Remora-PMC+

64-bit/66 MHz PCI Mezzanine Card with
Reconfigurable FPGA

User's Guide

BittWare

BittWare, Inc.
31 B South Main St.
Concord, NH 03301 USA
603.226.0404

If you have comments or suggestions about this manual or find any errors in it, please contact us via e-mail at techpubs@bittware.com.

For technical support, contact us using any of the following methods:
Phone: 603.226.0404
FAX: 603.226.6667
E-mail: support@bittware.com

BittWare also maintains the following Internet sites:

http://www.bittware.com	Contains product information, technical notes, support files available for download, and answers to frequently asked questions (FAQ).
ftp://ftp.bittware.com	Contains technical notes and support files. Login as "anonymous" and use your e-mail address for the password.

Remora-PMC+ Reference Manual

Hardware Revision 0

The information in this manual has been carefully checked and is believed to be accurate and reliable. However, BittWare assumes no responsibility for any inaccuracies, errors, or omissions that may be contained in this manual. In no event will BittWare be liable for direct, indirect, special, incidental, or consequential damages resulting from any defect or omission in this manual. BittWare reserves the right to revise this document and to make changes from time to time in the content hereof without obligation of BittWare to notify any person or persons of such revision or changes.

Printed in the USA
December 7, 2003 Edition UG-RMPM00-01

Contents

Chapter 1
Introduction

Chapter 2
Preparing the Remora-PMC+ for Operation

Chapter 3
Overview of the Hardware Components

Chapter 4
Functional Description

Chapter 5
Programming Details for the SharcFIN ASIC

Chapter 6
Configuring the Virtex-II FPGA

Appendix A
Example Programs for the Remora-PMC+

List of Figures

List of Tables

Chapter 1
Introduction

BittWare's Remora-PMC+ is a reconfigurable FPGA board that features the power of the XILINX Virtex-II FPGA on a BittWare Mezzanine Card (PMC). The Virtex-II features up to 1 million system gates, a flexible digital I/O interface, and reconfigurable computing blocks. The board supports 64–512 MB of SDRAM and features BittWare's Sharc®FIN™ ASIC, which flexibly interfaces the 64-bit, 66 MHz PCI interface to the SDRAM, CPLD, and the Virtex-II FPGA. The PMC+ interface provides a 64-bit, 66 MHz PCI interface to the host, and its I/O extensions provide connections from the Virtex-II link ports to the host board.

This chapter covers the following topics:

- The basic architecture of the Remora-PMC+ system
- An overview of each chapter in this user's guide
- Additional documents that provide more information about the Remora-PMC+'s components and software

1.1 Overview of the Remora-PMC+

This section gives a brief overview of the architecture of the Remora-PMC+ board and describes its software.

1.1.1 Remora-PMC+ Features

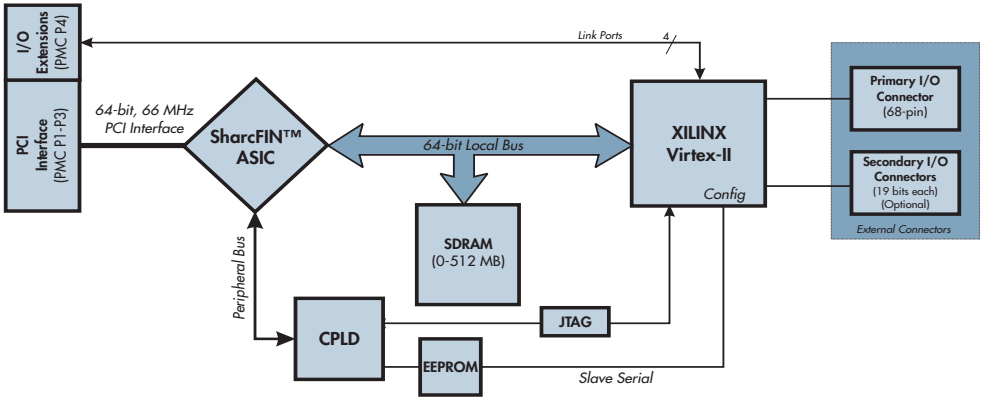
The Remora-PMC+ features:

- Virtex-II reconfigurable FPGA with up to 106 user-accessible I/Os
- BittWare SharcFIN ASIC-supported 64-bit, 66 MHz PCI interface
- 64 to 512 MB SDRAM (standard 144-pin SODIMM)
- One front panel 68-pin high-density connector with optional LVDS terminations provide direct inputs to the Virtex-II
- Four link ports extend from the Virtex-II to the PMC+ interface
- BittWare Mezzanine Card (PMC) form factor

1.1.2 Remora-PMC+ System Architecture

This section gives a basic overview of the Remora-PMC+ system, describing how all of its features work together. Figure 1–1 on page 2 is a detailed block diagram of the Remora-PMC+ board and its features.

Figure 1–1 Block Diagram of the Remora-PMC+



BittWare's Remora-PMC+ board features the XILINX Virtex-II reconfigurable FPGA, and BittWare's SharcFIN ASIC, which provides a

full 64-bit, 66 MHz master PCI interface. It also features up to 512 MB of SDRAM, and four link ports via P4 on the PMC interface. In addition to the PMC interface, a high-density 68-pin connector and two optional digital I/O connectors provide the I/O for the board.

SharcFIN ASIC

The Remora-PMC+ incorporates a BittWare SharcFIN ASIC to flexibly interface the Virtex-II FPGA to the 64-bit, 66 MHz PCI bus and the SDRAM. The SharcFIN also provides a feature-rich set of DMA functions and interrupt options to support very high-speed, real-time data flow with minimum processor overhead.

Virtex-II FPGA

The XILINX Virtex-II reconfigurable FPGA is a powerful tool for designing custom digital interfaces based on IP cores and customized modules. The Remora-PMC+ features a flexible digital I/O interface that provides up to 106 user I/O pins, each of which is individually configurable for any of nineteen single-ended I/O standards and six differential I/O standards, including LVDS, SSTL, HSTL II, and GTL+.

I/O Options

The Virtex-II can send and receive data via the SharcFIN, which provides the 64/66 PCI interface, or via a high-density 68-pin primary I/O connector on the front panel. The Virtex-II also supports four link ports, allowing it to move data to the host via the PMC+ (P4) interface. A local parallel bus gives the Virtex-II high-speed access to the SDRAM and the SharcFIN. In addition to the 68-pin front panel I/O connector, two optional connectors provide an additional 38 bits of digital I/O to the Virtex-II.

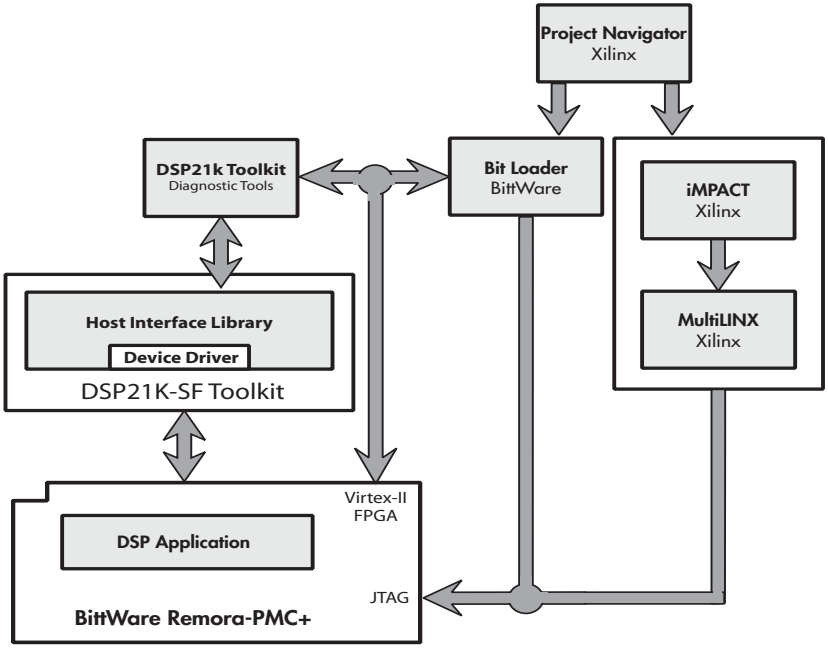
1.1.3 Remora-PMC+ Software Architecture

You will need two types of software development tools for the Remora-PMC+: host interface tools and development software for the FPGA. Figure 1–2 is a general block diagram of how the software development tools work together with the Remora-PMC+.

BittWare's DSP21k-SF Toolkit provides your host interface tools. The DSP21k-SF Toolkit is a complete software development kit that allows you to easily develop application code and integrate the Remora-PMC+ into your system. The software tools include a comprehensive host interface library (HIL), a standard I/O library, and diagnostic utilities.

To develop code for the FPGA, BittWare recommends the XILINX Project Navigator suite. This package provides a development platform for creating and debugging .bit and .mcs FPGA files. The iMPACT file loader, a subset of Project Navigator, works with the optional MultiLINX cable and allows you to easily port your FPGA code to the Virtex-II. BittWare also provides the BitLoader utility to efficiently load .bit files created with Project Navigator directly to the FPGA while the board is running.

Figure 1–2 Block Diagram of the Software Architecture



1.2 About This User's Guide

Note *This user's guide describes hardware that is fully populated with two optional 32-pin digital I/O connectors, SDRAM, JTAG header, P4 (PMC+) connector, and external power connector. Keep in mind that your board may be configured differently and that some information and settings may not apply to your hardware.*

1.2.1 Purpose of this Document

This user's guide covers hardware revision 0 of the Remora-PMC+ board, which supports one Virtex-II reconfigurable FPGA. The purpose of this document is to provide details about the Remora-PMC+'s major hardware components, to describe how to install and properly operate the Remora-PMC+, and to discuss important issues that relate to programming the board. We expect that you are familiar with the functionality of the FPGA, and that you will create and load its logic files.

1.2.2 Conventions Used Throughout This User's Guide

We have used the following conventions throughout this user's guide.:

- All signal names appear in small capitals (RESET).
- Active low signals appear in small capitals with an overline ($\overline{\text{RESET}}$), except signals listed in Table 6–7 on page 102, in which active low signals are described in the following format: xha_ms0_l, where the suffix "_l" represents active low.
- A "0x" prefix designates a number as a hexadecimal number (0x01).

1.2.3 Chapter Overviews

Chapter 2: Preparing the Remora-PMC+ for Operation

This chapter describes the tasks that you must perform to prepare the board for installation, install the software for the board, install the board, and test the installation.

Chapter 3: Overview of the Hardware Components

This chapter shows the location of the Remora-PMC+'s major components and connectors and briefly discusses their function.

Chapter 4: Functional Description

This chapter discusses the board's architecture, including link ports, bus interfaces, and the architecture of the SharcFIN ASIC. It also discusses how to reset the board.

Chapter 5: Programming Details for the SharcFIN ASIC

This chapter provides programming details for the SFIN-160 SharcFIN, briefly describing the chip's memory map and user-configurable registers.

Chapter 6: Configuring the Virtex-II FPGA

This chapter describes the methods of programming the Virtex-II FPGA. It is intended to give a general overview of configuring the FPGA, but it does not explain how to write the FPGA software.

Appendix A: Example Programs for the Remora-PMC+

This appendix provides general descriptions of the I/O software examples included with this board.

1.3 Other Helpful Documents and Tools

This section gives sources for additional information that applies to the Remora-pmc+, and it lists several third party software development tools that you may find useful.

1.3.1 Documents for Further Reference

The documents in the list below provide additional information about the Remora-PMC+ components and software.

- *Virtex-II Platform FPGA Handbook* – XILINX
- *SFIN-160 SharcFIN ASIC User's Guide* – BittWare, Inc.
- *DSP21k-SF Toolkit User's Guide* (Version 7.1) – BittWare, Inc.
- *MultiLINX Universal Cable Transceiver User's Guide* – XILINX

1.3.2 Software Development Tools

BittWare Host Interface Support

BittWare supplies host interface support for the Remora-PMC+ with the DSP21k-SF Toolkit. Using the Toolkit's C-callable library of routines for DOS and Windows programs, read from and write to the Remora-PMC+ memory, and control other board functions. The *DSP21k-SF Toolkit (Version 7.1* and greater*) User's Guide* from BittWare, Inc. contains complete information about the DSP21k-SF Toolkit.

1.4 Documentation History

Table 1–1 shows the history of the Remora-PMC+ user guide.

Table 1–1 D6PC User Guide History

Document Number	Hardware Revision	Document Revision	Date	Changes
UG-RMPC-00	0	0	6/10/03	
UG-RMPC-01	0	1	12/7/03	• Transposition and typographical errors in Tables 3-1 and 3-4, and section 6.2.1 • Graphical errors in Figures 3-1, 3-3, 3-6 • Addition of usage note to section 3.2 • Addition of this section (1.4)

Chapter 2

Preparing the Remora-PMC+ for Operation

This chapter describes the tasks necessary to prepare the board for installation, such as setting configuration switches, connecting external devices, installing the board, and installing development software. It also discusses different methods of resetting the board and how to test it using BittWare's software tools. After reading this chapter, you should be able to:

- Set the board's configuration switches
- Mount the Remora-PMC+ on a PMC-capable carrier board
- Attach any desired external signals to the board
- Locate the various software installation instructions and perform the installations
- Run diagnostic tests on the board to ensure that it is operating properly
- Reset the board by various methods

2.1 Unpacking the Remora-PMC+

Note *The Remora-PMC+ contains electro-static discharge (ESD) sensitive devices. Be sure to follow the standard handling procedures for ESD sensitive devices, taking proper precautions to ground yourself and the work area before removing the board from its anti-static bag. If you fail to follow proper handling procedures, you could damage the board.*

To unpack the Remora-PMC+ board,

1. Carefully remove the board from the shipping box. Save the box and packing materials in case you need to reship the board.
2. Remove the module from the plastic bag, observing all precautions described in the warning above to prevent damage from electro-static discharge (ESD).
3. Carefully examine the board, checking for damage. If the board is damaged, ***do not*** install it. Call BittWare technical support.

2.2 Setting the Configuration Switches on the Remora-PMC+

The Remora-PMC+ has three configuration switches that allow you to control and enable several of the board's features. Before mounting the Remora-PMC+ on a host board, make sure all of the configuration switches have been properly set. Figure 2–1 shows where each of the switches is located, and Figure 2–2 on page 12 shows an example of the orientation of the switch positions to ON and OFF. The sections that follow describe the settings of each of the board's three switches.

Figure 2–1 Layout of the Remora-PMC+ Configuration Switches

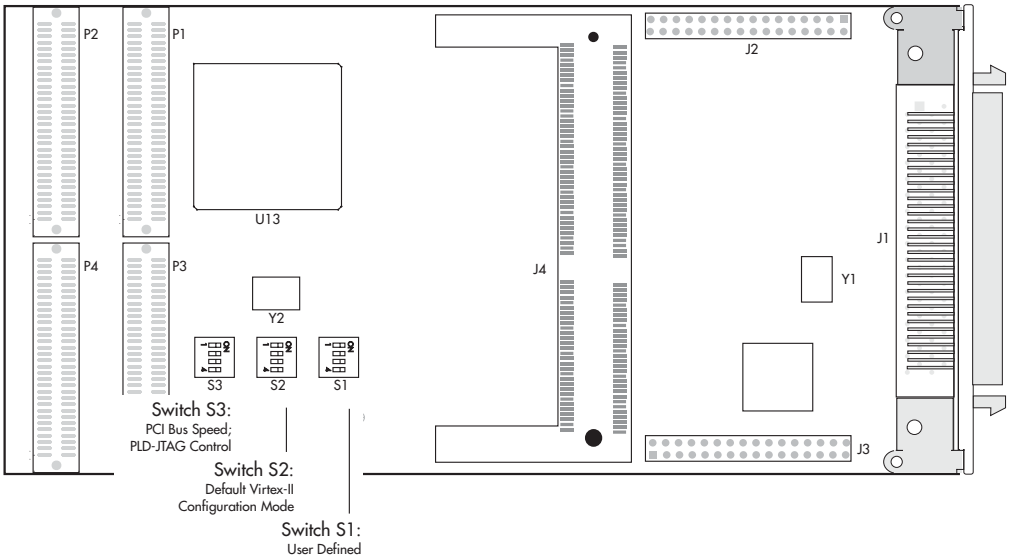
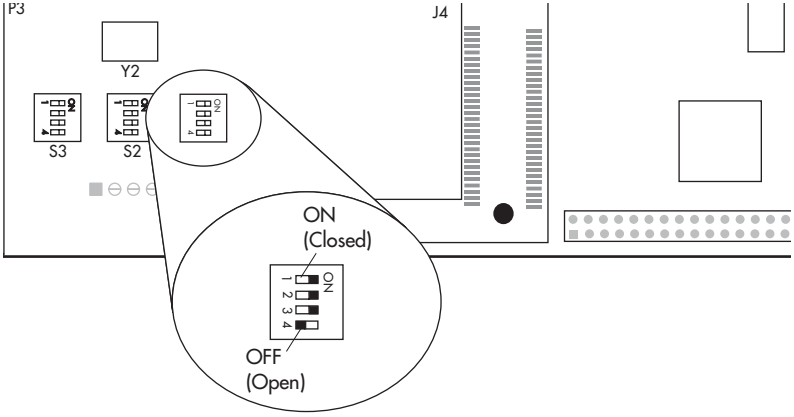


Figure 2–2 Orientation of Switch Positions



2.2.1 Setting the Inputs to the Virtex-II (S1)

Switch 1 (S1) sets modes for the user-configurable input signals on the Virtex-II chip. When the dips are in the OFF position, the inputs are routed to specific I/O pins on the Virtex-II chip. For example, you may choose to configure dip 1 as a manual reset switch by assigning the Virtex pin the appropriate reset signal value. In this case, 0=run and 1=reset, so that when the dip is set to OFF, the reset will occur. When configuring the pins, an input pull-up must also be defined in the chip file. Refer to the *Virtex-II Platform FPGA Handbook* by XILINX and Table 6–7 on page 102 for details on setting the user-configurable pins. Table 2–1 describes each dip setting and gives the associated Virtex pin name.

Table 2–1 Setting Inputs to the Virtex-II (S1)

Dip Switch	Description	ON	OFF	Virtex Pin Name
1	SW1_USR0	0	1	Y10
2	SW2_USR1	0	1	W10
3	SW3_USR2	0	1	AA13
4	SW4_USR3	0	1	Y13

2.2.2 Selecting the Default Virtex Configuration Mode (S2)

Before installing the Remora-PMC+, be sure to set switch 2 (S2), the Virtex-II configuration mode. The configuration mode is the method you will use to load the FPGA file onto the board:

- Master serial mode loads the file to the EEPROM
- JTAG boundary scan mode loads the file via the MultiLINX cable from Xilinx
- Slave serial mode loads the file from the host via the PCI bus

S2 differs from the other switches in that a combination of dips must be set for each mode. Table 2–2 shows the dip settings for each mode, and sections 6.1 and 6.2 describe their usage.

Note *While switch 2 sets the Virtex-II default configuration mode on power-up, the host can override the configuration mode.*

Table 2–2 Virtex-II Default Configuration Mode (S2)

Mode	Dip 4	Dip 3	Dip 2	Dip 1
EEPROM (Master serial mode)*	ON	ON	ON	ON
MultiLINX (JTAG boundary scan)	ON	OFF	ON	OFF
Host via PCI bus (slave serial mode)	ON	OFF	OFF	OFF
Manufacturer use only	OFF	ON	ON	ON

* The default configuration mode is set at the factory to EEPROM loading (master serial mode).

2.2.3 Setting the PCI Bus Speed and PLD-JTAG Control (S1)

Switch 3 (S3) configures the Remora-PMC+'s PCI bus speed, and sets the PLD-JTAG control (see the *Virtex-II Platform FPGA Handbook* for details). When dip 1 is set to OFF, the hot-swap mode is enabled. Dip 2 set to the ON position allows the SharcFIN to operate in standalone mode. Dip 3 controls the speed of the PCI bus. When dip 3 is set to ON, the PCI bus will operate at 33 MHz; when the dip is set to OFF, the bus will run at 66 MHz. Dip 4 controls the PLD-JTAG buffer. When this dip is set to ON, the JTAG interface is under software control. When this dip is set to OFF, the PLD is forced off the JTAG.

Table 2–3 PCI Bus Speed and PLD-JTAG Control (S3)

Dip Switch	Description	ON	OFF
1	Virtex-II Hot-swap Enable	Disabled	Enabled
2	SharcFIN Standalone mode	Enabled	Disabled
3	PCI 66 MHz Disable	PCI speed forced to 33 MHz	66 MHz
4	CPLD-JTAG buffer disable	EEPROM Programming via PCI Enabled*	Disabled

* The default setting for this switch is ON.

2.3 Connecting External Devices to the Remora-PMC+

2.3.1 Connecting a Cable to the I/O Connector

The Remora-PMC+ uses a 68-pin I/O connector to maximize throughput to the Virtex-II FPGA. I/O connector J1 is a 68-pin SCSI-type connector that is accessed from the front panel. It can be configured with 34 pairs of Voltage Differential Signalling (LVDS) input terminations, 17 pairs of LVDS input terminations, custom LVDS terminations, or no terminations.

To connect a cable to I/O connector J1, line up pin 1 on the cable with pin 1 on the connector. BittWare offers cables for use with the Remora-PMC+ board. See Table 2–4 for the part numbers for each connector, and contact your sales representative for ordering instructions. Please note that while BittWare uses SCSI-type connectors, a SCSI bus is not used on this board.

Table 2–4 Connector Part Numbers for J1

J1 Header Type	Part Number	Mating Part Number
68-pin D-shell female SCSI	AMP 787082-7	AMP 1-111196-7

2.3.2 Connecting Cables to the Digital I/O Connectors

Optional 32-pin external connectors J2 and J3 each provide 19 bits of digital I/O to the Virtex-II. Since the connectors are vertically mounted on the board, populating them violates the PMC specification and requires an adjacent slot in the chassis. To connect a cable to J2 or J3, align pin 1 on the cable to pin 1 on the connector (Figure 3–6). See section 3.2.2 for more details on the use of this connector.

