Data Generation and Configuration for Spartan Series FPGAs

XAPP 126 June 14, 1999 (Version 1.1)

Application Note by Ashok Chotai

Summary

This application note describes various methods to configure Spartan series FPGAs. Each configuration method is described in detail. Information on necessary software programs to run with input files required, output files produced, download cables used, and other hardware necessary to accomplish the task is discussed. This application note targets users who are new to Xilinx devices and Alliance/Foundation series software tools, and is intended to make the configuration and debugging flows easy to understand.

Xilinx Families

Spartan and SpartanXL

Introduction

A design is entered as a schematic or high level HDL description and then implemented (translated, partitioned, placed and routed) into a Spartan series FPGA using Xilinx software tools. As a result, a bitstream is produced which is used to configure the FPGA. Xilinx FPGAs are built with static RAM and hence they need a data source such as a PROM or other intelligent interface to load the configuration data upon power-up.

Configuration is the process of loading the bitstream into one or more FPGAs to define the functional operation of the internal blocks (function look-up table, flip-flop, multiplexer, buffers, pull up/down, slew rate etc.) and their interconnections (pass transistors). Each configuration bit for the FPGA defines the state of a static memory cell that controls either an internal block or its interconnections. The device can be configured through standard configuration pins or through the boundary scan (JTAG) interface.

Xilinx Alliance/Foundation series software programs, the files required by these programs, the download cables used, and other hardware necessary to accomplish the configuration and debugging task are discussed in the following section.

Software Programs

There are five different software programs that may be required to run the configuration flow. They are: BitGen, PROMGen, JTAG Programmer, Hardware Debugger and the HW-130 Programmer software. The first four programs are part of the Alliance/Foundation series software and the last program is a part of the HW-130 Programmer. The user may not need to run all of these programs. These programs are described briefly in this section.

BitGen (Configure - Flow Engine)

BitGen is a program that takes a mapped, placed and fully routed NCD design file as its input and produces a configuration bitstream - a BIT file with a .bit extension (see Figure 1). The program can also produce an LL file which is used for debugging the device. The program is a part of the implementation process. Additionally, the BitGen program has the following options:

Configuration Options: configuration clock rate, RBT and LL file generation and Cyclic Redundancy Check (CRC) error check.

I/O Programmability: TTL or CMOS thresholds for the inputs and outputs (Spartan family only), pull-up and pull-down control for the dedicated configuration pins (TDO, DONE, M0 and M1), Power Down Control (Spartan XL family only) and 5V Tolerant I/Os (SpartanXL family only).

Start-up Timing Options: User clock or configuration clock selection, control of the sequence of output events (DONE going High, user I/Os going active, and release of global Set/Reset).

Readback Options: User clock or configuration clock selection, bitstream verification and in-circuit debugging control.

Boundary Scan Options: Control of boundary scan based configuration and bitstream readback.

Other Options: Tie unused internal nodes.

The program can be run as a stand-alone tool from the UNIX/DOS command prompt or by running the **Configure** step in the Flow Engine of the Xilinx Design Manager Graphical User Interface. For information on the command-line mode usage and options, refer to the **BitGen** chapter of the Development System Reference Guide. For instructions on graphical mode usage and options, refer to the **Implementation Options** chapter of the Design Manager/Flow Engine Reference/User Guide.

PROMGen (PROM File Formatter)

PROMGen formats single or multiple BIT files into a PROM file or a straight HEX file (see Figure 1). The PROM file is used to program the PROM. The HEX file or the PROM file

is used by the microprocessor or other intelligent interface to configure the FPGA. PROMGen needs to be run in the following situations:

- The FPGA is configured using a serial or parallel PROM
- The FPGA is configured using a microprocessor
- To concatenate bitstream files for multiple devices, when they are to be configured in a daisy chain
- To split an existing PROM file into multiple PROM files

PROMGen can be run as a stand-alone program from the UNIX/DOS command prompt or by running the PROM File Formatter tool in the Xilinx Design Manager Graphical User Interface (GUI). For instructions on the command-line mode usage and options, refer to the PROMGen chapter in the Development System Reference Guide. For information on the graphical mode usage, refer to the PROM File Formatter Reference/User Guide.

