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

XAPP973 (v1.4) March 8, 2010

Indirect Programming of BPI PROMs with Virtex-5 FPGAs

Author: Stephanie Tapp

Summary

Virtex®-5 FPGAs and ISE® software support configuration from and programming of industry-standard, parallel NOR flash memory (BPI PROMs). Industry standard BPI PROMs are an alternate solution for Virtex-5 FPGA designs whose requirements are not met by the Platform Flash XL configuration and storage device (see [Ref 1] for more information on Platform Flash XL). The iMPACT software, included in the ISE® development software tools, provides indirect programming for select BPI PROMs during prototyping. This application note demonstrates how to program a Numonyx StrataFlash P30 BPI PROM indirectly using iMPACT 11.4 and a Xilinx cable. In this solution, the Virtex-5 FPGA serves as a bridge between the IEEE Std 1149.1 (JTAG) bus interface and the BPI bus interface. The required hardware setup, BPI-UP PROM file generation flow, and BPI indirect programming flow are shown. The Virtex-5 FPGA BPI-UP configuration sequence is also described.

Note: Parallel NOR flash memory is referred to by the term BPI PROM throughout this document.

Introduction

Xilinx FPGAs are CMOS configurable latch (CCL) based and must be configured at power-up from a non-volatile source. FPGA configuration is traditionally accomplished with a JTAG interface, a microprocessor, or the Xilinx PROMs (Platform Flash PROMs). In systems where the easiest solution is preferred, Master Serial mode with a Xilinx Platform Flash PROM is still the most popular configuration mode because it has:

- A direct JTAG interface for programming
- The smallest interface pin requirement for configuration
- Flexible I/O voltage support

Moreover, this solution is available for any Virtex-5 FPGA device (refer to [Ref 2] for more information).

In addition to the traditional methods, a direct configuration interface to third-party BPI PROMs is included on Virtex-5 FPGAs to address changing system requirements. Systems with a BPI PROM already onboard for random-access, non-volatile application data storage can benefit from consolidating the configuration storage into the same memory device.

Similar to the traditional configuration memories, BPI PROMs must be loaded with the configuration data. BPI PROMs have a single interface for programming, and three primary methods to deliver the data to this interface:

- Third-party programmers (off-board programming)
- In-system programming (ISP) with an embedded processor
- Indirect ISP (using JTAG or custom solution)

Production programming is often accomplished off-board with a third-party programmer or in-system with a JTAG tool vendor. During the prototyping phase, indirect ISP is preferred to easily accommodate design iterations.

Because BPI PROMs do not have a JTAG interface, extra logic is required to serve as a bridge between the iMPACT programmer (using a cable to drive the JTAG bus interface), and the BPI

PROM (connected to the FPGA's BPI bus interface). This extra logic must be downloaded into the FPGA by iMPACT before indirect programming is possible.

This application note is divided into three main sections. The first section discusses the hardware connections required for the indirect in-system programming of BPI PROMs for prototype designs. The second section shows the Xilinx software tool flows for generating a PROM file formatted for 16-bit BPI-UP mode and then for programming the select BPI PROMs. The third section provides a basic configuration flow overview for the FPGA after the BPI PROM is programmed and describes expectations when using this indirect setup.

iMPACT Indirect In-System Programming with a Virtex-5 FPGA

The basic hardware setup required for the iMPACT indirect BPI PROM programming method is shown in Figure 1.

Xilinx provides a proprietary IP module that must be downloaded into the FPGA. This IP converts JTAG instructions into the command/control signals needed to program the flash memory.
After downloading the configuration bitstream, the FPGA reads to flash to configure itself.