Table 2–5 Connector Part Numbers for User I/O Connectors (J2, J3)

Location	Header Type	Mating Part Number
J2, J3	32-pin male 2mm × 2mm	Most 32-pin female 2mm × 2mm straight headers

2.3.3 Connecting an External Power Supply

The Remora-PMC+ requires a +3.3 and +5V power supply for normal operation. The optional external power and reset connector (J6) supplies +3.3V and +5V to the Remora-PMC+. Table 3–5 on page 32 gives the pinout of the connector and Figure 3–8 shows where it is located.

To connect an external power source to the Remora-PMC+,

1. Plug a power adapter cable into the Remora-PMC+'s external power connector (J6). Be sure to align pin 1 on J6 with pin 1 on the cable.
2. Connect the remaining end of the cable to an external power source, such as a switching standalone power supply or the PC's power supply.
3. Apply power to the system.
4. Reset the Remora-PMC+. Section 4.5 explains in more detail how to reset the board.

2.3.4 Connecting the MultiLINX Cable

The JTAG interface on the Remora-PMC+ is accessed by an optional eight-pin header (J5). This header is used by the MultiLINX Universal Cable Transceiver to load programs onto either the Virtex-II or the EEPROM, as well as debug the Virtex-II. To connect the device to the Remora-PMC+, follow these basic steps and refer to the *MultiLINX Universal Cable Transceiver User's Guide* for complete installation instructions.

To connect a MultiLINX cable to the Remora-PMC+,

1. Depending on the host PC interface, connect the MultiLINX host interface to either the RS-232 port or the USB port on the PC.
2. The MultiLINX cable pod has two banks of flying-lead connectors. Connect only six leads of the first bank to the pins of J6 as described in Table 2–6. For the header's orientation, refer to Figure 3–7 on page 34 for the location of pin 1.

Table 2–6 MultiLINX Header Pin Arrangement (J6)

J6 Pin	Signal	Corresponding MultiLINX Lead	J6 Pin	Signal	Corresponding MultiLINX Lead
1	P33V	PWR	2	SD0	RD(TD0)
3	SDI	TDI	4	$\overline{\text{ISPEN}}$ (NC)	
5	Key		6	Mode	TMS
7	GND	GND	8	SCLK	TCK

2.4 Installing the Board and Its Software

This section describes how to mount the Remora-PMC+ on a host board and how to install the BittWare software development tools to interface with the hardware.

2.4.1 Mounting the Board on a PMC Carrier Board

The standard set-up for the Remora-PMC+ is mounted on a PMC-capable host board. Its PMC interface features three standard PMC connectors (P1–P3), which provide the 64-bit, 66 MHz PCI interface. An additional connector, the PMC+ connector (P4), supports a TDM serial connection, four link ports, and flags and interrupts directly to the DSPs on the host board.

Note *Connector P4 of the PMC connectors on the Remora-PMC+ is compatible only with BittWare's PMC+-capable boards. BittWare uses P4 (see section 3.2.6 on page 35) for our PMC+ extensions. If you are mounting the Remora-PMC+ card on a host board that is not from BittWare and uses the P4 connector, the host board may have incompatibilities with the Remora-PMC+'s PMC+ (P4) connector. Call BittWare technical support for assistance.*

Note *The PMC interface is auto-compatible with 32/33, 32/66, 64/33, and 64/66 PCI interfaces.*

To attach the Remora-PMC+ to a PMC-capable host board,

1. Plug the Remora-PMC+'s PMC connectors into the PMC interface on the carrier board, and press the connectors firmly together.
2. Secure the Remora-PMC+ to the mounting holes on the carrier board.
3. Check the required PCI signalling and width to ensure that they fall within the ranges in the note above.

Table 2–7 shows the different options for accessing the host via the PMC interface.

Table 2–7 PMC Interface Access Options

PMC Connectors	Host Access
P1, P2	32-bit PCI
P1, P2, P3	64-bit PCI
P1, P2, P4	32-bit PCI and PMC+ extensions
P1, P2, P3, P4	64-bit PCI and PMC+ extensions

2.4.2 Installing the BittWare DSP21k-SF Toolkit

This section gives a basic overview of installing the BittWare DSP21k-SF Toolkit. For detailed installation instructions, refer to the *DSP21k-SF Toolkit Installation Guide*

Overview of the DSP21k-SF Toolkit

BittWare's DSP21k-SF Toolkit is a set of libraries and utilities that enable you to develop DSP applications for the Remora-PMC+ more quickly and easily. It contains a host interface library of C-callable functions for PC-based programs, diagnostic utilities, and demo programs.

Libraries. The primary component of the DSP21k-SF Toolkit is the ***Host Interface Library*** (HIL). The HIL is a library of C-callable functions for programs that allows you to download and start programs on the board, read from and write to the Remora, and control other board functions.

Diagnostic Utilities. *Diag21k* is a character-based diagnostic utility that lets you interactively access the Remora-PMC+.

Installing the DSP21k-SF Libraries and Utilities

Run the DSP21k-SF Toolkit setup program to install the DSP21k-SF Toolkit libraries and utilities. The *DSP21k-SF Toolkit User's Guide* explains the procedure in more detail.

Verifying that the Board is Configured Properly

The BittWare Configuration Manager is a utility included with the DSP21k-SF Toolkit that allows you to install, uninstall, or get and set properties for the Remora-PMC+ board. The *DSP21k-SF Toolkit User's Guide* explains how to run the BittWare Configuration Manager. Run this program only after installing the Remora-PMC+ in your system.

2.4.3 Installing the BittWare BitLoader Utility

BittWare provides the BitLoader utility on the Remora-PMC+ example CD-ROM. BitLoader is used to load .bit files onto the Virtex-II, and to load the Virtex-II with existing EEPROM logic files. Three versions of the BitLoader utility are included on the DSP21k-SF Toolkit CD-ROM to accommodate different operating systems. Refer to Table 2–8 for the file paths for each operating system.

Table 2–8 BitLoader Versions for Various Operating Systems

Operating System	Program File Path	Driver Library File Path
Windows	DSP21K-SF Toolkit\bin\BitLoader.exe	DSP21K-SF Toolkit\bin\bitlib.dll
MS-DOS	DSP21K-SF Toolkit\bin\bitldr.exe	DSP21K-SF Toolkit\bin\bitlib.dll
Linux	DSP21K-SF Toolkit\linux\bitldr	n/a

To install BitLoader, copy the applicable program file from the Toolkit CD-ROM to your local drive using the following path: `%dsp21ksf%\bin`. For Windows and MS-DOS operating systems, you must also copy the driver library file `bitlib.dll` to `%dsp21ksf%\bin`. See section 6.1 for step-by-step instructions on loading files with the BitLoader utility.

2.4.4 Installing Recommended Development Tools

BittWare supports and strongly recommends the following development tools:

- XILINX Project Navigator suite
- XILINX MultiLINX cable

FPGA Development Software

BittWare provides support files for the XILINX Project Navigator suite, including .cdf files for use with the iMPACT interface. Project Navigator is used to create FPGA .mcs files, and iMPACT provides the interface to load programs to the EEPROM and Virtex-II devices. For details on the installation and use of the XILINX development suite, refer to the software's user documentation.

MultiLINX Cable

The Remora-PMC+ supports the optional XILINX MultiLINX Universal Cable Transceiver, which operates in conjunction with the Project Navigator suite. The MultiLINX cable uses the JTAG interface (J5) to download FPGA-logic files to the Virtex-II and EEPROM from the host computer. To install the MultiLINX, refer to the instructions in section 2.3.4 on page 16.

2.5 Testing the Board to Make Sure It is Operating Properly

This section discusses running diagnostic tests on the board after installation to make sure it is operating properly.

2.5.1 Testing the Board with Diag21k Toolkit Utilities

The DSP21k-SF Toolkit contains the Diag21k utility for testing the board to make sure it is operating properly. Diag21k is a character-based diagnostic utility that starts from the MS-DOS command prompt. It lets you interactively access the Remora-PMC+.

2.5.2 Testing the Board with the Example Files

The example software provided with the Remora-PMC+ contains examples that demonstrate how to use the various features of your board and software. The examples are located in the examples directory of the Remora-PMC+ CD-ROM. See the readme.txt file on the example disk for installation instructions.

Chapter 3
Overview of the Hardware Components

This chapter shows where the Remora-PMC+'s major components and connectors are located and briefly describes their function. Section 3.1 describes the layout and function of the Remora-PMC+'s major components, section 3.2 describes the external connectors, and section 3.3 describes the configuration switches. This chapter covers the following components and connectors:

- Virtex-II FPGA
- SDRAM
- on-board oscillators
- LEDs
- JTAG header
- external power connector
- configuration switches
- PMC connector
- I/O connectors
- SODIMM connector

3.1 Layout and Function of the Major Components

This section briefly describes the function of each major component on the Remora-PMC+ board and shows where each is located. Figure 3–1 and Figure 3–2 show the components on the board.

Figure 3–1 Location of the Remora-PMC+'s Major Components (Top)

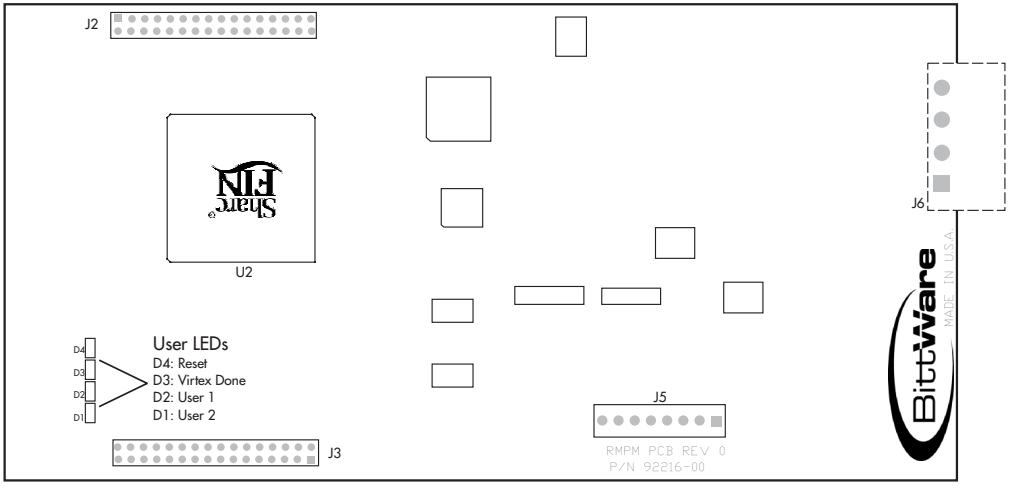
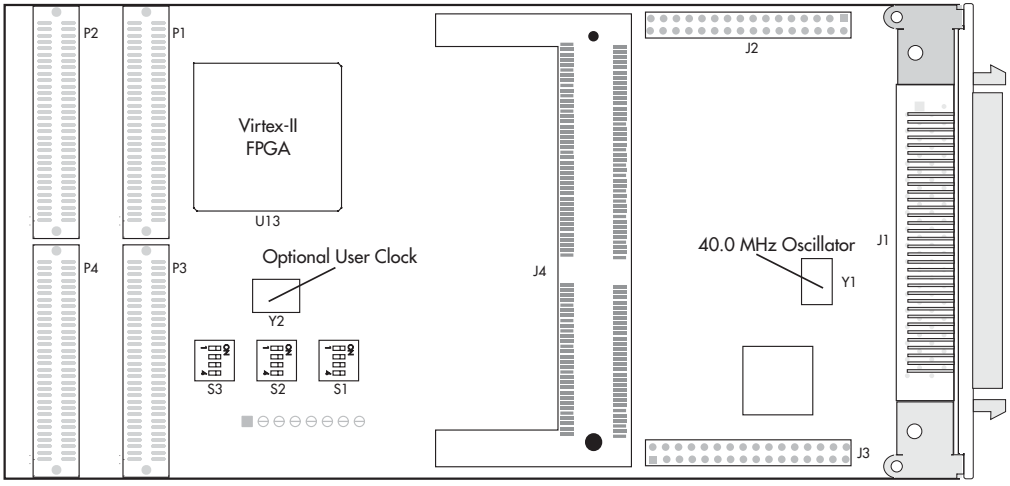


Figure 3–2 Location of the Remora-PMC+'s Major Components (Bottom)



3.1.1 SharcFIN ASIC

The SFIN-160 SharcFIN ASIC is a feature-rich, single device that combines the complex control of the PCI bus with an interface to the Virtex-II FPGA. The SharcFIN on the Remora-PMC+ board is a standard SFIN-160 SharcFIN for the ADSP-21160 SHARC DSP; however, although there are no DSPs on the board, the Virtex-II connects to the SharcFIN in the same manner as an ADSP-21160 DSP.

The SharcFIN ASIC flexibly interfaces the Virtex-II to a wide range of interfaces, including 64/66 MHz PCI bus (rev. 2.2 compliant), SDRAM, I^2C™ serial ports, and a general-purpose expansion bus. The SharcFIN also provides a feature-rich set of DMA functions and interrupt options to support very high-speed, real-time data flow with a minimum of processor overhead.

The following is a list of the SharcFIN's features:

- 64-bit, 66 MHz PCI rev. 2.2 compliant interface (528 MB/s burst)
- Interface to 64-bit, 40 MHz ADSP-21160 cluster bus
- Peripheral bus interface
  - 8 bits wide @ 20 MHz
  - Accessible from the ADSP-21160 cluster bus and the PCI bus
- Six independent FIFOs (2.4 KB total)
  - Four DMA buffers, 64×64 each (two transmit, two receive)
  - Two target buffers, 32×64 write, 16×64 read
- Direct, single PCI access from the ADSP-21160 cluster bus
- 16-byte configurable PCI mailbox registers
- I_2O™ V1.5 compliant
- Programmable interrupt multiplexer: 10 inputs, 7 outputs (one of each dedicated to PCI)
- SDRAM controller on ADSP-21160 cluster bus; supports up to 512 MB
- Standard I^2C interface

3.1.2 Virtex-II FPGA

The XC2V1000 Virtex-II FPGA from XILINX is a powerful tool for designing custom digital interfaces based on IP cores and customized modules. It features a flexible digital I/O interface that provides up to 106 inputs, each of which is individually configurable for any of nineteen

single-ended I/O standards and six differential I/O standards, including LVDS, SSTL, HSTL II, and GTL+. Any two I/O pins can be used as a differential pair, providing maximum board layout flexibility. The Virtex-II also offers configurable logic blocks (CLBs) that can be interconnected to create customized macros. It is configured on power-up by an on-board EEPROM, and it supports reconfiguration "on the fly" from the host.

3.1.3 SDRAM

The Remora-PMC+ has a standard 144-pin SODIMM that supports 64, 128, 256, or 512[1] MB SDRAM modules for banked external memory. The SDRAM is available via 64-bit parallel bus.

3.1.4 On-board Oscillators

The Remora-PMC+ has two on-board oscillators: one for the parallel bus, and one for user-specific applications.

Parallel Bus Oscillator

A 40 MHz system oscillator chip (Y2) provides the 1× clock for the parallel bus on the board. Figure 3–1 shows where the oscillator is located.

User Clock

The User Clock is populated by default with a 41.67 MHz oscillator for use by the TigerSharc link port core.

3.1.5 LEDs

The Remora-PMC+ has four user LEDs, D1–D4. These LEDs are connected directly to the Virtex-II; two are user-defined, and two provide feedback about the FPGA's status. The LEDs are ON when their corresponding signals are at logic-1. Table 3–1 describes the LEDs in more detail.

1. 512 MB modules are not yet available.

Table 3–1 Virtex-II LEDs

LED	Signal Name	Description	Virtex-II Pin Name
D1	USER_LED_2	User-defined	AA10
D2	USER_LED_1	User-defined	AB10
D3	V2_DONE	Virtex-II configuration status	
D4	$\overline{\text{V2\_RESET}}$	Virtex-II reset	W12

3.2 Layout and Function of the External Connectors

This section briefly describes the function of each external connector on the board and shows where they are located (see Figure 3–3 and Figure 3–4 below). It also provides the pinouts for the connectors.

Figure 3–3 Layout of the Remora-PMC+'s External Connectors (Top)

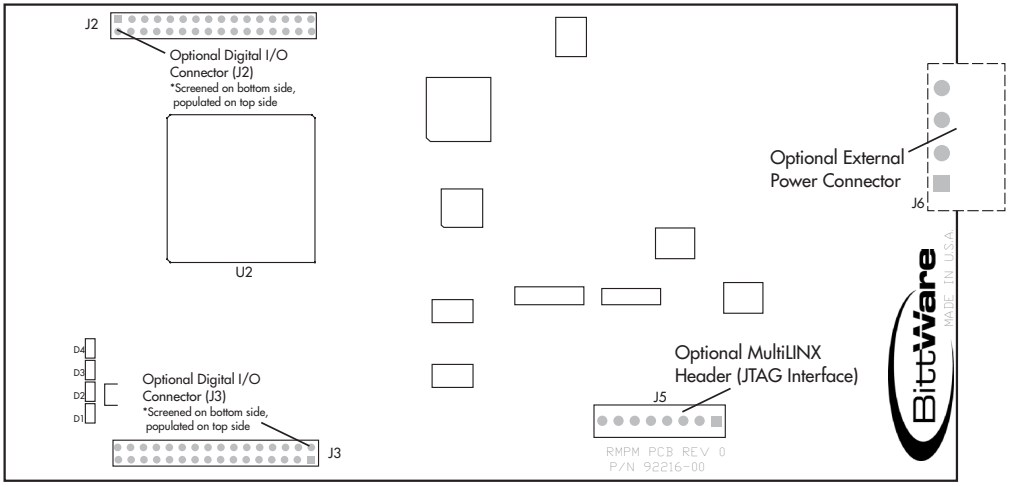


Figure 3–4 Layout of the Remora-PMC+'s External Connectors (Bottom)

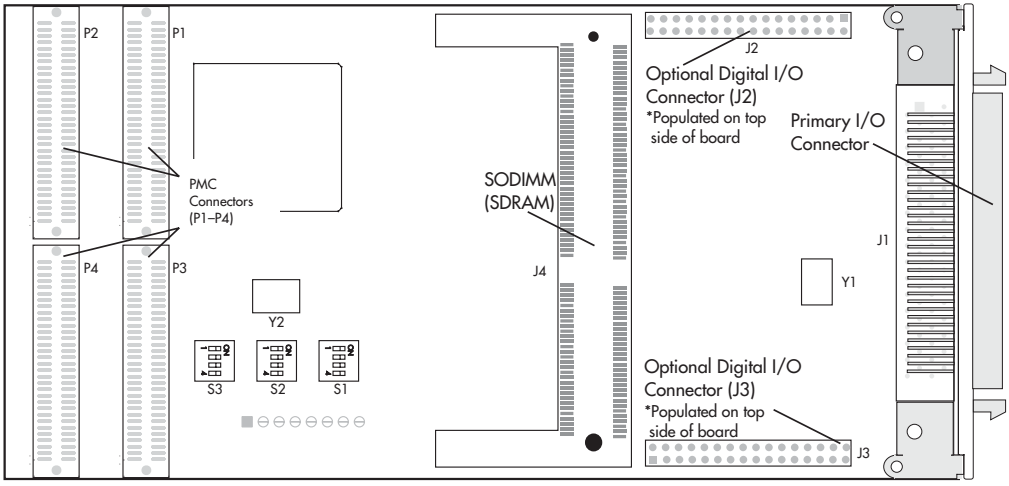


Table 3–2 Overview of the External Connectors

Connector	Ref Des	Type	Description
Primary I/O Connector	J1	68-pin	I/O connection to the Virtex-II FPGA
Secondary I/O Connectors	J2, J3	32-pin	Optional 19-bit digital I/O connectors, provide a total of 38 user bits*
SODIMM	J4	144-pin	Connection for standard 144-pin SODIMM SDRAM module
JTAG	J5	8-pin	Connection for optional MultiLINX device cable
External Power	J6	4-pin	Optional external power connector
PMC Connectors	P1–P4	64-pin	P1–P3: Standard PMC connectors to attach to BittWare PMC–capable carrier board P4: Provides BittWare's PMC+ I/O extensions, compatible only with a BittWare PMC+ capable carrier board.

* Optional digital I/O connectors J2 and J3 are screened on the bottom of the RMPC, but are populated on the top side of the board when ordered.

Note *If you are designing custom hardware to attach to any of the board's external connectors, we strongly recommend that you contact BittWare support first to ensure that you have the latest pinouts and design considerations.*

3.2.1 Primary I/O Connector

The primary I/O connector (J1) is a high-density, dual-use 68-pin connector. It is directly connected to the Virtex-II and therefore is limited to LVDS (differential) or 3.3V single-ended signalling. For LVDS use, as an ordering option, the pairs of pins (1 and 35, 2 and 36..., 34 and 68) can be individually configured with 100-Ohm termination. When populated, the 68-pin connector is a male shrouded header, but is easily converted to a female D-shell for use with the standard cable. See Table 2–4 on page 15 for a complete list of accessories and part numbers.

Table 3–3 on page 31 gives the primary I/O connector's pinout, and Figure 3–5 shows the location of pin 1. For details about each pin's corresponding signal on the Virtex-II, refer to Table 6–7 on page 102. The column labeled Resistors lists the resistor terminations for the optional LVDS input configuration.