JTAG Programmer

The JTAG Programmer software is used to load the configuration data into Xilinx FPGAs through boundary scan (JTAG) pins and perform functional (boundary scan) tests on a single FPGA or a daisy chain of FPGAs.

Hardware Debugger

The Hardware Debugger software is used to configure and debug a single FPGA, or configure a daisy chain of FPGAs. For a single device configuration, the software uses the BIT file, RBT file or a PROM file (MCS, TEK, EXO). For multiple devices, it uses only the PROM file. For debugging, which is available only for a single device, the software uses BIT and LL files. Also note that the software does not support configuration through the boundary scan (JTAG) interface.

HW-130 Programmer Software

As the name implies, the HW-130 Programmer software comes with the HW-130 Programmer. The program is used to load the configuration data into Xilinx serial PROMs.

Files for Configuration

There are various files that are used in the configuration flow. This section describes each file and its importance in the configuration flow.

The BIT file (configuration bitstream) is a binary representation of the logic design. The file is used to configure the Xilinx device. It can also be used to create a PROM file. Note that the bitstream file size is a constant for any given device, regardless of the amount of logic in it. Every bit in the device gets programmed, whether it is used or not. Also, note that the same bitstream is used for Master/Slave Serial Mode and the JTAG mode. However, the Express mode bitstream has a different size and hence it cannot be used for any other configuration mode. The Express Mode is available only in the SpartanXL family. The bitstream size for all the devices is listed in the **Spartan Program Data** table (Table 17) in the Spartan/XL data sheet.

The RBT (raw bit) file is an ASCII representation of the BIT file. The main benefit of using the RBT file instead of the BIT file is that this file can be viewed using a common text editor. Also, the order in which the 1's and 0's appear in this file is actually what shows up on the DOUT pin of the FPGA during configuration. Monitoring the DOUT pin during the configuration process helps in identifying whether the bitstream gets loaded into the correct device.

The LL (logic allocation) file contains the position information of CLB flip flops, latches and the IOBs. The file is used if the user wants to perform an additional step of performing the readback operation on the device, which is essentially reading the internal states of the configured device.

The HEX file has each hexadecimal digit representing four consecutive configuration bits in a BIT file. The file is typically used by the microprocessor based configuration.

The PROM file is a formatted BIT file that can be used to program the PROM. The file can be MCS, TEK or EXO format.

The BSD file is the Boundary scan description file. The file contains the description of test pins and test instructions. It is used for boundary scan based configuration and testing. A BSD file is required for every Xilinx and non-Xilinx device in the chain.

Programming Equipment

This section describes the hardware equipment that is required to accomplish the device configuration. It assumes that the user already has other equipment such as the system board with Spartan/XL device on it, the cables necessary for probing and connecting the power source and the 3.3/5V DC power supply.

Xilinx Download Cables

The XChecker cable is an RS232 serial cable which connects to the serial port of both the PC and the workstation. The cable supports configuration and debugging of FPGAs. The Parallel Download Cable connects to the parallel port of the PC. This cable supports only the configuration of FPGAs. It has a higher drive capability and is much faster than the XChecker cable. For more information on these cables, refer to the Hardware User Guide.

Xilinx HW-130/3rd Party Programmer

HW-130 Programmer is used to program the Spartan/XL series serial PROMs. The programmer supports MCS, EXO and TEK file formats. For more information, refer to the HW-130 Programmer Data sheet or the On-Line help which is available in the software.

The user can optionally use a 3rd party PROM programmer. For a complete list of the programmers supporting Xil-

inx FPGAs, refer to the **Programmer Solutions Technical Tips** on the Xilinx web site.

Configuration Modes

This section describes the various configuration modes in which the FPGA can be connected.

