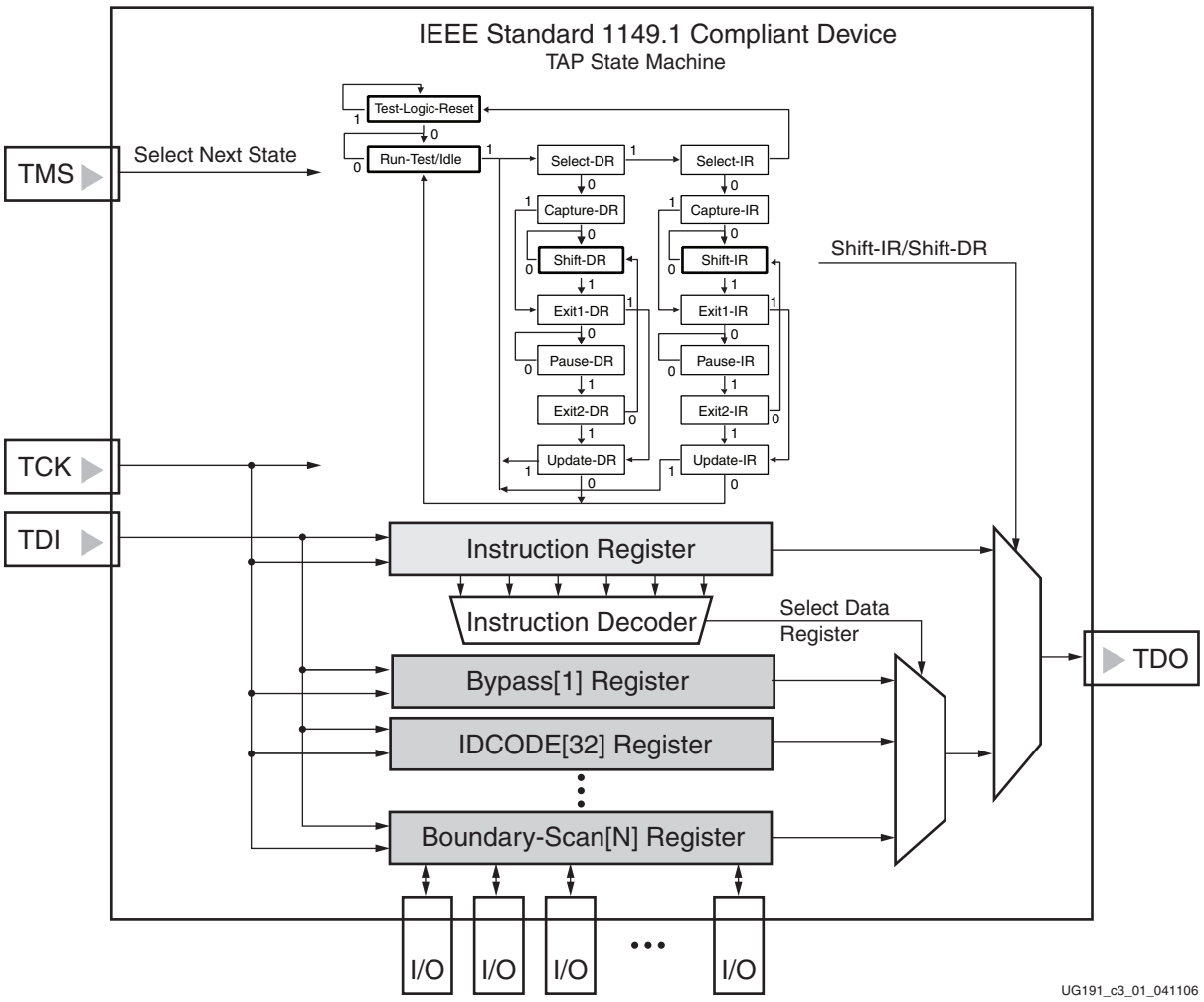


Figure 3-1: **Typical JTAG Architecture**

For JTAG configuration mode, JTAG inputs use the V_{CCO_CFG} supply.

TAP Controller

Figure 3-2 diagrams a 16-state finite state machine. The four TAP pins control how data is scanned into the various registers. The state of the TMS pin at the rising edge of TCK determines the sequence of state transitions. There are two main sequences, one for shifting data into the data register and the other for shifting an instruction into the instruction register.

A transition between the states only occurs on the rising edge of TCK, and each state has a different name. The two vertical columns with seven states each represent the Instruction Path and the Datapath. The data registers operate in the states whose names end with "DR," and the instruction register operates in the states whose names end in "IR." The states are otherwise identical.

The operation of each state is described below.

Test-Logic-Reset:
All test logic is disabled in this controller state, enabling the normal operation of the IC. The TAP controller state machine is designed so that regardless of the initial state of the controller, the Test-Logic-Reset state can be entered by holding TMS High and pulsing TCK five times. Consequently, the Test Reset (TRST) pin is optional.

Run-Test-Idle:
In this controller state, the test logic in the IC is active only if certain instructions are present. For example, if an instruction activates the self test, then it is executed when the controller enters this state. The test logic in the IC is idle otherwise.

Select-DR-Scan:
This controller state controls whether to enter the Datapath or the Select-IR-Scan state.

Select-IR-Scan:
This controller state controls whether or not to enter the Instruction Path. The controller can return to the Test-Logic-Reset state otherwise.

Capture-IR:
In this controller state, the shift register bank in the Instruction Register parallel loads a pattern of fixed values on the rising edge of TCK. The last two significant bits must always be `01`.

Shift-IR:
In this controller state, the instruction register gets connected between TDI and TDO, and the captured pattern gets shifted on each rising edge of TCK. The instruction available on the TDI pin is also shifted in to the instruction register.

Exit1-IR:
This controller state controls whether to enter the Pause-IR state or Update-IR state.

Pause-IR:
This state allows the shifting of the instruction register to be temporarily halted.

Exit2-DR:
This controller state controls whether to enter either the Shift-IR state or Update-IR state.

Update-IR:
In this controller state, the instruction in the instruction register is latched to the latch bank of the Instruction Register on every falling edge of TCK. This instruction becomes the current instruction after it is latched.

Capture-DR:
In this controller state, the data is parallel-loaded into the data registers selected by the current instruction on the rising edge of TCK.

Shift-Dr, Exit1-DR, Pause-DR, Exit2-DR, and Update-DR:
These controller states are similar to the Shift-IR, Exit1-IR, Pause-IR, Exit2-IR, and Update-IR states in the Instruction path.

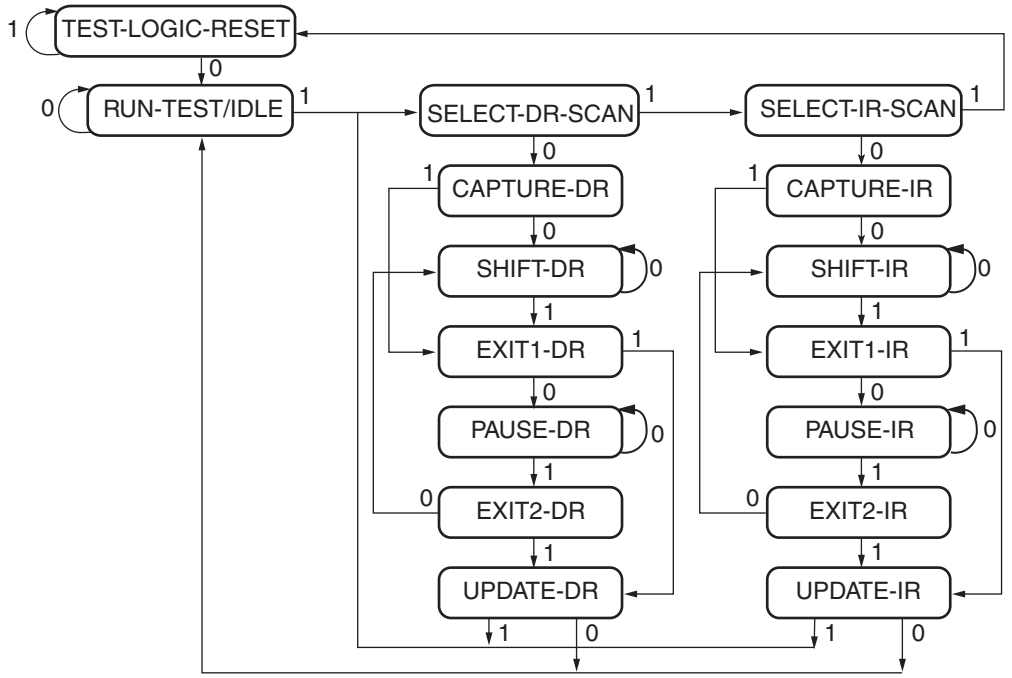


Figure 3-2: **Boundary-Scan TAP Controller**

Virtex-5 devices support the mandatory IEEE 1149.1 commands, as well as several Xilinx vendor-specific commands. The EXTEST, INTEST, SAMPLE/PRELOAD, BYPASS, IDCODE, USERCODE, and HIGHZ instructions are all included. The TAP also supports internal user-defined registers (USER1, USER2, USER3, and USER4) and configuration/readback of the device.

The Virtex-5 Boundary-Scan operations are independent of mode selection. The Boundary-Scan mode in Virtex-5 devices overrides other mode selections. For this reason, Boundary-Scan instructions using the Boundary-Scan register (SAMPLE/PRELOAD, INTEST, and EXTEST) must not be performed during configuration. All instructions except the user-defined instructions are available before a Virtex-5 device is configured. After configuration, all instructions are available.

JSTART and JSHUTDOWN are instructions specific to the Virtex-5 architecture and configuration flow. In Virtex-5 devices, the TAP controller is not reset by the PROGRAM_B pin and can only be reset by bringing the controller to the TLR state. The TAP controller is reset on power up.

For details on the standard Boundary-Scan instructions EXTEST, INTEST, and BYPASS, refer to the IEEE Standard.

Boundary-Scan Architecture

Virtex-5 device registers include all registers required by the IEEE 1149.1 Standard. In addition to the standard registers, the family contains optional registers for simplified testing and verification (Table 3-2).

Table 3-2: **Virtex-5 Device JTAG Registers**

Register Name	Register Length	Description
Boundary-Scan Register	3 bits per I/O	Controls and observes input, output, and output enable
Instruction Register	10 or 14 bits	Holds current instruction OPCODE and captures internal device status
BYPASS Register	1 bit	Bypasses the device
Identification Register	32 bits	Captures the Device ID
JTAG Configuration Register	32 bits	Allows access to the configuration bus when using the CFG_IN or CFG_OUT instructions
USERCODE Register	32 bits	Captures the user-programmable code
User-Defined Registers (USER1, USER2, USER3, and USER4)	Design-specific	Design-specific

Boundary-Scan Register

The test primary data register is the Boundary-Scan register. Boundary-Scan operation is independent of individual IOB configurations. Each IOB, bonded or unbonded, starts as bidirectional with 3-state control. Later, it can be configured to be an input, output, or 3-state only. Therefore, three data register bits are provided per IOB (Figure 3-3).

When conducting a data register (DR) operation, the DR captures data in a parallel fashion during the CAPTURE-DR state. The data is then shifted out and replaced by new data during the SHIFT-DR state. For each bit of the DR, an update latch is used to hold the input data stable during the next SHIFT-DR state. The data is then latched during the UPDATE-DR state when TCK is Low.

The update latch is opened each time the TAP controller enters the UPDATE-DR state. Care is necessary when exercising an INTEST or EXTEST to ensure that the proper data has been latched before exercising the command. This is typically accomplished by using the SAMPLE/PRELOAD instruction.

Internal pull-up and pull-down resistors should be considered when test vectors are being developed for testing opens and shorts. The HSWAPEN pin determines whether the IOB has a pull-up resistor. Figure 3-3 is a representation of Virtex-5 Boundary-Scan architecture.

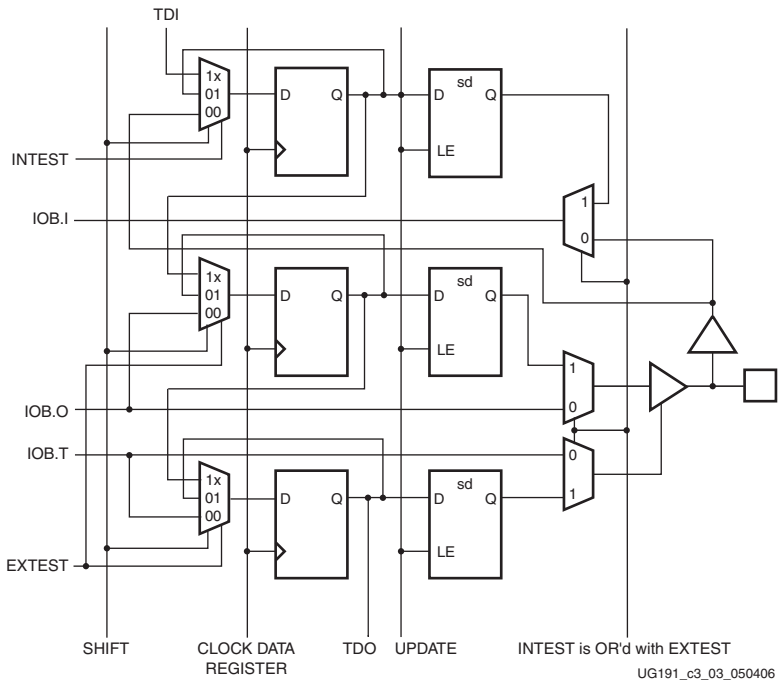


Figure 3-3: **Virtex-5 Family Boundary-Scan Logic**

Bit Sequence Boundary-Scan Register

The order of each non-TAP IOB is described in this section. The input is first, then the output, and finally the 3-state IOB control. The 3-state IOB control is closest to the TDO. The input-only pins contribute only the input bit to the Boundary-Scan I/O data register. The bit sequence of the device is obtainable from the *Boundary-Scan Description Language Files* (BSDL files) for the Virtex-5 family. (These files can be obtained from the Xilinx software download area.) The bit sequence always has the same bit order and the same number of bits and is independent of the design.

Instruction Register

The Instruction Register (IR) for the Virtex-5 device is connected between TDI and TDO during an instruction scan sequence. In preparation for an instruction scan sequence, the instruction register is parallel-loaded with a fixed instruction capture pattern. This pattern is shifted out onto TDO (LSB first), while an instruction is shifted into the instruction register from TDI.

To determine the operation to be invoked, an OPCODE necessary for the Virtex-5 Boundary-Scan instruction set is loaded into the Instruction Register. The length of the IR is device size-specific. The IR is 10 bits wide for the Virtex-5 LX, LXT, SXT, FXT, and TXT platform devices. The FX100T, FX130T, and FX200T have 14 bits of OPCODE because they contain 2 PowerPC processors. The 4 or 8 most significant bits support the PowerPC 440 embedded processor. The least significant 6 bits of the instruction code perform the same function for all Virtex-5 family members to support the new IEEE Standard 1532 for ISC devices. For PPC JTAG instruction, the least bits must be set to `100000` (20h). For PPC440 JTAG guidelines, refer to UG200, *Embedded Processor Block in Virtex-5 FPGAs Reference Guide*. Table 3-3 lists the available instructions for Virtex-5 devices.

Table 3-3: **Virtex-5 Device Boundary-Scan Instructions**

Boundary-Scan Command	Binary Code [9:0]	Description
EXTEST	1111000000	Enables Boundary-Scan EXTEST operation.
SAMPLE	1111000001	Enables Boundary-Scan SAMPLE operation.
USER1	1111000010	Access user-defined register 1.
USER2	1111000011	Access user-defined register 2.
USER3	1111100010	Access user-defined register 3.
USER4	1111100011	Access user-defined register 4.
CFG_OUT	1111000100	Access the configuration bus for readback.
CFG_IN	1111000101	Access the configuration bus for configuration.
INTEST	1111000111	Enables Boundary-Scan INTEST operation.
USERCODE	1111001000	Enables shifting out user code.
IDCODE	1111001001	Enables shifting out of ID code.
HIGHZ	1111001010	3-state output pins while enabling BYPASS Register.
JPROGRAM	1111001011	Equivalent to and has the same effect as PROGRAM.
JSTART	1111001100	Clocks the startup sequence when StartClk is TCK.
JSHUTDOWN	1111001101	Clocks the shutdown sequence.
SYSMON	1111110111	System Monitor DRP access through JTAG. See the DRP interface section in UG192: *Virtex-5 FPGA System Monitor User Guide.*
ISC_ENABLE	1111010000	Marks the beginning of ISC configuration. Full shutdown is executed.
ISC_PROGRAM	1111010001	Enables in-system programming.
ISC_PROGRAM_KEY	1111010010	Change security status from secure to non-secure mode and vice versa.
ISC_NOOP	1111010100	No operation.
ISC_READ	1111010101	Used to read back the device configuration data per IEEE std 1532 and the Battery-backed RAM (BBR). The BBR holds the encryption key for the AES bitstream encryption.
ISC_DISABLE	1111010111	Completes ISC configuration. Startup sequence is executed.
BYPASS	1111111111	Enables BYPASS.
RESERVED	All other codes	Xilinx reserved instructions.

Figure 3-4 shows the instruction capture values loaded into the IR as part of an instruction scan sequence.

TDI ∅	IR[9:6]	IR[5]	IR[4]	IR[3]	IR[2]	IR[1:0]	∅ TDO
	Reserved	DONE	INIT(1)	ISC_ENABLED	ISC_DONE	0 1	

Notes:

1. INIT is the status bit of the INIT_COMPLETE signal.

Figure 3-4: **Virtex-5 Device Instruction Capture Values Loaded into IR as Part of an Instruction Scan Sequence**

BYPASS Register

The other standard data register is the single flip-flop BYPASS register. It passes data serially from the TDI pin to the TDO pin during a bypass instruction. This register is initialized to zero when the TAP controller is in the CAPTURE-DR state.

Identification (IDCODE) Register

Virtex devices have a 32-bit identification register called the IDCODE register. The IDCODE is based on the IEEE 1149.1 standard, and is a fixed, vendor-assigned value that is used to identify electrically the manufacturer and the type of device that is being addressed. This register allows easy identification of the part being tested or programmed by Boundary-Scan, and it can be shifted out for examination by using the IDCODE instruction.

The last bit of the IDCODE is always `1` (based on JTAG IEEE 1149.1). The last three hex digits appear as `0x093`. IDCODEs assigned to Virtex-5 FPGAs are shown in Table 1-13, page 29.

JTAG Configuration Register

The JTAG Configuration register is a 32-bit register. This register allows access to the configuration bus and readback operations.

USERCODE Register

The USERCODE instruction is supported in the Virtex-5 family. This register allows a user to specify a design-specific identification code. The USERCODE can be programmed into the device and can be read back for verification later. The USERCODE is embedded into the bitstream during bitstream generation (BitGen **`-g`** UserID option) and is valid only after configuration. If the device is blank or the USERCODE was not programmed, the USERCODE register contains `0xFFFFFFFF`.

USER1, USER2, USER3, and USER4 Registers

The USER1, USER2, USER3, and USER4 registers are only available after configuration. These four registers must be defined by the user within the design. These registers can be accessed after they are defined by the TAP pins.

The BSCAN_VIRTEX5 library macro is required when creating these registers. This symbol is only required for driving internal scan chains (USER1, USER2, USER3, and USER4).

A common input pin (TDI) and shared output pins represent the state of the TAP controller (RESET, SHIFT, and UPDATE). Virtex-5 TAP pins are dedicated and do not require the BSCAN_VIRTEX5 macro for normal Boundary-Scan instructions or operations. For HDL, the BSCAN_VIRTEX5 macro must be instantiated in the design.

Using Boundary-Scan in Virtex-5 Devices

Characterization data for some of the most commonly requested timing parameters shown in Figure 3-5 are listed in DS202, *Virtex-5 Data Sheet: DC and Switching Characteristics*, in the Configuration Switching Characteristics table.

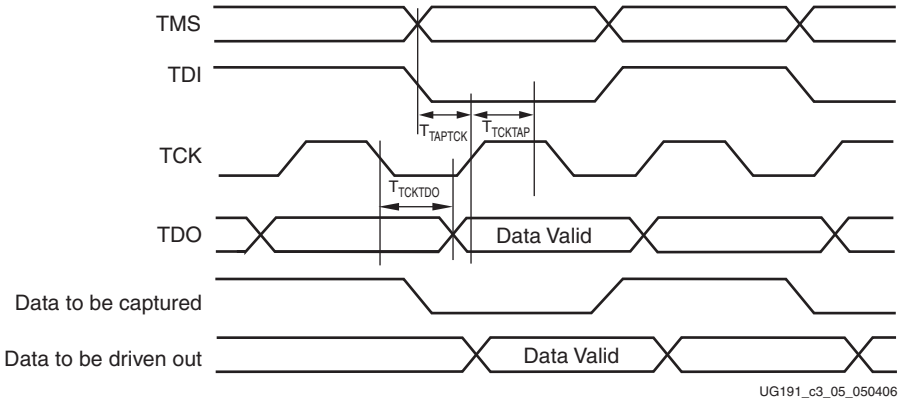


Figure 3-5: **Virtex-5 Device Boundary-Scan Port Timing Waveforms**

For further information on the startup sequence, bitstream, and internal configuration registers referenced here, refer to "Configuration Sequence" in Chapter 1.

Configuring through Boundary-Scan

One of the most common Boundary-Scan vendor-specific instructions is the configure instruction. If the Virtex-5 device is configured via JTAG on power-up, it is advisable to tie the mode pins to the Boundary-Scan configuration mode settings: `101` (M2 = `1`, M1 = `0`, M0 = `1`).

The configuration flow for Virtex-5 device configuration with JTAG is shown in Figure 3-6. The sections that follow describe how the Virtex-5 device can be configured as a single device through the Boundary-Scan or as part of a multiple-device scan chain.