Figure 3–5 Location of Pin 1 on Primary I/O Connector (J1)

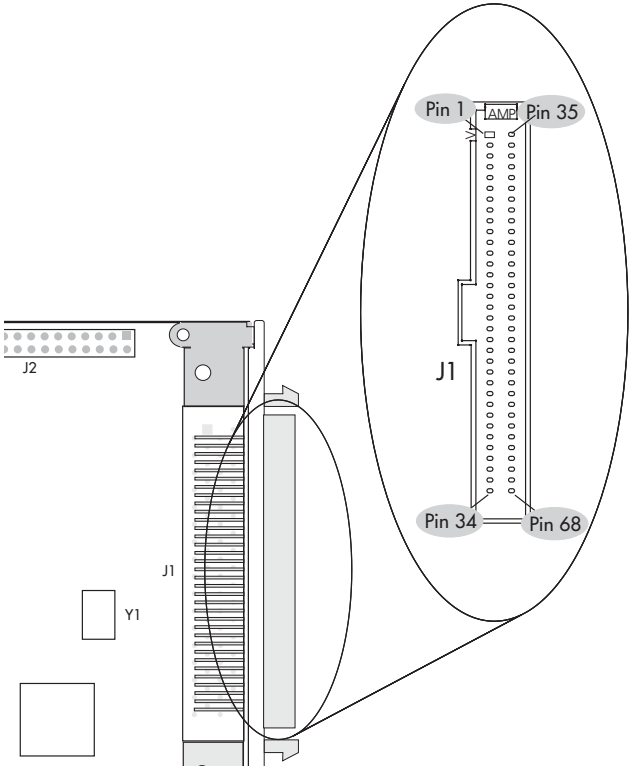


Table 3–3 Primary I/O Connector Pinout (J1)

Name	Pin	Pin	Name	Resistors
GND/EXIN_N1	1	35	GND/EXIN_P1	R20
EXEP_N2	2	36	EXEP_P2	R19
EXEP_N3	3	37	EXEP_P3	R37
EXEP_N4	4	38	EXEP_P4	R18
EXEP_N5	5	39	EXEP_P5	R36
EXEP_N6	6	40	EXEP_P6	R17
EXEP_N7	7	41	EXEP_P7	R35
EXEP_N8	8	42	EXEP_P8	R16
EXEP_N9	9	43	EXEP_P9	R34
EXEP_N10	10	44	EXEP_P10	R15
EXEP_N11	11	45	EXEP_P11	R33
EXEP_N12	12	46	EXEP_P12	R14
EXEP_N13	13	47	EXEP_P13	R32
EXEP_N14	14	48	EXEP_P14	R13
EXEP_N15	15	49	EXEP_P15	R31
EXEP_N16	16	50	EXEP_P16	R12
EXEP_N17	17	51	EXEP_P17	R30
EXEP_N18	18	52	EXEP_P18	R11
EXEP_N19	19	53	EXEP_P19	R29
EXEP_N20	20	54	EXEP_P20	R10
EXEP_N21	21	55	EXEP_P21	R28
EXEP_N22	22	56	EXEP_P22	R9
EXEP_N23	23	57	EXEP_P23	R27
EXEP_N24	24	58	EXEP_P24	R8
EXEP_N25	25	59	EXEP_P25	R26
EXEP_N26	26	60	EXEP_P26	R7
EXEP_N27	27	61	EXEP_P27	R25
EXEP_N28	28	62	EXEP_P28	R6
EXEP_N29	29	63	EXEP_P29	R24
EXEP_N30	30	64	EXEP_P30	R5
EXEP_N31	31	65	EXEP_P31	R23
EXEP_N32	32	66	EXEP_P32	R4
EXEP_N33	33	67	EXEP_P33	R22
GND / EXEP_N34	34	68	GND / EXEP_P34	R3

3.2.2 Secondary User I/O Connectors

The two 32-pin I/O connectors, J2 and J3, each have 19 user-configurable signals that pipe 19 bits of digital I/O to the Virtex-II. “Control Logic for I/O Connectors” on page 120 lists the signals on the connectors and describes a software example for their use. Note that when these optional headers are populated, they can be attached to either the top or bottom side of the board, depending on their usage and application. Figure 3–6 shows the location of pin 1 on each of the connectors based on a top-side installation, but note that this configuration violates the height limit of the PMC specification. Table 3–4 lists the connectors’ pinouts, and for details on the corresponding Virtex-II signals, refer to Table 6–7 on page 102.

Figure 3–6 Location of Pin 1 on User I/O Connectors – Top side (J2, J3)

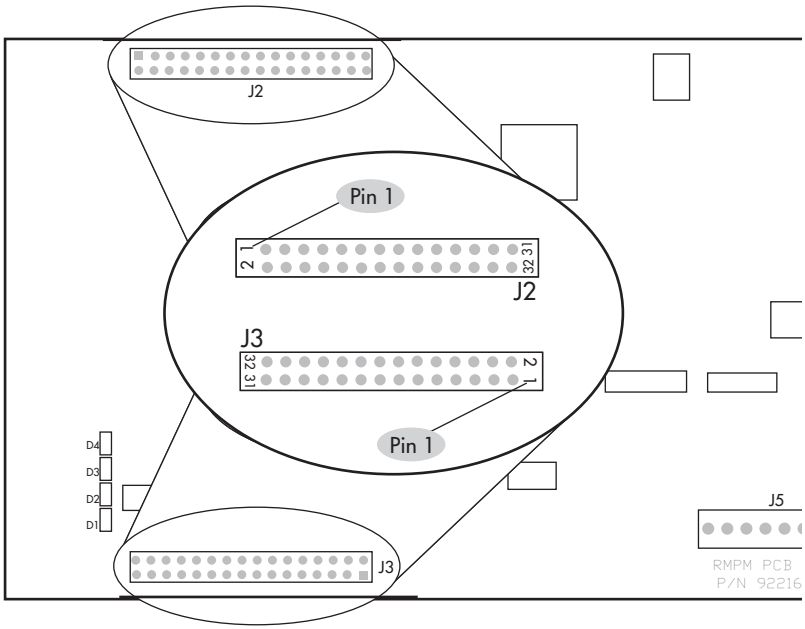


Table 3–4 Optional Digital I/O Connector Pinouts (J2 and J3)

J2 Pin	Header Signal	Virtex-II Pin Name	J2Pin	Header Signal	Virtex-II Pin Name
1	HDR_DIR	U12	2	HDR_D0	U18
3	GND		4	HDR_D1	U19
5	GND		6	HDR_D2	V16
7	GND		8	HDR_D3	V15
9	GND		10	HDR_D4	W16
11	HDR_CLK	AB12	12	HDR_D5	Y16
13	GND		14	HDR_D6	AA16
15	GND		16	HDR_D7	AB16
17	GND		18	HDR_D8	W15
19	GND		20	HDR_D9	Y15
21	HDR_ACK	AB13	22	HDR_D10	V8
23	GND		24	HDR_D11	AB6
25	GND		26	HDR_D12	AA6
27	GND		28	HDR_D13	Y7
29	GND		30	HDR_D14	W7
31	GND		32	HDR_D15	AB7
J3 Pin	**Header Signal**	**Virtex-II Pin Name**	**J3 Pin**	**Header Signal**	**Virtex-II Pin Name**
1	HDR2_DIR	C16	2	HDR2_D0	A16
3	GND		4	HDR2_D1	B16
5	GND		6	HDR2_D2	F14
7	GND		8	HDR2_D3	E15
9	GND		10	HDR2_D4	F19
11	HDR2_CLK	AA12	12	HDR2_D5	F20
13	GND		14	HDR2_D6	F21
15	GND		16	HDR2_D7	F22
17	GND		18	HDR2_D8	G18
19	GND		20	HDR2_D9	H18
21	HDR2_ACK	D16	22	HDR2_D10	G19
23	GND		24	HDR2_D11	G20
25	GND		26	HDR2_D12	T18
27	GND		28	GDR2_D13	U20
29	GND		30	HDR2_D14	U22
31	P33V		32	HDR2_D15	U21

3.2.3 SODIMM Connector

The Remora-PMC+ has an industry-standard 144-pin connection for a standard SODIMM module (J4). The SODIMMs are available in 64, 128, 256, and 512 MB modules.

3.2.4 MultiLINX Header

J5 is an optional 8-pin in-system programmable header for the XILINX MultiLINX Universal Cable Transceiver. The MultiLINX uses the JTAG interface to download FPGA files to the Virtex-II and EEPROM. Figure 3–7 shows the location of the pins on J5, and Table 3–5 gives the header's pinout. Refer to "Connecting the MultiLINX Cable" on page 17 for installation instructions. Section 6.2 describes how to configure the board for use with the MultiLINX and load files to the EEPROM and FPGA. Refer to the *MultiLINX Universal Cable Transceiver User's Guide* by XILINX for complete operation instructions.

Figure 3–7 Location of Pin 1 on the MultiLINX Header (J5)

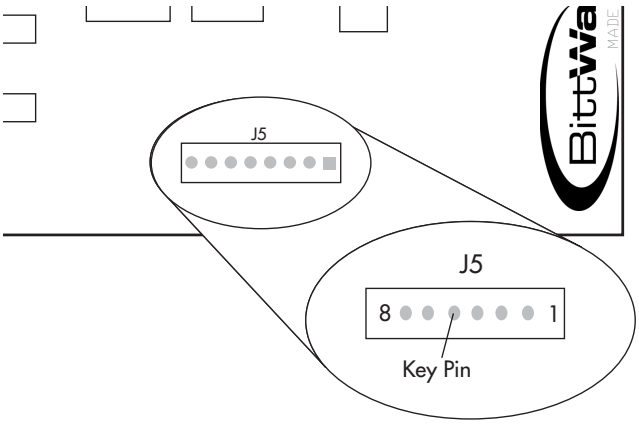


Table 3–5 MultiLINX Header Pinout (J5)

J6 Pin	Signal	Corresponding MultiLINX Lead	J6 Pin	Signal	Corresponding MultiLINX Lead
1	P33V	PWR	2	SDO	RD(TDO)
3	SDI	TDI	4	$\overline{\text{ISPEN}}$ (NC)	
5	Key		6	Mode	TMS
7	GND	GND	8	SCLK	TCK

3.2.5 External Power Connector

The Remora-PMC+ has an optional external power connector (J6) that, when populated, provides power to the board when it is operating in standalone mode. The external power connector is a 4-pin connector that supplies +3.3V and +5V to the Remora-PMC+. Figure 3–8 shows the location of the pins on the external power connector, and Table 3–6 gives the connector pinout. "Connecting an External Power Supply" on page 17 explains how to connect an external power supply to the connector.

Figure 3–8 Location of the External Power Connector Pins (J6)

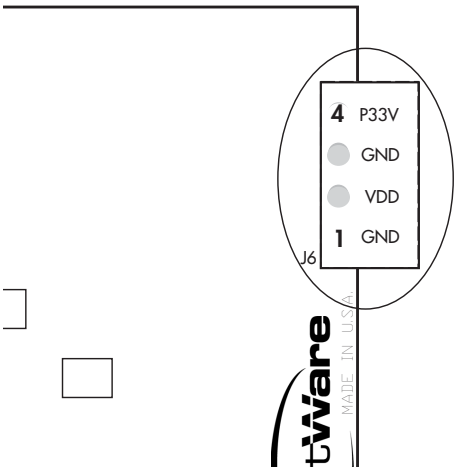


Table 3–6 External Power Connector Pinout (J6)

Pin Number	Power Signal
1	GND
2	VDD
3	GND
4	P33V

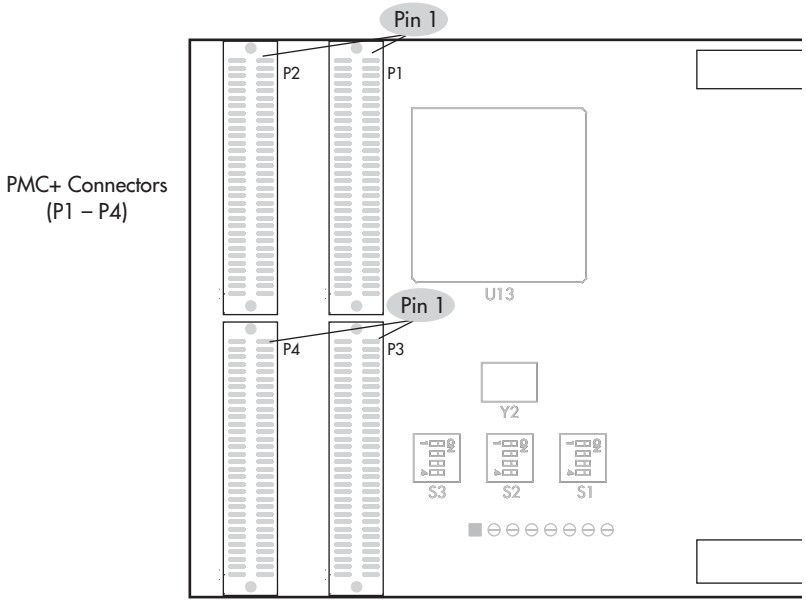
3.2.6 PMC+ Interface

The PMC+ interface allows you to mount the Remora-PMC+ on a standard PMC carrier board or on one of BittWare's PMC+ carrier boards. It consists of four 64-pin connectors. Three connectors (P1, P2, and P3) are standard 64-pin PMC connectors that provide the 64-bit, 66 MHz PCI interface. The fourth connector (P4)[1] is a 64-pin PMC+ connector that

provides BittWare's PMC+ I/O extensions. These extensions include a serial TDM bus, four link ports, an I^2C interface, and a reset line. Table 3–7 on page 37 gives the pinout for the PMC+ interface and Figure 3–9 on page 36 shows the location of pin 1 on the connectors.

Note *BittWare uses P4 of the PMC connectors for our PMC+ extensions. If you are mounting the Remora-PMC+ card on a host board that is not from BittWare and uses the P4 connector, the host board may have incompatibilities with the Remora-PMC+'s PMC+ (P4) connector. Call BittWare technical support for assistance.*

Figure 3–9 Location of the PMC Connectors (P1–P4)



1. P4 is the fourth (user-definable) PMC connector as defined in the *IEEE P1386.1 Standard Physical and Environmental Layers for PCI Mezzanine Cards: PMC* (PMC specification). It is referred to as Pn4/Jn4 in that document.

Table 3–7 PMC+ Interface Pinout (P1-P4)

P1 (PMC-J1)				P2 (PMC-J2)			
Pin	**Signal**	**Pin**	**Signal**	**Pin**	**Signal**	**Pin**	**Signal**
1	TCK	2	N12V	1	P12V	2	$\overline{\text{TRST}}$
3	GND	4	$\overline{\text{INTA}}$	3	TMS	4	TDO
5	$\overline{\text{INTB}}$	6	$\overline{\text{INTC}}$	5	TDI	6	GND
7	$\overline{\text{BUSMODE1}}$	8	VDD	7	GND	8	$\overline{\text{PCI-RSVD}}$
9	$\overline{\text{INTD}}$	10	$\overline{\text{PCI-RSVD}}$	9	$\overline{\text{PCI-RSVD}}$	10	$\overline{\text{PCI-RSVD}}$
11	GND	12	$\overline{\text{PCI-RSVD}}$	11	$\overline{\text{BUSMODE2}}$	12	P33V
13	CLK	14	GND	13	$\overline{\text{RST}}$	14	$\overline{\text{BUSMODE3}}$
15	GND	16	$\overline{\text{GNT}}$	15	P33V	16	$\overline{\text{BUSMODE4}}$
17	$\overline{\text{REQ}}$	18	VDD	17	$\overline{\text{PCI-RSVD}}$	18	GND
19	V(I/O)	20	AD[31]	19	AD[30]	20	AD[29]
21	AD[28]	22	AD[27]	21	GND	22	AD[26]
23	AD[25]	24	GND	23	AD[24]	24	P33V
25	GND	26	$\overline{\text{C/BE[3]}}$	25	IDSEL	26	AD[23]
27	AD[22]	28	AD[21]	27	P33V	28	AD[20]
29	AD[19]	30	VDD	29	AD[18]	30	GND
31	V(I/O)	32	AD[17]	31	AD[16]	32	$\overline{\text{C/BE[2]}}$
33	$\overline{\text{FRAME}}$	34	GND	33	GND	34	PMC-RSVD
35	GND	36	$\overline{\text{IRDY}}$	35	$\overline{\text{TRDY}}$	36	P33V
37	$\overline{\text{DEVSEL}}$	38	VDD	37	GND	38	$\overline{\text{STOP}}$
39	GND	40	$\overline{\text{LOCK}}$	39	$\overline{\text{PERR}}$	40	GND
41	$\overline{\text{SDONE}}$	42	$\overline{\text{SBO}}$	41	P33V	42	$\overline{\text{SERR}}$
43	PAR	44	GND	43	$\overline{\text{C/BE[1]}}$	44	GND
45	V(I/O)	46	AD[15]	45	AD[14]	46	AD[13]
47	AD[12]	48	AD[11]	47	M66EN	48	AD[10]
49	AD[09]	50	VDD	49	AD[08]	50	P33V
51	GND	52	$\overline{\text{C/BE[0]}}$	51	AD[07]	52	PMC-RSVD
53	AD[06]	54	AD[05]	53	P33V	54	PMC-RSVD
55	AD[04]	56	GND	55	PMC-RSVD	56	GND
57	V(I/O)	58	AD[03]	57	PMC-RSVD	58	PMC-RSVD
59	AD[02]	60	AD[01]	59	GND	60	PMC-RSVD
61	AD[00]	62	VDD	61	$\overline{\text{ACK64}}$	62	P33V
63	GND	64	$\overline{\text{REQ64}}$	63	GND	64	PMC-RSVD

P3 (PMC-J3)

Pin	Signal	Pin	Signal
1	PCI-RSVD	2	GND
3	GND	4	C/BE[7]
5	C/BE[6]	6	C/BE[5]
7	C/BE[4]	8	GND
9	V(I/O)	10	PAR64
11	AD[63]	12	AD[62]
13	AD[61]	14	GND
15	GND	16	AD[60]
17	AD[59]	18	AD[58]
19	AD[57]	20	GND
21	V(I/O)	22	AD[56]
23	AD[55]	24	AD[54]
25	AD[53]	26	GND
27	GND	28	AD[52]
29	AD[51]	30	AD[50]
31	AD[49]	32	GND
33	GND	34	AD[48]
35	AD[47]	36	AD[46]
37	AD[45]	38	GND
39	V(I/O)	40	AD[44]
41	AD[43]	42	AD[42]
43	AD[41]	44	GND
45	GND	46	AD[40]
47	AD[39]	48	AD[38]
49	AD[37]	50	GND
51	GND	52	AD[36]
53	AD[35]	54	AD[34]
55	AD[33]	56	GND
57	V(I/O)	58	AD[32]
59	PCI-RSVD	60	PCI-RSVD
61	PCI-RSVD	62	GND
63	GND	64	PCI-RSVD

P4 (PMC-J4, PMC+ Connections)

Pin	Signal	Pin	Signal
1	TDMRD	2	TDMRFS
3	TDMTD	4	TDMTRC
5	GND	6	GND
7	L1CLK/L1TXCLK	8	L1ACK/L1RXCLK
9	GND	10	GND/L1FSYNC
11	L1DAT0	12	L1DAT1
13	L1DAT2	14	L1DAT3
15	L1DAT4	16	L1DAT5
17	L1DAT6	18	L1DAT7
19	GND	20	GND
21	L2CLK/L2TXCLK	22	L2ACK/L2RXCLK
23	GND	24	GND/L2FSYNC
25	L2DAT0	26	L2DAT1
27	L2DAT2	28	L2DAT3
29	L2DAT4	30	L2DAT5
31	L2DAT6	32	L2DAT7
33	GND	34	GND
35	L3CLK/L3TXCLK	36	L3ACK/L3RXCLK
37	GND	38	GND/L3FSYNC
39	L3DAT0	40	L3DAT1
41	L3DAT2	42	L3DAT3
43	L3DAT4	44	L3DAT5
45	L3DAT6	46	L3DAT7
47	GND	48	GND
49	L4CLK/L4TXCLK	50	L4ACK/L4RXCLK
51	GND	52	GND/L4FSYNC
53	L4DAT0	54	L4DAT1
55	L4DAT2	56	L4DAT3
57	L4DAT4	58	L4DAT5
59	L4DAT6	60	L4DAT7
61	GND	62	RST
63	SCL	64	SDA

3.3 Layout and Function of the Configuration Switches

This section shows where each of the Remora-PMC+'s configuration switches is located (Figure 3–10) and gives a short description of each (Table 3–8). For details on the switch settings, refer to section 2.2.