The Spartan series can be configured through standard configuration pins or through the boundary scan interface. Using the standard configuration pins, the Spartan family has two modes and the SpartanXL family has three modes. The two modes common to both the families are Master Serial Mode and Slave Serial Mode. These modes are selected using the mode pin(s) on the device. In the Master Serial mode, the FPGA generates its own configuration clock (CCLK) to load the configuration data from an external memory source such as Xilinx serial PROM. In the Slave Serial mode, an external signal drives the CCLK input of the FPGA to load the configuration data from an external device such as a lead FPGA or an intelligent interface such as a microprocessor. The slave serial mode is described in the application note *XAPP098: The Low-Cost Efficient Serial Configuration of Spartan FPGAs.*

The SpartanXL family has an additional mode called Express Mode. It has the same timing as Slave Serial Mode, except that it is 8 times faster as the data is processed one byte instead of one bit per CCLK cycle. The Express Mode is described in the application note *XAPP122: The Express Configuration of SpartanXL FPGAs*. The bitstream file for this mode is created by the following command from the command prompt:

bitgen -g ExpressMode:Enable -g CRC:Disable -b infile outfile

Since the Express Mode does not support error detection using CRCs, it is necessary to disable this feature with the text "-g CRC: Disable". For more information on configuration modes, refer to the **Configuration and Test** section in the Spartan/XL data sheet.

Configuration through the boundary scan interface is discussed in detail in the section "JTAG Flow" on page 6 of this application note.

Configuration Flows

Xilinx FPGAs can be configured in several ways. The various possible flows are: Debug flow, Any_File flow, JTAG flow, Processor flow and PROM flow. The Debug flow is the only flow which has additional capability of reading the internal states of the configured device. The Debug, Any_File and the JTAG flows need a Xilinx cable. Any of these flows can be used based upon the requirements. These flows are summarized in Table 1 and shown graphically in Figure 1. To encounter problems during configuration during any of these flows, refer to **The Configuration Problem Solver** located on the Xilinx web site.

Table 1: Configuration Flows Overview

Flow Name	Designer Goal	Program Equipment[1]	Software Required[2]	Files Needed
Debug	Prototype for a single device	• XChecker Cable	• Flow Engine • Hardware Debugger	BIT and LL
Any_File	Prototype for a single device	• XChecker/Parallel Cable	• Flow Engine • Hardware Debugger	BIT/RBT/PROM file
JTAG	Production/Prototype for single device OR multiple devices in daisy chain using Boundary Scan interface[3]	• XChecker/Parallel Cable	• Flow Engine • JTAG Programmer	BIT and BSDL for each device
PROM	Prototype/Production stage for single device OR multiple devices in daisy chain	• PROM • HW-130/3rd Party PROM Programmer	• Flow Engine • PROM File Formatter • HW-130/3rd Party Programmer software	PROM file
Processor	Prototype/Production stage for single device OR multiple devices in daisy chain using Micro-processor	• Microprocessor with memory	• Flow Engine • PROM File Formatter • Microprocessor firmware[4]	HEX/PROM file

1. The target board which includes the Spartan/XL device(s) must be already present.
2. All the software programs are part of Xilinx Alliance/Foundation series software tools, unless otherwise stated.
3. Boundary-scan (BSCAN) symbol must be instantiated in the design and you must have a separate BIT file for each FPGA in the chain.
4. Not supplied by Xilinx.

Debug Flow

Debugging involves Readback Verify and Readback Capture. Readback Verify consists of reading the programming data that was sent to the device and comparing it against the original bitstream data to ensure that the device is loaded with the correct data. Readback Capture refers to