A configured device can be reconfigured by toggling the TAP and entering a CFG_IN instruction after pulsing the PROGRAM pin or issuing the shut-down sequence. (Refer to Figure 3-6.)

Designers who wish to implement the Virtex-5 JTAG configuration algorithm are encouraged to use the SVF-based flow provided in Xilinx application note XAPP058.

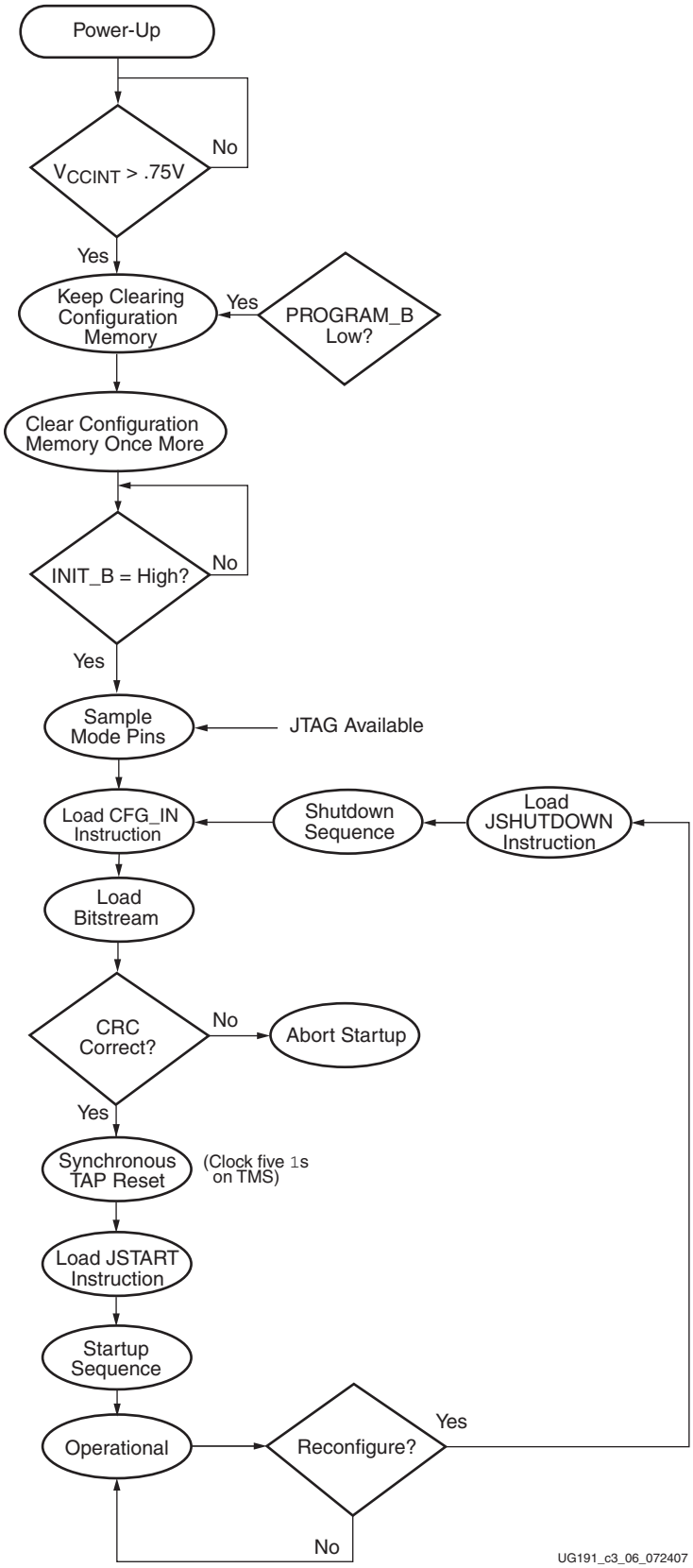


Figure 3-6: **Device Configuration Flow Diagram**

Single Device Configuration

Table 3-4 describes the TAP controller commands required to configure a Virtex-5 device. Refer to Figure 3-2 for TAP controller states. These TAP controller commands are issued automatically if configuring the part with the iMPACT software.

Table 3-4: **Single Device Configuration Sequence**

TAP Controller Step and Description		Set and Hold		# of Clocks
		TDI	TMS	TCK
1.	On power-up, place a logic 1 on the TMS, and clock the TCK five times. This ensures starting in the TLR (Test-Logic-Reset) state.	X	1	5
2.	Move into the RTI state.	X	0	1
3.	Move into the SELECT-IR state.	X	1	2
4.	Enter the SHIFT-IR state.	X	0	2
5.	Start loading the CFG_IN instruction, LSB first:	111000101	0	9
6.	Load the MSB of CFG_IN instruction when exiting SHIFT-IR, as defined in the IEEE standard.	1	1	1
7.	Enter the SELECT-DR state.	X	1	2
8.	Enter the SHIFT-DR state.	X	0	2
9.	Shift in the Virtex-5 bitstream. Bit_n (MSB) is the first bit in the bitstream(1).	$bit_1 ... bit_n$	0	(bits in bitstream)-1
10.	Shift in the last bit of the bitstream. Bit_0 (LSB) shifts on the transition to EXIT1-DR.	bit_0	1	1
11.	Enter UPDATE-DR state.	X	1	1
12.	Reset TAP by clocking five 1s on TMS	X	1	5
13.	Move into RTI state.	X	1	1
14.	Enter the SELECT-IR state.	X	1	2
15.	Move to the SHIFT-IR state.	X	0	2
16.	Start loading the JSTART instruction. The JSTART instruction initializes the startup sequence.	111001100	0	9
17.	Load the last bit of the JSTART instruction.	1	1	1
18.	Move to the UPDATE-IR state.	X	1	1
19.	Move to the RTI state and clock the startup sequence by applying a minimum of 12 clock cycles to the TCK.	X	0	12
20.	Move to the TLR state. The device is now functional.	X	1	3

Notes:

1. In the Configuration Register, data is shifted in from the right (TDI) to the left (TDO), MSB first. (Shifts into the Configuration Register are different from shifts into the other registers in that they are MSB first.)

Multiple Device Configuration

It is possible to configure multiple Virtex-5 devices in a chain. (See Figure 3-7.) The devices in the JTAG chain are configured one at a time. The multiple device configuration steps can be applied to any size chain.

Refer to the state diagram in Figure 3-2 for the following TAP controller steps:

1. On power-up, place a logic 1 on the TMS and clock the TCK five times. This ensures starting in the TLR (Test-Logic-Reset) state.
2. Load the CFG_IN instruction into the target device (and BYPASS in all other devices). Go through the RTI state (RUN-TEST/IDLE).
3. Load in the configuration bitstream per step 7 through step 11 in Table 3-4.
4. Repeat step 2 and step 3 for each device.
5. Reset all TAPs by clocking five 1s on TMS.
6. Load the JSTART command into all devices.
7. Go to the RTI state and clock TCK 12 times.

All devices are active at this point.

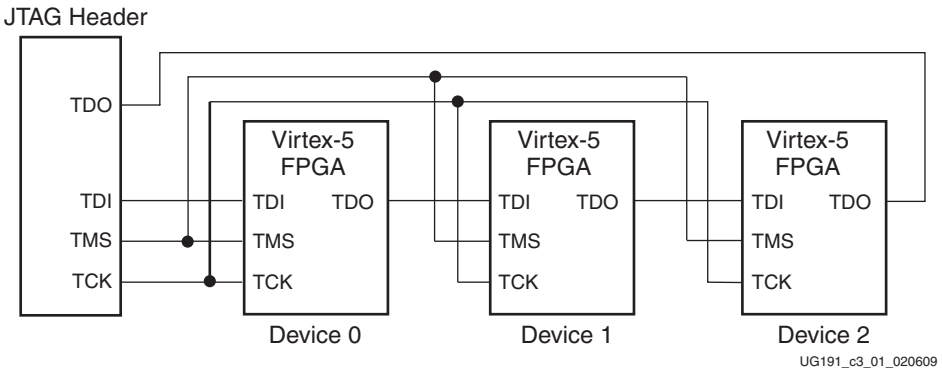


Figure 3-7: **Boundary-Scan Chain of Devices**

If JTAG is the only configuration mode, then PROGRAM_B, INIT_B, and DONE can each be tied High to separate resistors as shown in the Master serial or Master/Slave Serial Mode Daisy Chain Configuration (see Figure 2-3 and Figure 2-4).

Reconfiguring through Boundary-Scan

The ability of Virtex-5 devices to perform partial reconfiguration is the reason that the configuration memory is not cleared when reconfiguring the device. When reconfiguring a chain of devices, refer to step 3 in Table 3-4. There are two methods to reconfigure Virtex-5 devices without possible internal contention. The first method is to pulse the PROGRAM_B pin, resetting the internal configuration memory. The alternate method is to perform a shutdown sequence, placing the device in a safe state. The following shutdown sequence includes using internal registers. (For details on internal registers, refer to Chapter 7, "Readback and Configuration Verification.")

1. Load the CFG_IN instruction.
2. In the SHIFT-DR state, load the synchronization word followed by the Reset CRC Register (RCRC) command.

```
1111 1111 1111 1111 1111 1111 1111 1111Ø Dummy word
1010 1010 1001 1001 0101 0101 0110 0110Ø Synchronization word
0011 0000 0000 0000 1000 0000 0000 0001Ø Header: Write to CMD register
0000 0000 0000 0000 0000 0000 0000 0111Ø RCRC command
0010 0000 0000 0000 0000 0000 0000 0000Ø NO-OP
0000 0000 0000 0000 0000 0000 0000 0000Ø flush pipe
```

3. Load JSHUTDOWN.
4. Go to the RTI state and clock TCK at least 12 times to clock the shutdown sequence.
5. Proceed to the SHIFT-IR state and load the CFG_IN instruction again.
6. Go to the SHIFT-DR state and load the configuration bits. Make sure the configuration bits contain the AGHIGH command, asserting the global signal GHIGH_B. This prevents contention while writing configuration data.

```
0011 0000 0000 0000 1000 0000 0000 0001Ø Header: Write to CMD
0000 0000 0000 0000 0000 0000 0000 1000Ø AGHIGH command asserts
GHIGH_B
0000 0000 0000 0000 0000 0000 0000 0000Ø flush pipe
```

7. When all configuration bits have been loaded, reset the TAP by clocking five 1s on TMS.
8. Go to the SHIFT-IR state and load the JSTART instruction.
9. Go to the RTI state and clock TCK at least 12 times to clock the startup sequence.
10. Go to the TLR state to complete the reconfiguration process.

Boundary-Scan for Virtex-5 Devices Using IEEE Standard 1532

ISC Modal States

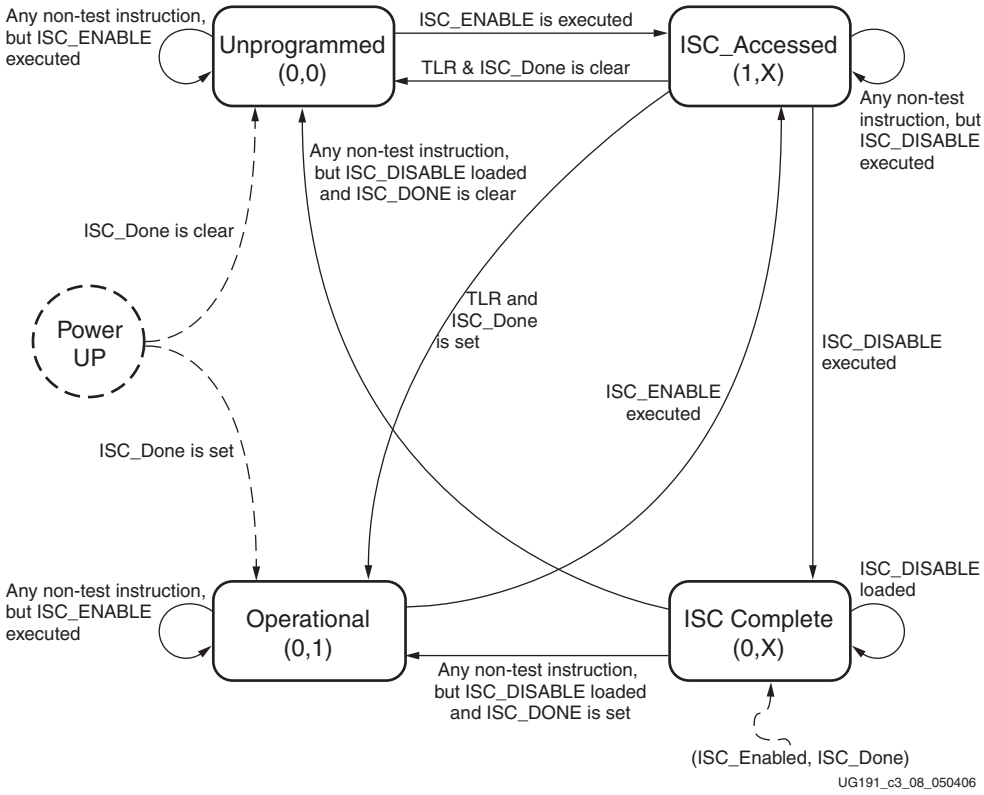


Figure 3-8: **ISC Modal States**

Once the device is powered up, it goes to the Unprogrammed state. The I/Os are all either 3-stated or pulled up. When ISC_ENABLE is successfully executed, the ISC_Enabled signal is asserted, and the device moves to the ISC_Accessed state. When the device moves to the ISC_Accessed state from the Operational state, the shutdown sequence is executed. The I/Os are all either 3-stated or pulled up.

The startup sequence is executed when in the ISC_Accessed state. At the end of the startup sequence, ISC_Enabled is cleared and the device moves to ISC_Complete. The minimum clock cycle requirement is the number of clock cycles required to complete the startup sequence. At the completion of the minimum required clock cycles, ISC_Enabled is deasserted.

Whether the startup sequence is successful or not is determined by CRC or configuration error status from the configuration processor. If the startup is completed, ISC_Done is asserted; otherwise, ISC_Done stays Low. The I/Os are either 3-stated or pulled up.

When ISC_Done is set in ISC_Complete state, the device moves to the Operational state. Otherwise, if ISC_Done is clear, the device moves to the Unprogrammed state. However, if the TAP controller goes to the TLR state while the device is in ISC_Accessed state, and if ISC_Done is set, then the device moves to the Operational state.

Though Operational, the I/O is not active yet because the startup sequence has not been performed. The startup sequence has to be performed in the Operational state to bring the I/O active.

Clocking Startup and Shutdown Sequences (JTAG)

There are three clock sources for startup and shutdown sequence: CCLK, UserCLK, and JTAGCLK. Clock selection is set by BitGen. The startup sequence is executed in the ISC_Accessed state. When it is clocked by JTAGCLK, the startup sequence receives the JTAGCLK in TAP Run/Test Idle state while ISC_DISABLE is the current JTAG instruction. The number of clock cycles in Run/Test Idle state for successful completion of ISC_DISABLE is determined by the number of clock cycles needed to complete the startup sequence.

When UserCLK or CCLK is used to clock the startup sequence, the user should know how many JTAGCLK cycles should be spent in Run/Test Idle to complete the startup sequence successfully.

The shutdown sequence is executed when the device transitions from the Operational to the ISC_Accessed state. Shutdown is done while executing the ISC_ENABLE instruction. When the shutdown sequence is clocked using JTAGCLK, the clock is supplied in the Run/Test Idle state of the ISC_ENABLE instruction. The number of clock cycles in Run/Test Idle is determined by the number of clock cycles needed to complete the shutdown sequence.

When the shutdown sequence is clocked by CCLK or UserCLK, the user is responsible for knowing how many JTAGCLK cycles in Run/Test Idle are needed to complete the shutdown sequence. The shutdown sequence is the startup sequence in reverse order.

Note:

When configuring the device through JTAG, the startup and shutdown clock should come from TCK, regardless of the selection in BitGen. In IEEE 1532 configuration mode, the startup and shutdown clock source is always TCK.

Configuration Flows Using JTAG

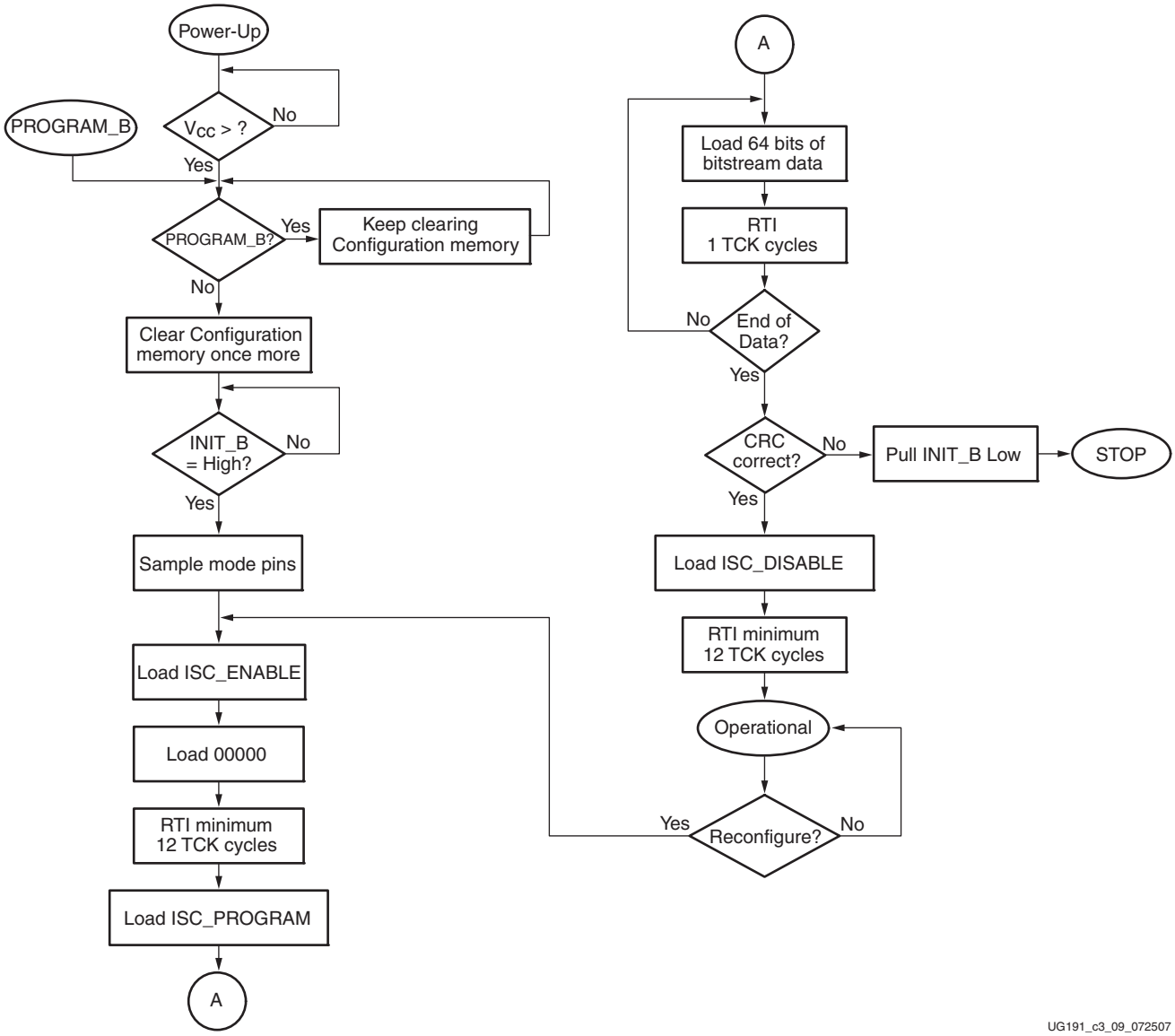


Figure 3-9: **IEEE 1532 Configuration Flow**

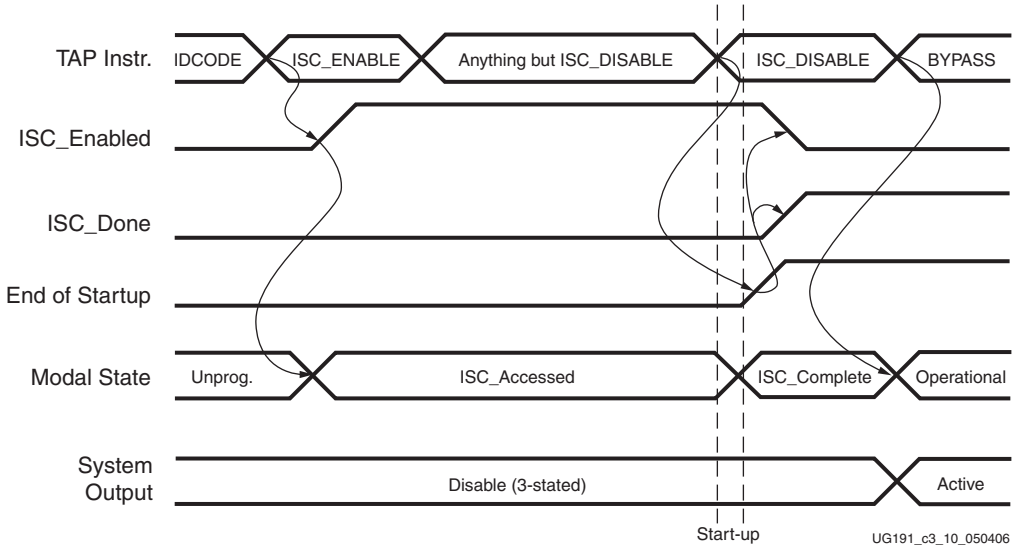


Figure 3-10: **Signal Diagram for Successful First Time ISC Configuration**

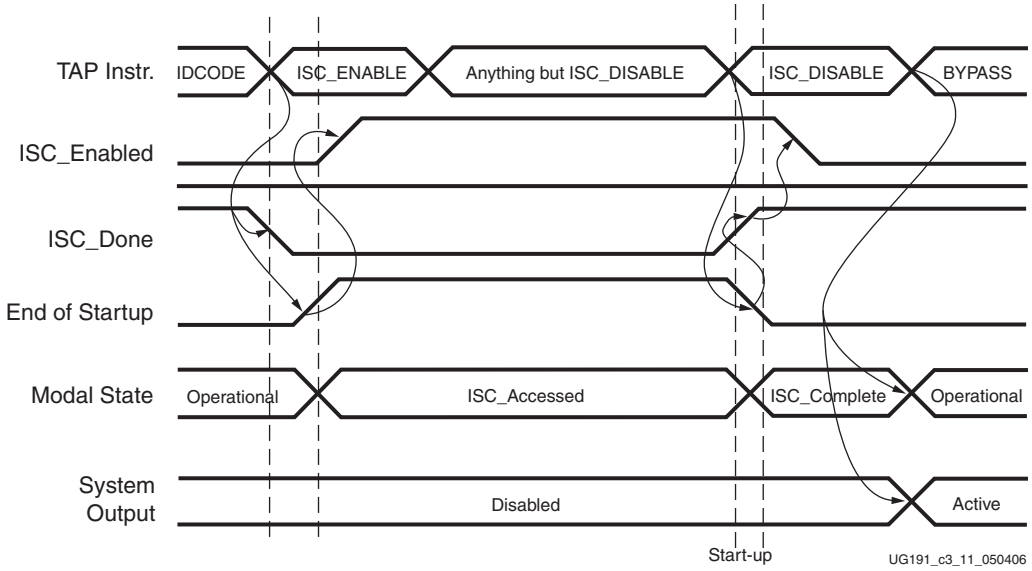


Figure 3-11: **Signal Diagram for Successful ISC Partial and Full Reconfiguration**

Chapter 4

User Primitives

The following configuration primitives are provided for users to access FPGA configuration resources during or after FPGA configuration.