Figure 3–10 Location of the Configuration Switches (Top)

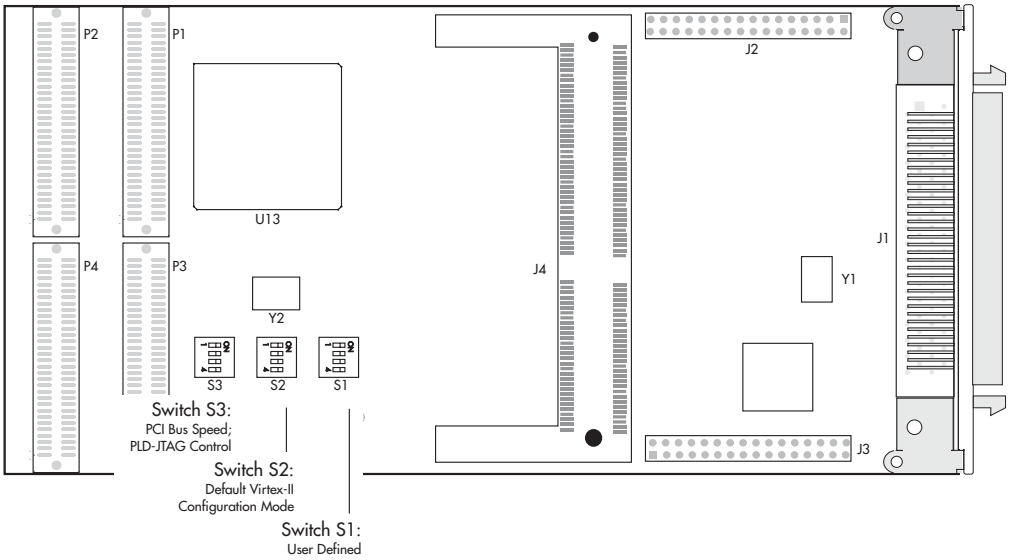


Table 3–8 Overview of the Configuration Switches (S1, S2, S3)

Switch	Name	Description
S1	User inputs to the Virtex	Four user-defined inputs to the Virtex-II FPGA
S2	Default Virtex configuration mode	Sets the configuration mode of the Virtex-II FPGA to EEPROM, MultiLINX, or host via PCI bus
S3	• PLD-JTAG control • PCI bus speed • Standalone • Hot-swap enable	• Places JTAG interface under software control • Sets PCI bus speed to either 33 or 66 MHz • Configures board for standalone mode • Configures Virtex for hot-swap

Chapter 4
Functional Description

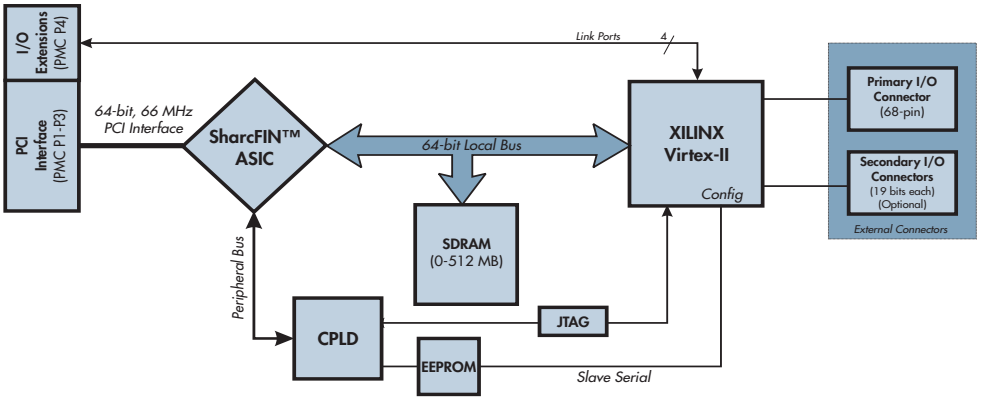
This chapter discusses how the Remora-PMC+ board functions. It covers the following topics:

- Overview of the board's architecture
- PCI interface architecture
- Connections available via the PMC+ interface
- The function of the Virtex-II FPGA
- Resetting the board
- The board's power requirements

4.1 Board Architecture Overview

This section briefly describes how data flows through the Remora-PMC+ board. The sections that follow discuss the board's architecture in more detail.

Figure 4–1 Block Diagram of the Remora-PMC+ System Architecture



The 64-bit, 40 MHz parallel bus provides an interface between the bank of SDRAM, the Virtex-II, and the SharcFIN ASIC. The SharcFIN provides a bridge between the Remora's parallel bus and the 64-bit, 66 MHz PCI bus on the PMC interface. A peripheral bus also extends off of the SharcFIN, providing access to the EEPROM.

The Remora also features a PMC+ connector. When attached to a BittWare PMC-capable host board, the PMC+ interface provides a TDM serial bus, an I^2C interface, and four link ports that extend to the Virtex-II FPGA.

The Virtex-II resides on the 64-bit, 40 MHz parallel bus and has direct access to the SDRAM, SharcFIN, and the EEPROM. In addition, the Virtex-II can be accessed by the host via the PCI interface (and the SharcFIN), or directly via the link ports on the PMC+ interface. For I/O connections to the Virtex-II, the Remora-PMC+ features up to three external I/O connectors.

4.1.1 PCI Interface Overview

A SharcFIN ASIC provides the 64-bit, 66 MHz PCI interface between the Virtex-II local bus and the PMC connectors. It also provides the SDRAM controller for the on-board 64-512 MB bank of SDRAM.

4.1.2 I/O Interface

The board features a primary I/O connector and two optional secondary digital I/O connectors mapped directly to the Virtex-II. The primary I/O connector is a 68-pin header that is mapped to specific I/O pins on the Virtex-II. The two optional secondary connectors each provide 19 user-configurable bits of digital I/O.

4.1.3 PMC Interface

The PMC provides a 64 bit/66 MHz PCI interface to the host board on connectors P1–P3. The optional fourth connector, P4, contains BittWare's PMC+ I/O extensions. These extensions include a serial TDM bus, four link ports, an I2C interface, and a reset line.

4.1.4 Link Port Interface

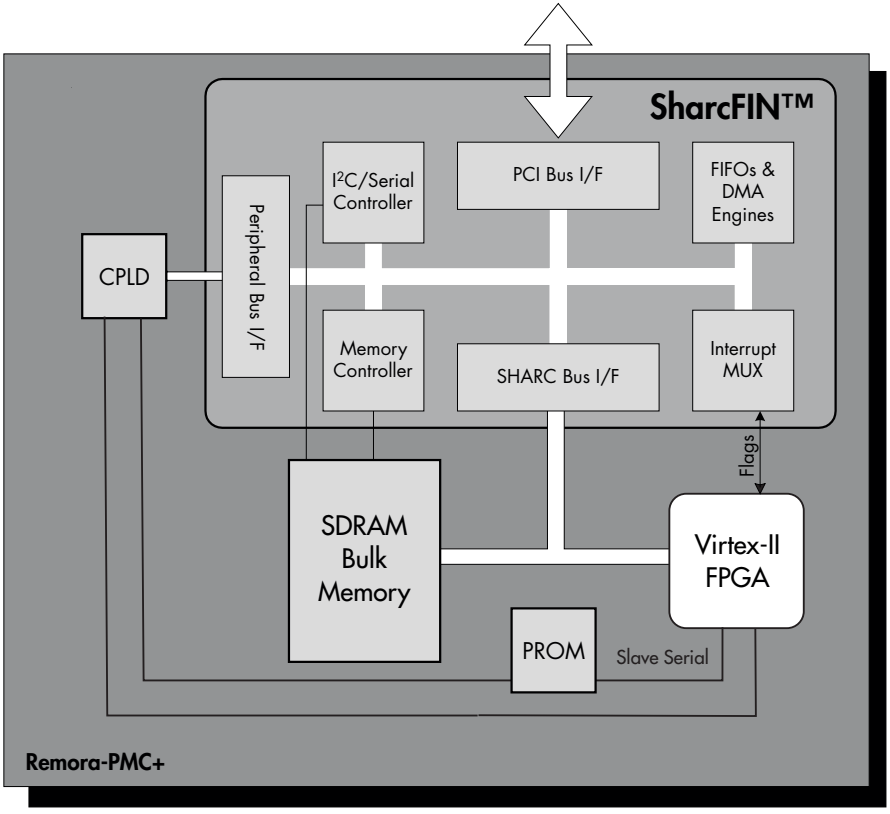
The Virtex-II has a link port core that provides four link ports connected directly to the PMC+ interface. The PMC+ interface connects the links directly to the DSPs on a BittWare Tiger carrier board. The links are configurable for either ADSP-TS101S or ADSP-21160 DSPs.

4.2 PCI Interface Architecture

4.2.1 Overview of the SharcFIN Architecture

The Remora-PMC+ features an SFIN-160 SharcFIN ASIC to provide the PCI interface and SDRAM controller for the Remora-PMC+. The SharcFIN can function as both a target and a master to the PCI bus specification 2.2. This section provides an overview of the SharcFIN architecture. Figure 4–2 below is a simplified block diagram of the SharcFIN architecture as it is implemented on the Remora-PMC+ board.

Figure 4–2 Simplified Block Diagram of the SharcFIN Architecture on the Remora-PMC+



Note *Although it does not have any DSPs on board, the Remora-PMC+ features an SFIN-160 SharcFIN for the ADSP-21160 SHARC DSP. The Virtex-II connects to the SharcFIN in the same manner as an ADSP-21160 DSP.*

Interface to Cluster Bus

The first function of the SharcFIN is to interface to the ADSP-21160 processor bus. On the Remora-PMC+, the Virtex-II is fully connected to the cluster bus and acts as an ADSP-21160 DSP.

Interface to PCI

The SharcFIN implements a full 64-bit/66MHz master PCI interface. The PCI interface is PCI rev 2.2 compliant.

Interface to Peripheral Bus

A third bus interface is provided by the SharcFIN's peripheral bus. The peripheral bus is a general-purpose utility bus that allows easy interface to standard microprocessor peripherals such as Flash memory. It provides a simple, glueless way to add additional functionality to the Remora-PMC+.

SDRAM Controller

The SharcFIN integrates a full-featured SDRAM controller, which allows the Virtex-II to access SDRAM using burst mode.

I^2C Serial Interface

The SharcFIN's I^2C/Serial controller integrates some of the most common peripheral requirements right into the SharcFIN. Uses include data communications, SharcFIN interconnection, and hardware configuration and identification. The I^2C controller is used on the Remora-PMC+ for a factory EEPROM that contains board configuration information.

Interrupt Multiplexer

The SharcFIN integrates an extensive interrupt and flag multiplexer to facilitate system-level control and coordination of multiprocessors. This

programmable resource allows you to select the sources of hardware interrupts to the Virtex-II, PCI host, and PMC interface.

On the Remora-PMC+, a few signals from the Virtex-II are connected to the interrupt multiplexer to allow PCI and PMC+ interrupts from the Virtex-II.

4.2.2 PCI Bus Interface

The Remora-PMC+'s 64-bit 66 MHz PCI bus interfaces the PMC interface and the SharcFIN ASIC, which provides the PCI controller. The PCI bus operates at 66 MHz with a 64-bit data path and a 64-bit address and is compliant with the rev 2.2 specification.

4.2.3 Peripheral Bus

The peripheral bus extends off of the SharcFIN ASIC and connects to a CPLD that allows reprogramming of the Virtex-II via its slave serial interface over PCI.

4.3 PMC Interface Architecture

The Remora features a BittWare Mezzanine Card (PMC) interface to allow you to attach the card to a PMC- or PMC+-capable carrier board. The PMC interface features four connectors which provide standard PMC connections and BittWare's PMC+ extensions.

4.3.1 PMC-to-PCI Interface

The first three connectors (P1, P2, P3) on the PMC interface provide a connection for standard PMC modules and are connected to the 64-bit, 66 MHz PCI interface. For complete pinouts of these three connectors, refer to Table 3–7 on page 37.

4.3.2 PMC+ Extensions

The fourth connector (P4) provides BittWare's PMC+ extensions. This connector is optional, and is compatible only with the BittWare's PMC+ capable host boards. The PMC+ interface provides link ports, a TDM serial bus, a reset line, and an I^2C interface directly to the DSPs on the host board. The PMC+ extensions are detailed in Table 3–7 on page 37.

4.4 Virtex-II FPGA Architecture

4.4.1 Link Interface

The Virtex-II has a link port core to provide four link port connections directly to the PMC+ interface. The PMC+ interface provides a direct connection to the DSPs on a BittWare Tiger-6U-cPCI carrier board. The link port connections are as follows:

- PMC+ Link 1 --> Virtex-II Link 1
- PMC+ Link 2 --> Virtex-II Link 2
- PMC+ Link 3 --> Virtex-II Link 3
- PMC+ Link 4 --> Virtex-II Link 4

4.4.2 Local Bus

The Virtex-II has a local bus core to provide a 64-bit local bus (parallel bus) to the SharcFIN ASIC, which connects to the PCI bus.

4.4.3 Digital I/O

An additional 19 pins of digital I/O are available to the Virtex-II on each of two optional 32-pin connectors (see "Secondary User I/O Connectors" on page 32).

4.4.4 JTAG

An optional MultiLINX header is available to allow the Virtex-II to be configured via JTAG using the Xilinx MultiLINX cable. The Remora-PMC+ has a single JTAG chain as follows (instruction widths are in parentheses):

TDI > Xilinx PROM (8) > Virtex-II (6) > PLD (5) > SharcFIN (3) > TDO

Note *By default, the digital I/O connectors and MultiLINX header are not populated because they violate the PMC height specification and are not accessible in-system while maintaining a single slot.*

Note that only users who are doing custom configurations of the Virtex-II will need to use the MultiLINX cable. For details on using the MultiLINX cable with the Remora-PMC+, refer to section 2.3.4 on page 16 and section 6.2 on page 92.

4.4.5 User Configurable Blocks

Space is available in the Virtex-II for user configurable blocks to allow you to customize the Remora-PMC+ for nearly any application. A variety of third party IP cores are available for use with the Remora-PMC+. For additional details on customizing the Virtex-II, go to Chapter 6.

4.5 Resetting the Board

The Remora-PMC+ will reset under the following conditions:

- Power on
- Under voltage
- PMC+ reset input[1]
- PCI bus reset
- Software reset via PCI and SharcFIN[1]

The sections that follow describe each method in greater detail. For more information, refer to the *SharcFIN ASIC User's Manual* by BittWare.

4.5.1 Resetting via PMC+ Interface

The PMC+ interface (via P4) provides a reset line from the host board to the Remora-PMC+ board. If the Remora-PMC+ is mounted on a BittWare PMC capable carrier board, you can reset the Remora-PMC+ via that reset line. Refer to the manual for the BittWare host board for details on issuing a PMC+ reset signal.

4.5.2 Resetting via PCI Bus

A register bit in the SharcFIN ASIC allows you to perform a total reset on the Remora-PMC+. The bit is B0 of the register located at offset 0xE8 from the base of Base Address Register 0 (BAR0).

4.5.3 Resetting via Software Reset

Either the SharcFIN or the PCI interface can issue a software reset to reset the Remora-PMC+. Refer to the *SFIN-160 SharcFIN ASIC User's Manual* for details on software resets.

1. PMC+ and software resets do not reset the PCI interface.

4.6 Power Requirements

The Remora-PMC+ requires a +3.3 and +5V power supply for normal operation. The optional external power and reset connector (J4) supplies +3.3V and +5V to the Remora-PMC+. Table 4–1 gives the worst-case power requirements for the board.

Table 4–1 Power Requirements for the Remora-PMC+

Voltage	Maximum current	Power*
+5V	1.6A	8W
+3.3V	3A	10W
	Total	**18W**

*This column provides worst case for all supplies and should be used for power supply specification. Actual power usage is likely to be much lower but depends heavily on the Virtex-II configuration.

Chapter 5

Programming Details for the SharcFIN ASIC

This chapter provides a brief description of how the SharcFIN functions, overviews the SharcFIN memory map, and describes how to set the SharcFIN's user-configurable registers. In addition to the information in the chapter, you will also need to refer to the SharcFIN user's manual.

Note *Although the Remora-PMC+ has no DSPs, it features a standard SFIN-160 SharcFIN, which is designed for BittWare's ADSP-21160 SHARC DSP boards. The Virtex-II FPGA resides on the ADSP-21160 cluster bus and uses the space normally occupied by DSP 1 on a BittWare ADSP-21160 board.*

5.1 Overview of the SharcFIN

5.1.1 The Two Sides of the SharcFIN

The SharcFIN consists of two main sections: the PCI interface and the SHARC interface. The PCI interface consists of a full 64-bit, 66 MHz bus mastering PCI interface and includes two DMA transmit channels, two DMA receive channels, and a single PCI read/write channel. Also included in the PCI interface is full I_2O support with the associated mailboxes.

The SHARC interface provides the SHARC specific functionality, which includes the SDRAM interface and control, the Virtex-II interface, the interrupt multiplexers, and the I^2C interface.

5.1.2 How the SharcFIN Maps to the PCI and ADSP-21160 Buses

The SharcFIN maps into PCI and the ADSP-21160 cluster bus. It uses PCI Base Address Registers (BARs) to map its various parts onto the PCI bus. BAR0 maps the SharcFIN's control registers, BAR1 maps to the peripheral bus (Virtex-II programming CPLD), BAR2 maps to the ADSP-21160's MMS space, BAR3 is unused in a 32-bit environment and used for the upper 32 bits of address in a 64-bit addressable system, and BAR4 maps to the SDRAM.

The SharcFIN maps into the ADSP-21160 cluster bus space using the MS (memory select) lines of the ADSP-21160s. MS0 maps to the SDRAM, MS1 maps to the peripheral bus, and MS2 maps to the SharcFIN control registers.

5.2 Function of the SharcFIN PCI Interface

The PCI side of the SharcFIN provides the complete PCI interface. It interfaces the PCI bus and the SHARC side of the SharcFIN and moves data between them. The SHARC side of the SharcFIN completes the interface, whether it be to SDRAM or to the Virtex-II.

The PCI side provides four DMA channels for performing PCI bus mastering DMAs: two are receive (for reads), and two are transmit (for writes). These channels can be run independently and will self-arbitrate for bus access. Along with the DMA channels, the PCI side provides a single PCI access channel for doing single PCI reads and writes and supports interrupts both to and from the PCI bus.

All control registers for the PCI interface are in Base Address Register 0 (BAR0) and occupy byte addresses from 0x00 to 0x100 off of BAR0. For complete details on these registers, refer to the *SharcFIN ASIC User's Manual* (BittWare). From the ADSP-21160, these control registers are at MS2 and are 32-bit addressable, so that they occupy word addresses 0x00 to 0x40 off of MS2. Section 5.3.1 gives an overview of how to access these registers.

5.2.1 Performing PCI Side DMAs

To perform a PCI bus mastering DMA, program a DMA channel in the PCI side of the SharcFIN. Next, program the DMA on the SHARC side to work in conjunction with the PCI side DMA. The PCI side DMA will move the data between the PCI bus and an internal FIFO, and the SHARC side DMA will move data between the internal FIFO and the actual source or destination on the board.

The PCI side DMAs are designed for 64-bit based transfers and expect 64-bit aligned data, regardless of the actual width of the PCI bus. The PCI address used in the transfer is a standard PCI byte-based address. For complete details on how to perform PCI side and SHARC side DMAs, refer to the section on "Bus Mastering and DMAs" in the S*harcFIN ASIC User's Manual.*

5.2.2 Performing a PCI Side Single Access

The SharcFIN supports a single PCI access channel for performing single PCI reads and writes. To perform PCI reads and writes, tell the SharcFIN which address to read or write, provide the data (for a write), and then request the transfer. On a read completion, the data is available in a buffer to be read. As with the DMAs, the SharcFIN is designed for 64-bit transfers and alignment. You can make it perform any number of byte width transfers by specifying which of the 8 bytes of the 64-bit access are to be enabled. However, you will need to align the data in the 64-bit word and use the 64-bit aligned address. For complete details on single PCI accesses, refer to the section on "Bus Mastering and DMAs" in the *SharcFIN ASIC User's Manual*.

5.2.3 Performing PCI Side Interrupts

The SharcFIN provides full I_2O support with the associated mailboxes. To generate PCI side interrupts, either write to PCI outgoing mailboxes or use the SHARC side PCI interrupt multiplexer, which generates the PCI side user interrupt bit.

Note *When reading the PCI side documentation of these registers, take careful note of whether you are looking at them from a PCI side or the "user" side. Phrases such as "PCI outgoing" have different meaning depending on your viewpoint, and several mailboxes and registers are duplicated – one for each direction.*

5.2.4 Function of the SharcFIN SHARC Interface

The SHARC side of the SharcFIN consists of the ADSP-21160 bus interface, the SDRAM controller, the peripheral bus, the I^2C interface, and the interrupt multiplexers. The SharcFIN control registers for the SHARC side are mapped into PCI in BAR0, starting at byte offset 0x100. On the ADSP-21160 side, they are mapped into MS2, starting at word offset 0x40.

5.2.5 ADSP-21160 Bus Interface

The SharcFIN interfaces to the ADSP-21160 cluster bus as a synchronous host. It sits on the ADSP-21160 bus and will request the bus to complete a PCI side initiated transfer. It also monitors the bus for any accesses to memory spaces it controls, including SDRAM, and the SharcFIN registers.

5.2.6 SDRAM Interface and Control

The SharcFIN's SDRAM controller supports up to 512 Mbytes of SDRAM. It refreshes the SDRAM and controls all of the interfacing from the ADSP-21160s to the SDRAM. In the ADSP-21160 memory space, the SDRAM is

mapped into MS0, and the ADSP-21160s have full access to all of the SDRAM.

Accessing SDRAM from the PCI Side

From the PCI side, the SDRAM is mapped into BAR4 with a 16 Mbyte window viewable at a time. Because the SDRAM is so large, this window exists to keep the entire SDRAM from being mapped into PCI memory. A SharcFIN control register (the MD Window Control register at word offset 0x4A), which provides the upper address bits for a PCI initiated SDRAM access, sets the window location.

The SDRAM window has the following two limitations:

1. Window boundaries must be crossed carefully.
2. The window register is a shared resource.

The Host Interface Library (in BittWare's DSP21k-SF Toolkit) takes care of the first limitation. The second limitation is a system issue that you must consider. Because the SharcFIN uses the window register for every PCI access to SDRAM, be careful to coordinate SDRAM accesses from PCI if you have multiple threads on the host or multiple PCI bus masters accessing the SDRAM.

SDRAM Timing from the ADSP-21160

SDRAM timing from the ADSP-21160 is synchronous, 1 wait state. A single write access takes two bus cycles. Since each additional write is single cycle, using the ADSP-21160's burst mode, you can achieve a four word burst write in five bus cycles. Reads require additional setup in the SDRAM, resulting in four bus cycles for the first access and a four word burst read in seven cycles. Because the SDRAM is page based, you will encounter additional latencies when page boundaries are crossed.

5.2.7 Peripheral Bus Interface

The peripheral bus is an 8-bit wide bus containing the Virtex-II programming CPLD. The peripheral bus is mapped into the ADSP-21160 space as MS1 and into PCI space as BAR1.

5.2.8 I²C Interface

The SDRAM and configuration EEPROM sit on an I^2C bus that is connected to the SharcFIN. The SDRAM is interrogated over the I^2C to determine its size and type so that the SDRAM configuration registers can be written. The Host Interface Library (included with BittWare's DSP21k-SF Toolkit) sets up the SDRAM on a board reset command.

The EEPROM contains factory programmed board information, including a serial number and factory build date. You can use the BittWare Configuration Manger (bwcfg) to view this information. Space for the user is also reserved in the EEPROM. The I^2C interface in the SharcFIN is the low level clock and data lines for the I^2C available in a control register. Perform all bit manipulation through software. Along with the on-board I^2C, the SharcFIN supports a second I^2C bus called the PMC I^2C, which is pinned out to the PMC+ connector.