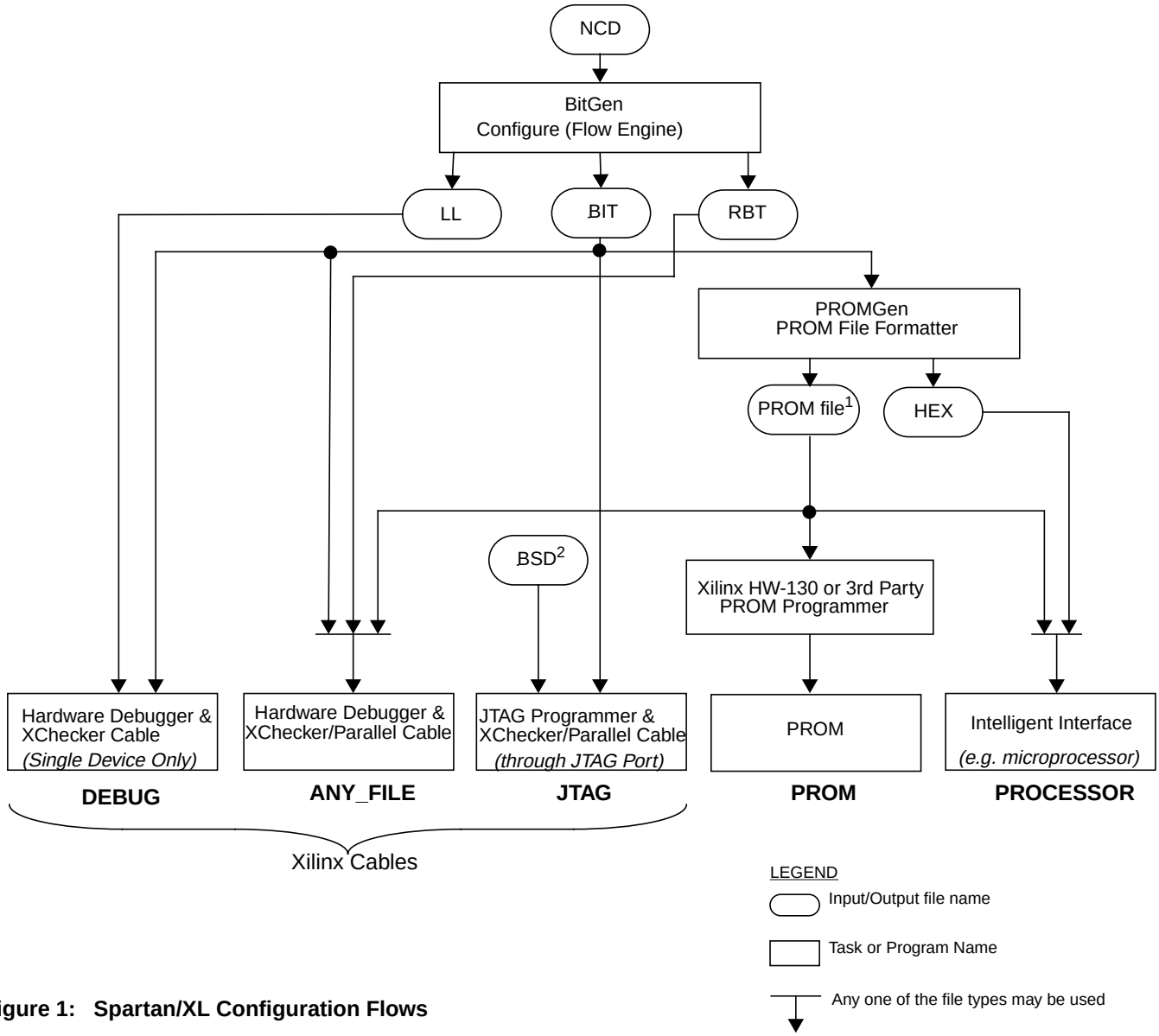

Figure 1: Spartan/XL Configuration Flows

NOTE:

1. The PROM file can be MCS, TEK or EXO file
2. Call the Xilinx hotline to get the BSD file for the Spartan/XL device

reading the internal states (flip flop outputs, CLB outputs, IOB outputs and the RAM/ROM bits) of the device to make sure the design performs correctly. Note that debug process does not interfere with the normal operation of the device. To get familiar with the configuration and debugging flow, use the Watch tutorial on the Xilinx web site.

The Debug Flow is useful when the designer wants to configure and debug the prototype of the design. The Hardware Debugger software and the XChecker cable are used for this task. This method neither requires external storage for configuration data nor an external clock to synchronize the configuration process. The XChecker cable itself contains static RAM and an internal oscillator circuit for clock generation. The FPGA is configured in Slave Serial mode. For more information on this mode, refer to the **Slave Serial Mode** description under the **Configuration and Test** section in the Spartan/XL data sheet. The following steps are required to perform the task:

1. *Instantiate the READBACK symbol in your schematics or HDL based design (Figure 2).*

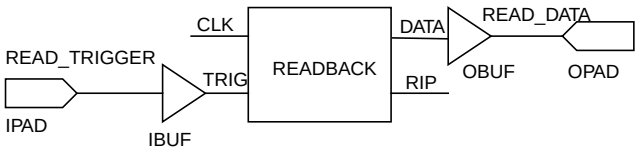

Figure 2: Example of Readback Schematic

For more information on the READBACK symbol and its description, refer to the **Configuration and Test** section in the Spartan/XL data sheet.