BSCAN_VIRTEX5

JTAG is a standard four-pin interface: TCK, TMS, TDI, and TDO. Many applications are built around this interface. The JTAG TAP controller is a dedicated state machine inside the configuration logic. BSCAN_VIRTEX5 provides access between the JTAG TAP controller and user logic in fabric. There are up to four instances of BSCAN_VIRTEX5 for each device, each instance is controlled with the JTAG_CHAIN parameter. Table 4-1 lists the BSCAN_VIRTEX5 fabric pins.

Table 4-1: **BSCAN_VIRTEX5 Pin Table**

Pin Name	Type	Description
SEL	Output	Active-High interface selection output. SEL = 1 when the JTAG instruction register holds the corresponding USER1-4 instruction. Change in Update_IR state. SEL changes on the falling edge of TCK in the UPDATE_IR state of the TAP controller.
RESET	Output	Active-High reset output. RESET = 1 during the TEST-LOGIC-RESET state, PROGRAM_B, or during power up. This signal is deasserted on the falling edge of TCK.
TDI	Output	Fed through directly from the FPGA TDI pin.
DRCK	Output	DRCK is the same as TCK in the Capture_DR and Shift_DR states. If the interface is not selected by the instruction register, DRCK remains High.
CAPTURE	Output	Active-High pulse indicating the Capture_DR state. This signal is asserted on the falling edge of TCK.
UPDATE	Output	Active-High pulse indicating the Update_DR state. This signal is asserted on the falling edge of TCK.
SHIFT	Output	Active-High pulse indicating the Shift_DR state. This signal is asserted on the falling edge of TCK.
TDO	Input	TDO input driven from the user fabric logic. This signal is internally sampled on the falling edge before being driven out to the FPGA TDO pin.

CAPTURE_VIRTEX5

The CAPTURE_VIRTEX5 primitive is used to capture I/O, CLB, and block RAM output flip-flop status, and then read back through the configuration interface. The CAP input is sampled by CLK to generate an internal gcap signal. The I/O and CLB flip-flop status are captured into an FPGA configuration memory cell when the gcap signal is High. There are operation modes, a one-shot mode, or a continuous mode.

In one-shot mode, after the first CAP falling edge, gcap is held to 0 to avoid further capturing. An explicit RCAP command is required to re-arm the capture circuit.

In continuous mode, the CAP input is simply sampled by CLK, and becomes the gcap signal, allowing the user to control when to capture.

CAPTURE_VIRTEX5 should not operate simultaneously with the FRAME_ECC_VIRTEX5 primitive or the Readback CRC function (see Chapter 9, "Readback CRC") because capturing a value into configuration memory might cause a false error.

Table 4-2: **CAPTURE_VIRTEX5 Pin Table**

Pin Name	Type	Description
CLK	Input	Clock for sampling the CAP input.
CAP	Input	Active-High capture enable. The CAP input is sampled by the rising edge of CLK.

ICAP_VIRTEX5

The ICAP_VIRTEX5 primitive works the same way as the SelectMAP configuration interface except it is on the fabric side, and ICAP has a separate read/write bus, as opposed to the bidirectional bus in SelectMAP. The general SelectMAP timing diagrams and the SelectMAP bitstream ordering information as described in the "SelectMAP Configuration Interface" section of this user guide are also applicable to ICAP. It allows the user to access configuration registers, readback configuration data, or partially reconfigure the FPGA after configuration is done.

ICAP has three data width selections through the ICAP WIDTH parameter: x8, x16, and x32.

The two ICAP ports cannot be operated simultaneously. The design must start from the top ICAP, then switch back and forth between the two.

Table 4-3: **ICAP_VIRTEX5 Pin Table**

Pin Name	Type	Description
CLK	Input	ICAP interface clock
CE	Input	Active-Low ICAP interface select. Equivalent to CS_B in the SelectMAP interface.
WRITE	Input	0=WRITE, 1=READ. Equivalent to the RDWR_B signal in the SelectMAP interface.
I[31:0]	Input	ICAP write data bus. The bus width depends on ICAP_WIDTH parameter. The bit ordering is identical to the SelectMAP interface. See SelectMap Data Ordering in Figure 2-19.

Table 4-3: **ICAP_VIRTEX5 Pin Table *(Continued)***

Pin Name	Type	Description
O[31:0]	Output	Unregistered ICAP read data bus. The bus width depends on the ICAP_WIDTH parameter. The bit ordering is identical to the SelectMAP interface. See SelectMap Data Ordering in Figure 2-19.
BUSY	Output	Active-High busy status. Only used in read operations. BUSY remains Low during writes.

FRAME_ECC_VIRTEX5

The Virtex-5 Frame error correction code (ECC) logic is designed to detect single- or double-bit errors in configuration frame data. It uses SECDED (Hamming code) parity values based on the frame data generated by BitGen. During readback, the Frame ECC logic calculates a *syndrome* value using all the bits in the frame, including the ECC bits. If the bits have not changed from the original programmed values, then the syndrome bits are all 0s. If a single bit has changed, including any of the ECC bits, then the location of the bit is indicated by syndrome bits 10:0 and the syndrome bit 11 is 1. If two bits have changed, then syndrome bit 11 is 0 and the remaining bits are non-zero and meaningless. If more than two bits have changed, then the syndrome bits is indeterminate. The *error* output of the block is asserted if one or two bits have changed, indicating that action needs to be taken.

To use the Frame ECC logic, FRAME_ECC_VIRTEX5 must be instantiated in the user's design, and readback must be performed through SelectMAP, JTAG, or ICAP. At the end of each frame of readback, the *syndrome_valid* signal is asserted for one cycle of the readback clock (CCLK, TCK, or ICAP_CLK). The number of cycles required to read back a frame varies with the interface used. Refer to "Readback and Configuration Verification" in Chapter 7 for further information.

The FRAME_ECC_VIRTEX5 logic does not repair changed bits; this requires a user design. The design must be able to store at least one frame of data, or be able to fetch original frames of data for reload. A single frame is 1,312 bits. Following is an example of a simple repair implementation:

1. A frame is read out through ICAP and stored in block RAM. The frame address must be generated as each frame is read.
2. If an error is indicated by the *error* output of the FRAME_ECC block, then the readback is halted and the syndrome value is saved. If bit 11 is 0, then the whole frame must be restored. If bit 11 is 1, then bits 10:0 are used to locate the error bit in the saved frame, and the bit is inverted.
3. The repaired frame is then written back into the frame address generated in step 1.
4. Readback then begins again with the next frame address.

The syndrome bits S[10:0] are derived from the Hamming parity bits, while S[11] is derived from the overall parity bit. The syndrome bit is interpreted as follows:

`S[11] = 0, S[10:0] = 0:` no error.

`S[11] = 1, S[10:0] ≠ 0:` single bit (SED) error; S[10:0] denotes location of bit to patch (indirectly).

`S[11] = 1, S[10:0] = 0:` single-bit error; overall parity bit p[11] is in error.

`S[11] = 0, S[10:0] ≠ 0:` double-bit error, not correctable.

In case of a single-bit error in the frame data, the syndrome bits S[10:0] points to the flipped bit in the address space from 704 (location of the first bit in the frame) to 2047 (last bit in the frame). To convert the syndrome value S[10:0] to the index of the flipped bit in the range 0 to 1311, subtract 704 decimal (`2C0` hexadecimal or `01011000000` binary) if the syndrome is less than 1,024 decimal; otherwise, subtract 736 decimal (`2E0` hexadecimal or `01011100000` binary). This is equivalent to subtracting 22 or 23 decimal from S[10:5], and can be calculated as `bit_index = {S[10:5] -6'd22-S[10],S[5:0]}`.

If S[10:0] is `0` or a power of 2, however, an error in a parity bit has occurred. The Hamming parity bits are stored in locations 640–651. If bit S[11] indicates a single-bit error, then (in the case of a Hamming code parity bit error) a `1` is presented in the appropriate power-of-2 bit position, with the other syndrome bits set to `0`:

```
100000000001 ->  640
100000000010 ->  641
100000000100 ->  642
100000001000 ->  643
100000010000 ->  644
100000100000 ->  645
100001000000 ->  646
100010000000 ->  647
100100000000 ->  648
101000000000 ->  649
110000000000 ->  650
100000000000 ->  651
```

Table 4-4 defines the FRAME_ECC_VIRTEX5 pins. See Chapter 9, "Readback CRC" for more information.

Table 4-4: **FRAME_ECC_VIRTEX5 Pin Table**

Pin Name	Type	Description
SYNDROMEVALID	Output	Frame ECC syndrome valid pulse. Active one cycle for each frame. Used to sample ERROR and SYNDROME[11:0].
ECCERROR	Output	When SYNDROMEVALID is active, this output signals if a frame error was detected or not. • ERROR = 1 when SYNDROME[11:0] is non-zero. • ERROR = 0 when SYNDROME[11:0] is all zeros.

Table 4-4: **FRAME_ECC_VIRTEX5 Pin Table *(Continued)***

Pin Name	Type	Description
SYNDROME[11:0]	Output	When SYNDROMEVALID is active, this output reflects the frame error condition: • No bit error: [11]==0, [10:0] ==0 • One bit error: [11]==1, [10:0] !=0 • Two bit errors: [11]==0, [10:0] != 0 • More than two bits: SYNDROME is unpredictable • Parity Bit Error: [11]==1, [10:0]==0
CRCERROR	Output	RBCRC error. See Chapter 9, "Readback CRC."

USR_ACCESS_VIRTEX5

The User Access Register (USR_ACCESS_VIRTEX5) is a 32-bit register that allows data from the bitstream to be directly accessible by the FPGA fabric. The register has three outputs: CFG_CLK, the 32-bit D bus, and the DATAVALID signal that is asserted for one cycle of the configuration data source clock whenever a new value is available. The configuration data source clock can be CCLK or TCK.

The use model for this block allows data from a bitstream data storage source (e.g., PROM) to be accessed by the fabric after the FPGA has been configured. To accomplish this, the STARTUP_VIRTEX5 block should also be instantiated. The STARTUP_VIRTEX5 block has inputs that allow the user to take over the CCLK and DONE pins after the EOS (End-Of-Startup) signal is asserted. These pins are: USRCCLKO, USRCCLKTS, USRDONEO, and USRDONETS. The BitGen option **`-g DONE_cycle:Keep`** is used to prevent the DONE pin from going High and resetting the PROM. The USR_CCLK_O pin should be connected to a controlled clock in the fabric. In the bitstream, a write to the USR_ACCESS register that follows the normal configuration bitstream provides the data. After EOS is asserted, the data packet is loaded by clocking the USR_CCLK_O pin while keeping USR_CCLK_TS Low (it can be tied Low in this usage).

Alternatively, the User Access register can provide a single 32-bit constant value to the fabric (using the "AXSS Register," page 115) as an alternative to using a block RAM or a LUT RAM to hold a constant.

Table 4-5: **USR_ACCESS_VIRTEX5 Pin Table**

Pin Name	Type	Description
CFG_CLK	Output	Configuration clock.
DATA[31:0]	Output	User access data from bitstream
DATAVALID	Output	DATA[31:0] is ready for read signal

STARTUP_VIRTEX5

The STARTUP_VIRTEX5 primitive provides a fabric interface to allow users to control some of global signals after End of Startup (EOS).

Table 4-6: **STARTUP_VIRTEX5 Pin Table**

Pin Name	Type	Description
EOS	Output	High True. Absolute end of startup.
CLK	Input	User startup clock
GSR	Input	High True. When this input is asserted, all flip-flops are restored to their initial value in the bitstream.
GTS	Input	High True. When this input is asserted, all user I/Os are 3-stated.
USRCCLKO	Input	CCLK pin output
USRCCLKTS	Input	User CCLK 3-state enable to CCLK pin. When this input is High, CCLK is 3-stated.
USRDONEO	Input	DONE pin output value
USRDONETS	Input	User DONE 3-state enable to DONE pin. When this input is High, DONE is 3-stated.
TCKSPI	Output	Direct from the TCK pin
DINSPI	Output	Direct from the D_IN pin
CFGMCLK	Output	Configuration internal oscillator clock output
CFGCLK	Output	Configuration logic main clock output

TCKSPI and DINSPI are added to allow fabric access to dedicated TCK and D_IN pins. These pins can be used for indirect SPI Flash programming and access to an SPI Flash post configuration from the fabric.

CFGMCLK is driven by the configuration internal oscillator. Its rate is approximately 50 MHz and can be used as a generic clock source instead of a ring oscillator in the fabric.

CFGCLK is the main configuration clock.

Any unused input can be left unconnected or tied to ground.

Chapter 5

Dynamic Reconfiguration Port (DRP)

Dynamic Reconfiguration of Functional Blocks

Background

In the Virtex-5 family of FPGAs, the configuration memory is used primarily to implement user logic, connectivity, and I/Os, but it is also used for other purposes. For example, it is used to specify a variety of static conditions in functional blocks, such as clock management tiles (CMTs).

Sometimes an application requires a change in these conditions in the functional blocks while the block is operational. This can be accomplished by partial dynamic reconfiguration using the JTAG, ICAP, or SelectMAP ports. However, the dynamic reconfiguration port that is an integral part of each functional block simplifies this process greatly. Such configuration ports exist in the CMTs.

Overview

This document describes the addressable, parallel write/read configuration memory that is implemented in each functional block that might require reconfiguration. This memory has the following attributes:

- It is directly accessible from the FPGA fabric. Configuration bits can be written to and/or read from depending on their function.
- Each bit of memory is initialized with the value of the corresponding configuration memory bit in the bitstream. Memory bits can also be changed later through the ICAP.
- The output of each memory bit drives the functional block logic, so the content of this memory determines the configuration of the functional block.

The address space can include status (read-only) and function enables (write-only). Read-only and write-only operations can share the same address space. Figure 5-1 shows how the configuration bits drive the logic in functional blocks directly in earlier FPGA families, and Figure 5-2 shows how the reconfiguration logic changes the flow to read or write the configuration bits.

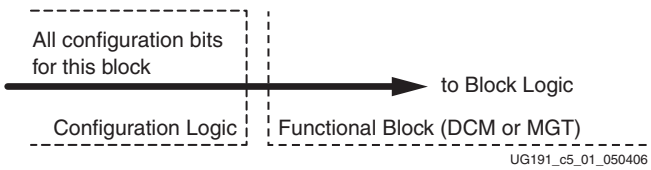


Figure 5-1: **Block Configuration Logic without Dynamic Interface**

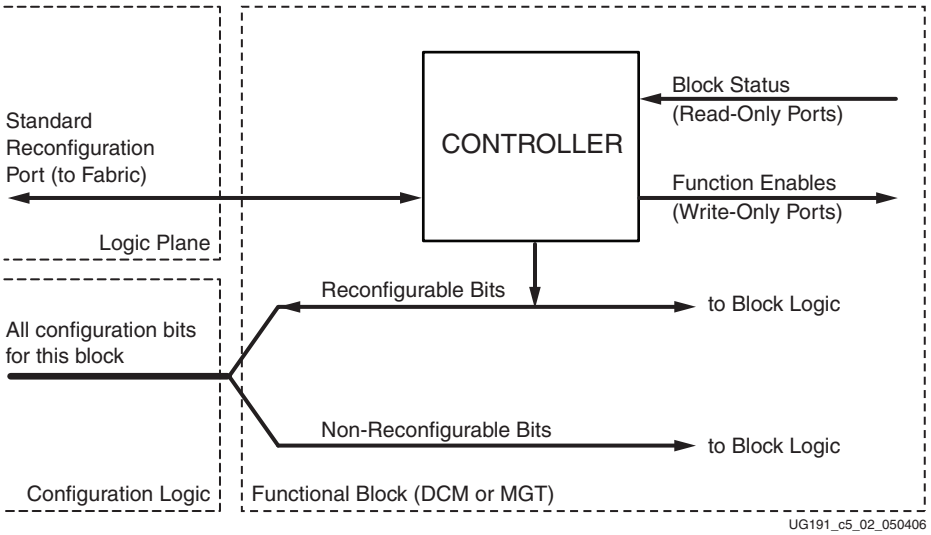


Figure 5-2: **Block Configuration Logic with Dynamic Interface**

Figure 5-3 is the same as Figure 5-2, except the port between the Logic Plane and Functional Block is expanded to show the actual signal names and directions.

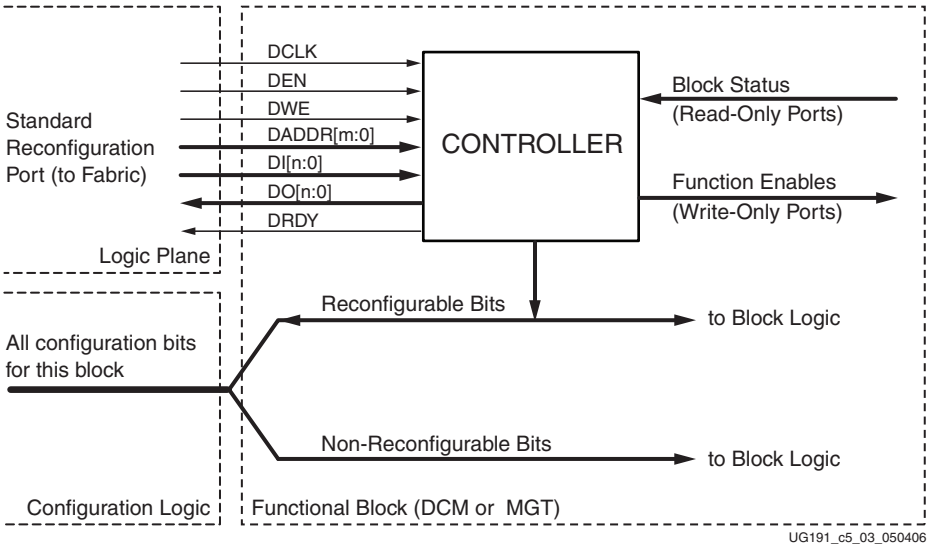


Figure 5-3: **Block Configuration Logic Expanded to Show Signal Names**

FPGA Fabric Port Definition

Table 5-1, page 108, lists each signal on the FPGA Fabric port. The individual functional blocks can implement all or only a subset of these signals. The DCM chapter in the *Virtex-5 User Guide* shows the signals and functions implemented for the specific blocks. In general, the port is a synchronous parallel memory port, with separate read and write buses similar to the block RAM interface. Bus bits are numbered least-significant to most-significant, starting at 0. All signals are active High.

Synchronous timing for the port is provided by the DCLK input, and all the other input signals are registered in the functional block on the rising edge of DCLK. Input (write) data

is presented simultaneously with the write address and DWE and DEN signals prior to the next positive edge of DCLK. The port asserts DRDY for one clock cycle when it is ready to accept more data. The timing requirements relative to DCLK for all the other signals are the same. The output data is not registered in the functional blocks. Output (read) data is available after some cycles following the cycle that DEN and DADDR are asserted. The availability of output data is indicated by the assertion of DRDY.

Figure 5-4 and Figure 5-5 show the timing relationships between the port signals for write and read operations. Absolute timing parameters, such as maximum DCLK frequency, setup time, etc., are defined in DS202, *Virtex-5 Data Sheet: DC and Switching Characteristics*.