5.2.9 Interrupt Multiplexer

The SharcFIN contains a flexible interrupt multiplexer that you can use to create complex interrupt schemes on the Remora-PMC+ board. The interrupt multiplexer contains an interrupt multiplexer for each ADSP-21160, the PCI, and the PMC. Inputs to the multiplexer are flags from the ADSP-21160, a PCI side flag, PMC interrupts, and Virtex interrupts. Outputs from the multiplexer are an interrupt line to each ADSP-21160, the PCI side, and the PMC site.

Note *The Remora-PMC+ features an interrupt multiplexer for generating PCI interrupts and one for generating PMC+ interrupts.*

How the Interrupt Multiplexer Functions

The interrupt multiplexer for each output is completely independent and can handle multiple sources. Each interrupt multiplexer consists of a 32-bit configuration register that selects the desired interrupt sources and then masks the results (see section 5.4.10 for a description of the registers). To generate an interrupt, both the flag input from the desired source and its corresponding bit in the configuration register must be high. Any other flag input and its corresponding bit in the register can also be high to generate an interrupt. The interrupt multiplexer ANDs the mask with the sources; it then ORs the result together to create the output.

Note *The interrupt multiplexer is level sensitive and does not latch interrupt sources. Therefore, the interrupt is active as long as the source is driven.*

Creating PCI Side Interrupts

To create PCI side interrupts, configure the multiplexer, which will generate the “user side” flag into the PCI side interrupt mechanism. The PCI side must then “open” the interrupt.

5.3 Overview of the SharcFIN Memory Map

The SharcFIN maps into PCI and the ADSP-21160 cluster bus. The following section provides an overview of the PCI and ADSP-21160 memory mapping of the SharcFIN. All addresses are shown as offsets from the appropriate BAR or MS. Table 5–1 gives an overview of how the SharcFIN maps to the PCI and ADSP-21160 cluster buses.

Note *The SharcFIN ASIC User's Manual contains descriptions of the registers listed in this section. Refer to it for specifics. If you cannot find sufficient information, contact BittWare for more detail.*

Table 5–1 Overview of How the SharcFIN Maps to the PCI and ADSP-21160 Buses

PCI Base Address Register	Description	21160 Memory Select	Description
BAR0	SharcFIN control registers	MS0	SDRAM
BAR1	Peripheral bus (Virtex-II programming CPLD)	MS1	Peripheral Bus (Virtex-II programming CPLD)
BAR2	ADSP-21160 MMS (Virtex-II I/O)	MS2	SharcFIN control registers
BAR3	Unused		
BAR4	SDRAM	MMS	Virtex-II

Even though the following sections list both 32-bit (word) and byte addresses, some BARs should be accessed in specific ways from the PCI side. Table 5–2 shows how to access those BARs.

Table 5–2 Accessing BAR0–BAR4 From the PCI Side

BAR	Access	Description
BAR0	Read/Write	Byte or 32-bit word accesses for all registers
BAR1	Read/Write	Byte accesses for all registers*
BAR2	Read Only Write Only	Byte or word accesses Word accesses only†
BAR4	Read Only Write Only	Byte or word accesses Word accesses only†

* Word accesses will produce erroneous data.
† Byte writes will corrupt the rest of the word.

5.3.1 Accessing System Settings and Configuration Registers

BAR0 = MS2 = system settings and configuration registers

BAR0 from the PCI interface and MS2 from the ADSP-21160 cluster bus map to system settings and configuration registers in the SharcFIN. Table 5–3 gives the PCI and ADSP-21160 offset addresses for accessing system settings and configuration registers.

Table 5–3 PCI and ADSP-21160 Addresses for System Settings and Configuration Registers

PCI 32-bit offset from BAR0	PCI byte offset from BAR0	ADSP-21160 offset from MS2	Description
0x00 – 0x5F	0x000 – 0x17F	0x00 – 0x5F	Chip control registers (PCI and ADSP-21160)

5.3.2 Accessing the Peripheral Bus

BAR1 = MS1 = Peripheral bus

BAR1 from the PCI interface maps to the peripheral bus. MS1 from the ADSP-21160 cluster bus also maps to the peripheral bus. Table 5–4 gives the PCI and ADSP-21160 offset addresses for accessing them. The Virtex-II programming CPLD is at offset 0x200000 on this bus.

Note *BAR1 must be accessed a byte at a time from the PCI side. Word accesses will produce erroneous data.*

Table 5–4 PCI and ADSP-21160 Addresses for Peripheral Bus

PCI byte Offset from BAR1	ADSP-21160 offset from MS1	Description
0x000000 – 0x1FFFFF	0x000000 – 0x1FFFFF	Reserved
0x200000 – 0x20000F	0x200000 – 0x20000F	Virtex-II programming CPLD
0x200010 – 0x2FFFFF	0x200010 – 0x3FFFFF	Reserved
0x300000 – 0x3FFFFF	0x400000 – 0x5FFFFF	Peripheral Bus

5.3.3 Accessing Multiprocessor Memory Space

BAR 2 = MMS = flat map of Multiprocessor Memory Space

BAR2 from the PCI and MMS from the ADSP-21160 cluster bus allow access to the ADSP-21160 multiprocessor memory space. Table 5–5 gives the PCI and ADSP-21160 offset addresses for accessing the MMS.

Note *On the Remora-PMC+, the Virtex-II occupies the space for the ADSP-21160-1. Refer to Appendix A for descriptions of the Virtex-II registers and software examples.*

Table 5–5 PCI and ADSP-21160 Addresses for Multiprocessor Memory Space

PCI 32-bit offset from BAR2	PCI byte offset from BAR2	ADSP-21160 address	Description
0x000000 – 0x0FFFFF	0x0000000 – 0x03FFFFF	0x000000 – 0x0FFFFF	Reserved
0x100000 – 0x1FFFFF	0x0400000 – 0x07FFFFF	0x100000 - 0x1FFFFF	21160-1 MMS space (Virtex-II FPGA)
0x200000 – 0x2FFFFF	0x0800000 – 0x0BFFFFF	0x200000 – 0x2FFFFF	21160-2 MMS space*
0x300000 – 0x3FFFFF	0x0C00000 – 0x0FFFFFF	0x300000 – 0x3FFFFF	21160-3 MMS space*
0x400000 – 0x4FFFFF	0x1000000 – 0x13FFFFF	0x400000 – 0x4FFFFF	21160-4 MMS space*
0x500000 – 0x7FFFFF	0x1400000 – 0x1FFFFFF	0x500000 – 0x7FFFFF	Reserved

* Unused on the Remora-PMC+.

5.3.4 Accessing SDRAM

BAR 4 = MS0 = SDRAM[1]

You can see a window of 16 Mbytes of SDRAM from the PCI bus. The SD Window register allows you to select which 16 MB window is currently visible. The register is located at word/ADSP-21160 offset 0x4A in BAR0/MS2. Table 5–6 gives the PCI and ADSP-21160 offset addresses for

accessing a 64 MB bank of SDRAM, and Table 5–7 on page 64 gives the addresses for a 128 MB bank.

Note *The addresses listed in Table 5–6 and Table 5–7 only apply to the given 64 MB and 128 MB SDRAM cases.*

Table 5–6 PCI and ADSP-21160 Addresses for 64 MB SDRAM

SD Window Register value*	PCI 32-bit offset from BAR4	PCI byte offset from BAR4	ADSP-21160 offset from MS0	Description
0x02	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x800000 – 0xBFFFFF	First 16 MB block of SDRAM
0x03	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0xC00000 – 0xFFFFFF	Second 16 MB block of SDRAM
0x00	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x1000000 – 0x13FFFFF	Third 16 MB block of SDRAM
0x01	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x1400000 – 0x17FFFFF	Fourth 16 MB block of SDRAM

* Window register values of 0x00 and 0x01 always access the last two 16 MB windows of SDRAM. Refer to section 5.2.6 for details.

1. Some caveats apply. Refer to the SFIN-160 SharcFIN ASIC User's Guide for complete details.

Table 5–7 PCI and ADSP-21160 Addresses for 128 MB SDRAM

SD Window Register value*	PCI 32-bit offset from BAR4	PCI byte offset from BAR4	ADSP-21160 offset from MS0	Description
0x02	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x800000 – 0xBFFFFF	First 16 MB block of SDRAM
0x03	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0xC00000 – 0xFFFFFF	Second 16 MB block of SDRAM
0x04	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x1000000 – 0x13FFFFF	Third 16 MB block of SDRAM
0x05	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x1400000 – 0x17FFFFF	Fourth 16 MB block of SDRAM
0x06	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x1800000 – 0x1BFFFFF	Fifth 16 MB block of SDRAM
0x07	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x1C00000 – 0x1FFFFF	Sixth 16 MB block of SDRAM
0x00	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x2000000 – 0x23FFFFF	Seventh 16 MB block of SDRAM
0x01	0x00000 – 0x3FFFFF	0x000000 – 0xFFFFFF	0x2400000 – 0x27FFFFF	Eighth 16 MB block of SDRAM

* Window register values of 0x00 and 0x01 always access the last two 16 MB windows of SDRAM. Refer to section 5.2.6 for details.

5.4 Setting the SharcFIN User-Configurable Registers

The SharcFIN has two sets of registers. One set, the PCI configuration registers, configures the PCI interface. The other set, the chip control registers, configures both the PCI and SHARC interfaces. The PCI configuration registers are only accessible by PCI (configuration access) and are documented in the SharcFIN ASIC User's Manual. The chip control registers are broken into two groups: the PCI control registers (which are documented in the SharcFIN ASIC User's Manual) and the SHARC interface control registers; both groups are accessible by PCI (BAR0) and the ADSP-21160 cluster bus (MS2).

This section describes the memory locations and settings for the SharcFIN's SHARC interface control registers. All addresses described in this section are 32-bit addresses and are accessible from the ADSP-21160 side (via MS2) and from the PCI interface (via BAR0). Table 5–8 on page 66 gives the memory mapping for the SHARC interface control registers in the SharcFIN.

Note *Most of the SHARC interface control registers are already set and do not require you to program them. You will only need to set them if you are writing your own host interface programs.*

Note *SharcFIN registers that are unused on the Remora-PMC+ are grayed-out in Table 5–8 on page 66.*

Table 5–8 Memory Map for the SharcFIN User-Configurable Registers

Address	Register	Type	Description
0x40	Address Override	R/W	Allows addressing of IOP registers when ADSP-21160 is using a host packing mode
0x41	Status	R/O	General set of status registers. Indicates ADSP-21160 cluster bus status, and last reset source
0x42	Peripheral Bus Configuration	R/W	Configures and shows status of wait cycles of the 8-bit peripheral bus
0x43	Watchdog Configuration	WORM*	Enables and disables the watchdog timer
0x44	PMC+ Configuration	R/W	Configures the PMC+ interface
0x45	SD Size Config	R/W	Resets and reinitializes the SDRAM controller
0x46	Onboard I^2C Control	R/W	Controls the I^2C interface
0x47	PMC I^2C Control	R/W	Controls the I^2C interface to the PMC+ interface
0x48	DMA Address	R/W	Sets the address of DMA to be performed
0x49	DMA Configuration	R/W	Controls various features of the SharcFIN DMA engine: size of DMA, increment size of DMA, other various configuration bits
0x4A	SD Window	R/W	Selects which 16 MB of SDRAM the PCI interface will view
0x4B–4F	Unused		
0x50, 0x51	H1I0, H1I	R/W	Configure ADSP-21160 DSPs' interrupts; show status of interrupts
0x51, 53, 55, 57	Reserved		
0x58	PCInt	R/W	Configures PCI interrupts
0x5A	PMCI0	R/W	Configures PMC interrupts
0x59, - 0x5B–5D	Unused		
0x5E	Flag Status	R/O	Shows state of all flags
0x5F	IRQ Status	R/O	Shows state of all interrupts

* Write Once Read Many

5.4.1 Address Override Register

The Address Override register (0x40) configures how the ADSP-21160 DSPs access the least significant 32 bits on the ADSP-21160 cluster bus. It allows access to the DSPs' IOP space before the SYSCON register has been configured.

Note *Only use this register if you are writing your own host interface programs.*

Table 5–9 Address Override Register Description

Bit	Name	Type	Reset Value	Function
0	A0 Override En*	R/W	0	Address override enable for booting across the PCI bus
1	Overridden A0	W/O	0	Overridden address
2	BusLockReq	W/O	0	Bus Lock Request. Requests the SharcFIN to acquire the ADSP-21160 cluster bus and locks access to the bus so that only the SharcFIN can access it. 0 = Disabled 1 = Requests that the SharcFIN acquire the ADSP-21160 cluster bus and not give it up
3	Destructive FIFO Read Enable	W/O	1	Determines whether a read to the DMA FIFOs will cause the FIFOs to advance 0 = A read to the DMA FIFOs does not cause the FIFOs to advance 1 = A read to the DMA FIFOs causes the FIFOs to advance
4	Host Clock Disable	R/W	0	Disables the clock to the ADSP-21160 DSPs. This is used to address powerup anomalies on current editions of ADSP-21160 processors. 1 = Clock disabled 0 = Clock enabled

* Do not use the Address Override bits (B0 and B1) under normal setup conditions. If you are running the board in standalone mode and are booting across the PCI bus, you can change these bits. However, exercise extreme caution since data loss or corruption will occur if you set the bits improperly.

5.4.2 Status Register

The Status register (offset 0x41) is a 16-bit read only register that gives information about various features on the board.

Table 5–10 Contents of the Status Register

Bit	Name	Type	Reset Value	Function
0	PMCHostCfg	R/O	Set in hardware	Indicates whether SharcFIN is configured for a PMC host site or a PMC daughter card. 1 = On baseboard 0 = On PMC
1	StandAlone	R/O	Set by jumper	Determines whether PCI side resets are accepted by the SharcFIN 1 = PCI resets ignored 0 = PCI resets accepted
2	Bus Locked	R/O	0	Indicates whether the SharcFIN has locked and acquired the ADSP-21160 cluster bus 0 = Cluster bus is not locked 1 = Cluster bus is locked
3	Last Reset Source	R/O	0	Indicates whether the PCI interface or the watchdog was the source of the last board reset 0 = PCI reset 1 = Watchdog/external reset
4	SpareInput Pin	R/O	1	Spare external input signal

5.4.3 Peripheral Bus Configuration Register

The Peripheral Bus Configuration register (offset 0x42) allows you to configure the wait cycles of the Reef-PMC+'s 8-bit peripheral bus. Table 5–11 shows the contents of the Peripheral Bus Configuration register.

Table 5–11 Contents of the Peripheral Bus Configuration Register

Bit	Name	Type	Reset Value	Function
3:0	PCI to Pbus Wait	R/W	0101 B0: 1 B1: 0 B2: 1 B3: 0	Select the number of wait cycles the SharcFIN will wait before completing a transaction on the peripheral bus. The actual value of wait cycles is one greater than the value in the register (for example, if the register value = 0, the number of wait cycles = 1). 0101 =Default setting (6 wait cycles)
4	Pbus Ack Enable	R/W	0	Selects whether the SharcFIN will monitor the peripheral bus Ack line after the peripheral bus wait time has expired. 0 = SharcFIN will wait the selected number of wait cycles and consider the transaction complete* 1 = SharcFIN will wait the selected number of wait cycles and then monitor the Ack line
5	Pbus Reset	R/W	0	Resets the peripheral bus reset line, the CPLD. The reset stays active until cleared by another write to the register.† 0 = No reset 1 = Resets CPLD, and all devices on the peripheral bus

* Five wait cycles is the minimum amount of wait cycles required to talk to the Flash memory.

† You can also reset the Virtex-II, and all devices on the peripheral bus via a board reset.

5.4.4 PMC+ Configuration Register

The PMC Configuration register (offset 0x44) is a read/write register that configures the bus mode lines of the PMC+ interface and allows you to read their status. Table 5–12 below shows the contents of the PMC Configuration register.

Table 5–12 Contents of the PMC Configuration Register

Bit	Name	Type	Reset Value	Function
0	PMC Flg/Int En	R/W	0	Configures the PMC+ interface's bus mode lines to be used as flag interrupts.* 1 = The PMC+ interface's bus mode lines will be used as flag interrupts 0 = The PMC+ interface's bus mode lines will be used as bus mode lines
3:1	BusMode2, BusMode3, BusMode4	R/W	BusMode2: 1 BusMode1: 0 BusMode3: 0	Tells the PMC site whether or not it should drive BusMode1 to indicate its presence. When the flag interrupts are disabled (by *PMC Flg/Int En* bit), the BusMode lines work according to the PMC specification. Bits 1–3 are Bus Mode lines 2–4. B1 = 1 Bus Mode line 2 B2 = 0 Bus Mode line 3 B3 = 0 Bus Mode line 4
4	BusMode 1	R/O		Bus Mode line 1 is an input†. It indicates that a PMC card is present on the board.‡ 0 = PMC board is present
5	BusMode2	R/O		Current status of BusMode2 line
6	BusMode3	R/O		Current status of BusMode3 line
7	BusMode4	R/O		Current status of BusMode4 line

* The option of using the bus mode lines as flag interrupts is a feature of BittWare's PMC+ form factor; to work properly, it must be enabled on both the PMC+ card and the host board.

† Bits 3:1 are outputs. Bit 4 is an input.

‡ Refer to the *IEEE P138.1 Standard Physical and Environmental Layers for PCI Mezzanine Cards: PMC* (PMC Specification) for details on the operation of these lines.

5.4.5 SDRAM Configuration Registers

The SharcFIN contains two registers that configure the SDRAM:

- SDRAM Size Configuration Register (offset 0x45)
- SDRAM Window Register (offset 0x4A)

SDRAM Size Configuration Register

The SDRAM Size Configuration register sets the size of the SDRAM. The settings for this register depend on the type of SDRAM modules used on the DSP board. Table 5–13 below shows the contents of the register.

Table 5–13 Contents of the SDRAM Size Configuration Register

Bit	Name	Type	Reset Value	Function
1:0	SD Bank Size 1:0	R/W	0	Determine how the SDRAM controller uses the SharcFIN's CS0 and CS1 (chip select 0 and 1) pins. The chip select pins allow the SharcFIN to seamlessly connect to two banks of SDRAM. CS0 selects SDRAM bank 0, and CS1 selects SDRAM bank 1. **B0 B1 Result** 0 0 CS0 always active 0 1 CS1 active when 32-bit word address bit 24 is high; otherwise CS0 is active 1 0 CS1 active when 32-bit word address bit 25 is high; otherwise CS0 is active 1 1 CS1 active when 32-bit word address bit 26 is high; otherwise CS0 is active
2	SD RF Size	R/W	1	Sets the refresh rate of the SDRAM 0 = 4K refreshes every 64 milliseconds 1 = 8K refreshes every 64 milliseconds
3	SD Reset	W/O	0	Writing a 1 resets the SDRAM controller and reinitializes the SDRAM

SDRAM Window Register

The SDRAM Window register (offset 0x4A) is a 5-bit register that lets you select which 16 MB section of memory in the SDRAM to view from the host over the PCI interface. From the PCI side, the SDRAM is mapped into BAR4 with a 4 Mword (16 Mbyte) window viewable at a time. The offset into BAR4 provides bits 0 through 21 of the address of the SDRAM word to be accessed. The contents of the 5-bit SDRAM Window register are appended to bits 22 through 26 of the address to complete the address and thereby select the 4 Mword window to be accessed. Table 5–14 lists the bits included in this register.

Note *You will not need to configure this register since the Diag21k utility, which is included with the DSP21k-SF Toolkit, will set these bits.*

Table 5–14 Contents of the SDRAM Window Register

Bit	Name	Type	Reset Value	Description
0	Window A0	W/O	0	Selects window A0 of the SDRAM
1	Window A1	W/O	1	Selects window A1 of the SDRAM
2	Window A2	W/O	0	Selects window A2 of the SDRAM
3	Window A3	W/O	0	Selects window A3 of the SDRAM
4	Window A4	W/O	0	Selects window A4 of the SDRAM

5.4.6 Onboard I^2C Control Register

The Onboard I^2C Control register (offset 0x46) controls the I^2C interface. The I^2C interface is a two-wire bus; one wire is a clock signal and the other is a data signal. Both the clock and the data lines are pulled up. Table 5–15 on page 73 shows the contents of the register.

As per standard I^2C, both the clock and data lines are pulled high. Devices on the I^2C bus either do not drive the bus or they drive it low. Any device

on the I^{2}C bus can drive either the clock or data line low when required. You can also read the actual status of the lines.

Table 5–15 Contents of the I^{2}C Control Register

Bit	Name	Type	Reset Value	Description
0	Clock	R/W	1	On write, drives the clock line. On read, shows the state of the clock line.
1	Data	R/W	1	On write, drives the data line. On read, shows the state of the data line.
2	Clock Drive	R/O	1	
3	Data Drive	R/O	1	

When you write a 1 to either the clock or data line in this register, the SharcFIN does not drive the corresponding line. When you write a 0 to either the clock or the data line, the SharcFIN drives the corresponding line to 0. When you read either line, you read the actual state of the line rather than what you have written to it. If you are not driving the line, it will be 0 if another device is driving it and 1 if nothing is driving it. Table 5–16 below shows the effect of the values written to the Clock and Data bits.

Table 5–16 Effects of Values Written to the Clock and Data Bits (B0, B1)

Value Written	Description
0	Drives the line low; when read back, shows 0
1	When read back, shows the actual state of the I^{2}C line

5.4.7 Setting the PMC I^{2}C Control Register

The PMC I^{2}C Control register (offset 0x47) controls the I^{2}C interface to the PMC+ interface. The settings for this register are the same as the settings for the Onboard I^{2}C Register, except that all settings apply to the PMC+ interface I^{2}C instead of the on-board I^{2}C.