2. *Create EDIF netlist for your design.*

3. *Implement the design to get the LL and BIT files.*

Set the Readback Clock to CCLK and check the field for "Enable Bitstream Verification and In-Circuit Hardware Debugging" in the Configuration Options window to use the internal clock in the FPGA to synchronize the readback data and create an LL file for readback. The Hardware Debugger uses this file for the readback operation. Implement the design using Xilinx Foundation/Alliance implementation tools to create the bitstream. For more information on implementing the design, refer to **Implementation Options** chapter in the Design Manager/Flow Engine Reference/User Guide.

4. *XChecker Cable Connections*

Make sure the mode pin(s) are set for slave serial mode. Although these pins have weak pull-up resistor during configuration, it is recommended to attach an external pull-up of 4.7 KΩ to make sure these pins are not floating. Connect the cable connectors 1 and 2 to the target system as shown in Figure 3. This setup is used to perform synchronous debugging where the cable is used to control the external system clock and the logic states of the design. The XChecker cable needs 5V DC power which is drawn from the target system. The power supply required is 5V DC for the system with Spartan device(s) and 3.3V DC for the system that contains SpartanXL device(s). Since the XChecker cable must be powered only with a 5V supply, a separate 5V supply may be used for the SpartanXL system or an 3.3V XChecker adapter (Order Number: HW-XCH3V) may be used that accepts 2.9-5.25V DC as input and provides the output stepped up to 5V needed by the cable.

Power up the board using the DC power supply. Invoke Hardware Debugger from the Design Manager window. For description of XChecker pin connections, refer to Tables 4_3 and 4_4 in the **Connecting your Cable** chapter of the Hardware Debugger Reference/User Guide.

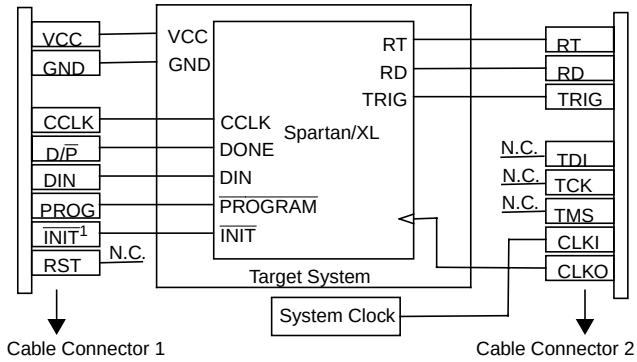

Figure 3: XChecker Cable Connections

NOTE: 1. The $\overline{\text{INIT}}$ pin on the connector is marked as INIT by mistake

5. *Configuration and Debugging using Hardware Debugger*

Configure the device using the Hardware Debugger software. Set the synchronous debug mode, clock options, trigger type settings and the signals to display and then perform the device readback and verify operations using the software. For more information on these operations, refer to the **Programming a Device or a Daisy Chain** and **Debugging a Device** chapters in the Hardware Debugger Reference/User Guide or the online help which is available in the Hardware Debugger software.

Any_File Flow

Here, any of the two cables - XChecker or Parallel cable can be used to configure the device. The XChecker cable must be connected to the serial port of the PC or workstation and the Parallel cable to the parallel port of the PC. Also, you can use any of these files: BIT, RBT or any PROM file (MCS, TEK, EXO). The flows for these three files are described in the next section.

Using the BIT file

1. *Implement the design to get the BIT file.*

For more information, refer to **Using the Design Manager** chapter in the Design Manager/Flow Engine Reference/User Guide.

2. *XChecker/Parallel Cable Connections*

Make sure the mode pin(s) are set for slave serial mode. Although these pins have weak pull-up resistor during configuration, it is recommended to attach an external pull-up of 4.7 KΩ to make sure these pins are not floating. Connect the XChecker cable connector 1 to the target system as shown in Figure 4. The Parallel cable pin connections are the same as that of the XChecker cable except that there is no connection to the $\overline{\text{INIT}}$ pin as the cable does not have it. Power up the board using a 5V DC supply if it has Spartan device(s) and 3.3V supply if it has SpartanXL device(s).