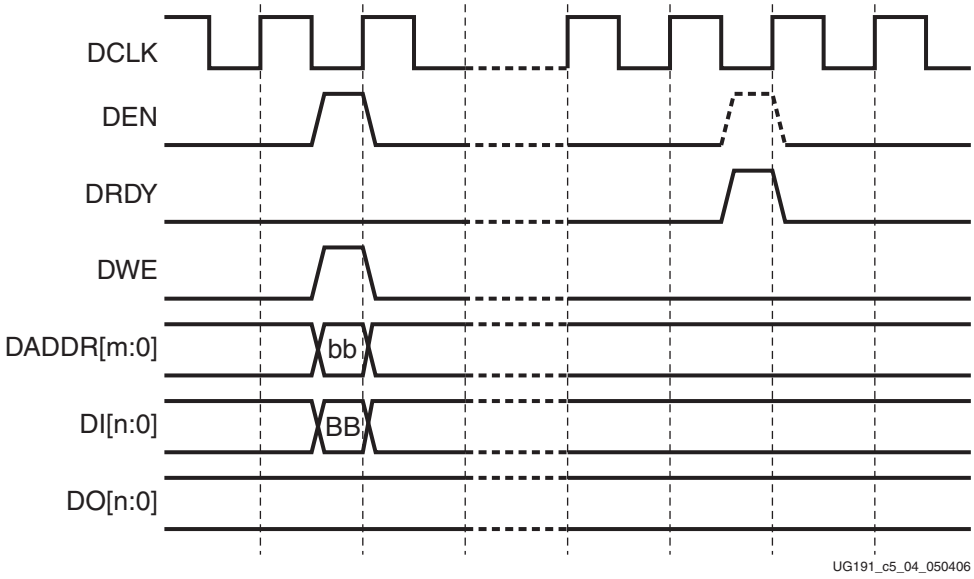


Figure 5-4: **Write Timing with Wait States**

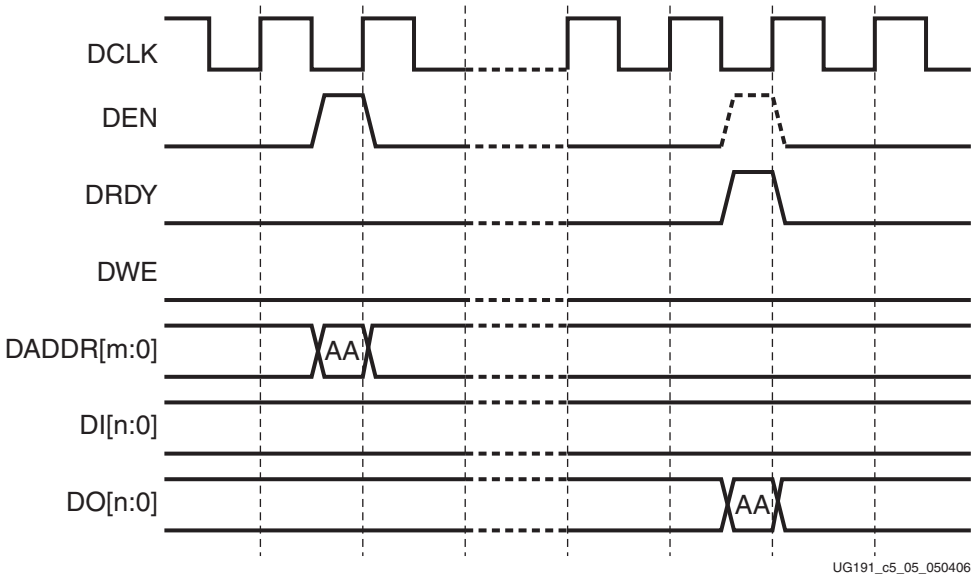


Figure 5-5: **Read Timing with Wait States**

Table 5-1: **Port Signal Definitions**

Signal Name	Direction(1)	Description
DCLK	Input	The rising edge of this signal is the timing reference for all the other port signals. The required hold time for the other input signals relative to the rising edge of DCLK is zero (maximum). Normally, DCLK is driven with a global clock buffer.
DEN	Input	This signal enables all port operations. If DWE is FALSE, it is a read operation, otherwise a write operation. For any given DCLK cycle, all other input signals are *don't care* if DEN is not active. DEN should only be pulsed for one DCLK cycle.
DWE	Input	When active, this signal enables a write operation to the port (see DEN, above). DWE should only be pulsed for one DCLK cycle.
DADDR[m:0]	Input	The value on this bus specifies the individual cell that is written or read on the next cycle of DCLK. The address is presented in the cycle that DEN is active.
DI[n:0]	Input	The value on this bus is the data that is written to the addressed cell. The data is presented in the cycle that DEN and DWE are active, and is captured in a register at the end of that cycle, but the actual write occurs at an unspecified time before DRDY is returned.
DO[n:0]	Output	If DWE was inactive when DEN was activated, the value on this bus when DRDY goes active is the data read from the addressed cell. At all other times, the value on DO[n:0] is undefined.
DRDY	Output	This signal is a response to DEN to indicate that the DRP cycle is complete and another DRP cycle can be initiated. In the case of a port read, the DO bus must be captured on the rising edge of DCLK in the cycle that DRDY is active. The earliest that DEN can go active to start the next port cycle is the same clock cycle that DRDY is active.

Notes:

1. Input denotes input (write) to the DRP.

Changing the Multiply and Divide Values

The Multiply and Divide (M/D) values can be directly programmed in the DCM through the DRP by reading from and writing to hex address `50h`. The user application must read from address `50h` first if either the M or the D value is changed. The M value must be masked into the most significant byte and the D value must be masked into the least significant byte of the 16-bit DRP data word. The entire word is then written back. If both values are changed at the same time, only a write cycle is required.

The DCM must be held in reset by activating input RST while changing the M/D values. At some point after RST is released, signal LOCKED goes true, indicating that the clock outputs of the DCM are valid.

Table 5-2: **Multiplier Settings**

DADDR[15:0]	DEC	DI[15:8]	Function
50h	00	00h (00000000)	N/A
50h	01	01h (00000001)	Multiply by 2
50h	02	02h (00000010)	Multiply by 3
50h	03	03h (00000011)	Multiply by 4
50h	04	04h (00000100)	Multiply by 5
• • •	• • •	• • •	• • •
50h	30	1Eh (00011110)	Multiply by 31
50h	31	1Fh (00011111)	Multiply by 32
50h	32	20h (00100000)	Multiply by 33

Table 5-3: **Divider Settings**

DADDR[15:0]	DEC	DI[7:0]	Function
50h	0000	0000h (0000000000000000)	Divide by 1
50h	0001	0001h (0000000000000001)	Divide by 2
50h	0002	0002h (0000000000000010)	Divide by 3
50h	0003	0003h (0000000000000011)	Divide by 4
50h	0004	0004h (0000000000000100)	Divide by 5
• • •	• • •	• • •	• • •
50h	0030	001Eh (0000000000011110)	Divide by 31
50h	0031	001Fh (0000000000011111)	Divide by 32

If the M or D values are dynamically changed, then in some cases, the frequency mode must also be changed to comply with the specifications in the data sheet. For the DFS_FREQUENCY_MODE DRP address, 41h must be read and bit 3 (DI[2]) is then set to:

- 0 for low frequency mode
- 1 for high frequency mode

All other bits remain unchanged.

For the DLL_FREQUENCY_MODE DRP address, 51h must be read and bits 3 and 4 (DI[3:2]) is then set to:

- 00 for low frequency mode
- 11 for high frequency mode

Again, all other bits must be left undisturbed.

After a DRP read or write access, a read from address 00h must be issued. This read restores the default DCM status output.

XILINX®

Chapter 6

Configuration Details

Configuration Memory Frames

Virtex®-5 FPGA configuration memory is arranged in frames that are tiled about the device. These frames are the smallest addressable segments of the Virtex-5 configuration memory space, and all operations must therefore act upon whole configuration frames. Virtex-5 frame counts and configuration sizes are shown in Table 6-1. Depending on BitGen options, additional overhead exists in the configuration bitstream. The exact bitstream length is available in the rawbits file (.rbt) created by using the "-b" option with bitgen or selecting "Create ASCII Configuration File" in the Generate Programming File options popup in ISE. Bitstream length (words) are roughly equal to the configuration array size (words) plus configuration overhead (words). Bitstream length (bits) are roughly equal to the bitstream length in words times 32.

Table 6-1: **Virtex-5 Device Frame Count, Frame Length, Overhead, and Bitstream Size**

Device	Non-Configuration Frames(1)	Configuration Frames	Total Device Frames	Frame Lengths in Words(2)	Configuration Array Size in Words(3)	Bitstream Overhead in Words(4)
LX30	172	6,376	6,548	41	261,416	272
LX50	258	9,564	9,822	41	392,124	272
LX85	426	16,644	17,070	41	682,404	272
LX110	568	22,192	22,760	41	909,872	272
LX155	800	32,544	33,344	41	1,334,304	272
LX220	1,040	40,496	41,536	41	1,660,336	272
LX330	1,560	60,744	62,304	41	2,490,504	272
LX20T	126	3,762	3,888	41	154,242	272
LX30T	184	7,136	7,320	41	292,576	272
LX50T	276	10,704	10,980	41	438,864	272
LX85T	444	17,784	18,228	41	729,144	272
LX110T	592	23,712	24,304	41	972,192	272
LX155T	808	32,800	33,608	41	1,344,800	272
LX220T	1,064	42,016	43,080	41	1,722,656	272
LX330T	1,596	63,024	64,620	41	2,583,984	272
SX35T	244	10,168	10,412	41	416,888	272
SX50T	366	15,252	15,618	41	625,332	272

Table 6-1: **Virtex-5 Device Frame Count, Frame Length, Overhead, and Bitstream Size *(Continued)***

Device	Non-Configuration Frames(1)	Configuration Frames	Total Device Frames	Frame Lengths in Words(2)	Configuration Array Size in Words(3)	Bitstream Overhead in Words(4)
SX95T	648	27,216	27,864	41	1,115,856	272
SX240T	1,440	60,672	62,112	41	2,487,552	272
FX30T	244	10,296	10,540	41	422,136	272
FX70T	488	20,592	21,080	41	844,272	272
FX100T	696	30,016	30,712	41	1,230,656	272
FX130T	870	37,520	38,390	41	1,538,320	272
FX200T	1,236	54,000	55,236	41	2,214,000	272
TX150T	810	32,980	33,790	41	1,352,180	272
TX240T	1,236	50,112	51,348	41	2,054,592	272

1. Non-configuration frames do not contribute to the bitstream size.
2. All Virtex-5 configuration frames consist of 41 32-bit words.
3. Configuration array size equals the number of configuration frames times the number of words per frame.
4. Configuration overhead consists of commands in the bitstream that are needed to perform configuration but do not themselves program any memory cells. Configuration overhead contributes to the overall bitstream size.

Configuration Registers

All Virtex-5 FPGA bitstream commands are executed by reading or writing to the configuration registers.

Packet Types

The FPGA bitstream consists of two packet types: Type 1 and Type 2. These packet types and their usage are described below.

Type 1 Packet

The Type 1 packet is used for register reads and writes. Only 5 out of 14 register address bits are used in Virtex-5 FPGAs. The header section is always a 32-bit word.

Following the Type 1 packet header is the Type 1 Data section, which contains the number of 32-bit words specified by the word count portion of the header.

Table 6-2: **Type 1 Packet Header Format**

Header Type	Opcode	Register Address	Reserved	Word Count
[31:29]	[28:27]	[26:13]	[12:11]	[10:0]
001	xx	RRRRRRRRRxxxxx	RR	xxxxxxxxxxx

Notes:

1. "R" means the bit is not used and reserved for future use.

Table 6-3: **Opcode Format**

Opcode	Function
00	NOP
01	Read
10	Write
11	Reserved

Type 2 Packet

The Type 2 packet, which must follow a Type 1 packet, is used to write long blocks. No address is presented here because it uses the previous Type 1 packet address. The header section is always a 32-bit word.

Following the Type 2 packet header is the Type 2 Data section, which contains the number of 32-bit words specified by the word count portion of the header.

Table 6-4: **Type 2 Packet Header**

Header Type	Opcode	Word Count
[31:29]	[28:27]	[26:0]
010	RR	xxxxxxxxxxxxxxxxxxxxxxxxxx

For details on the bitstream format, refer to Chapter 6, "Configuration Details."

Type 1 Packet Registers

Table 6-5 summarizes the Type 1 Packet registers. A detailed explanation of selected registers follows.

Table 6-5: **Type 1 Packet Registers**

Reg. Name	Read/Write	Address	Description
CRC	Read/Write	00000	CRC Register
FAR	Read/Write	00001	Frame Address Register
FDRI	Write	00010	Frame Data Register, Input Register (write configuration data)
FDRO	Read	00011	Frame Data Register, Output Register (read configuration data)
CMD	Read/Write	00100	Command Register
CTL0	Read/Write	00101	Control Register 0
MASK	Read/Write	00110	Masking Register for CTL0 and CTL1
STAT	Read	00111	Status Register
LOUT	Write	01000	Legacy Output Register (DOUT for daisy chain)
COR0	Read/Write	01001	Configuration Option Register 0
MFWR	Write	01010	Multiple Frame Write Register

Table 6-5: **Type 1 Packet Registers *(Continued)***

Reg. Name	Read/Write	Address	Description
CBC	Write	01011	Initial CBC Value Register
IDCODE	Read/Write	01100	Device ID Register
AXSS	Read/Write	01101	User Bitstream Access Register
COR1	Read/Write	01110	Configuration Option Register 1
CSOB	Write	01111	Used for daisy chain parallel interface, similar to LOUT
WBSTAR	Read/Write	10000	Warm Boot Start Address Register
TIMER	Read/Write	10001	Watchdog Timer Register
BOOTSTS	Read	10110	Boot History Status Register
CTL1	Read/Write	11000	Control Register 1

CRC Register

Writes to this register perform a CRC check against the bitstream data. If the value written matches the current calculated CRC, the CRC_ERROR flag is cleared and startup is allowed.

FDRI Register

Writes to this register configure frame data at the frame address specified in the FAR register. See "Bitstream Composition," page 126.

FDRO Register

This read-only register provides readback data for configuration frames starting at the address specified in the FAR register. See "Readback Command Sequences," page 136.

MASK Register

Writes to the CTL0 and CTL1 registers are bit-masked by the MASK register.

LOUT Register

Software uses this register to drive data to the DOUT pin during serial daisy-chain configuration.

MFWR Register

This register is used by the bitstream compression option.

CBC Register

This register is used by the bitstream compression option to hold the Initial Vector (IV) for AES decryption.

IDCODE Register

Any writes to the FDRI register must be preceded by a write to this register. The provided IDCODE must match the device's IDCODE. See "Configuration Sequence," page 23.

A read of this register returns the device IDCODE.

AXSS Register

Software uses this register to support the USR_ACCESS_VIRTEX5 primitive (see "USR_ACCESS_VIRTEX5," page 103).

CSOB Register

Software uses this register to assert the CSO_B pin for parallel daisy-chain operation.

Command Register (CMD)

The Command Register is used to instruct the configuration control logic to strobe global signals and perform other configuration functions. The command present in the CMD register is executed each time CMD or FAR is loaded. The code bits are located in the LSB bits of the 32-bit CMD register, with the remaining MSB bits set to 0. Table 6-6 lists the Command Register commands and codes.

Table 6-6: **Command Register Codes**

Command	Code	Description
NULL	00000	Null command.
WCFG	00001	Writes Configuration Data: Used prior to writing configuration data to the FDRI.
MFW	00010	Multiple Frame Write: Used to perform a write of a single frame data to multiple frame addresses.
DGHIGH/ LFRM	00011	Last Frame: Deasserts the GHIGH_B signal, activating all interconnects. The GHIGH_B signal is asserted with the AGHIGH command.
RCFG	00100	Reads Configuration Data: Used prior to reading configuration data from the FDRO.
START	00101	Begins the Startup Sequence: Initiates the startup sequence. The startup sequence begins after a successful CRC check and a DESYNC command are performed.
RCAP	00110	Resets the CAPTURE signal after performing readback-capture in single-shot mode (see "Readback Capture," page 151).
RCRC	00111	Resets CRC: Resets the CRC register.
AGHIGH	01000	Asserts the GHIGH_B signal: Places all interconnect in a High-Z state to prevent contention when writing new configuration data. This command is only used in shutdown reconfiguration. Interconnect is reactivated with the LFRM command.
SWITCH	01001	Switches the CCLK frequency: Updates the frequency of the Master CCLK to the value specified by the OFSEL bits in the COR0 register.

Table 6-6: **Command Register Codes *(Continued)***

Command	Code	Description
GRESTORE	01010	Pulses the GRESTORE signal: sets/resets (depending on user configuration) IOB and CLB flip-flops.
SHUTDOWN	01011	Begin Shutdown Sequence: Initiates the shutdown sequence, disabling the device when finished. Shutdown activates on the next successful CRC check or RCRC instruction (typically an RCRC instruction).
GCAPTURE	01100	Pulses GCAPTURE: Loads the capture cells with the current register states (see "Readback Capture," page 151).
DESYNCH	01101	Resets the DALIGN signal: Used at the end of configuration to desynchronize the device. After desynchronization, all values on the configuration data pins are ignored.
CRCC	10000	When Readback CRC is selected, the CRC is calculated after full configuration and reconfiguration. Toggling GHIGH also calculates the CRC. This command can be used when GHIGH is not toggled during reconfiguration (active partial reconfiguration).
IPROG	01111	Internal PROG for triggering a warm boot.
LTIMER	10001	Reload watchdog timer.

Control Register 0 (CTL0)

CTL0 and CTL1 registers are used to configure the Virtex-5 device. Writes to the CTL0 register are masked by the value in the MASK Register (this allows the GTS_USR_B signal to be toggled without re-specifying the SBITS and PERSIST bits). The name of each bit position in the CTL0 register is given in Figure 6-1 and described in Table 6-7.

Description	Reserved	ICAP_SELECT	Reserved	OverTempPowerDown	Reserved	ConfigFallback	SelectMAPAbort	GLUTMASK_B	Reserved	DEC	SBITS[1:0]	PERSIST	Reserved	GTS_USR_B
Bit Index	31	30	29:13	12	11	10	9	8	7	6	5:4	3	2:1	0
Value	X	0	X	X	0	0	0	1	X	0	0:0	0	X:X	1

Figure 6-1: **Control Register 0 (CTL0)**

Table 6-7: **Control Register 0 Description**

Name	Bit Index	Description
ICAP_SELECT	30	ICAP Port Select. 0: Top ICAP Port Enabled (default) 1: Bottom ICAP Port Enabled
OverTempPowerDown	12	Enables the System Monitor Over-Temperature power down.
ConfigFallback	10	Stops when CFG fails and disables fallback to the default bitstream. The BitGen option is **ConfigFallback: Enable*/Disable**
SelectMAPAbort	9	Disable abort in SelectMAP when RDWR_B toggles when CS_B is asserted. The BitGen option is **SelectMAPAbort: Enable*/Disable**
GLUTMASK_B	8	Global LUT mask signal. Masks any changeable memory cell readback value.
DEC	6	AES Decryptor enable bit.
SBITS[1:0]	[5:4]	Security Level. 00: Read/Write OK (default) 01: Readback disabled 1x: Both Writes and Reads disabled
PERSIST	3	The configuration interface defined by M2:M0 remains after configuration. Typically used only with the SelectMAP interface to allow reconfiguration and readback. See also "SelectMAP Reconfiguration." 0: No (default) 1: Yes
GTS_USR_B	0	Active-Low global 3-state I/Os. Turns off pull-ups if GTS_CFG_B is also asserted. 0: I/Os 3-stated 1: I/Os active

Control Register 1 (CTL1)

CTL0 and CTL1 registers are used to configure the Virtex-5 device. This register is reserved.

Description	Reserved																															
Bit Index	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Figure 6-2: **Control Register 1 (CTL1)**

Frame Address Register (FAR)

The Virtex-5 devices are divided into two halves, the top and the bottom. All frames in Virtex-5 devices have a fixed, identical length of 1312 bits (41 32-bit words).

The FAR is divided into five fields: block type, top/bottom bit, row address, column address, and minor address. See Table 6-8. The address can be written directly or can be auto-incremented at the end of each frame. The typical bitstream starts at address 0 and auto-increment to the final count.

Table 6-8: **Frame Address Register Description**

Address Type	Bit Index	Description
Block Type	[23:21]	Block types are: Interconnect and Block Configuration (000), Block RAM Content (001), Interconnect and Block Special Frames (010 - typically not used by users), and Block RAM Non-Configuration Frames (011 - not used by users).
Top_B Bit	20	Select between top-half rows (0) and bottom-half rows (1).
Row Address	[19:15]	Selects the current row. The row addresses increase from bottom to top.
Column Address	[14:7]	Selects a major column, such as a column of CLBs. Column addresses start at 0 on the left and increase to the right.
Minor Address	[6:0]	Selects a frame within a major column.

Status Register (STAT)

The Status Register indicates the value of numerous global signals. The register can be read through the SelectMAP or JTAG interfaces. Figure 6-3 gives the name of each bit position in the STAT register; a detailed explanation of each bit position is given in Table 6-9.

Description	Reserved					BUS_WIDTH		FS			Reserved	STARTUP_STATE			Reserved	DEC_ERROR	ID_ERROR	DONE	RELEASE_DONE	INIT_B	INIT_COMPLETE	MODE			GHIGH_B	GWE	GTS_CFG_B	EOS	DCI_MATCH	DCM_LOCK	PART_SECURED	CRC_ERROR
Bit Index	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Figure 6-3: **Status Register**

Table 6-9: **Status Register Description**

Name	Bit Index	Description
BUS_WIDTH	[26:25]	CFG bus width auto detection result. If ICAP is enabled, this field reflects the ICAP bus width after configuration is done. 00 = x1 01 = x8 10 = x16 11 = x32
FS	[24:22]	SPI Flash type select
STARTUP_STATE	[20:18]	CFG startup state machine (0 to 7). Phase 0 = 000 Phase 1 = 001 Phase 2 = 011 Phase 3 = 010 Phase 4 = 110 Phase 5 = 111 Phase 6 = 101 Phase 7 = 100
DEC_ERROR	16	FDRI write attempted before or after decrypt operation: 0: No DEC_ERROR 1: DEC_ERROR
ID_ERROR	15	Attempt to write to FDRI without successful DEVICE_ID check. 0: No ID_ERROR 1: ID_ERROR
DONE	14	Value on DONE pin
RELEASE_DONE	13	Value of internal DONE signal: 0: DONE signal not released (pin is actively held Low) 1: DONE signal released (can be held Low externally)
INIT_B	12	Value on INIT_B pin
INIT_COMPLETE	11	Internal signal indicating initialization has completed: 0: Initialization has not finished 1: Initialization finished
MODE	[10:8]	Status of the Mode pins (M[2:0]).
GHIGH_B	7	Status of GHIGH_B: 0: GHIGH_B asserted 1: GHIGH_B deasserted
GWE	6	Status of GWE: 0: FFs and block RAM are write disabled 1: FFs and block RAM are write enabled

Table 6-9: **Status Register Description *(Continued)***

Name	Bit Index	Description
GTS_CFG_B	5	Status of GTS_CFG_B: 0: All I/Os are placed in High-Z state 1: All I/Os behave as configured
EOS	4	End of Startup signal from Startup Block: 0: Startup sequence has not finished 1: Startup sequence has finished
DCI_MATCH	3	0: DCI not matched 1: DCI is matched This bit is a logical AND function of all the MATCH signals (one per bank). If no DCI I/Os are in a particular bank, the bank's MATCH signal = 1.
DCM_LOCK	2	0: DCMs not locked 1: DCMs are locked This bit is a logical AND function of all DCM LOCKED signals. Unused DCM LOCKED signals = 1.
PART_SECURED	1	0: Decryptor security not set 1: Decryptor security set
CRC_ERROR	0	0: No CRC error 1: CRC error

Configuration Options Register 0 (COR0)

The Configuration Options Register 0 is used to set certain configuration options for the device. The name of each bit position in the COR0 is given in Figure 6-4 and described in Table 6-10.