5.4.8 DMA Address Register

The DMA Address register (offset 0x48) configures the address of the current DMA location. This register is incremented as the DMA progresses, allowing you to monitor the DMA engine's current address. You must reset this register each time if you wish to repeat a DMA on the same address range as last time. This register cannot be written to while the DMA start bit is set, but it can be read from at any time.

Table 5–17 Contents of the DMA Address Register

Bit	Name	Type	Reset Value	Function
0	Unused	R/O	0	
27:1	A[27:1]	R/W	0	Indicates the current DMA location
31:28	Unused	R/O	0	

5.4.9 DMA Configuration Register

The DMA Configuration register (offset 0x49) controls various features of the SharcFIN DMA engine, including starting a DMA, size of the DMA, increment size of the DMA, and other various configuration bits. Table 5–18 on page 74 describes the contents of the register. The *SharcFIN ASIC User's Manual* explains this register in more detail. This register can not be written while the DMA start bit is set, but it can be read from at any time.

Table 5–18 Contents of the DMA Configuration Register

Bit	Name	Type	Reset Value	Description
15:0	DMACnt B[15:0]	R/W	X*	DMA transfer count (in 64-bit words) bits. All are settable.
22:16	DMA Stride B[6:0]	R/W	X	Stride or address increment bits. These bits will typically be set to 0x01.
23	Unused			Unwritable; fixed at 0.
(Sheet 1 of 3)				

Table 5–18 Contents of the DMA Configuration Register (Continued)

Bit	Name	Type	Reset Value	Description
24	DMAStart	R/W	0	Setting the bit to 1 starts the DMA; it resets to 0 when the DMA is complete.
25	DMA Channel Select	R/W	0	Selects the PCI channel being operated on (0 or 1)
26	DMA Direction	R/W	0	Selects whether a PCI transmit or receive DMA is being performed. 0 = PCI receive DMA (PCI to 21160/ SDRAM) 1 = PCI transmit DMA (21160/SDRAM to PCI)
27	DMA Interrupt	R/W	0	If this bit is set at the start of the DMA, The SharcFIN generates an interrupt in the interrupt multiplexer on completion of the DMA. Any write to this register clears the interrupt.
28	Burst Disable	R/W	0	1 = Disable bursting on the ADSP-21160 side. Must be set to 1 when the address increment > 0x01. If it is set when the address increment <= 0x01, it functions but will slow things down. 0 = Bursting enabled
29	DMA Buslock		0	1 = SharcFIN requests the ADSP-21160 cluster bus when the start bit is set, and once it obtains the bus, keeps it until the DMA completes.
(Sheet 2 of 3)				

Table 5–18 Contents of the DMA Configuration Register (Continued)

Bit	Name	Type	Reset Value	Description
31:30	DMA Xfer Length B[1:0]		00	Set the DMA transfer length. These bits must be set along with the bits in the PCI control register at 0x1A and 0x1B (0x68 byte address). **B30 B31 Result** 0 0 Data transferred during each access of the bus = eight 64-bit words 1 0 Data transferred during each access of the bus = sixteen 64-bit words 0 1 Data transferred during each access of the bus = thirty-two 64-bit words 1 1 Data transferred during each access of the bus = sixty-four 64-bit words Corresponding Receive FIFO almost empty or Transmit FIFO almost full flags must be set with corresponding value (length –1, so for B30, B31 must be set to 7).
(Sheet 3 of 3)				

* X = Unknown

5.4.10 Interrupt Configuration Registers

The SFIN-160 SharcFIN features an interrupt multiplexer for four ADSP-21160s, the PCI interface, and the PMC+ interface.[1] Inputs to the multiplexers are flags from the ADSP-21160s, a PCI side flag, PMC flags, UART flags, a DMA flag, and a peripheral bus flag[2]. The registers at offsets 0x50 to 0x5A (see Table 5–19 on page 78) provide the interrupt multiplexers (see the *SharcFIN ASIC User's Manual* for additional details on the interrupt multiplexer).

On the Remora-PMC+ board, the Virtex-II is connected to the Flag0 and Flag1 signals for ADSP-21160-1 in the SharcFIN. The "done" output of the Virtex-II, which indicates that the Virtex-II has been configured, is connected as the Flag0 signal for ADSP-21160-2 in the SharcFIN. The flags are tied to the interrupt logic in the Virtex-II (Chapter 6). You can use the PCI interrupt multiplexer for PCI interrupts or the PMC interrupt multiplexer for PMC+ interrupts. Note that you can only use the PMC+ interrupts if the Remora-PMC+ is on a BittWare PMC+ baseboard.

Table 5–20 on page 78 lists the settings for the PCI interrupt multiplexer, and Table 5–21 on page 78 lists the settings for the PMC+ interrupt multiplexer.

The interrupt multiplexer registers are 32-bit registers that allow you to select the desired input sources. The first 16 bits (15:0) are read/write and select the source that will generate an interrupt to the processor; each of the bits corresponds to one of the flag inputs to the multiplexer. The second 16 bits (31:16) are read only and show which of the enabled interrupts are generating an interrupt, each bit corresponding to one of the flag inputs. Bits 31:16 are masked interrupt lines; when one of the flag inputs and its corresponding bit in bits 15:0 of the configuration register is high, the corresponding bit in bits 31:16 is also set to indicate the source of the input. Bits 31:16 are masked by 21160-1 IRQ0's interrupt mask.

1. On the Remora-PMC+, only the PCI and PMC+ interrupt multiplexers are used.
2. On the Remora-PMC+, the only inputs to the interrupt multiplexers are from the Virtex-II.

Table 5–19 SharcFIN Interrupt Configuration Registers

Address	Register	Description
0x50 – 0x57	Unused	
0x58	PCInt	Configures the direction of the PCI interrupt
0x59	Unused	
0x5A	PMCI0	Configures the direction of PMC+ IRQ0

Table 5–20 Contents of the PCI Interrupt Configuration Register (0x58)

Bit	Name	Type	Reset Value	Description*
0	H1F0	R/W	1	Enables Virtex-II flag0 to cause an interrupt
1	H1F1	R/W	1	Enables Virtex-II flag1to cause an interrupt
2	H2F0	R/W	1	Enables Virtex-II Done flag to cause an interrupt
31:3	Unused			

*All descriptions in this column apply when bits are set to 1.

Table 5–21 Contents of the PMC+ Interrupt Configuration Register (0x5A)

Bit	Name	Type	Reset Value	Description*
0	H1F0	R/W	1	Enables Virtex-II flag0 to cause an interrupt
1	H1F1	R/W	1	Enables Virtex-II flag1to cause an interrupt
2	H2F0	R/W	1	Enables Virtex-II Done flag to cause an interrupt
31:3	Unused			

*All descriptions in this column apply when bits are set to 1.

5.4.11 Flag and Interrupt Status Registers

The registers at offsets 0x5E and 0x5F are 16-bit unmasked registers that show the status of all flags and interrupts. The register at 0x5E shows the status of the flags, and 0x5F shows the status of the interrupts. Table 5–22 on page 79 and Table 5–23 on page 80 describe the bits in the registers (unused bits are grayed out). For additional explanation of these registers, refer to the *SharcFIN ASIC User's Manual*.

Table 5–22 Contents of the Flag Status Register

Bit	Name	Type	Description
0	H1F0	R/O	Status of Virtex-II FLAG0
1	H1F1	R/O	Status of 21160-1 FLAG1
2	H2F0	R/O	Status of Virtex-II FLAG0
3	H2F1	R/O	Status of 21160-2 FLAG1
4	H3F0	R/O	Status of 21160-3 FLAG0
5	H3F1	R/O	Status of 21160-3 FLAG1
6	H4F0	R/O	Status of 21160-4 FLAG0
7	H4F1	R/O	Status of 21160-4 FLAG1
8	PCFlg	R/O	Status of PCI flag
9	PMCFlg0	R/O	Status of PMC+ FLAG0
11	PRFlg	R/O	Status of peripheral bus flag
12	UART0	R/O	Status of UART0 flag
13	UART1	R/O	Status of UART1 flag
14	DMAInterrupt	R/O	Status of DMA interrupt
15	Unused		

Table 5–23 Contents of the Interrupt Status Register

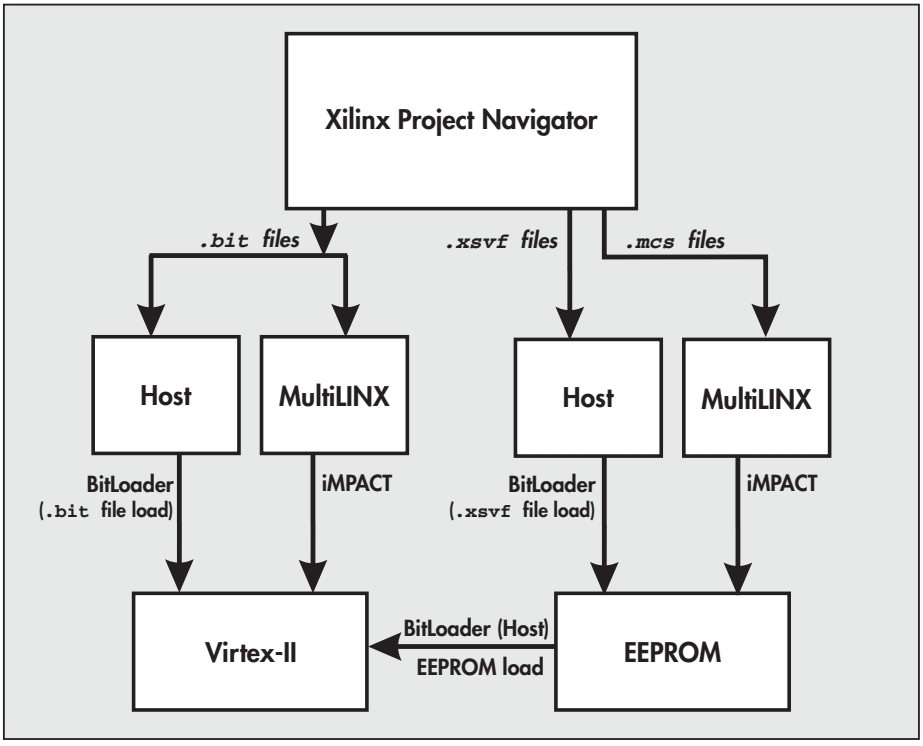
Bit	Name	Type	Description
0	H1I0	R/O	Status of 21160-1 IRQ0
1	H1I1	R/O	Status of 21160-1 IRQ1
2	H2I0	R/O	Status of 21160-2 IRQ0
3	H2I1	R/O	Status of 21160-2 IRQ1
4	H3I0	R/O	Status of 21160-3 IRQ0
5	H3I1	R/O	Status of 21160-3 IRQ1
6	H4I0	R/O	Status of 21160-4 IRQ0
7	H4I1	R/O	Status of 21160-4 IRQ1
8	PCInt	R/O	Status of PCI interrupt
9	PMCI0	R/O	Status of PMC+ IRQ0
15:10	Unused		

Chapter 6

Configuring the Virtex-II FPGA

This chapter describes the steps involved in configuring the Virtex-II FPGA. It does not instruct how to create the FPGA program, but rather how to load the logic onto the FPGA. We assume that you are familiar with the FPGA software tools, including the XILINX Project Navigator and iMPACT loader utility. You will also need to refer to the *Virtex-II Platform FPGA Handbook* and the *MultiLINX Universal Cable Transceiver User's Guide* both by XILINX. The diagram below illustrates the file configuration methods, and the sections that follow describe each method in detail.

Figure 6–1 Overview of the FPGA-Logic File Configuration Methods



6.1 Loading Files with the BitLoader Utility

This section describes the two ways you can control the FPGA-logic using BittWare's BitLoader utility:

- load files directly to the Virtex-II FPGA from the host via the PCI bus
- load the EEPROM's existing FPGA-logic files to the Virtex-II

6.1.1 Getting Ready to Load from the Host

Before you load files to the FPGA, complete the following steps.

1. The configuration switches for Remora-PMC+ are set at the factory to boot from the EEPROM. As the software can override the configuration switch settings, it is not necessary for you to change them.

2. Install the BitLoader utility.
   To install the BitLoader utility, follow the instructions given in "Installing the BittWare BitLoader Utility" on page 19.

3. Create the FPGA-logic.
   To write the FPGA-logic, use the XILINX Project Navigator tools to create a .bit file. Refer to the XILINX Project Navigator user's manual for details on creating this type of file.

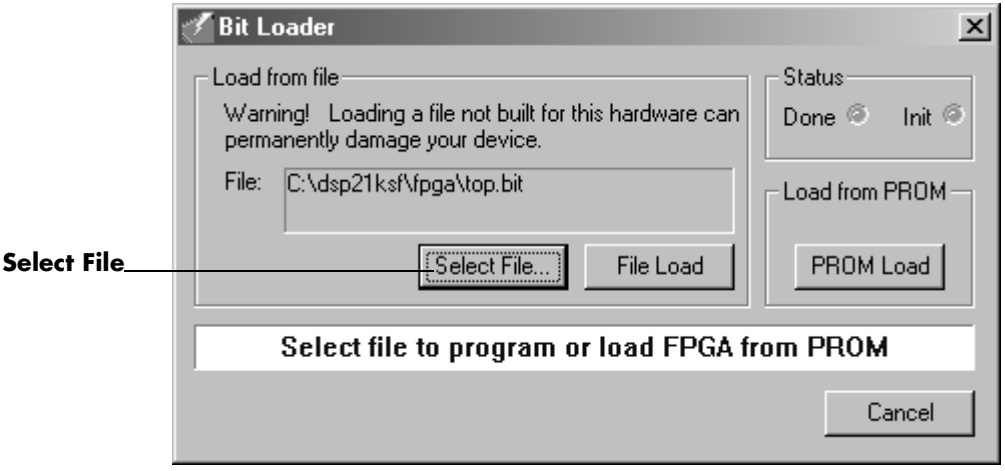
6.1.2 Loading the Virtex-II from the Host with BitLoader

This section describes the steps for loading a .bit file into the FPGA using the BittWare BitLoader. This utility loads the .bit file directly from the host to the Virtex-II FPGA via the PCI bus.

To load the FPGA .bit file with BitLoader,

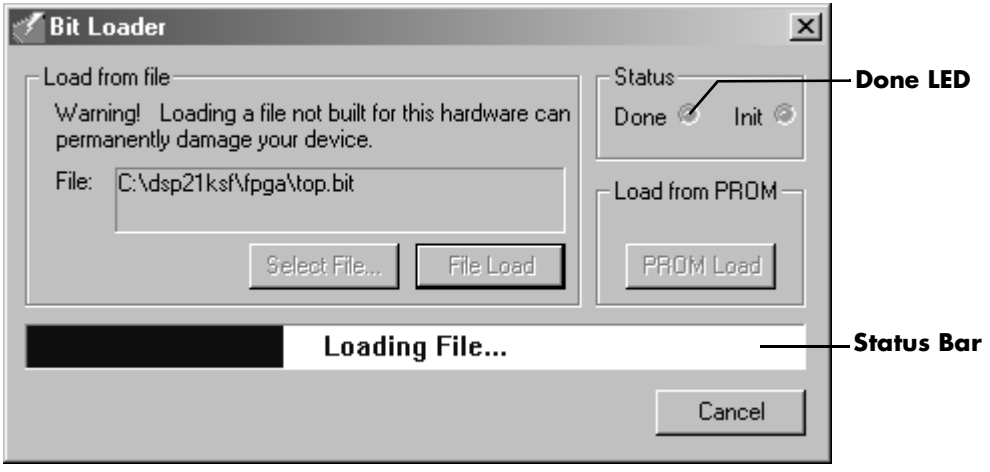
1. Install and run the BitLoader utility as instructed in section 6.1.1. The BitLoader main menu will appear. Click on the **Select Fil e** button to choose the .bit file to be loaded.

Figure 6–2 Selecting a File to Load



2. Select the .bit file from the browser window and click **Open**. Click on **File Load** in the Virtex Utility window. The status bar will reflect the file loading status, and the **Done** LED will light up once the file load is complete.

Figure 6–3 Loading the File



3. If BitLoader does not successfully load on the first attempt, it will automatically try again up to two times.

Figure 6–4 Timeout Error

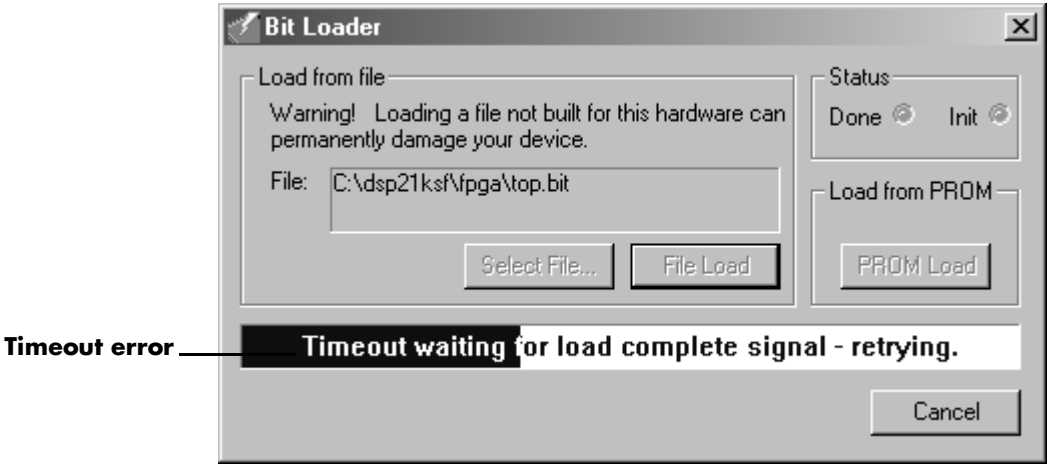
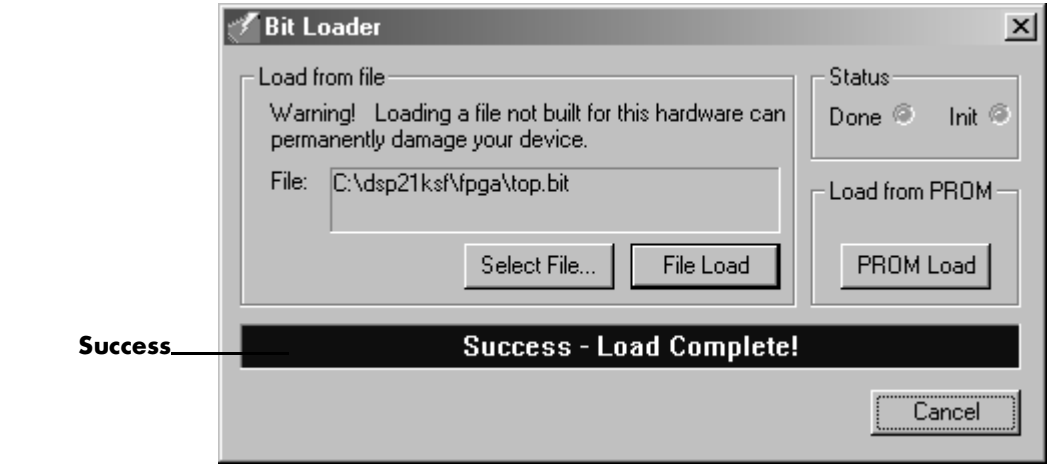


Figure 6–5 File Loaded Successfully



6.1.3 Loading the EEPROM from the Host with BitLoader

There are two steps in loading the EEPROM from the host:

- Create the .xsvf file using Xilinx's iMPACT
- Load the .xsvf file using BitLoader

Create or Obtain the .xsvf files with iMPACT

SVF files contain both programming and configuration data and are used to perform boundary scan operations. Instead of physically programming a device over a Xilinx cable, the JTAG output can be re-routed to a .svf file. This compressed recording of an iMPACT programming session becomes the .xsvf file.

We assume that you are already familiar with .svf file creation using the Xilinx iMPACT tools (refer to the *iMPACT User Guide* by Xilinx). This section explains how to create the .xsvf file that the host uses to program the EEPROM via the PCI bus.

To create the `.xsvf` file with iMPACT,

1. Follow the directions in the *iMPACT User Guide* by Xilinx to create and capture the .svf file.
2. Use the BitLoader svf2xsvf.exe tool to convert the .svf file to a .xsvf file.

To obtain the `.xsvf` file

Contact BittWare technical support for the latest .xsvf files available for download.

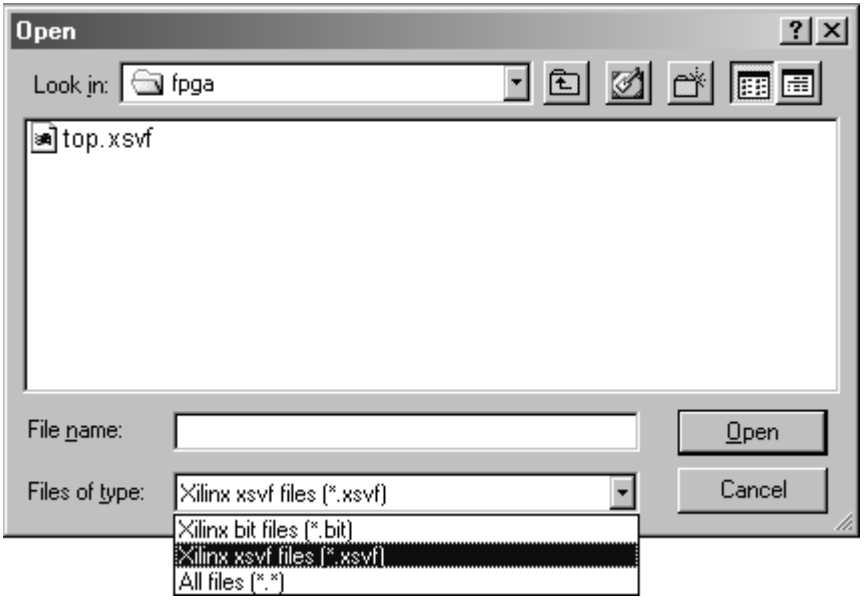
Load the .xsvf files with BitLoader

Once the .xsvf file has been created, BitLoader is used to download the file from the host to the EEPROM via the PCI bus.