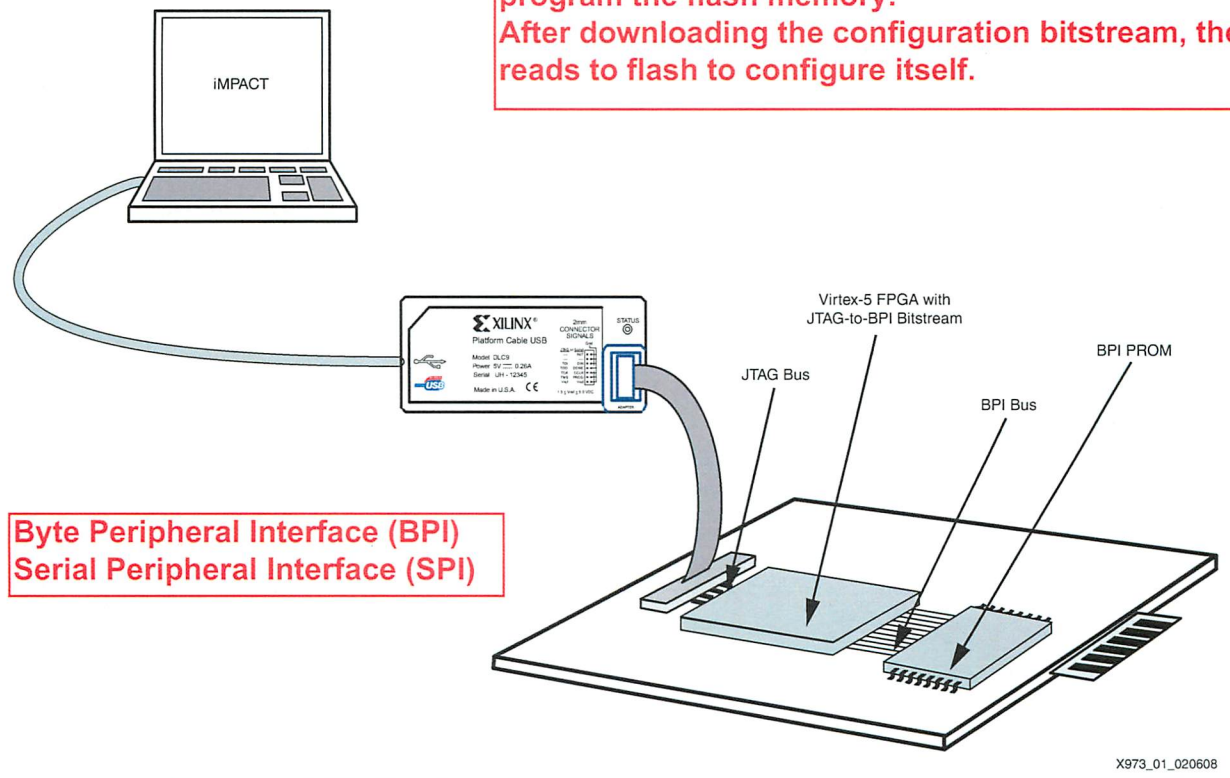


Figure 1: **iMPACT Indirect BPI PROM Programming with a Virtex-5 FPGA**

Minimum Requirements

- Virtex-5 FPGA
- BPI PROM (refer to Table 1)
- Xilinx Cable and Connector (refer to Table 4, page 7)
- ISE iMPACT Software 11.4

Note: Indirect BPI PROM programming was introduced with limited device support in iMPACT 9.2i. This application note demonstrates the software flow and lists the device support in iMPACT 11.4.

JTAG programming software, such as iMPACT by Xilinx, uses sequences of JTAG instructions to perform specific programming and verification operations selected by the user.

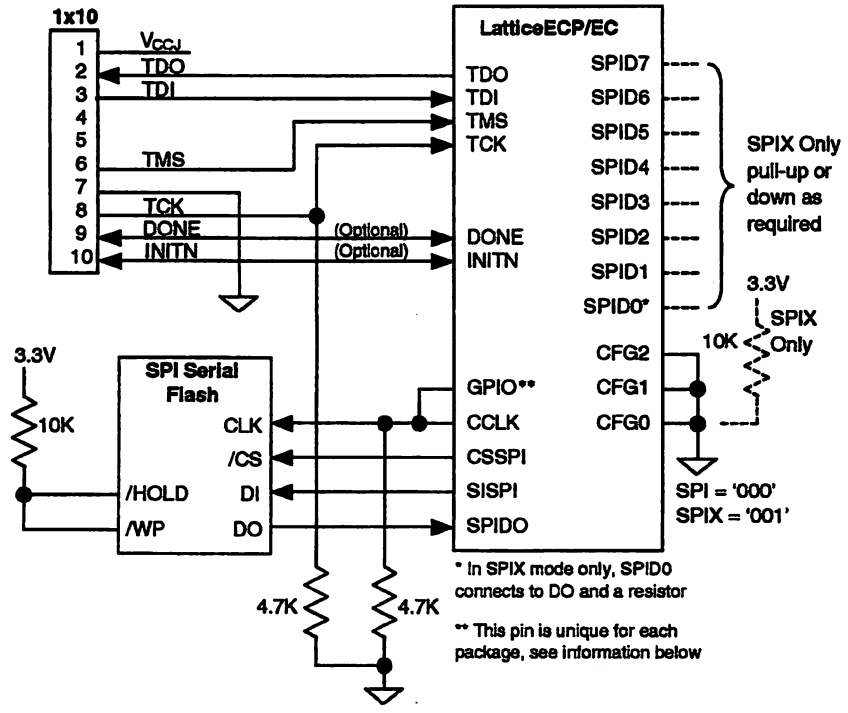
However, these software tools do *not* support execution of individual JTAG instructions. In other words, the user can't write custom sequences of JTAG instructions.