Description	Reserved	Reserved	Reserved	CRC_BYPASS	PWRDWN_STAT	Reserved	DONE_PIPE	DRIVE_DONE	SINGLE	OSCFSEL							SSCLKSRC		DONE_CYCLE			MATCH_CYCLE			LOCK_CYCLE			GTS_CYCLE			GWE_CYCLE		
Bit Index	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	X	0	0	0	0	0	0	0	0	0	0	x	0	1	1	1	1	1	1	1	1	1	0	1	1	0	0

Figure 6-4: **Configuration Options Register 0**

Table 6-10: **Configuration Options Register 0 Description**

Name	Bit Index	Description
CRC_BYPASS	28	Allows bypass of CRC when a special CRC value is loaded (0xDEFC): 0: CRC enabled 1: CRC disabled
PWRDWN_STAT	27	Changes the DONE pin to a Powerdown status pin: 0: DONE pin 1: Powerdown pin
DONE_PIPE	25	0: No pipeline stage for DONEIN 1: Add pipeline stage for DONEIN The FPGA waits on DONE that is delayed by one StartupClk cycle. Use this option when StartupClk is running at high speeds.
DRIVE_DONE	24	0: DONE pin is open drain 1: DONE is actively driven High
SINGLE	23	0: Readback is not single-shot New captured values are loaded on each successive CAP assertion on the CAPTURE_VIRTEX5 primitive. Capture can also be performed with the GCAPTURE instruction in the CMD register. 1: Readback is single-shot. The RCAP instruction must be loaded into the CMD register between successive readbacks.
OSCFSEL	[22:17]	Select CCLK frequency in Master modes (2 MHz–60 MHz)
SSCLKSRC	[16:15]	Startup-sequence clock source. 00: CCLK 01: UserClk (per connection on the CAPTURE_VIRTEX5 block) 1x: JTAGClk
DONE_CYCLE	[14:12]	Startup cycle to release the DONE pin. 000: Startup phase 1 001: Startup phase 2 010: Startup phase 3 011: Startup phase 4 100: Startup phase 5 101: Startup phase 6 110: Startup phase 7 111: Keep

Table 6-10: **Configuration Options Register 0 Description *(Continued)***

Name	Bit Index	Description
MATCH_CYCLE	[11:9]	Startup cycle to stall in until DCI matches. 000: Startup phase 0 001: Startup phase 1 010: Startup phase 2 011: Startup phase 3 100: Startup phase 4 101: Startup phase 5 110: Startup phase 6 111: No Wait
LOCK_CYCLE	[8:6]	Startup cycle to stall in until DCMs lock. 000: Startup phase 0 001: Startup phase 1 010: Startup phase 2 011: Startup phase 3 100: Startup phase 4 101: Startup phase 5 110: Startup phase 6 111: No Wait
GTS_CYCLE	[5:3]	Startup cycle to deassert the Global Three-State (GTS) signal. 000: Startup phase 1 001: Startup phase 2 010: Startup phase 3 011: Startup phase 4 100: Startup phase 5 101: Startup phase 6 110: GTS tracks DONE pin. BitGen option **-g GTS_cycle:Done** 001: Keep
GWE_CYCLE	[2:0]	Startup phase to deassert the Global Write Enable (GWE) signal. 000: Startup phase 1 001: Startup phase 2 010: Startup phase 3 011: Startup phase 4 100: Startup phase 5 101: Startup phase 6 110: GWE tracks DONE pin. BitGen option **-g GWE_cycle:Done** 111: Keep

Configuration Options Register 1 (COR1)

The Configuration Options Register 1 is used to set certain configuration options for the device. The name of each bit position in the COR1 is given in Figure 6-5 and described in Table 6-11.

Description	Reserved	PERSIST_DEASSERT_AT_DESYNCH	Reserved	Reserved	RBCRC_NO_PIN	RBCRC_EN	Reserved	BPI_1ST_READ_CYCLES	BPI_PAGE_SIZE
Bit Index	31 30 29 28 27 26 25 24 23 22 21 20 19 18	17	16	15 14 13 12 11 10	9	8	7 6 5 4	3 2	1 0
Value	0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0 0 0 0 0 0	0	0	0 0 0 0	0 0	0 0

Figure 6-5: **Configuration Options Register 1**

Table 6-11: **Configuration Options Register 1 Description**

Name	Bit Index	Description
PERSIST_DEASSERT_AT_DESYNCH	17	Enables deassertion of PERSIST with the DESYNCH command
RBCRC_NO_PIN	9	Disables INIT_B as read back CRC error status output pin
RBCRC_EN	8	Continuous readback CRC enable
BPI_1ST_READ_CYCLES	[3:2]	First byte read timing: 00: 1 CCLK 01: 2 CCLKs 10: 3 CCLKs 11: 4 CCLKs
BPI_PAGE_SIZE	[1:0]	Flash memory page size: 00: 1 byte/word 01: 4 bytes/words 10: 8 bytes/words 11: Reserved

Warm Boot Start Address Register (WBSTAR)

The name of each bit position in the WBSTAR is given in Figure 6-6 and described in Table 6-12.

Description	Reserved			RS[1:0]		RS_TS_B	START_ADDR																									
Bit Index	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6-6: **WBSTAR Register**

Table 6-12: **WBSTAR Register Description**

Name	Bit Index	Description
RS[1:0]	[28:27]	RS[1:0] pin value on next warm boot
RS_TS_B	26	RS[1:0] pins 3-state enable. 0: Disabled 1: Enabled
START_ADDR	[25:0]	Next bitstream start address. The default start address is 0x0000000.

Watchdog Timer Register (TIMER)

The Watchdog timer is automatically disabled for fallback bitstreams. The name of each bit position in the TIMER register is given in Figure 6-7 and described in Table 6-13.

Description	Reserved	TIMER_USR_MON	TIMER_CFG_MON	TIMER_VALUE
Bit Index	31 30 29 28 27 26	25	24	23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
Value	0 0 0 0 0 0	0	0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Figure 6-7: **TIMER Register**

Table 6-13: **TIMER Register Description**

Name	Bit Index	Description
TIMER_USR_MON	25	Watchdog is enabled during user mode: 0: Disabled 1: Enabled
TIMER_CFG_MON	24	Watchdog is enabled during configuration: 0: Disabled 1: Enabled
TIMER_VALUE	[23:0]	Watchdog time-out value, CFG_MCLK is used for this counter. CFG_MCLK is about 50 MHz by default, and is pre-divided by 256.

Boot History Status Register (BOOTSTS)

This register can only be reset by POR, asserting PROGRAM_B, or issuing a JPROGRAM instruction. At EOS or an error condition, status (_0) is shifted to status (_1), and status (_0) is updated with the current status. The name of each bit position in the BOOTSTS register is given in Figure 6-8 and described in Table 6-14.

Description	Reserved	RBCRC_ERROR_1	WRAP_ERROR_1	CRC_ERROR_1	ID_ERROR_1	WTO_ERROR_1	IPROG_1	FALLBACK_1	VALID_1	RBCRC_ERROR_0	WRAP_ERROR_0	CRC_ERROR_0	ID_ERROR_0	WTO_ERROR_0	IPROG_0	FALLBACK_0	VALID_0
Bit Index	31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Value	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 6-8: **BOOTSTS Register**

Table 6-14: **BOOTSTS Register Description**

Name	Bit Index	Description
RBCRC_ERROR_1	15	RBCRC error cause reconfiguration
WRAP_ERROR_1	14	BPI address counter wraparound error
CRC_ERROR_1	13	CRC error
ID_ERROR_1	12	ID error
WTO_ERROR_1	11	Watchdog time-out error
IPROG_1	10	Internal PROG triggered configuration
FALLBACK_1	9	1: Fallback to default reconfiguration, RS[1:0] actively drives 2'b00 0: Normal configuration
VALID_1	8	Status valid
RBCRC_ERROR_0	7	RBCRC error cause reconfiguration
WRAP_ERROR_0	6	BPI address counter wraparound error
CRC_ERROR_0	5	CRC error
ID_ERROR_0	4	ID error
WTO_ERROR_0	3	Watchdog time-out error
IPROG_0	2	Internal PROG triggered configuration
FALLBACK_0	1	1: Fallback to default reconfiguration, RS[1:0] actively drives 2'b00 0: Normal configuration
VALID_0	0	Status valid

Bitstream Composition

Configuration can begin after the device is powered and initialization has finished, as indicated by the INIT_B pin being released. After initialization, the packet processor ignores all data presented on the configuration interface until it receives the synchronization word. After synchronization, the packet processor waits for a valid packet header to begin the configuration process.

Default Initial Configuration Process

Initial configuration using a *default* bitstream (a bitstream generated using the default BitGen settings) begins by pulsing the PROGRAM_B pin for SelectMAP and Serial configuration modes or by issuing the JPROGRAM instruction for JTAG configuration mode. Configuration proceeds as shown in Table 6-15.

Table 6-15: **Configuration Sequence**

Configuration Data (hex)	Explanation
FFFFFFFF	Dummy Word
000000BB	BusWidth Word
11220044	8/16/32 BusWidth
FFFFFFFF	Dummy Word
FFFFFFFF	Dummy Word
AA995566	Sync Word
20000000	Type 1 NO OP
30020001	Type 1 write 1 words to WBSTAR
00000000	Warm Boot Start Address
30008001	Type 1 write 1 words to CMD
00000000	NULL
20000000	Type 1 NO OP
20000000	Type 1 NO OP
30008001	Type 1 write 1 words to CMD
00000007	RCRC
20000000	Type 1 NO OP
20000000	Type 1 NO OP
30022001	Type 1 write 1 words to TIMER
00000000	TIMER value
30026001	Type 1 write 1 words to CBC
00000000	CBC value
30012001	Type 1 write 1 words to COR0
0000401D	done@4 m@0 l@0 gts@3 gwe@5
3001C001	Type 1 write 1 words to COR1
00000000	COR1 value
30018001	Type 1 write 1 words to ID
02896093	ID code
30008001	Type 1 write 1 words to CMD
00000009	SWITCH
20000000	Type 1 NO OP
3000C001	Type 1 write 1 words to MASK
00000100	MASK value

Table 6-15: **Configuration Sequence *(Continued)***

Configuration Data (hex)	Explanation
3000A001	Type 1 write 1 words to CTL0
00000100	CTL0 value
3000C001	Type 1 write 1 words to MASK
00000000	MASK value
30030001	Type 1 write 1 words to CTL1
00000000	CTL1 value
20000000	Type 1 NO OP
20000000	Type 1 NO OP
20000000	Type 1 NO OP
20000000	Type 1 NO OP
20000000	Type 1 NO OP
20000000	Type 1 NO OP
20000000	Type 1 NO OP
20000000	Type 1 NO OP
30002001	Type 1 write 1 words to FAR
00000000	FAR value
30008001	Type 1 write 1 words to CMD
00000001	WCFG
20000000	Type 1 NO OP
30004000	Type 1 write 0 words to FDRI
50000400	Type 2 write 1024 words to FDRI
00000000	Data word 1
	Data word n
	Data word n + 1
00000000	Data word last (1024)
30000001	Type 1 write 1 words to CRC
F1927570	CRC value
30008001	Type 1 write 1 words to CMD
0000000A	GRESTORE
20000000	Type 1 NO OP
30008001	Type 1 write 1 words to CMD
00000003	LFRM

Table 6-15: **Configuration Sequence *(Continued)***

Configuration Data (hex)	Explanation
3000C001	Type 1 write 1 words to MASK
00000000	Data word 1
3000A001	Type 1 write 1 words to CTL0
00000000	CTL0 register value
20000000	Type 1 NO OP
	…
20000000	Type 1 NO OP
30008001	Type 1 write 1 words to CMD
00000005	START
20000000	Type 1 NO OP
30002001	Type 1 write 1 words to FAR
00FFFF80	FAR value
30000001	Type 1 write 1 words to CRC
845EAE07	CRC value
30008001	Type 1 write 1 words to CMD
0000000D	DESYNCH
20000000	Type 1 NO OP
	…
20000000	Type 1 NO OP

Frame Addressing

Frame Bits

A frame is the smallest amount of configuration information that can be accessed. It can be thought of as a vertical stack of 1312 bits that spans the whole height of a row.

A row consists of a stack of basic blocks (20 CLBs, 40 IOBs, 4 block RAMs, etc.) with a row of HCLK tiles passing through the middle. Thus, out of the 1312 bits in a frame, 640 bits are found in the basic blocks located above the row of HCLK tiles, 640 bits are found below, and 32 bits are used inside the HCLK tiles. Of the 32 bits in the HCLK tile, the 16 MSBs are unused, the 12 LSBs are used for the ECC bits, and the 4 remaining bits are used as miscellaneous configuration bits for circuits inside the HCLK tiles.

A frame of configuration bits is shifted inside the Frame Data Register (FDR) by words of 32 bits. Thus, 41 words are required for a frame of 1312 bits. Figure 6-9 shows the 41 words in the bitstream, numbered from 1 to 41, and how they correspond to the bits inside the 1312-bit frame. This correspondence is true for all rows inside the FPGA.

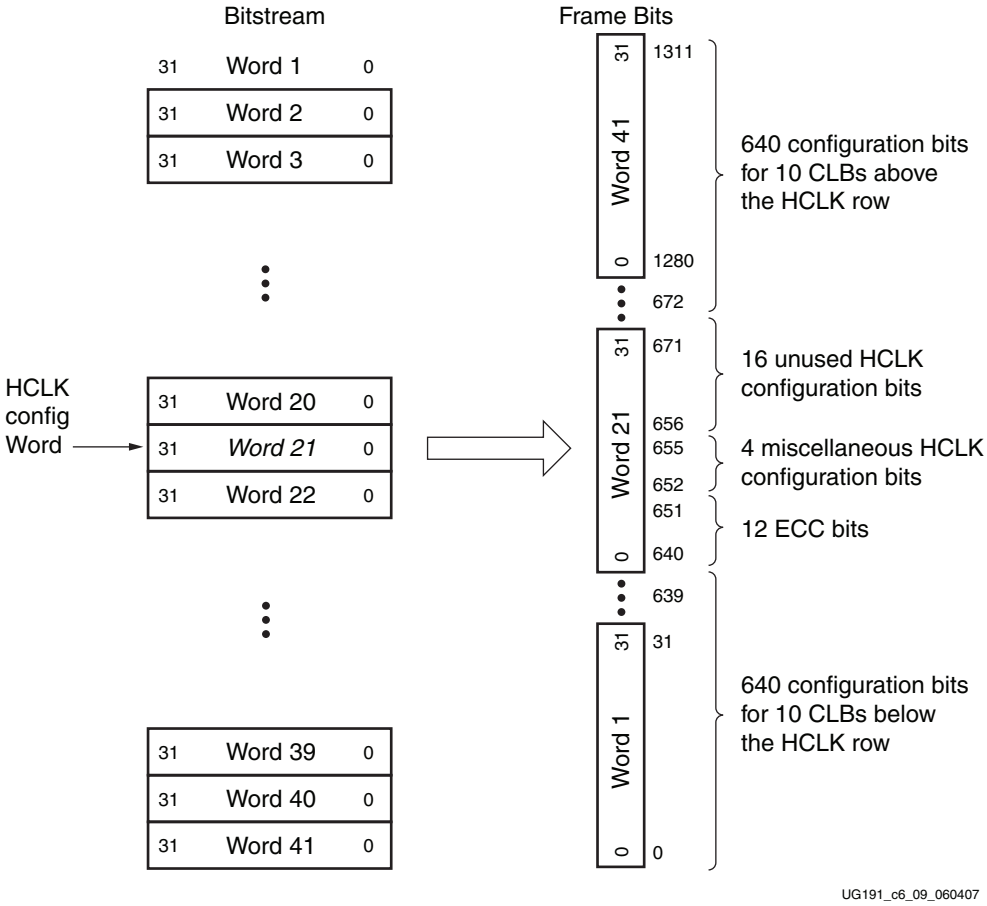


Figure 6-9: **Configuration Words in the Bitstream and Configuration Bits in a Frame**

Frame Address

Each configuration frame in the FPGA has a unique 32-bit address that can be divided into 5 parts, as shown in Figure 6-10:

- Block type
- Top/Bottom indicator
- Row address
- Major address (column address)
- Minor address (frame address inside a column)

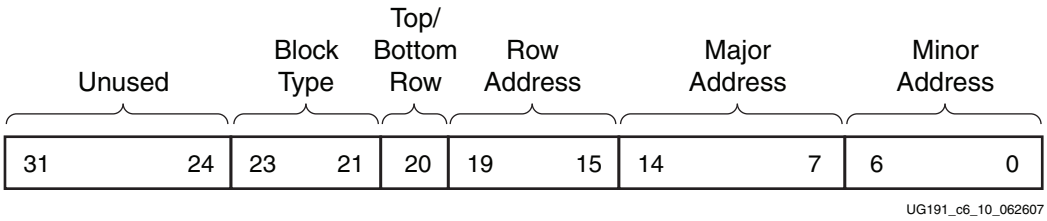


Figure 6-10: **Division of 32-Bit Frame Address into 5 Parts**

The easiest way to explain the meaning of each part is to start with the row address. The block type is discussed last.

Row Address (with Top/Bottom Indicator)

From a configuration point of view, it is best to first divide the FPGA fabric into different rows (and then into columns). The rows are numbered from 0 (up to 9) in the top and bottom halves of the FPGA, starting from the center, as shown in Figure 6-11. During configuration, the row addresses are scanned from 0 upward in the top half, and then from 0 upward in the bottom half.

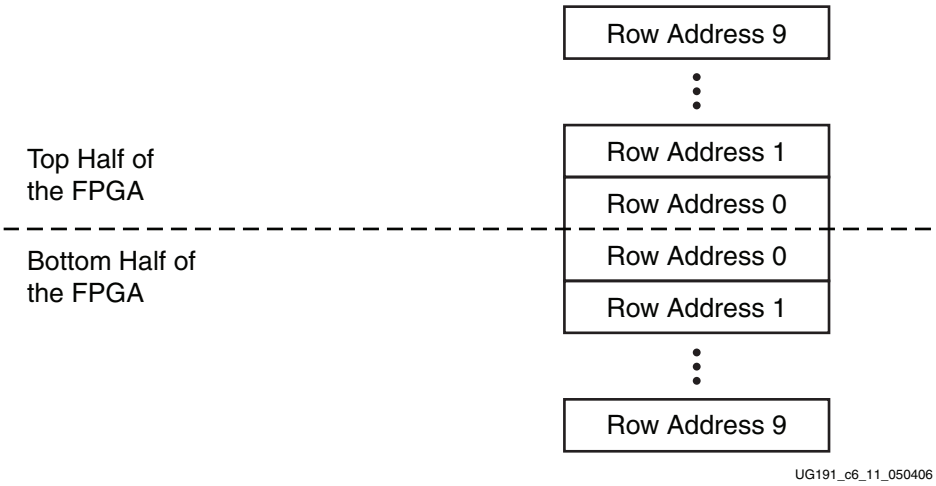


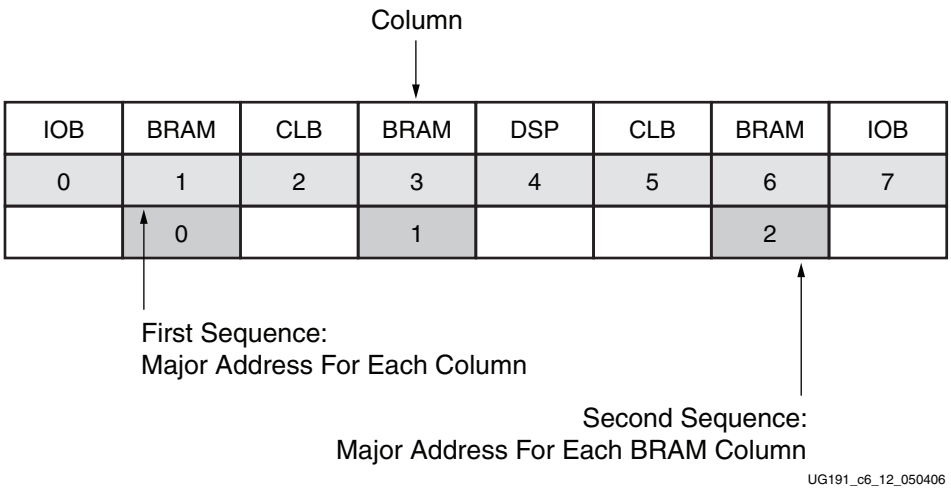
Figure 6-11: **Row Addresses in the FPGA**

The largest Virtex-5 device has a total of 12 rows (6 in each half), even though the hardware design can support up to 20 rows (10 in each half).