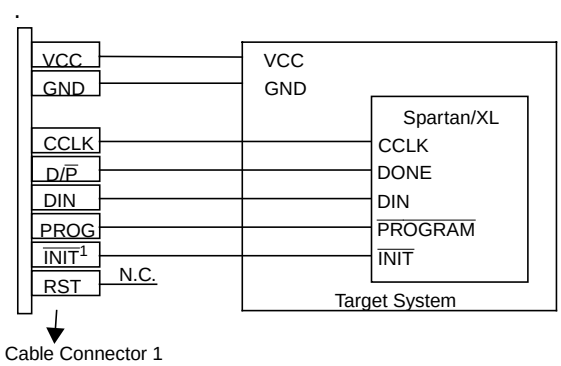

Figure 4: XChecker Cable Connections

NOTE: 1. The INIT pin on the connector is marked as INIT by mistake

3. *Configuration using Hardware Debugger*

Invoke the Hardware Debugger program from the Design Manager window and configure the device. For more information, refer to the **Programming a Device or a Daisy Chain** and **Debugging a Device** chapters in the Hardware Debugger Reference/User Guide or the online help which is available in the software.

Using the RBT file

1. *Implement the design to get the BIT and RBT files.*

For more information, refer to **Using the Design Manager** chapter in the Design Manager/Flow Engine Reference/User Guide.

2. *Follow the steps 2 and 3 described in the previous section "Using the BIT file" to configure the device.*

Using the PROM file

For a single device, the user may use the PROM file. But, for multiple devices in daisy chain, the PROM file must be used. No other file format (BIT, RBT) is acceptable.

1. *Implement the design to get the BIT file.*
2. *Create the PROM file using the PROM File Formatter in the Design Manager*

For more information, refer to the PROM File Formatter Reference/User Guide.

3. *XChecker/Parallel Cable Connections*

Connect the XChecker or Parallel cable as shown in Figure 4. Follow the instructions described in the step 2 of the previous section "Using the BIT file" to power up the board.

4. *Configuration using Hardware Debugger*

Invoke the Hardware Debugger program from the Design Manager window. The program looks for the MCS file by default. Optionally, the PROM format files TEK and EXO may also be used. Configure the device. For more information, refer to the **Programming a Device or a Daisy Chain** and **Debugging a Device** chapters in the Hardware Debugger Reference/User Guide or the online help which is available in the software.

JTAG Flow

If the user is testing a system using Boundary Scan, then it is possible to configure the FPGA through the same Boundary scan (JTAG) pins. This will minimize additional configuration circuitry. The JTAG Programmer software and one of the two cables, XChecker or Parallel, are used for this purpose. The programming file used is a BIT file. The BSDL files for all the Xilinx and non-Xilinx devices in the chain also must be available. The steps to perform configuration are described below:

1. *Instantiate the BSCAN symbol in your schematics or HDL based design.*

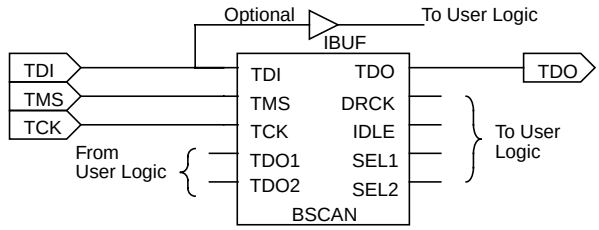

Figure 5: Example of Boundary Scan

2. *Implement the design to get the BIT file.*

For more information, refer to **Using the Design Manager** chapter in the Design Manager/Flow Engine Reference/User Guide.

2. *XChecker/Parallel Cable Connections*

Connect one end of the Parallel cable to the PC port (see Figure 6) and the other end to the target system. The XChecker cable pin connections are same as the Parallel cable except that the RD pin on the cable must be connected to the TDO pin of the target system. Power up the board using a 5V DC supply if it has Spartan device(s) and 3.3V supply if it has SpartanXL device(s). For more information, refer to the **Hardware** chapter of the Hardware User Guide.