Application Note: TN-1078

Schematic

The schematic in Figure 20-2 illustrates how to wire the ispJTAG connector, FPGA, and SPI Serial Flash.

Figure 20-2. Hardware Schematic



- The download header has standard 0.1 inch pin-to-pin spacing.
- The 4.7K pull-down resistors prevent spurious clock pulses during V_{CC} ramp-up. Place the resistors close to their clock line to keep the stub length as short as possible.
- The CCLK frequency can be as high as 50MHz, so keep this trace fairly short.
- V_{CCIO} for the bank that drives the signals to the SPI Serial Flash must match the SPI Serial Flash V_{CC} (today that voltage is 3.3V but this will change over time as Flash chip geometries decrease). For all packages these signals are located in bank 3.
- During configuration CCLK drives the SPI Serial Flash CLK pin, but once the FPGA completes configuration CCLK goes into tri-state. The CCLK pin is not accessible by user code so CCLK needs to be wired to a nearby General Purpose I/O pin (GPIO) to allow the FPGA fabric to supply a clock to the SPI Serial Flash. This pin is part of the Soft SPI Interface and is unique to each package. If the user embeds the Soft SPI Interface into their code then this pin, along with the other pins wired to the SPI Serial Flash, must be locked using the Pre-Map Preference Editor in ispLEVER. A complete list of these pins is found in Table 20-5.
- In addition to standard decoupling practices, place a decoupling capacitor close to the connector's V_{CCJ} pin. Any standard ceramic capacitor value may be used, for example 0.1 µF, 0.01 µF, etc.

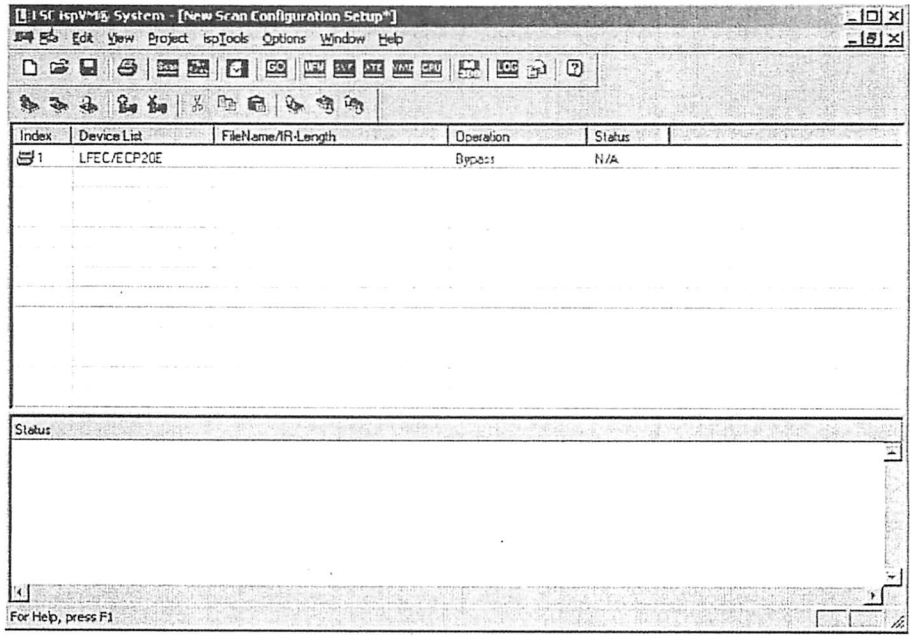
Programming Procedure

In order to program SPI Serial Flash via ispJTAG the FPGA must contain the Soft SPI Interface. Programming the SPI Serial Flash with ispVM System and an ispDOWNLOAD cable makes this transparent to the user. The software simply loads a default Soft SPI Interface bitstream into the FPGA and then loads the user bitstream into the Flash. Once programing of the SPI Serial Flash is complete the FPGA configures itself by reading the Flash. Again, software makes all of this transparent to the user so that it is no different than programming any other serial boot device.

The following instructions describe the process of selecting the FPGA, selecting the SPI Serial Flash, and programming the SPI Serial Flash. The following screen shots are from ispVM System 15.0.

1. Connect the ispDOWNLOAD Cable to the appropriate header and apply power to the board.
2. Start the ispVM System software.
3. From the main window click on the **Scan** button located on the toolbar. The LatticeECP/EC device should be detected automatically (if it's not detected, check the ispJTAG connections and make sure the board is powered up). The resulting screen should be similar to Figure 20-3.

Figure 20-3. Main Window, Scan Complete



4. Double-click the number in the **Index** column and select the appropriate device to open the Device Information window, shown in Figure 20-4.