Major Address

Each row is divided into the same number of columns, where a column corresponds to a block in the array (CLB, DSP, block RAM, IOB, etc.). The major addresses are numbered from left to right starting with 0.

There are two sequences of major address per row, as shown in Figure 6-12. The first sequence (in the light gray) assigns a major address to each column, and it is used to select a column for normal configuration. The second sequence (in darker gray) assigns a major address to each block RAM column, which is used to access configuration frames that are specific to the block RAM contents. This means that block RAMs have two major addresses: one to access their normal configuration, and the other to access their contents.



Notes:

1. The sequence of normal major addresses is in light gray, and the sequence of block RAM major addresses is in dark gray.

Figure 6-12: **Assignment of Major Addresses in a Row**

Minor Address

Each column contains a certain number of frames, which are accessed using the minor address. The number of frames inside a column depends on the block type and on the column, as explained in "Block Type."

Block Type

The block type divides the configuration address space into eight sections, but only four are used in Virtex-5 devices. This is done to separate configuration frames depending on their function or on how they are accessed.

The four block types used in Virtex-5 devices are:

- Interconnect and Block Configuration
- Block RAM contents
- Interconnect and block special frames
- Block RAM non-configuration frame

Interconnect and Block Configuration

This section of the configuration address space contains all the normal interconnect and block configuration frames (CLB, DSP, IOB, etc.). It also includes the frames that configure the parameters of the block RAM (such as port widths and FIFO operation). However, it does not include the contents of the block RAM.

Table 6-16 lists the number of frames (minor addresses) per column when using this block type.

Table 6-16: **Frames (Minor Addresses) per Column**

Block	Number of Frames
CLB	36
DSP	28
Block RAM	30
IOB	54
Clock Column	4
Clock Column	4

The frames are numbered from left to right, starting with 0. For each block, except the clock column, frames numbered 0 to 25 access the Interconnect for that column. For all blocks, except the CLB and the clock column, frames numbered 26 and 27 access the Interface for that column. All other frames are specific to that block.

Block RAM Contents

The actual memory contents of the block RAM is configured in a different section of the address space for two reasons. First, access to the configuration frames is done differently for block RAM contents than for regular configuration frames. Second, it is easy to skip configuration of the block RAM contents if it is not required, which can significantly reduce the size of the bitstream.

When accessing the block RAM contents, the block RAM major address must be used (from the second sequence of major addresses). This major address is probably different from the normal major address used to access the configuration of the block RAM and of its interconnect. The block RAM contains 128 frames per column and HCLK row.

Interconnect and Block Special Frame

There is one special frame per column, which contains configuration bits used for partial reconfiguration. Because few designs use partial reconfiguration, this section of the address space can be skipped for normal use.

This special frame is accessed with a minor address of 0. Only the 16 HCLK bits are used, and all other bits are assumed to be 0. Thus a special frame contains only 4 bits of data, because the other 12 HCLK bits are the 12 ECC bits, as shown in Table 6-17.

Table 6-17: **Special Frame Bits**

Bit Numbers	Use
<15>	Unused
<14>	Gates GTS_CFG_B in IOB column only
<13>	Gates GCAP and GRESTORE in all columns
<12>	Gates GHIGH_B and GWE in all columns
<11:0>	12 ECC bits (assumes that all other bits are 0)

The "Partial Reconfiguration" section provides more details.

XILINX®

Chapter 7

Readback and Configuration Verification

Virtex®-5 devices allow users to read configuration memory through the SelectMAP, ICAP, and JTAG interfaces. There are two styles of readback: Readback Verify and Readback Capture. During Readback Verify, the user reads all configuration memory cells, including the current values on all user memory elements (LUT RAM, SRL16, and block RAM). Readback Capture is a superset of Readback Verify—in addition to reading all configuration memory cells, the current state of all internal CLB and IOB registers is read, and is useful for design debugging.

To read configuration memory, users must send a sequence of commands to the device to initiate the readback procedure. Once initiated the device dumps the contents of its configuration memory to the SelectMAP or JTAG interface. The "Accessing Configuration Registers through the SelectMAP Interface" section, IEEE 1149.1 JTAG, and IEEE 1532 JTAG describe the steps for reading configuration memory.

Users can send the readback command sequence from a custom microprocessor, CPLD, or FPGA-based system, or use iMPACT to perform JTAG-based readback verify. iMPACT, the device programming software provided with the Xilinx Integrated Software Environment (ISE), can perform all readback and comparison functions for Virtex-5 devices and report to the user whether there were any configuration errors. iMPACT cannot perform capture operations, although Readback Capture is seldom used for design debugging because the ChipScope™ ILA, sold separately through the Xilinx website, provides superior design debugging functionality in a user-friendly interface.

Once configuration memory is read from the device, the next step is to determine if there are any errors by comparing the readback bitstream to the configuration bitstream. The "Verifying Readback Data" section explains how this is done.

Preparing a Design for Readback

There are two mandatory bitstream settings for readback: the BitGen security setting must not prohibit readback (`-g security:none`), and bitstream encryption must not be used. Additionally, if readback is to be performed through the SelectMAP interface, the port must be set to retain its function after configuration by setting the *persist* option in BitGen (`-g Persist:Yes`), otherwise the SelectMAP data pins revert to user I/O, precluding further configuration operations. Beyond these security and encryption requirements, no special considerations are necessary to enable readback through the Boundary-Scan port.

If capture functionality is needed, the CAPTURE_VIRTEX5 primitive can instantiated in the user design (Figure 7-7, page 151). Alternatively, writing the GCAPTURE command to the CMD register can be used (see Readback Capture). To capture the state of user registers, the user design triggers the CAP input on this primitive, storing the current

register values in configuration memory. The register values are later read out of the device along with all other configuration memory.

Readback Command Sequences

Virtex-5 configuration memory is read from the FDRO (Frame Data Register - Output) configuration register and can be accessed from the JTAG, SelectMAP, and ICAP interfaces. For the JTAG and SelectMAP interfaces, readback is possible while the FPGA design is active or in a shutdown state, although block RAMs cannot be accessed by the user design while they are being accessed by the configuration logic.

Accessing Configuration Registers through the SelectMAP Interface

To read configuration memory through the SelectMAP interface, users must set the interface for write control to send commands to the FPGA, and then switch the interface to read control to read data from the device. Write and read control for the SelectMAP interface is determined by the RDWR_B input: the SelectMAP data pins are inputs when the interface is set for Write control (RDWR_B = 0); they are outputs when the interface is set for Read control (RDWR_B = 1).

The CS_B signal must be deasserted (CS_B =1) before toggling the RDWR_B signal, otherwise the user causes an abort (refer to "SelectMAP ABORT" in Chapter 2 for details).

The procedure for changing the SelectMAP interface from Write to Read Control, or vice versa, is:

1. Deassert CS_B.
2. Toggle RDWR_B.

   RDWR_B = 0: Write control
   RDWR_B = 1: Read control
3. Assert CS_B.
4. CS_B and RDWR_B is synchronous to CCLK.
5. This procedure is illustrated in Figure 7-1.

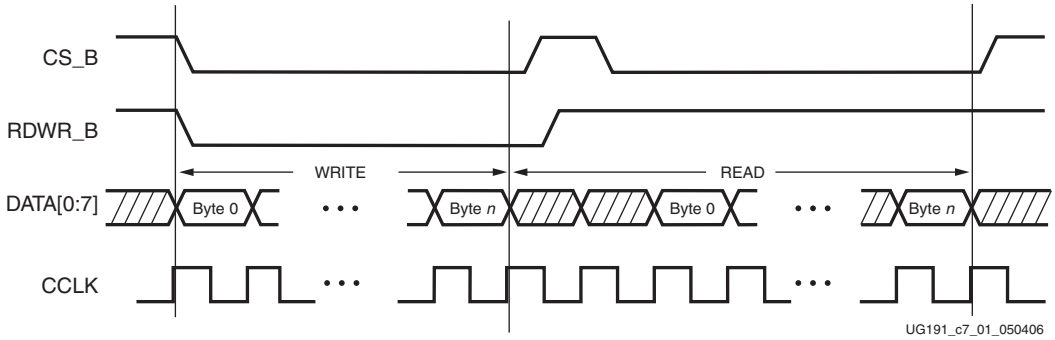


Figure 7-1: **Changing the SelectMAP Port from Write to Read Control**

Configuration Register Read Procedure (SelectMAP)

The simplest read operation targets a configuration register such as the COR0 or STAT register. Any configuration register with read access can be read through the SelectMAP interface, although not all registers offer read access. The procedure for reading the STAT register through the SelectMAP interface follows:

1. Write the Bus Width detection sequence Dummy word and Synchronization word to the device followed by a NOOP.
2. Write the *read STAT register* packet header to the device.
3. Write two NOOP words to the device to flush the packet buffer.
4. Read one word from the SelectMAP interface; this is the Status register value.
5. Write the DESYNCH command to the device.
6. Write two dummy words to the device to flush the packet buffer.

Table 7-1: **Status Register Readback Command Sequence (SelectMAP)**

Step	SelectMAP Port Direction	Configuration Data	Explanation
1	Write	FFFFFFFF	Dummy Word
2	Write	000000BB	Bus Width Sync Word
3	Write	11220044	Bus Width Detect
4	Write	FFFFFFFF	Dummy Word
5	Write	AA995566	Sync Word
6	Write	20000000	NOOP
7	Write	2800E001	Write Type1 packet header to read STAT register
8	Write	20000000	NOOP
9	Write	20000000	NOOP
10	Read	SSSSSSSS	Device writes one word from the STAT register to the configuration interface
11	Write	30008001	Type 1 Write 1 Word to CMD
12	Write	0000000D	DESYNC Command
13	Write	20000000	NOOP
14	Write	20000000	NOOP

The user must change the SelectMAP interface from write to read control between steps 8 and 9, and back to write control after step 9, as illustrated in Figure 7-2.

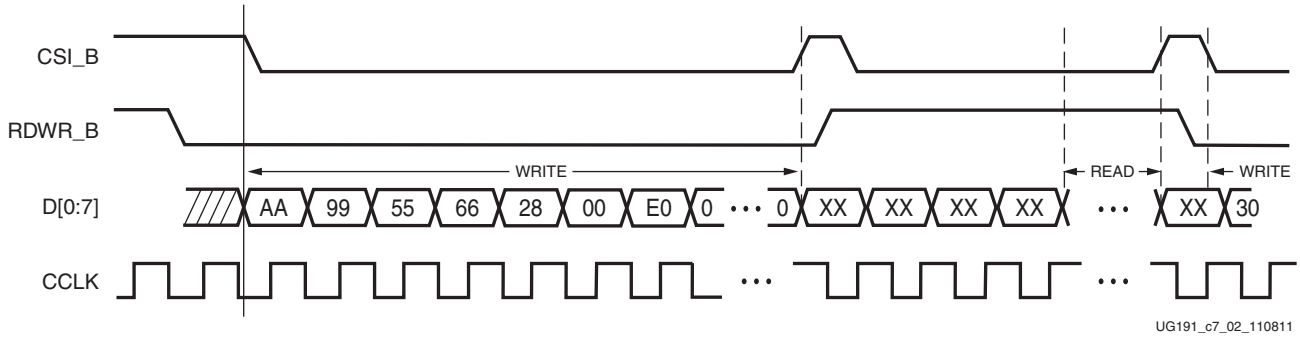


Figure 7-2: **SelectMAP Status Register Read**

To read registers other than STAT, the address specified in the Type-1 packet header in step 2 of Table 7-1 should be modified and the word count changed if necessary. Reading from the FDRO register is a special case that is described in "Configuration Memory Read Procedure (SelectMAP)."

Configuration Memory Read Procedure (SelectMAP)

The process for reading configuration memory from the FDRO register is similar to the process for reading from other registers. Additional steps are needed to accommodate the configuration logic. Configuration data coming from the FDRO register passes through the frame buffer. The first frame of readback data should be discarded.

1. Write the Bus Width detection sequence and Synchronization word to the device.
2. Write one NOOP command.
3. Write the Shutdown command, and write one NOOP command.
4. Write the RCRC command to the CMD register, and write one NOOP command.
5. Write five NOOP instructions to ensure the shutdown sequence has completed. DONE goes Low during the shutdown sequence.
6. Write the RCFG command to the CMD register, and write one NOOP command.
7. Write the Starting Frame Address to the FAR (typically `0x00000000`).
8. Write the *read FDRO register* packet header to the device. The FDRO read length is given by:

   FDRO Read Length = (words per frame) x (frames to read + 1)

   One extra frame is read to account for the frame buffer. Users should strobe readback data while DOUT_BUSY is Low. The frame buffer produces one dummy frame at the beginning of the read. Also, one extra word is read in SelectMap8 mode.
9. Write 32 NOOP commands to the device to flush the packet buffer.
10. Read the FDRO register from the SelectMAP interface. The FDRO read length is the same as in step 9 above.
11. Write one NOOP instruction.
12. Write the START command, and write one NOOP command.
13. Write the RCRC command, and write one NOOP command.
14. Write the DESYNCH command.

15. Write at least 64 bits of NOOP commands to flush the packet buffer. Continue sending CCLK pulses until DONE goes High.

Table 7-2 shows the readback command sequence.

Table 7-2: **Shutdown Readback Command Sequence (SelectMAP)**

Step	SelectMAP Port Direction	Configuration Data	Explanation
1	Write	FFFFFFFF	Dummy Word
		000000BB	Bus Width Sync Word
		11220044	Bus Width Detect
		FFFFFFFF	Dummy Word
		AA995566	Sync Word
2	Write	20000000	Type 1 NOOP Word 0
3	Write	30008001	Type 1 Write 1 Word to CMD
		0000000B	SHUTDOWN Command
		20000000	Type 1 NOOP Word 0
4	Write	30008001	Type 1 Write 1 Word to CMD
		00000007	RCRC Command
		20000000	Type 1 NOOP Word 0
5	Write	20000000	Type 1 NOOP Word 0
		20000000	Type 1 NOOP Word 0
		20000000	Type 1 NOOP Word 0
		20000000	Type 1 NOOP Word 0
		20000000	Type 1 NOOP Word 0
6	Write	30008001	Type 1 Write 1 Word to CMD
		00000004	RCFG Command
		20000000	Type 1 NOOP Word 0
7	Write	30002001	Type 1 Write 1 Word to FAR
		00000000	FAR Address = 00000000
8	Write	28006000	Type 1 Read 0 Words from FDRO
		48024090	Type 2 Read 147,600 Words from FDRO
9	Write	20000000	Type 1 NOOP Word 0
		. . .	Type 1 31 more NOOPs Word 0
10	Read	00000000	Packet Data Read FDRO Word 0
		. . .	
		00000000	Packet Data Read FDRO Word 147599
11	Write	20000000	Type 1 NOOP Word 0
12	Write	30008001	Type 1 Write 1 Word to CMD
		00000005	START Command
		20000000	Type 1 NOOP Word 0

Table 7-2: **Shutdown Readback Command Sequence (SelectMAP) *(Continued)***

Step	SelectMAP Port Direction	Configuration Data	Explanation
13	Write	30008001	Type 1 Write 1 Word to CMD
		00000007	RCRC Command
		20000000	Type 1 NOOP Word 0
14	Write	30008001	Type 1 Write 1 Word to CMD
		0000000D	DESYNCH Command
15	Write	20000000	Type 1 NOOP Word 0
		20000000	Type 1 NOOP Word 0

User logic should strobe readback data while DOUT_BUSY is Low after switching from a write to a read (both CS_B and RDWR_B are Low). DOUT_BUSY must be monitored to determine when the readback data is valid.

When readback is initiated, and after BUSY is deasserted, 42 dummy words are read back. In x16 and x8 modes, the readback cycles multiply by 2 and 4 respectively.

Table 7-3: **Readback DOUT_BUSY Latency (SelectMAP)**

	x8	x16	x32
Read to DOUT_BUSY Latency	1 clock	3 clocks	4 clocks

Notes:

These latencies assume CS_B is deasserted for one cycle when changing from write to read (RDWR_B deassertion). It is best to monitor the BUSY signal for valid readback data.

Accessing Configuration Registers through the JTAG Interface

JTAG access to the Virtex-5 configuration logic is provided through the JTAG CFG_IN and CFG_OUT registers. The CFG_IN and CFG_OUT registers are not configuration registers, rather they are JTAG registers like BYPASS and BOUNDARY_SCAN. Data shifted into the CFG_IN register go to the configuration packet processor, where they are processed in the same way commands from the SelectMAP interface are processed.

Readback commands are written to the configuration logic by going through the CFG_IN register; configuration memory is read through the CFG_OUT register. The JTAG state transitions for accessing the CFG_IN and CFG_OUT registers are described in Table 7-4.

Table 7-4: **Shifting in the JTAG CFG_IN and CFG_OUT Instructions**

Step	Description	Set and Hold		# of Clocks (TCK)
		TDI	TMS	
1	Clock five *1*s on TMS to bring the device to the TLR state	X	1	5
2	Move into the RTI state	X	0	1
3	Move into the Select-IR state	X	1	2
4	Move into the Shift-IR State	X	0	2

Table 7-4: **Shifting in the JTAG CFG_IN and CFG_OUT Instructions *(Continued)***

Step	Description	Set and Hold		# of Clocks (TCK)
		TDI	TMS	
5	Shift the first nine bits of the CFG_IN or CFG_OUT instruction, LSB first	111000101 (CFG_IN) 111000100 (CFG_OUT)	0	9
6	Shift the MSB of the CFG_IN or CFG_OUT instruction while exiting SHIFT-IR	1	1	1
7	Move into the SELECT-DR state	X	1	2
8	Move into the SHIFT-DR state	X	0	2
9	Shift data into the CFG_IN register or out of the CFG_OUT register while in SHIFT_DR, MSB first	X	0	X
10	Shift the LSB while exiting SHIFT-DR	X	1	1
11	Reset the TAP by clocking five *1*s on TMS	X	1	5

Configuration Register Read Procedure (JTAG)

The simplest read operation targets a configuration register such as the COR0 or STAT register. Any configuration register with read access can be read through the JTAG interface, although not all registers offer read access. The procedure for reading the STAT register through the JTAG interface follows:

1. Reset the TAP controller.
2. Shift the CFG_IN instruction into the JTAG Instruction Register through the Shift-IR state. The LSB of the CFG_IN instruction is shifted first; the MSB is shifted while moving the TAP controller out of the SHIFT-IR state.
3. Shift packet write commands into the CFG_IN register through the Shift-DR state:
   a. Write the Synchronization word to the device.
   b. Write one NOOP instruction to the device.
   c. Write the *read STAT register* packet header to the device.
   d. Write two dummy words to the device to flush the packet buffer.

   The MSB of all configuration packets sent through the CFG_IN register must be sent first. The LSB is shifted while moving the TAP controller out of the SHIFT-DR state.
4. Shift the CFG_OUT instruction into the JTAG Instruction Register through the Shift-IR state. The LSB of the CFG_OUT instruction is shifted first; the MSB is shifted while moving the TAP controller out of the SHIFT-IR state.
5. Shift 32 bits out of the Status register through the Shift-DR state.
6. Reset the TAP controller.

Table 7-5: **Status Register Readback Command Sequence (JTAG)**

Step	Description	Set and Hold		# of Clocks (TCK)
		TDI	TMS	
1	Clock five *1*s on TMS to bring the device to the TLR state.	X	1	5
	Move into the RTI state.	X	0	1
	Move into the Select-IR state.	X	1	2
	Move into the Shift-IR state.	X	0	2
2	Shift the first nine bits of the CFG_IN instruction, LSB first.	111000101 (CFG_IN)	0	9
	Shift the MSB of the CFG_IN instruction while exiting SHIFT-IR.	1	1	1
	Move into the SELECT-DR state.	X	1	2
	Move into the SHIFT-DR state.	X	0	2
3	Shift configuration packets into the CFG_IN data register, MSB first.	a:0xAA995566 b:0x20000000 c:0x2800E001 d:0x20000000 e:0x20000000	0	159
	Shift the LSB of the last configuration packet while exiting SHIFT-DR.	0	1	1
	Move into the SELECT-IR state.	X	1	3
	Move into the SHIFT-IR state.	X	0	2
4	Shift the first nine bits of the CFG_OUT instruction, LSB first.	111000100 (CFG_OUT)	0	9
	Shift the MSB of the CFG_OUT instruction while exiting Shift-IR.	1	1	1
	Move into the SELECT-DR state.	X	1	2
	Move into the SHIFT-DR state.	X	0	2
5	Shift the contents of the STAT register out of the CFG_OUT data register.	0xSSSSSSSS	0	31
	Shift the last bit of the STAT register out of the CFG_OUT data register while exiting SHIFT-DR.	S	1	1
	Move into the Select-IR state.	X	1	3
	Move into the Shift-IR State.	X	0	2
6	Reset the TAP Controller.	X	1	5

The packets shifted in to the JTAG CFG_IN register are identical to the packets shifted in through the SelectMAP interface when reading the STAT register through SelectMAP.

Configuration Memory Read Procedure (1149.1 JTAG)

The process for reading configuration memory from the FDRO register through the JTAG interface is similar to the process for reading from other registers. However, additional steps are needed to accommodate frame logic. Configuration data coming from the FDRO register pass through the frame buffer, therefore the first frame of readback data is *dummy data* and should be discarded (refer to the FDRI and FDRO register description). The 1149.1 JTAG readback flow is recommended for most users.