To load the `.xsvf` file with BitLoader,

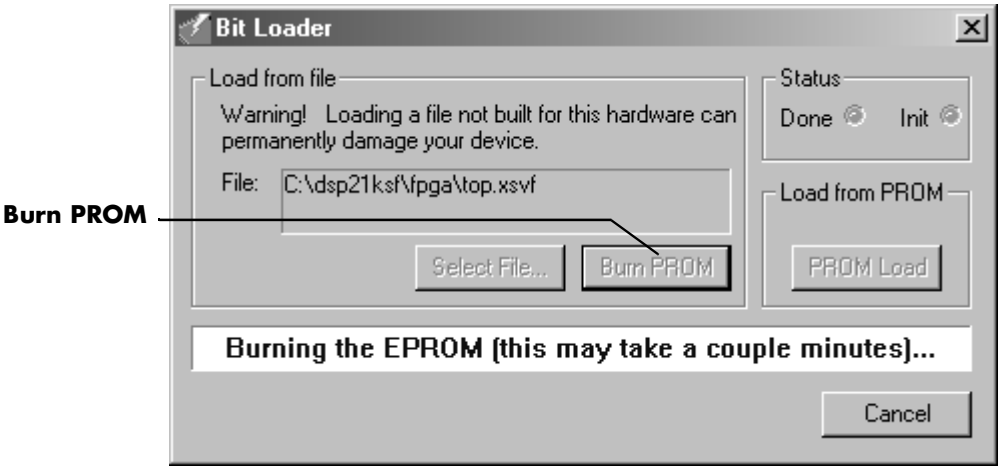
1. The configuration switches for Remora-PMC+ are set at the factory to boot from the EEPROM. As the software can override the configuration switch settings, it is not necessary for you to change them..
2. Install and run the BitLoader utility as instructed in section 6.1.1. The BitLoader main window will open. Choose the **Select File** button to select the .xsvf file to be loaded. Note that you may need to select the **Files of type** drop-down list to change the view to .xsvf files.

Figure 6–6 Selecting a .xsvf File to Load



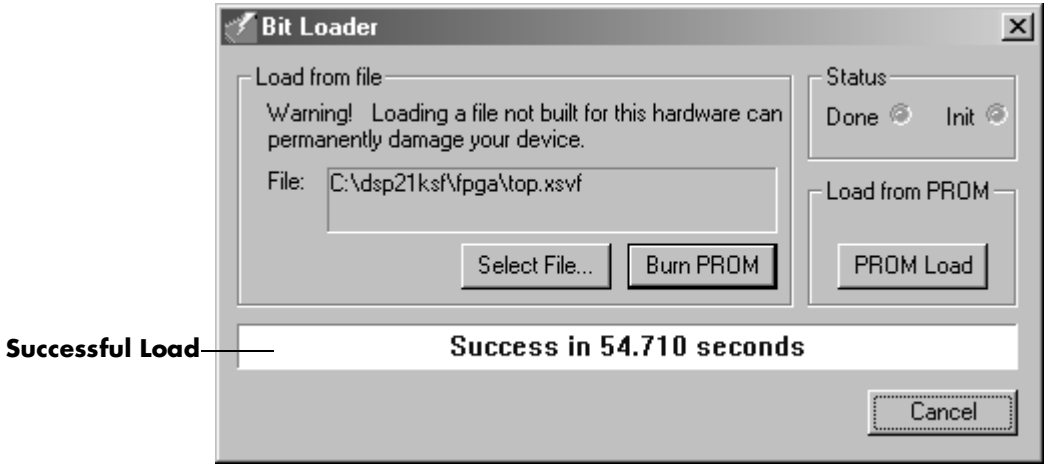
3. Select the .xsvf file from the browser window and click **Open**. The **File Load** button on the BitLoader window will change to **Burn PROM**. Click the **Burn PROM** button to load the .xsvf file. The status bar will reflect the file loading status.

Figure 6–7 PROM File Burned Successfully



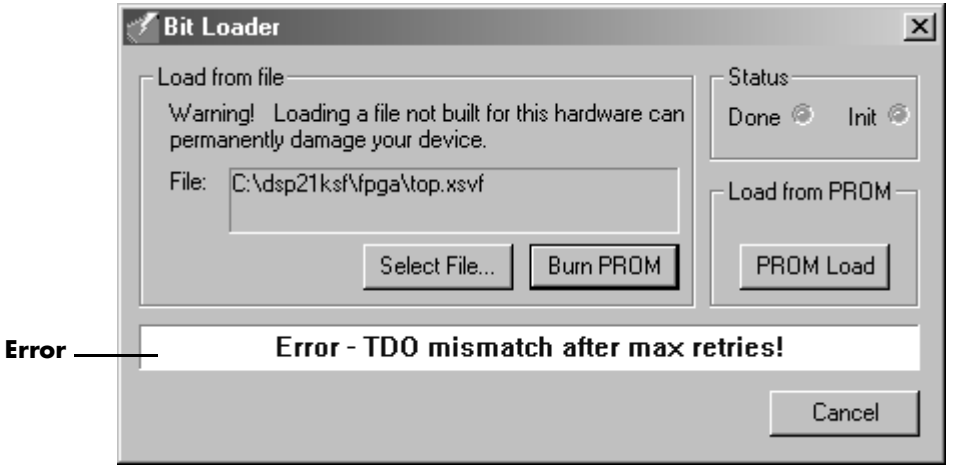
Upon completion of the file load, a success message will be displayed.

Figure 6–8 EEPROM File Load Successful



Note *In the event that the CPLD-JTAG control configuration switch S1 dip 4 is set to OFF, the file load process will fail and the "TDO mismatch" error message will be displayed in the status bar, as shown in Figure 6–9. For more details, see "Setting the PCI Bus Speed and PLD-JTAG Control (S1)" on page 13*

Figure 6–9 Error Loading File



6.1.4 Loading the Virtex-II from the EEPROM with BitLoader

There are two ways to program the FPGA directly from an .mcs file already loaded on the EEPROM:

- Run BittWare's BitLoader utility while the board is in operation.
- Load automatically on board power-up.

Loading the File with the BittWare BitLoader Utility

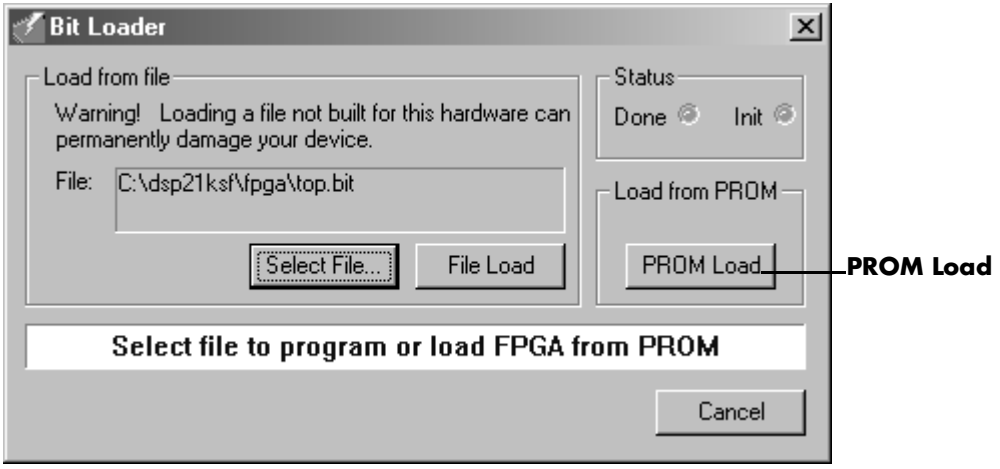
If you have already loaded your FPGA-logic file to the EEPROM, BitLoader can download the file to the Virtex-II while the board is in operation.

To load the FPGA .mcs file with BitLoader,

1. Install and run the applicable BitLoader utility as instructed in section 6.1.1.

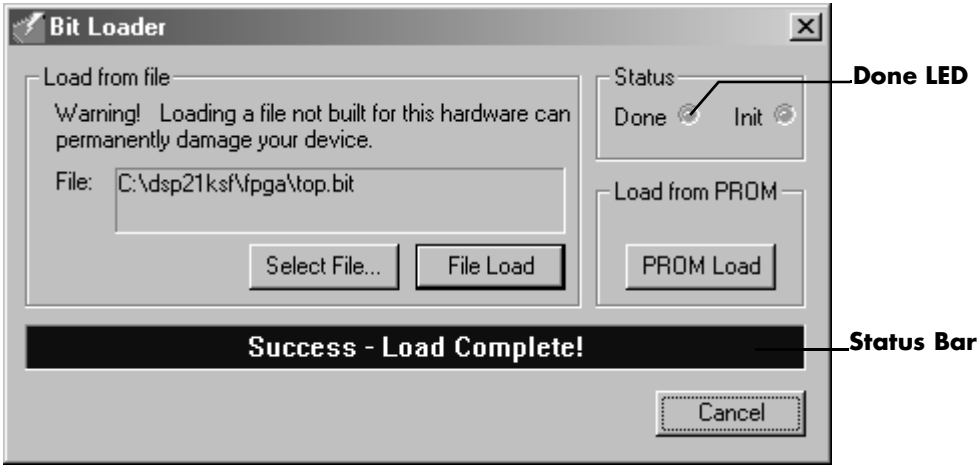
2. Click the **PROM Load** button on the BitLoader menu to load the existing FPGA-logic file from the EEPROM to the FPGA.

Figure 6–10 Loading a File with the PROM Load Button



3. The status bar will reflect the file loading status, and the **Done** LED will light up once the file load is complete.

Figure 6–11 EEPROM Load Successful.



Loading the FPGA File on Start-Up

To automatically load the FPGA-logic file from the EEPROM on power-up, set S2 to EEPROM loading (see Table 2–2 and "Selecting the Default Virtex Configuration Mode (S2)" on page 13). When the board is powered up, the boot process will cause the EEPROM to load the FPGA.

6.2 Loading Files with the MultiLINX Cable

This section describes how to load FPGA files via the optional MultiLINX cable. The MultiLINX is an external device that allows you to load files onto the board as well as debug the Virtex-II. It works in conjunction with the Project Navigator suite. There are two destinations for files downloaded to the board via the MultiLINX device:

- EEPROM
- Virtex-II

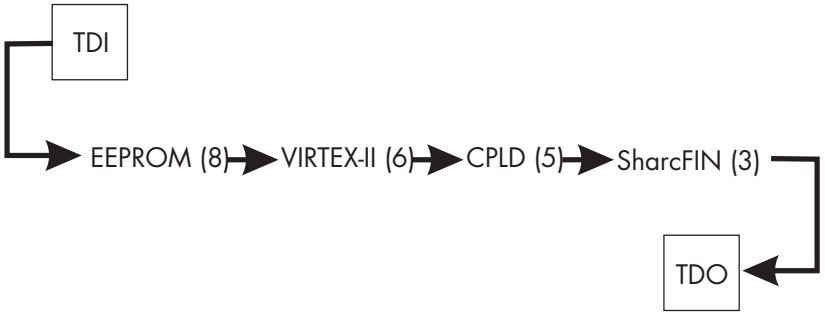
The MultiLINX connects to header J5, as described in "Connecting the MultiLINX Cable" on page 16. We assume that you are already familiar with the MultiLINX device's setup and operation, therefore this section will instead focus on the file-load methods specific to the Remora-PMC+.

We also describe the steps for loading your .mcs FPGA-logic file with the XILINX iMPACT utility. IMPACT is part of the XILINX Project Navigator suite, and provides the interface to program the EEPROM and Virtex-II devices. BittWare supplies the common data format (.cdf) files needed to load the FPGA software with iMPACT. They are on the Remora-PMC+ example CD-ROM as

- `jtag config\jtagprom.cdf`
- `jtag config\v2pmjtag.cdf`

The MultiLINX cable uses the JTAG interface to program the devices on the Remora-PMC+. The iMPACT utility requires that the number of devices on the JTAG interface (the JTAG chain) be defined. BittWare has defined the chain in each .cdf file as shown in Figure 6–12, with JTAG instruction width noted in parentheses.

Figure 6–12 JTAG Chain Defined for iMPACT



6.2.1 Getting Ready to Use the MultiLINX

Before you load files onto the board via the MultiLINX device, complete the following steps.

1. Set the configuration switches for MultiLINX operation.
   To configure the Remora-PMC+ board for operation with the MultiLINX device, set the default configuration mode on S2 to MultiLINX (JTAG boundary scan). Refer to "Selecting the Default Virtex Configuration Mode (S2)" on page 13 for operation details, and Table 2–2 on page 14 for the switch settings.

2. Attach the MultiLINX to the Remora-PMC+.
   Refer to "Connecting the MultiLINX Cable" on page 17 for installation instructions.

3. Create the FPGA-logic.
   To write the FPGA-logic, use the XILINX Project Navigator tools to create an .mcs file. Refer to the XILINX user's manual for details on creating this type of file.

6.2.2 Loading the EEPROM with the MultiLINX

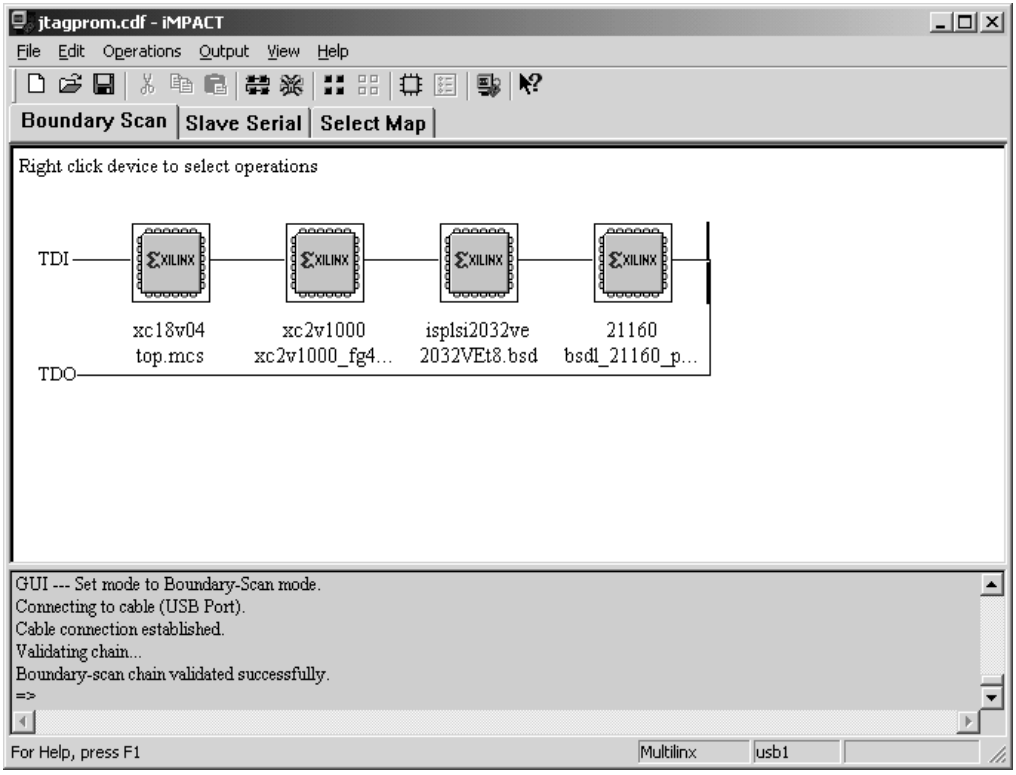
This section describes how to use the MultiLINX device to load the configuration file onto the EEPROM. This method preserves the FPGA program in case of a power-down event. On power-up, the EEPROM will automatically load the program onto the FPGA.

To load the FPGA via MultiLINX to the EEPROM,

1. Once the MultiLINX device has been installed, start the iMPACT utility from either the Xilinx Project Navigator or from the Xilinx Start menu. If you get an error message indicating the non-usage of a JTAGClk, the EEPROM file needs to be regenerated.[1]

2. Select **File** on the tool bar and click **Open**. Select and open `jtagprom.cdf` from the `RMPM Tools\examples` directory on the Remora-PMC+ example CD-ROM. This file contains the JTAG chain and should look like Figure 6–13.

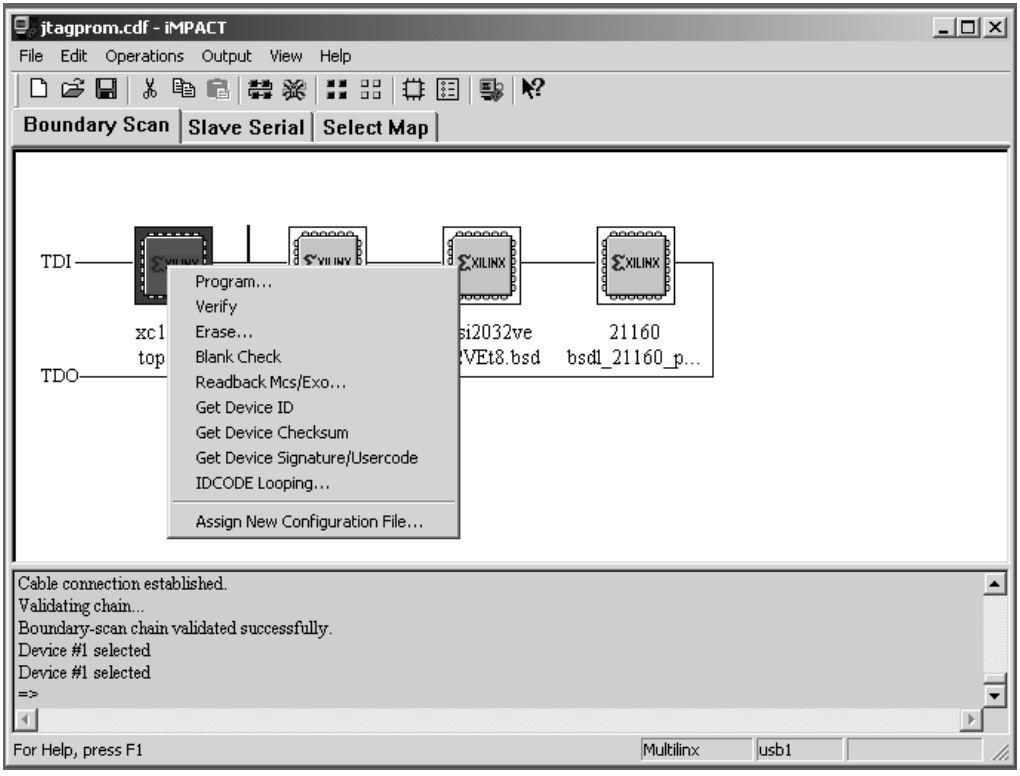
Figure 6–13 iMPACT JTAG Chain



1. Right-click on **Generate Programming File** in Project Navigator and choose **Properties**. Choose **Startup Options** in the Process Properties window and check **JTAG clock** for the StartUp-Clock option.

3. Right-click on the first device icon in the JTAG chain (xc18v04) and select **Assign New Configuration File** from the menu. Select the .mcs file from the menu that you created with the XILINX Project Navigator and click **Open**.

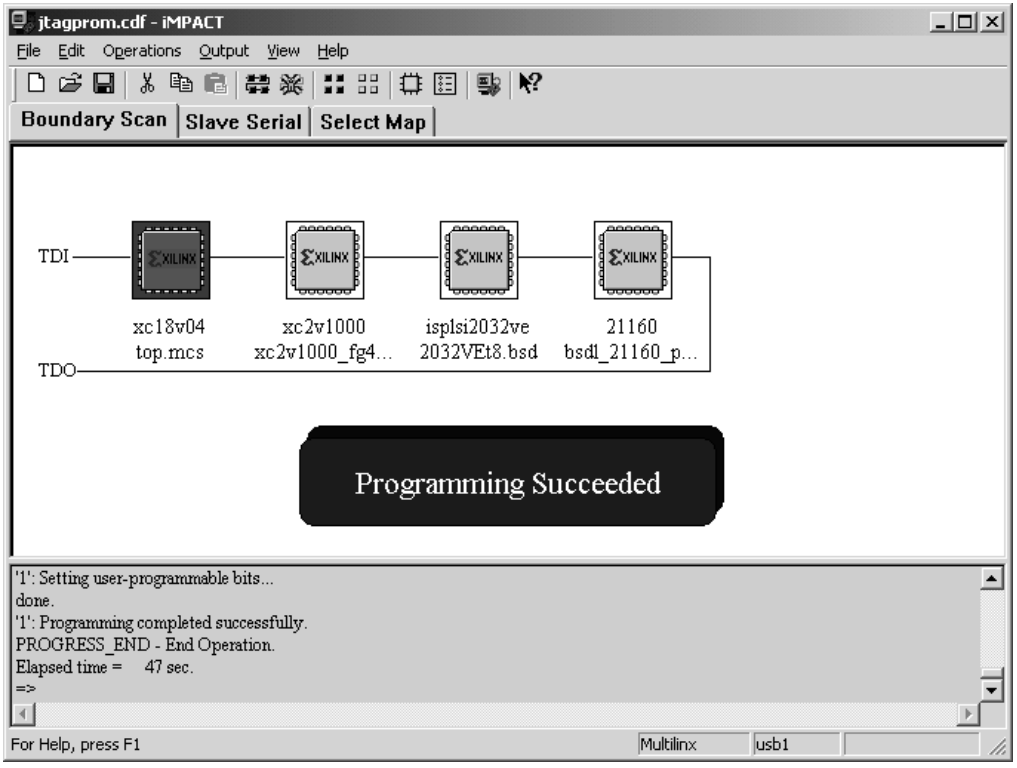
Figure 6–14 Assign New Configuration File



4. If the devices in the chain are initialized, skip to the next step. Otherwise, to initialize the devices, select **File** on the tool bar and then select **Initialize Chain**. Follow the on-screen menus to select the appropriate files for the corresponding devices.