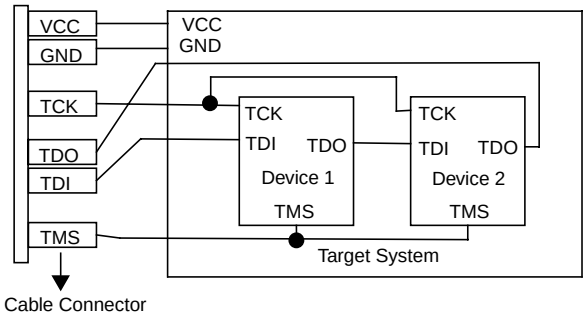

Figure 6: Parallel Cable Connections

3. *Configuration*

Invoke the JTAG Programmer program in the Design Manager and configure the devices in the chain. For

more information, refer to the JTAG Programmer Guide or the online help which is available in the software.

For more information on boundary scan, refer to the application note *XAPP017: Boundary Scan in XC4000 and XC5200 Series Devices*, which also applies to the Spartan series.

PROM Flow

Here, the PROM is used to store the configuration data for an FPGA or a daisy chain of FPGAs.

1. *Implement the design to get the BIT file.*
2. *Create the PROM file using PROM File Formatter in the Design Manager*

   For more information, refer to the PROM File Formatter Reference/User Guide.
3. *Loading the PROM file into a PROM*

   To load the PROM file into a Xilinx serial PROM, use the HW-130 programmer or any third-party PROM programmer. The byte-wide PROM can be used only for the Express mode configuration. It can be from any third party PROM manufacturer. For information on programmers, refer to the HW-130 Programmer Users Manual or 3rd party supplied documentation, if you are using a non-Xilinx programmer.
3. *Connecting PROM to an FPGA*

   Select the appropriate configuration mode according to the requirements. Connect the PROM to the FPGA by looking at corresponding circuit diagram under the **Configuration and Test** section in Spartan/XL data sheet.
4. *Configuration*

   Power up the board using a 5V DC supply if it has Spartan device(s) and 3.3V DC power supply if it has SpartanXL device(s). Configure the device.

Processor Flow

Note that this section covers the topic in a very high level without going into specific technical details. The user is advised to learn more based on the type of processor that is being used. This method is also described in the application note *XAPP098: The Low-Cost Efficient Serial Configuration of Spartan FPGAs*.

1. *Implement the design to get the BIT file.*
2. *Create the PROM file using PROM File Formatter in the Design Manager*

   Make sure that you create the file format that is acceptable by the intelligent interface. For more information, refer to the PROM File Formatter Reference/User Guide.
3. *Loading the PROM file*

   Load the PROM file into RAM of the processor or any other configuration data storage device you are using.
4. *Connecting the processor to an FPGA*

   Select the appropriate configuration mode according to your requirements. Connect the intelligent interface to the FPGA as shown in the corresponding circuit diagram under the **Configuration and Test** section in the Spartan/XL data sheet.
5. *Configuration*

   Power up the board using a 5V DC supply if it has Spartan device(s) and 3.3V DC supply if it has SpartanXL device(s). Configure the device.

References

Data Sheets

- Spartan/XL data sheet

  (http://www.xilinx.com/partinfo/spartan.pdf)
- Spartan/XL series Serial PROMs data sheet

  (http://www.xilinx.com/partinfo/
- HW-130 Programmer Data Sheet

  (http://www.xilinx.com/partinfo/program.pdf)

Application Notes

- XAPP017: Boundary Scan in XC4000 and XC5200 Series Devices

  (http://www.xilinx.com/xapp/xapp017.pdf)
- XAPP098: The Low-Cost Efficient Serial Configuration of Spartan FPGAs
  (http ://www.xilinx.com/xapp/xapp098.pdf)
- XAPP122: The Express Configuration of SpartanXL FPGAs

  (http://www.xilinx.com/xapp/xapp122.pdf)
- FPGA Configuration Application Notes

  (http://support.xilinx.com/apps/config.htm)

Technical Support

- The Configuration Problem Solver

  (http://support.xilinx.com/support/techsup/journals/config/cps.htm)
- Hardware Debugger Watch Tutorial

  (http://support.xilinx.com/support/techsup/tutorials)
- Programmer Solutions Technical Tips

  (http://support.xilinx.com/support/programr/ps.htm)

Online Software Documents

(http://support.xilinx.com/support/sw_manuals/)

- Design Manager/Flow Engine Reference/User Guide
- Development System Reference Guide
- PROM File Formatter Reference/User Guide
- JTAG Programmer User Guide
- Hardware Debugger Reference/User Guide
- Hardware User Guide