1. Reset the TAP controller.
2. Shift the CFG_IN instruction into the JTAG Instruction Register. The LSB of the CFG_IN instruction is shifted first; the MSB is shifted while moving the TAP controller out of the SHIFT-IR state.
3. Shift packet write commands into the CFG_IN register through the Shift-DR state:
   a. Write a dummy word to the device.
   b. Write the Synchronization word to the device.
   c. Write a NOOP instruction to the device.
   d. Write the RCRC command to the device.
   e. Write two dummy words to flush the packet buffer.
4. Shift the JSHUTDOWN instruction into the JTAG Instruction Register.
5. Move into the RTI state; remain there for 12 TCK cycles to complete the Shutdown sequence. The DONE pin goes Low during the Shutdown sequence.
6. Shift the CFG_IN instruction into the JTAG Instruction Register.
7. Move to the Shift-DR state and shift packet write commands into the CFG_IN register:
   a. Write a dummy word to the device.
   b. Write the Synchronization word to the device.
   c. Write a NOOP instruction to the device.
   d. Write the *write CMD register* header.
   e. Write the RCFG command to the device.
   f. Write the *write FAR register* header.
   g. Write the starting frame address to the FAR register (typically `0x0000000`).
   h. Write the *read FDRO register* Type-1 packet header to the device.
   i. Write a Type-2 packet header to indicate the number of words to read from the device.
   j. Write two dummy words to the device to flush the packet buffer.

   The MSB of all configuration packets sent through the CFG_IN register must be sent first. The LSB is shifted while moving the TAP controller out of the SHIFT-DR state.
8. Shift the CFG_OUT instruction into the JTAG Instruction Register through the Shift-DR state. The LSB of the CFG_OUT instruction is shifted first; the MSB is shifted while moving the TAP controller out of the SHIFT-IR state.
9. Shift frame data from the FDRO register through the Shift-DR state.
10. Reset the TAP controller.

Table 7-6: **Shutdown Readback Command Sequence (JTAG)**

Step	Description	Set and Hold		# of Clocks (TCK)
		TDI	TMS	
1	Clock five *1*s on TMS to bring the device to the TLR state.	X	1	5
	Move into the RTI state.	X	0	1
	Move into the Select-IR state.	X	1	2
	Move into the Shift-IR State.	X	0	2
2	Shift the first nine bits of the CFG_IN instruction, LSB first.	111000101	0	9
	Shift the MSB of the CFG_IN instruction while exiting Shift-IR.	1	1	1
	Move into the SELECT-DR state.	X	1	2
	Move into the SHIFT-DR state.	X	0	2
3	Shift configuration packets into the CFG_IN data register, MSB first.	a:0xFFFFFFFF b:0xAA995566 c:0x20000000 d:0x30008001 e:0x00000007 f:0x20000000 g:0x20000000	0	223
	Shift the LSB of the last configuration packet while exiting SHIFT-DR.	0	1	1
	Move into the SELECT-IR State.	X	1	3
	Move into the SHIFT-IR State.	X	0	2
4	Shift the first 9 bits of the JSHUTDOWN instruction, LSB first.	111001101	0	9
	Shift the MSB of the JSHUTDOWN instruction while exiting SHIFT-IR.	1	1	1
5	Move into the RTI state; remain there for 12 TCK cycles.	X	0	12
	Move into the Select-IR state.	X	1	2
	Move into the Shift-IR State.	X	0	2
6	Shift the first nine bits of the CFG_IN instruction, LSB first.	111000101	0	9
	Shift the MSB of the CFG_IN instruction while exiting SHIFT-IR.	1	1	1
	Move into the SELECT-DR state.	X	1	2
	Move into the SHIFT-DR state.	X	0	2

Table 7-6: **Shutdown Readback Command Sequence (JTAG) *(Continued)***

Step	Description	Set and Hold		# of Clocks (TCK)
		TDI	TMS	
7	Shift configuration packets into the CFG_IN data register, MSB first.	a:0xFFFFFFFF b:0xAA995566 c:0x20000000 d:0x30008001 e:0x00000004 f:0x30002001 g:0x00000000 h:0x28006000 i:0x48024090 j:0x20000000 k:0x20000000	0	351
	Shift the LSB of the last configuration packet while exiting SHIFT-DR.	0	1	1
	Move into the SELECT-IR state.	X	1	3
	Move into the SHIFT-IR state.	X	0	2
8	Shift the first nine bits of the CFG_OUT instruction, LSB first.	111000100 (CFG_OUT)	0	9
	Shift the MSB of the CFG_OUT instruction while exiting Shift-IR.	1	1	1
	Move into the SELECT-DR state.	X	1	2
	Move into the SHIFT-DR state.	X	0	2
9	Shift the contents of the FDRO register out of the CFG_OUT data register.	…	0	number of readback bits – 1
	Shift the last bit of the FDRO register out of the CFG_OUT data register while exiting SHIFT-DR.	X	1	1
	Move into the Select-IR state.	X	1	3
	Move into the Shift-IR state.	X	0	2
10	End by placing the TAP controller in the TLR state.	X	1	3

Configuration Memory Read Procedure (1532 JTAG)

The IEEE 1532 JTAG readback procedure differs slightly from the IEEE 1149.1 JTAG readback procedure in that readback commands are not sent to the configuration logic through the CFG_IN JTAG register, rather the ISC_READ JTAG register is used to read configuration memory directly.

At the end of 1532 JTAG readback, the CRC Error status must be cleared by issuing a Reset CRC command or writing the correct CRC value to the CRC register. The 1532 JTAG readback procedure is illustrated in Figure 7-3.

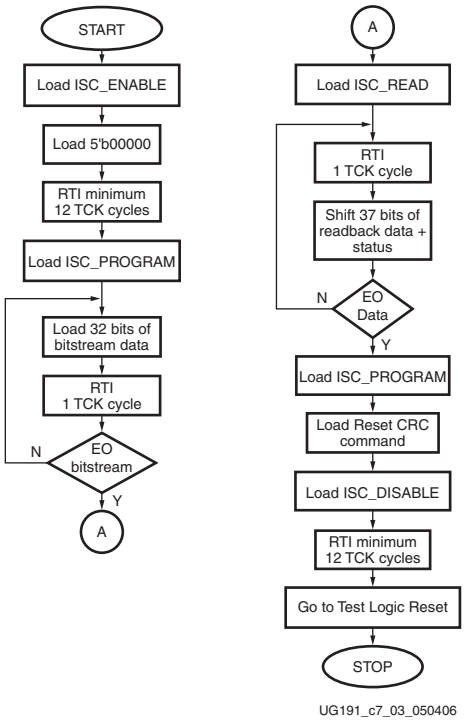


Figure 7-3: **IEEE 1532 JTAG Readback Flow**

Table 7-7 lists the readback files.

Table 7-7: **Readback Files**

File Extension	File Type	BitGen Setting	Description
RBA	ASCII	**-b** and **-g** Readback	An ASCII file that contains readback commands, rather than configuration commands, and expected readback data where the configuration data normally is. This file must be used with the MSK file
RBB	Binary	**-g** Readback	Binary version of the RBA file. This file must be used with the MSK file.
RBD	ASCII	**-g** Readback	An ASCII file that contains only expected readback data, including the initial pad frame. No commands are included. This file must be used with the MSD file.

Table 7-7: **Readback Files *(Continued)***

File Extension	File Type	BitGen Setting	Description
MSK	Binary	**-m**	A binary file that contains the same configuration commands as a BIT file, but replaces the contents of the FDRI write packet with mask data that indicate whether the corresponding bits in the BIT file should be compared. If a mask bit is 0, the corresponding bits in the readback data stream should be compared. If a mask bit is 1, the corresponding bit in the readback data stream should be ignored.
MSD	ASCII	**-g** readback	An ASCII file that contains only mask bits. The first bit in the MSD file corresponds to the first bit in the RBD file. Pad data in the actual readback stream are accounted for in the MSD and RBD files. If a mask bit is 0, that bit should be verified against the bit stream data. If a mask bit is 1, that bit should not be verified.
LL	ASCII	**-l**	An ASCII file that contains information on each of the nodes in the design that can be captured for readback. The file contains the absolute bit position in the readback stream, frame address, frame offset, logic resource used, and name of the component in the design.

The design.rba and design.rbb files combine readback commands with expected readback data and the RBD file contains only expected readback data. Systems that use an RBD file for readback must store readback commands elsewhere. The actual readback data must be masked against an MSK or MSD mask file, as certain bits in the expected readback stream in the RBA, RBB, and RBD files should be ignored.

The readback command set files do not indicate when users must change the SelectMAP or JTAG interface from write to read control; the user must handle this based on the Readback Command Sequences described above.

Verifying Readback Data

The readback data stream contains configuration frame data that are preceded by one frame of pad data, as described in the "Configuration Memory Read Procedure (SelectMAP)." The readback stream does not contain any of the commands or packet information found in the configuration bitstream and no CRC calculation is performed during readback. The readback data stream is shown in Figure 7-4.

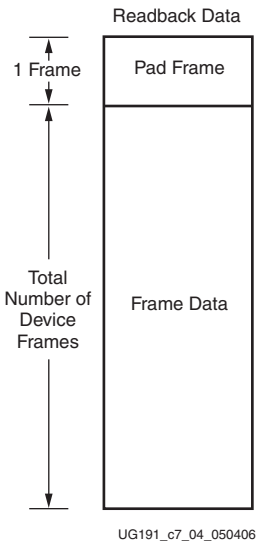


Figure 7-4: **Readback Data Stream**

The readback data stream is verified by comparing it to the original configuration frame data that were programmed into the device. Certain bits within the readback data stream must not be compared, because these can correspond to user memory or null memory locations. The location of *don't care* bits in the readback data stream is given by the mask files (`MSK` and `MSD`). These files have different formats although both convey essentially the same information. Once readback data have been obtained from the device, either of the following comparison procedures can be used:

1. Compare readback data to the `RBD` *golden* readback file. Mask by using the `MSD` file (see Figure 7-5).

   The simplest way to verify the readback data stream is to compare it to the `RBD` *golden* readback file, masking readback bits with the `MSD` file. This approach is simple because there is a 1:1 correspondence between the start of the readback data stream and the start of the `RBD` and `MSD` files, making the task of aligning readback, mask, and expected data easier.

   The `RBD` and `MSD` files contain an ASCII representation of the readback and mask data along with a file header that lists the file name, etc. This header information should be ignored or deleted. The ASCII 1s and 0s in the `RBD` and `MSD` files correspond to the binary readback data from the device. Take care to interpret these files as text, not binary sources. Users can convert the `RBD` and `MSD` files to a binary format using a script or text editor, to simplify the verify procedure for some systems and to reduce the size of the files by a factor of eight.

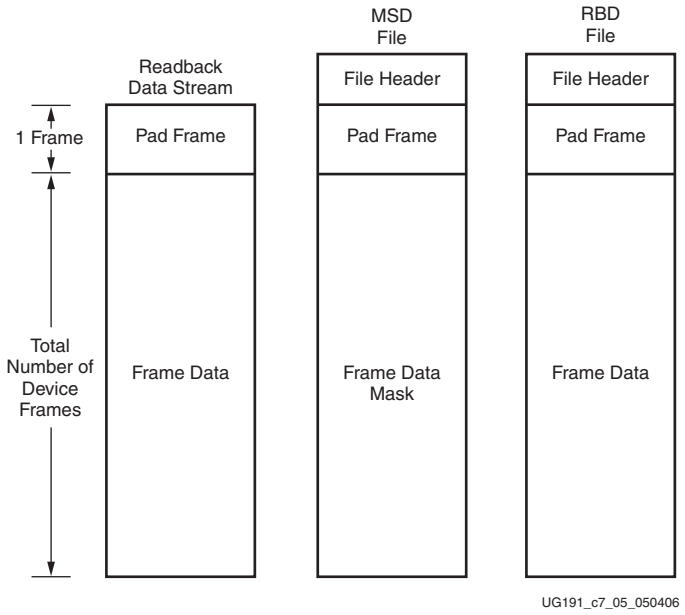


Figure 7-5: **Comparing Readback Data Using the MSD and RBD Files**

The drawback to this approach is that in addition to storing the initial configuration bitstream and the MSD file, the golden RBD file must be stored somewhere, increasing the overall storage requirement.

2. Compare readback data to the configuration BIT file, mask using the MSK file (see Figure 7-6).

   Another approach for verifying readback data is to compare the readback data stream to the frame data within the FDRI write in the original configuration bitstream, masking readback bits with the MSK file.

   After sending readback commands to the device, comparison begins by aligning the beginning of the readback frame data to the beginning of the FDRI write in the BIT and MSK files. The comparison ends when the end of the FDRI write is reached.

   This approach requires the least in-system storage space, because only the BIT, MSK, and readback commands must be stored.

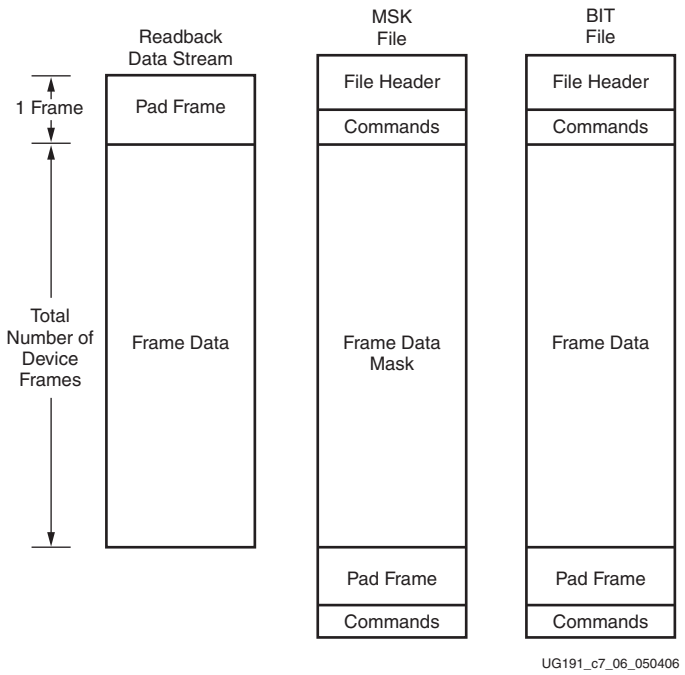


Figure 7-6: **Comparing Readback Data Using the MSK and BIT Files**

The RBA and RBB files contain expected readback data along with readback command sets. They are intended for use with the MSK file, although they are better suited to readback for Virtex devices (see Xilinx application note XAPP138) than for Virtex-5 devices.

Readback Capture

The configuration memory readback command sequence is identical for both Readback Verify and Readback Capture. However, the Capture sequence requires an additional step to sample internal register values.

Users can sample block RAM outputs, and CLB and IOB registers by instantiating the CAPTURE_VIRTEX5 primitive in their design (Figure 7-7) and asserting the CAP input on that primitive while the design is operating. On the next rising clock edge on the CAPTURE_VIRTEX5 CLK input, the internal GRDBK signal is asserted, storing all CLB and IOB register values into configuration memory cells. These values can then be read out of the device along with the IOB and CLB configuration columns by reading configuration memory through the readback process. Register values are stored in the same memory cell that programs the register's init state configuration, thus sending the GRESTORE command to the Virtex-5 configuration logic after the Capture sequence can cause registers to return to an unintended state.

Alternatively, the GRDBK signal can be asserted by writing the GCAPTURE command to the CMD register. This command asserts the GRDBK signal for two CCLK or TCK cycles, depending on the startup clock setting.

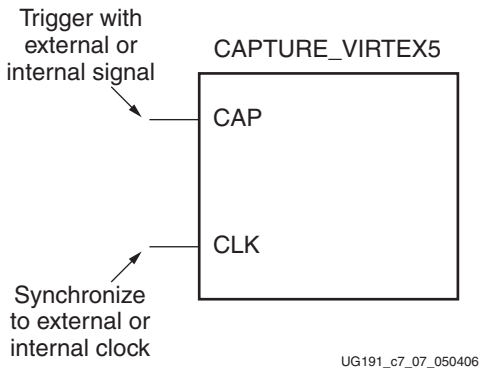


Figure 7-7: **Virtex-5 Device Library Primitive**

Table 7-8: **Capture Signals**

Signal	Description	Access
GCAPTURE	Captures the state of all slice and IOB registers. Complement of GRESTORE.	GCAPTURE command through the CMD register or *CAP* input on capture block, user controlled.
GRESTORE	Initializes all registers as configured.	CMD register and STARTUP_VIRTEX5 block.

If the CAP signal is left asserted over multiple clock cycles, the Capture cell is updated with the new register value on each rising clock edge. To limit the capture operation to the first rising clock edge, the user can add the ONESHOT attribute to the CAPTURE_VIRTEX5 primitive. More information on the ONESHOT attribute can be found in the Constraints Guide.

Once the configuration memory frames have been read out of the device, the user can pick the captured register values out of the readback data stream. The capture bit locations are given in the logic allocation file (`design.ll`) as described in Table 7-9.

Table 7-9 shows a snippet from a logic allocation file for the ISE *jc2_top* example design. The line from the header comments explaining the line format has been moved to the start of the bit offset data for clarity. The Offset field gives the absolute bit offset from the beginning of the readback frame data. The Frame Address field gives the frame address that the capture bit is located in, and the Frame Offset field gives the bit offset from the start of the frame. The Information field gives the mapping between the bit and the user design—for example, the *DIR* register (Table 7-9) that is located in Slice X8Y15 is located at bit offset `100790`. Captured DFF values are stored in their inverted sense.

Table 7-9: **Logic Allocation File Format**

Offset	Frame Address	Frame Offset	Information
100714	0x000e0400	42	Block=B7, Latch=I, Net=RIGHT_IBUF
100734	0x000e0400	62	Block=A8, Latch=I, Net=RIGHT_IBUF
100754	0x000e0400	82	Block=B8, Latch=I, Net=RIGHT_IBUF
100790	0x000e0400	118	Block=SLICE_X8Y15, Latch=YQ, Net=DIR
119038	0x00100400	62	Block=C8, Latch=I, Net=STOP_IBUF
119132	0x00100400	156	Block=SLICE_X11Y14, Latch=YQ, Net=RUN
136566	0x00120200	118	Block=SLICE_X12Y15, Latch=XQ, Net=Q_3
136606	0x00120200	158	Block=SLICE_X12Y14, Latch=XQ, Net=Q_1
137300	0x00120400	20	Block=C9, Latch=O2, Net=Q_3
137320	0x00120400	40	Block=D9, Latch=O2, Net=Q_1
137398	0x00120400	118	Block=SLICE_X12Y15, Latch=YQ, Net=Q_2
137438	0x00120400	158	Block=SLICE_X12Y14, Latch=YQ, Net=Q_0
155662	0x00140400	78	Block=D10, Latch=O2, Net=Q_1
174046	0x00160400	158	Block=D13, Latch=O2, Net=Q_2

XILINX®

Chapter 8

Reconfiguration and MultiBoot

This chapter focuses on full bitstream reconfiguration methods introduced in the Virtex®-5 family. The Virtex-5 family supports the industry's only partial reconfiguration solution. For additional details on partial reconfiguration, refer to Chapter 5 in the *Development System Reference Guide*:

http://toolbox.xilinx.com/docsan/xilinx8/books/docs/dev/dev.pdf

Fallback MultiBoot

Fallback Overview

Virtex-5 FPGAs have dedicated MultiBoot logic, which is used for both fallback and warm boot (IPROG) reconfiguration. When fallback or IPROG happens, an internally generated pulse resets the entire configuration logic, except for the dedicated MultiBoot logic and the WBSTAR and BOOTSTS registers. This reset pulse pulls INIT_B and DONE Low, and restarts the configuration process by clearing configuration memory. See "Clear Configuration Memory (Step 2, Initialization)," page 25.

During fallback reconfiguration, the FPGA drives new values on the two dual-mode pins RS[1:0] (Revision Select). RS[1:0] are 3-stated and weakly pulled up during the first configuration, and weakly pulled down after configuration by default. The user can use external pull-up/pull-down resistors to override the FPGA weak pull-up resistor during first configuration to load the desired bitstream (see Figure 8-2, page 154). When a configuration error is detected, the configuration logic generates an internal reset pulse and actively drives RS[1:0] to 00 for loading the fallback bitstream. Indirect BPI programming is supported with iMPACT version 11.3 and later when using the RS pin (see XAPP973, *Indirect Programming of BPI PROMs with Virtex-5 FPGAs*). One example of fallback reconfiguration is described in "Fallback Example."

During configuration, the following errors can trigger fallback: an IDCODE error, a CRC error, a Watchdog Timer time-out error, or a BPI address wraparound error. After configuration the Watchdog Timer, enabled in the user monitor mode, can also trigger fallback.

After successful fallback reconfiguration, the user design should readback the STATUS or BOOTSTS registers (see "Status Register for Fallback and IPROG Reconfiguration") to verify the fallback was successful.

If fallback reconfiguration fails a second time, configuration stops and both INIT_B and DONE are held Low.

Fallback is disabled if AES is enabled and for Slave SelectMAP mode. Fallback can also be disabled with BitGen option **`-g ConfigFallback:Disable`**.

Embedded IPROG (see "IPROG Embedded in the Bitstream") is ignored during fallback reconfiguration. The Watchdog (see "Watchdog") is disabled during and after fallback reconfiguration. A successful IPROG reconfiguration after fallback reconfiguration can re-enable the Watchdog.

Fallback Example

The simplest way to use the RS pins with a BPI Flash is to connect them to the Flash high address lines. In this example, the Flash address space is divided into four sections, and each can be used to store one application bitstream (see Figure 8-1). The fallback bitstream (safe bitstream) must be stored at the RS[1:0]=00 location. Refer to "MultiBoot Bitstream Spacing" for more information.

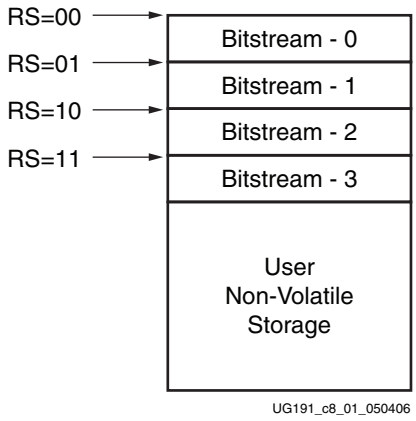


Figure 8-1: **BPI Flash Address Space for MultiBoot**

The initial bitstream can be selected using external resistors to override the internal weak pull-up resistor, as shown in Figure 8-2.

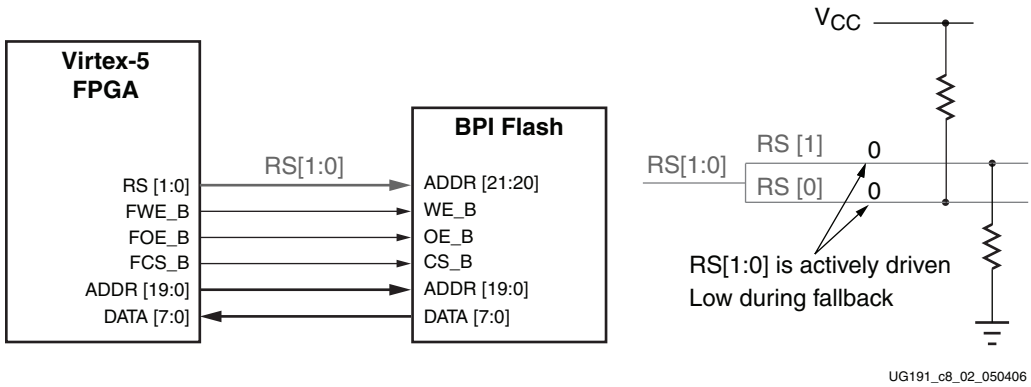


Figure 8-2: **Fallback Reconfiguration Usage for BPI**

Notes related to Figure 8-2:

- Initial configuration is triggered by PROGRAM_B, JPROGRAM instruction, or POR.
- RS[1:0] is 3-stated during initial configuration.
- This example selects the initial bitstream using RS[1:0] = 01, as determined by the pull-up and pull-down resistors.
- If initial configuration fails, RS[1:0] are actively driven Low:

- For BPI-Up mode, ADDR[25:0] starts from 0.
- For BPI-Down mode, ADDR[25:0] equals 0x3FFFFFF. ADDR[21:20] are driven Low by the RS pins, and the resulting address for the fallback bitstream is 0x000FFFFF.
- For SPI mode, the address sent over MOSI always starts from 0.

- Configuration stops if the fallback bitstream fails a second time.

MultiBoot Bitstream Spacing

The Virtex-5 FPGA keeps loading the bitstream until the DONE pin goes High. When DCI lock or DCM lock wait is enabled before the DONE cycle, padding (all 0s or all 1s) is required between bitstreams to compensate for the total lock time. Otherwise, the following bitstream can potentially overwrite the previous bitstream. The DCI lock time is typically less than 1 ms. For the DCM lock time, refer to the DCM section in DS202, *Virtex-5 Data Sheet: DC and Switching Characteristics*. The formula for the padding size is cfg_bus_width * total_lock_time / CCLK_period.

IPROG Reconfiguration

The IPROG (Internal PROGRAM_B) command has similar effect as a pulsing PROGRAM_B pin, except IPROG does not reset the dedicated reconfiguration logic. The start address set in WBSTAR ("Warm Boot Start Address Register (WBSTAR)," page 123) is used during reconfiguration instead of the default address. The default is zero in BPI-Up and SPI modes, and the default is 0x3FFFFFF in BPI-Down mode. The IPROG command can be sent through ICAP_VIRTEX5 or the bitstream. The "IPROG using ICAP_VIRTEX5" and "IPROG Embedded in the Bitstream" sections describe these two usages.

IPROG using ICAP_VIRTEX5

The IPROG command can also be sent using the ICAP_VIRTEX5 primitive. After a successful configuration, the user design determines the start address of the next bitstream, and sets the WBSTAR register, and then issues an IPROG command using ICAP.

The sequence of commands are:

1. Send the Sync word.
2. Program the WBSTAR register for the next bitstream start address (see "Warm Boot Start Address Register (WBSTAR)," page 123).
3. Send the IPROG command.

Table 8-1 shows an example bitstream for the IPROG command using ICAP.

Table 8-1: **Example Bitstream for IPROG through ICAP**

Configuration Data (hex)	Explanation
FFFFFFFF	Dummy Word
AA995566	Sync Word
20000000	Type 1 NO OP
30020001	Type 1 Write 1 Words to WBSTAR
00000000	Warm Boot Start Address (Load the Desired Address)

Table 8-1: **Example Bitstream for IPROG through ICAP *(Continued)***

Configuration Data (hex)	Explanation
30008001	Type 1 Write 1 Words to CMD
0000000F	IPROG Command
20000000	Type 1 NO OP

After the configuration logic receives the IPROG command, the FPGA resets everything except the dedicated reconfiguration logic, and the INIT_B and DONE pins go Low. After the FPGA clears all configuration memory, INIT_B goes High again. Then the value in WBSTAR is used for the bitstream starting address. The configuration mode determines which pins are controlled by WBSTAR.

Table 8-2: **WBSTAR Controlled Pins According to Configuration Mode**

Configuration Mode	Pins Controlled by WBSTAR
Master Serial	RS[1:0]
Master SPI	START_ADDR[23:0] are sent to the SPI device serially. The RS pins are set but not used.
Master BPI-Up	RS[1:0], ADDR[25:0]
Master BPI-Down	RS[1:0], ADDR[25:0]
Master SelectMAP	RS[1:0]
JTAG	RS[1:0]
Slave SelectMAP	RS[1:0]
Slave Serial	RS[1:0]

In all configuration modes, RS[1:0] is always controllable by WBSTAR. The START_ADDR field is only meaningful for the BPI-Up, BPI-Down, and SPI modes.

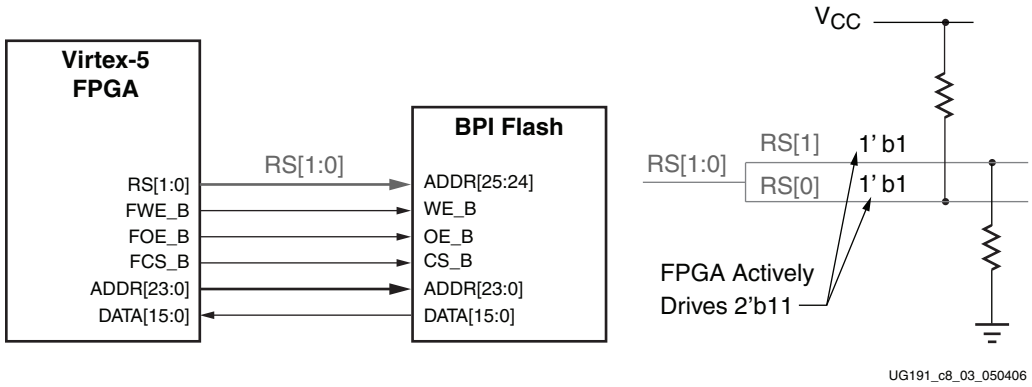


Figure 8-3: **IPROG in BPI Modes**

Notes relevant to Figure 8-3:

1. All BPI pins are dual mode I/Os. After configuration is DONE, these pins become user I/Os and can be controlled by user logic to access BPI Flash for user data storage and programming.

2. In this example, RS[1:0] is set to 2'b11. During IPROG reconfiguration, the RS[1:0] pins override the external pull-up and pull-down resistors. The user can specify any RS[1:0] value in the WBSTAR register.

IPROG Embedded in the Bitstream

WBSTAR and the IPROG command can be embedded inside a bitstream. A safe bitstream is stored at address 0 (in BPI-Up or SPI mode). Later a new application bitstream can be added to Flash, simply by modifying the WBSTAR and the IPROG command in first bitstream. The FPGA directly loads the new bitstream. If the new bitstream fails, configuration falls back to the original bitstream (see "Fallback MultiBoot"). ISE software inserts the blank write into WBSTAR and a place holder for the IPROG command in every Virtex-5 bitstream (see Table 6-15, page 127). For example, WBSTAR can be modified to a user-desired start address (see "Warm Boot Start Address Register (WBSTAR)," page 123). A NULL command after WBSTAR can be modified to IPROG by setting the four LSB bits to all ones (see "Command Register (CMD)," page 115).

Figure 8-4 illustrates this use model. Refer to "MultiBoot Bitstream Spacing" for more information.

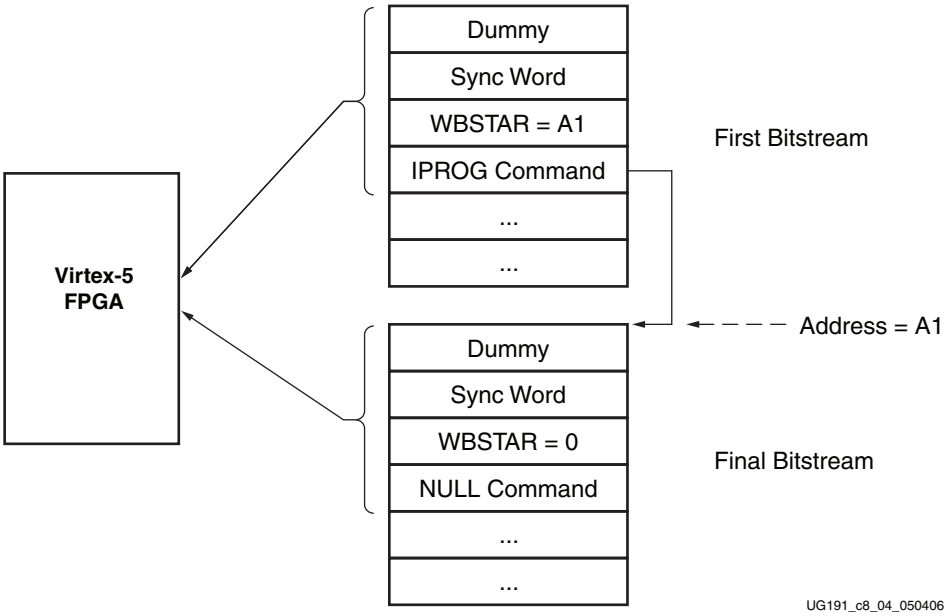


Figure 8-4: **IPROG Embedded in the Bitstream**

Status Register for Fallback and IPROG Reconfiguration

Virtex-5 devices contain a BOOTSTS that stores configuration history. BOOTSTS operates similar to a two-entry FIFO. The most recent configuration status is stored in Status_0, and the current value for Status_0 is shifted into Status_1. The Valid_0 bit indicates if the rest of Status_0 is valid or not. See "Boot History Status Register (BOOTSTS)," page 125.)

Table 8-3 through Table 8-5 show the BOOTSTS values in some common situations.

Table 8-3: **Status after First Bitstream Configuration without Error**

	RBCRC_ERROR	WRAP_ERROR	CRC_ERROR	ID_ERROR	WTO_ERROR	IPROG	FALLBACK	VALID
Status_1	0	0	0	0	0	0	0	0
Status_0	0	0	0	0	0	0	0	1

Table 8-4: **First Configuration followed by IPROG**

	RBCRC_ERROR	WRAP_ERROR	CRC_ERROR	ID_ERROR	WTO_ERROR	IPROG	FALLBACK	VALID
Status_1	0	0	0	0	0	0	0	1
Status_0	0	0	0	0	0	1	0	1

Table 8-5: **IPROG Embedded in First Bitstream, Second Bitstream CRC Error, Fallback Successfully**

	RBCRC_ERROR	WRAP_ERROR	CRC_ERROR	ID_ERROR	WTO_ERROR	IPROG	FALLBACK	VALID
Status_1	0	0	1	0	0	1	0	1
Status_0	0	0	0	0	0	1	1	1

Notes for Table 8-5:

1. Status_1 shows IPROG was attempted, and a CRC_ERROR was detected for that bitstream.
2. Status_0 shows a fallback bitstream was loaded successfully. The IPROG bit was also set in this case, because the fallback bitstream contains an IPROG command. Although the IPROG command is ignored during fallback, the status still records this occurrence.

Watchdog

The Virtex-5 Watchdog can be used to monitor configuration steps or user logic operation in the FPGA fabric. When the Watchdog times out, the configuration logic loads the fallback bitstream. The "Fallback MultiBoot" section provides more details.

The Watchdog uses a dedicated internal clock, CFG_MCLK, which has a nominal frequency of 50 MHz (±50%). The clock is predivided by 256, so that the Watchdog clock period is about 5120 ns. Given the watchdog counter is 24 bits wide, the maximum possible Watchdog value is about 86 seconds.

The Watchdog can be enabled in the bitstream or through any configuration port by writing to the TIMER register. The Watchdog is disabled during and after fallback reconfiguration. A successful IPROG reconfiguration initiated by a successful fallback reconfiguration is necessary to re-enable the Watchdog.

FPGA End of Startup

To use the Watchdog to monitor the bitstream configuration, set TIMER_CFG_MON to 1 and the desired TIMER_VALUE in a write to the TIMER register in the bitstream. The TIMER_VALUE should be adequate to cover the entire FPGA configuration time until startup is complete. Any wait time in startup for DCI match, DCM lock, or DONE should also be included.

Once enabled, the watchdog timer starts to count down. If the timer reaches 0 and the FPGA has not reached the final state of startup, a watchdog time-out error occurs and triggers a fallback configuration.

User Operation

To use the Watchdog to monitor the user logic, set TIMER_USR_MON to 1 and the desired TIMER_VALUE in a write to the TIMER register in the bitstream. The user must constantly reset the watchdog counter before it times out, either by the LTIMER command or by directly accessing the TIMER register. The watchdog is automatically disabled when the device is shut down or on power down (including shutdown).

Table 8-6 shows an example bitstream for reloading the Watchdog using the LTIMER command.

Table 8-6: **Example Bitstream for Reloading the Watchdog with LTIMER**

Configuration Data (hex)	Explanation
FFFFFFFF	Dummy Word
AA995566	Sync Word
20000000	Type 1 NO OP
30008001	Type 1 Write 1 Words to CMD
00000000	NULL
20000000	Type 1 NO OP
30008001	Type 1 Write 1 Words to CMD
00000011	LTIMER Command
20000000	Type 1 NO OP
30008001	Type 1 Write 1 Words to CMD
0000000D	DESYNCH
20000000	Type 1 NO OP

Table 8-7 shows an example bitstream for directly accessing the TIMER register:

Table 8-7: **Example Bitstream for Accessing the TIMER Register**

Configuration Data (hex)	Explanation
FFFFFFFF	Dummy Word
AA995566	Sync Word

Table 8-7: **Example Bitstream for Accessing the TIMER Register *(Continued)***

Configuration Data (hex)	Explanation
20000000	Type 1 NO OP
30022001	Type 1 write 1 words to TIMER
00000000	TIMER value
20000000	Type 1 NO OP
30008001	Type 1 write 1 words to CMD
0000000D	DESYNCH
20000000	Type 1 NO OP

XILINX®

Chapter 9

Readback CRC

Virtex®-5 devices include a new feature to do continuous readback of configuration data in the background of a user design. This feature is aimed at simplifying detection of Single Event Upsets (SEUs) that cause a configuration memory bit to flip and can be used in conjunction with the FRAME ECC feature for advanced operations such as SEU corrections. To enable Readback CRC, the CONFIG user constraint POST_CRC is set to **Enable**. Once enabled, the configuration dedicated logic reads back continuously in the background to check the CRC of the configuration memory content. The first round of readback CRC value is latched as the golden value for later comparison. The subsequent rounds of readback CRC value are compared against the golden value. When a CRC mismatch is found, the crc_error pin of the FRAME_ECC_VIRTEX5 primitive is driven High, the INIT_B pin is driven Low, and the DONE pin remains High. If a CRC error is detected, the device must be reconfigured. The CONFIG user constraint POST_CRC_SIGNAL can be optionally set to FRAME_ECC_ONLY to turn off INIT_B as the readback CRC flag. The error flag remains asserted until the next comparison if the error was not corrected. If no new golden CRC was calculated, then the error flag remains asserted. When the user logic accesses the configuration logic through an ICAP command, JTAG, or a SelectMAP **`persist`**, the readback CRC error automatically stops without affecting the user configuration access, and the error flag is cleared. When the user finishes accessing the configuration logic, readback CRC automatically resumes.

Readback CRC logic runs under these conditions:

- Any configuration operation must finish with a DESYNC command to release the configuration logic. If a DESYNC command is not issued, the readback CRC logic cannot access the configuration logic and cannot run. The DESYNC command resets the readback CRC circuit and clears all error conditions and flags.
- In addition, the JTAG instruction register (IR) must not contain any configuration instructions (CFG_IN, CFG_OUT, or ISC_ENABLE). When these instructions are present, at any time, the readback CRC logic can not access the configuration logic and cannot run. Any configuration operation performed via the JTAG interface should finish by loading the IR with a value other than these three configuration instructions.

The following dynamically changeable memory locations are masked during background readback:

- MLUT (RAM or SRL)
- Block RAM content is skipped during readback to avoid interfering with user functions. Block RAM is covered by its own ECC circuit during operation.
- Dynamic Reconfigure Port (DRP) memories are masked.

When enabled, the readback CRC logic automatically runs in the background after configuration is DONE, and when the following conditions hold:

- The FPGA is started up successfully, as indicated by the DONE pin going High.
- The configuration interface has been parked correctly. A normal bitstream has a DESYNC command at the end that signals to the configuration interface that it is no longer being used.
- If the JTAG interface is in use, the JTAG instruction register must not be set to CFG_IN, CFG_OUT, or ISC_ENABLE.

Readback CRC runs on different clock sources in different modes as indicated in Table 9-1.

Table 9-1: **Readback CRC Clock Sources**

ICAP Primitive	STARTUP Primitive	Master Modes	Slave Modes	JTAG Mode	Clock Source
Instantiated	x	x	x	x	CLK input of the ICAP primitive
Not Instantiated	Instantiated	x	x	x	USRCCLKO input of the STARTUP primitive
Not Instantiated	Not Instantiated	Yes	No	No	Master CCLK controlled by the BitGen option **`-g ConfigRate`**
Not Instantiated	Not Instantiated	No	Yes	No	CCLK pin input
Not Instantiated	Not Instantiated	No	No	Yes	No clock (see paragraph below this table).

Because JTAG has the highest priority in the configuration mode, it takes over the configuration bus whenever it needs to. M[2:0] are recommended to be set to Master Serial mode when only JTAG configuration is intended, so that the internal oscillator provides a continuous clock. The JTAG Instruction Register must not be parked at the CFG_IN, CFG_OUT, or ISC_ENABLE instructions.

In a partial reconfiguration application, the configuration memory content changes, so the golden signature must be recalculated. This is automatically done when a partial reconfiguration is performed using the shutdown sequence, where AGHIGH and DGHIGH commands are used. For partial reconfiguration without the shutdown sequence, the CRCC command must be sent at the end of bitstream to trigger recalculation of the golden signature.

Post_CRC Constraints

There are two Virtex-5 constraints used for signalling Single Event Upset (SEU) events. Both constraints have the same propagation rule. They are placed as a CONFIG constraint and then propagated to the physical design object.

POST_CRC

POST_CRC uses the FRAME_ECC_VIRTEX5 primitive's CRC_ERROR pin for signalling SEU events. In addition, INIT can be reserved as an SEU CRC error indicator by using the POST_CRC_SIGNAL constraint. During configuration, the INIT pin operates normally. After configuration, if SEU analysis is enabled, and INIT is reserved, the INIT pin (default) serves as an SEU status pin. An SEU is detected when a comparison of the real-time computed CRC differs from the pre-computed CRC, the CRC_ERROR pin is driven High and the INIT pin is driven Low.

The POST_CRC constraint is the best way to convey this information. It attaches to the CONFIG constraint. POST_CRC can be used by PACE, PAR, and BitGen to reserve the INIT pin by not programming the IOB to drive the INIT pin.

POST_CRC can take two values:

- ENABLE
  SEU detection is enabled.
- DISABLE
  SEU detection is disabled.

POST_CRC_SIGNAL

POST_CRC_SIGNAL determines whether the Virtex-5 INIT_B pin is a source of the SEU error signal. Virtex-5 devices support the POST_CRC SEU detection mode.

The readback CRC feature compares a pre-computed CRC on the configuration bitstream or post-computed CRC against a CRC computed by internal logic based on periodic readback of the configuration memory cells. If a bit flips in the configuration memory cells, then an SEU is detected. Single bit flips are typically caused by background radiation.

Use POST_CRC to disable the INIT_B pin as the readback CRC error status output pin. The error condition is still available from the FRAME_ECC_VIRTEX5 site. The best way to convey this information is to create the POST_CRC_SIGNAL constraint, which attaches to the CONFIG constraint. POST_CRC_SIGNAL can be used by BitGen to set a single bit in the COR1 register.

POST_CRC_SIGNAL can take two values:

- FRAME_ECC_ONLY
  Disable the use of the INIT_B pin, with the FRAME_ECC site as the sole source of the SEU error signal
- INIT_AND_FRAME_ECC
  Leave the INIT_B pin enabled as a source of the SEU error signal

Syntax Examples

This section lists the supported syntax examples for each constraint.

POST_CRC

NCF Syntax Example

```
CONFIG POST_CRC = [ENABLE|DISABLE]
```

UCF Syntax Example

```
CONFIG POST_CRC = [ENABLE|DISABLE]
```

PCF Syntax Example

```
CONFIG POST_CRC = [ENABLE|DISABLE]
```

POST_CRC_SIGNAL

NCF Syntax Example

```
CONFIG POST_CRC_SIGNAL = [FRAME_ECC_ONLY|INIT_AND_FRAME_ECC]
```

UCF Syntax Example

```
CONFIG POST_CRC_SIGNAL = [FRAME_ECC_ONLY|INIT_AND_FRAME_ECC]
```

PCF Syntax Example

```
CONFIG POST_CRC_SIGNAL = [FRAME_ECC_ONLY|INIT_AND_FRAME_ECC]
```

Index

R

S

T

U

V

W