5. Once the device initializations are verified, select device icon xc18v04, which represents the EEPROM. Right-click to open up this device's menu options. Click **Program** from the menu, and your .mcs file will be loaded onto the EEPROM. Upon completion of the fileload, a successful programming message should appear, as shown in Figure 6–15.

Figure 6–15 Successful Programming Session



6. You can now program the Virtex-II from the EEPROM with the BitLoader as described in section 6.1.4, or by setting the EEPROM to automatically load as described in “Loading the FPGA File on Start-Up” on page 91.

6.2.3 Loading the Virtex-II with the MultiLINX

The MultiLINX loads FPGA-logic files directly onto the Virtex-II the same way it loads the EEPROM. Note that .bit files loaded into the Virtex-II will be lost in case of a power-down event, and must be reloaded on power-up.

To load the FPGA files to the Virtex-II with the MultiLINX,

1. Follow the configuration and installation instructions as described in section 6.2.1. In this case, you will be loading a .bit file, not an .mcs file.

2. Once the MultiLINX device has been installed, start the iMPACT utility from either the XILINX Project Navigator or from the XILINX Start menu. If you get an error message that indicates the non-usage of a JTAGClk, the EEPROM file needs to be regenerated.[1]

3. Select **File** on the tool bar and click **Open**. Select and open `jtagprom.cdf` from the `RMPM Tools\examples` directory on the RMPM example CD-ROM. This file contains the JTAG chain and should look like the figure on page 94.

4. Right-click on the second device icon in the JTAG chain (xc2v1000), which represents the Virtex-II FPGA. Select **Assign New Configuration File** from the menu. Select the .bit file that you created with the XILINX Project Navigator from the menu and click **Open**.

5. If the devices in the chain are initialized, skip to the next step. Otherwise, to initialize the devices, select **File** from the tool bar and then select **Initialize Chain**. Follow the on-screen menus to select the appropriate files for the corresponding devices.

6. Once the device initializations are verified, select device icon xc18v04, which represents the Virtex-II FPGA. Right-click to open up this device's menu options. Click **Program** on the menu and your .bit file will be loaded onto the FPGA. Upon completion of the fileload, you should see the "Programming Succeeded" message as shown on page 96.

1. Right-click on **Generate Programming File** in Project Navigator and click on **Properties.** Click on **Startup Options** in the Process Properties window and check **JTAG clock** for the StartUp-Clock option.

6.3 Configuring the CPLD and its Control Registers

The control registers in the CPLD allow you to program certain control and mode features of the Virtex-II. These registers are currently set based on our expectations of their use; however, you may change them to suit your programming needs.

For details about each register's associated Virtex-II pin, refer to the *Virtex-II Platform FPGA Handbook* by XILINX.

6.3.1 Overview of the CPLD Control Registers

The control registers are located within the CPLD to provide access to some of the Virtex-II's control pins and the host-controlled JTAG pins. The base address is the peripheral bus "unbanked" space (see the footnote of Table 6–1) and is accessible by either the on-board DSP or the PCI host. Table 6–1 describes the register map, and the sections that follow detail the individual 4-bit registers.

Table 6–1 CPLD Control Register Map

Offset*	Name	Access
0	Control bit read register	Read only
0	JTAG control register	Write only
1	Virtex mode pin register	Write only
2	Virtex control pin register	Write only
3	Miscellaneous control register	Write only

* Offset from PCI byte offset 0x20 0000 from the base of BAR1 and 32 bit word offset 0x20 0000 from the base of MS1

6.3.2 Configuring the Control Bit Read Register

The Control Bit Read register provides read-only status of select JTAG Virtex pins. Table on page 99 lists the contents of the control read register.

Table 6–2 Control Bit Read Register

Bit	Description	Type
3	State of on-board JTAG TDO pin	Output
2	State of Virtex DOUT pin	Output
1	State of Virtex INIT_B pin	Bidirectional (open-drain)
0	State of Virtex DONE pin	Input/Output

6.3.3 Configuring the JTAG Control Register

The JTAG Control register is a write-only register that allows you to drive the JTAG interface from the CPLD as well as set clock, data input, and data output functions while the board is in boundary scan mode. Table on page 99 lists the contents of the JTAG register, and gives the reset value for each bit.

Table 6–3 Contents of the JTAG Control Register

Bit	Description	Reset Value
3	Enables the JTAG to be driven from the CPLD	1: Enables JTAG 0: Disables JTAG
2	On-board JTAG TMS	1: Enables Boundary Scan Mode Select 0: Normal operation
1	On-board JTAG TCK	1: Enables Boundary Scan Clock 0: Normal operation
0	On-board JTAG TDI	1: Enables Boundary Scan Data Input 0: Normal operation

6.3.4 Configuring the Virtex Mode Pin Register

The Virtex Mode Pin register allows you to set the bits either to dip-switch control or software (register) control. Bit 3 must be set first and selects either dip switch or register control. Bits 2:0 correspond to the modes associated with switch S2 (see "Selecting the Default Virtex Configuration Mode (S2)" on page 13).

Table 6–4 Contents of the Virtex Mode Register

Bit	Description	Control Value
3	Sets Virtex mode pins to dip switch control or register control	1: Register control 0: Switch S2 control
2	Virtex-II M2 pin, Slave serial mode	1: Register control 0: Switch S2 control
1	Virtex-II M1pin, JTAG boundary scan	1: Register control 0: Switch S2 control
0	Virtex-II M0 pin, Master serial mode	1: Register control 0: Switch S2 control

6.3.5 Configuring the Virtex Control Pin Register

This register controls the several programming features on the Virtex-II, including enabling the EEPROM, clearing the Virtex-II, and setting the serial programming inputs.

Table 6–5 Contents of the Virtex Control Pin Register

Bit	Description	Reset Value
3	Enables EEPROM or CPLD to drive Virtex-II serial programming pins	1: Enables CPLD 0: Enables EEPROM
2	Virtex-II PROG_B pin	1: Clear Virtex-II 0: Normal operation
1	Virtex-II CCLK pin serial programming input	1: Enables Configuration Clock; output in master mode or input in slave mode. 0: Normal operation
0	Virtex-II data-In serial programming input (pin IO_L02N_4/D0)	1: Enables user I/O pin DIN 0: Normal operation

6.3.6 Configuring the Miscellaneous Control Register

The Miscellaneous Control register contains the Virtex-II power-down setting and three user-configurable bits.

Table 6–6 Contents of the Miscellaneous Control Register

Bit	Description	Reset Value
3	Reserved (write 0)	1
2	Reserved (write 0)	1
1	Reserved (write 0)	1
0	Virtex power down pin (pin PWRDWN_B)	1: Power down 0: Normal operation

6.4 Virtex-II Signal Descriptions

The following table lists all of the Virtex-II FPGA I/O pins and shows how they are connected on the Remora-PMC+. This file is also included on the Remora-PMC+ examples disk in RMPM `Tools\examples\RMPM.ucf`. For additional details about the FPGA, refer to the *Virtex-II Platform FPGA Handbook* by XILINX. Note that the signal names are case-sensitive.

Note *XILINX requires that the signals being used in an application be defined in a user constraint file (.ucf). Table 6–7 and `RMPM.ucf` are provided for completeness; however, your file should include only the signals you will use in your project.*

Note *Active low signals are described in the following format: xHA_MS0_l, where the suffix "_l" represents active low.*

Table 6–7 Virtex-II Signal Descriptions

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xexep_n<1>	3	IO_L01N_3	J1, pin 1	W20
xexep_n<2>	3	IO_L03N_3/VF_3	J1, pin 2	W22
xexep_n<3>	3	IO_L04N_3	J1, pin 3	V20
xexep_n<4>	3	IO_L06N_3	J1, pin 4	V22
xexep_n<5>	3	IO_L43N_3	J1, pin 5	T22
xexep_n<6>	3	IO_L45N_3/VF_3	J1, pin 6	R18
xexep_n<7>	3	IO_L46N_3	J1, pin 7	R20
xexep_n<8>	3	IO_L48N_3	J1, pin 8	R22
xexep_n<9>	3	IO_L91N_3	J1, pin 9	N22
(Sheet 1 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xexep_n<10>	3	IO_L93N_3/VF_3	J1, pin 10	M17
xexep_n<11>	3	IO_L94N_3	J1, pin 11	M19
xexep_n<12>	3	IO_L96N_3	J1, pin 12	M21
xexep_n<13>	2	IO_L01N_2	J1, pin 13	C21
xexep_n<14>	2	IO_L03N_2	J1, pin 14	D21
xexep_n<15>	2	IO_L04N_2	J1, pin 15	E19
xexep_n<16>	2	IO_L06N_2	J1, pin 16	E21
xexep_n<17>	2	IO_L54N_2	J1, pin 17	K19
xexep_n<18>	3	IO_L49N_3	J1, pin 18	P20
xexep_n<19>	3	IO_L51N_3/VF_3	J1, pin 19	P22
xexep_n<20>	3	IO_L52N_3	J1, pin 20	N18
xexep_n<21>	3	IO_L54N_3	J1, pin 21	N20
xexep_n<22>	1	IO_L01N_1	J1, pin 22	A19
xexep_n<23>	1	IO_L92N_1	J1, pin 23	C13
xexep_n<24>	1	IO_L93N_1	J1, pin 24	A13
xexep_n<25>	1	IO_L94N_1	J1, pin 25	C12
xexep_n<26>	2	IO_L49N_2	J1, pin 26	J19
xexep_n<27>	2	IO_L96N_2	J1, pin 27	L21
xexep_n<28>	2	IO_L94N_2	J1, pin 28	L19
xexep_n<29>	3	IO_L93N_2	J1, pin 29	L17
xexep_n<30>	2	IO_L91N_2	J1, pin 30	K21
xexep_n<31>	2	IO_L48N_2	J1, pin 31	J17
xexep_n<32>	2	IO_L46N_2	J1, pin 32	H21
xexep_n<33>	2	IO_L45N_2	J1, pin 33	H19
xexep_n<34>	2	IO_L43N_2	J1, pin 34	G21
xexep_p<1>	4	IO_L01P_3	J1, pin 35	AA20
xexep_p<2>	3	IO_L03P_3	J1, pin 36	W21
(Sheet 2 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xexep_p<3>	4	IO_L04P_3	J1, pin 37	V19
xexep_p<4>	3	IO_L06P_3	J1, pin 38	V21
xexep_p<5>	3	IO_L43P_3	J1, pin 39	T21
xexep_p<6>	3	IO_L45P_3	J1, pin 40	P17
xexep_p<7>	3	IO_L46P_3	J1, pin 41	R19
xexep_p<8>	3	IO_L48P_3	J1, pin 42	R21
xexep_p<9>	3	IO_L91P_3	J1, pin 43	N21
xexep_p<10>	3	IO_L93P_3	J1, pin 44	N17
xexep_p<11>	3	IO_L94P_3	J1, pin 45	M18
xexep_p<12>	3	IO_L96P_3	J1, pin 46	M20
xexep_p<13>	2	IO_L01P_2	J1, pin 47	C22
xexep_p<14>	2	IO_L03P_2/VF_2	J1, pin 48	D22
xexep_p<15>	2	IO_L04P_2	J1, pin 49	E20
xexep_p<16>	2	IO_L06P_2	J1, pin 50	E22
xexep_p<17>	2	IO_L54P_2	J1, pin 51	K20
xexep_p<18>	3	IO_L49P_3	J1, pin 52	P19
xexep_p<19>	3	IO_L51P_3	J1, pin 53	P21
xexep_p<20>	3	IO_L52P_3	J1, pin 54	P18
xexep_p<21>	3	IO_L54P_3	J1, pin 55	N19
xexep_p<22>	1	IO_L01P_1	J1, pin 56	B19
xexep_p<23>	1	IO_L92P_1	J1, pin 57	D13
xexep_p<24>	1	IO_L93P_1	J1, pin 58	B13
xexep_p<25>	1	IO_L94P_1/VF_1	J1, pin 59	B12
xexep_p<26>	2	IO_L49P_2	J1, pin 60	J20
xexep_p<27>	2	IO_L96P_2	J1, pin 61	L22
xexep_p<28>	2	IO_L94P_2	J1, pin 62	L20
xexep_p<29>	2	IO_L93P_2/VF_2	J1, pin 63	L18
(Sheet 3 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xexep_p<30>	2	IO_L91P_2	J1, pin 64	K22
xexep_p<31>	2	IO_L48P_2	J1, pin 65	J18
xexep_p<32>	2	IO_L46P_2	J1, pin 66	H22
xexep_p<33>	2	IO_L45P_2/VF_2	J1, pin 67	H20
xexep_p<34>	2	IO_L43P_2	J1, pin 68	G22
xflag0	4	IO_L91N_4/VF_4	21160 bus	U13
xflag1	4	IO_L91P_4	21160 bus	V13
xha_a<0>	6	IO_L01N_6	21160 bus	U5
xha_a<1>	5	IO_L01P_6	21160 bus	V5
xha_a<2>	6	IO_L03N_6/VF_6	21160 bus	V3
xha_a<3>	6	IO_L03P_6	21160 bus	V4
xha_a<4>	6	IO_L04N_6	21160 bus	W1
xha_a<5>	6	IO_L04P_6	21160 bus	W2
xha_a<6>	6	IO_L06N_6	21160 bus	U3
xha_a<7>	6	IO_L06P_6	21160 bus	U4
xha_a<8>	6	IO_L43N_6	21160 bus	T1
xha_a<9>	6	IO_L43P_6	21160 bus	T2
xha_a<10>	6	IO_L45N_6/VF_6	21160 bus	R3
xha_a<11>	6	IO_L45P_6	21160 bus	R4
xha_a<12>	6	IO_L46N_6	21160 bus	R1
xha_a<13>	6	IO_L46P_6	21160 bus	R2
xha_a<14>	6	IO_L48N_6	21160 bus	P5
xha_a<15>	6	IO_L48P_6	21160 bus	P6
xha_a<16>	6	IO_L91N_6	21160 bus	N1
xha_a<17>	6	IO_L91P_6	21160 bus	N2
xha_a<18>	6	IO_L93N_6/VF_6	21160 bus	M5
xha_a<19>	7	IO_L93P_6	21160 bus	M6
(Sheet 4 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xha_a<20>	6	IO_L94N_6	21160 bus	M3
xha_a<21>	6	IO_L94P_6	21160 bus	M4
xha_a<22>	6	IO_L96N_6	21160 bus	M1
xha_a<23>	6	IO_L96P_6	21160 bus	M2
xha_a<24>	6	IO_L19N_6	21160 bus	V1
xha_a<25>	6	IO_L19P_6	21160 bus	V2
xha_a<26>	6	IO_L21N_6/VF_6	21160 bus	U1
xha_a<27>	6	IO_L21P_6	21160 bus	U2
xha_a<28>	6	IO_L22N_6	21160 bus	R5
xha_a<29>	6	IO_L22P_6	21160 bus	T5
xha_a<30>	6	IO_L24N_6	21160 bus	T3
xha_a<31>	6	IO_L24P_6	21160 bus	T4
xha_ack	4	IO_L02P_4/D1	21160 bus	V17
xha_brst_l	5	IO_L94P_5/VF_5	21160 bus	U10
xha_d<0>	4	IO_L49N_4	21160 bus	AA15
xha_d<1>	4	IO_L49P_4	21160 bus	AB15
xha_d<2>	4	IO_L51N_4	21160 bus	U14
xha_d<3>	4	IO_L51P_4/VF_4	21160 bus	V14
xha_d<4>	4	IO_L52N_4	21160 bus	W14
xha_d<5>	4	IO_L52P_4	21160 bus	Y14
xha_d<6>	4	IO_L54N_4	21160 bus	AA14
xha_d<7>	4	IO_L54P_4	21160 bus	AB14
xha_d<8>	5	IO_L49N_5	21160 bus	Y8
xha_d<9>	5	IO_L49P_5	21160 bus	W8
xha_d<10>	5	IO_L51N_5/VF_5	21160 bus	AB8
xha_d<11>	5	IO_L51P_5	21160 bus	AA8
xha_d<12>	5	IO_L52N_5	21160 bus	Y9
(Sheet 5 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xha_d<13>	5	IO_L52P_5	21160 bus	W9
xha_d<14>	5	IO_L54N_5	21160 bus	AB9
xha_d<15>	5	IO_L54P_5	21160 bus	AA9
xha_d<16>	6	IO_L49N_6	21160 bus	P3
xha_d<17>	6	IO_L49P_6	21160 bus	P4
xha_d<18>	6	IO_L51N_6/VF_6	21160 bus	P1
xha_d<19>	6	IO_L51P_6	21160 bus	P2
xha_d<20>	6	IO_L52N_6	21160 bus	N5
xha_d<21>	6	IO_L52P_6	21160 bus	N6
xha_d<22>	6	IO_L54N_6	21160 bus	N3
xha_d<23>	6	IO_L54P_6	21160 bus	N4
xha_d<24>	7	IO_L49N_7	21160 bus	J4
xha_d<25>	7	IO_L49P_7	21160 bus	J3
xha_d<26>	7	IO_L51N_7	21160 bus	J2
xha_d<27>	7	IO_L51P_7/VREF_7	21160 bus	J1
xha_d<28>	7	IO_L52N_7	21160 bus	J5
xha_d<29>	7	IO_L52P_7	21160 bus	K5
xha_d<30>	7	IO_L54N_7	21160 bus	K6
xha_d<31>	7	IO_L54P_7	21160 bus	L6
xha_d<32>	5	IO_L04N_5	21160 bus	AB5
xha_d<33>	5	IO_L04P_5/VF_5	21160 bus	AA5
xha_d<34>	5	IO_L05N_5	21160 bus	V7
xha_d<35>	5	IO_L05P_5	21160 bus	V6
xha_d<36>	5	IO_L06N_5	21160 bus	Y6
xha_d<37>	5	IO_L06P_5	21160 bus	W6
xha_d<38>	5	IO_L91N_5	21160 bus	V10
xha_d<39>	5	IO_L91P_5/VF_5	21160 bus	V9
(Sheet 6 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xha_d<40>	0	IO_L01N_7	21160 bus	E6
xha_d<41>	0	IO_L01P_7	21160 bus	E5
xha_d<42>	7	IO_L03N_7	21160 bus	D2
xha_d<43>	7	IO_L03P_7/VF_7	21160 bus	D1
xha_d<44>	0	IO_L04N_7	21160 bus	E4
xha_d<45>	7	IO_L04P_7	21160 bus	E3
xha_d<46>	7	IO_L06N_7	21160 bus	E2
xha_d<47>	7	IO_L06P_7	21160 bus	E1
xha_d<48>	7	IO_L43N_7	21160 bus	G2
xha_d<49>	7	IO_L43P_7	21160 bus	G1
xha_d<50>	7	IO_L45N_7	21160 bus	H5
xha_d<51>	7	IO_L45P_7/VREF_7	21160 bus	J6
xha_d<52>	7	IO_L46N_7	21160 bus	H4
xha_d<53>	7	IO_L46P_7	21160 bus	H3
xha_d<54>	7	IO_L48N_7	21160 bus	H2
xha_d<55>	7	IO_L48P_7	21160 bus	H1
xha_d<56>	7	IO_L91N_7	21160 bus	K4
xha_d<57>	7	IO_L91P_7	21160 bus	K3
xha_d<58>	7	IO_L93N_7	21160 bus	K2
xha_d<59>	7	IO_L93P_7/VREF_7	21160 bus	K1
xha_d<60>	7	IO_L94N_7	21160 bus	L5
xha_d<61>	7	IO_L94P_7	21160 bus	L4
xha_d<62>	7	IO_L96N_7	21160 bus	L3
xha_d<63>	7	IO_L96P_7	21160 bus	L2
xha_hbr_l	4	IO_L06N_4	21160 bus	AA17
xha_ms0_l	4	IO_L04N_4/VF_4	21160 bus	AA18
xha_ms1_l	4	IO_L04P_4	21160 bus	AB18
(Sheet 7 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xha_ms2_l	4	IO_L05N_4	21160 bus	W17
xha_ms3_l	4	IO_L05P_4	21160 bus	Y17
xha_page	5	IO_L94N_5	21160 bus	U11
xha_rdh_l	5	IO_L02P_5/D7	21160 bus	AA4
xha_rdl_l	5	IO_L02N_5/D6	21160 bus	AB4
xha_sbts_l	4	IO_L94P_4	21160 bus	V12
xha_tck1		TCK		C19
xha_tms1		TMS		B20
xha_wrh_l	5	IO_L01P_5/CS_B	21160 bus	AA3
xha_wrl_l	5	IO_L01N_5/RDWR_B	21160 bus	Y4
xha1_dmar1_l	4	IO_L92N_4	21160 bus	W13
xhdr_ack	4	IO_L93P_4	J3, pin 21	AB13
xhdr_clk	4	IO_L96P_4/GCLK0P	J3, pin 11	AB12
xhdr_d<0>	4	IO_L19P_3	J3, pin 2	U18
xhdr_d<1>	3	IO_L21P_3	J3, pin 4	U19
xhdr_d<2>	4	IO_L19N_4	J3, pin 6	V16
xhdr_d<3>	4	IO_L19P_4	J3, pin 8	V15
xhdr_d<4>	4	IO_L21N_4	J3, pin 10	W16
xhdr_d<5>	4	IO_L21P_4/VF_4	J3, pin 12	Y16
xhdr_d<6>	4	IO_L22N_4	J3, pin 14	AA16
xhdr_d<7>	4	IO_L22P_4	J3, pin 16	AB16
xhdr_d<8>	4	IO_L24N_4	J3, pin 18	W15
xhdr_d<9>	4	IO_L24P_4	J3, pin 20	Y15
xhdr_d<10>	5	IO_L24P_5	J3, pin 22	V8
xhdr_d<11>	5	IO_L19N_5	J3, pin 24	AB6
xhdr_d<12>	5	IO_L19P_5	J3, pin 26	AA6
xhdr_d<13>	5	IO_L21N_5/VF_5	J3, pin 28	Y7
(Sheet 8 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xhdr_d<14>	5	IO_L21P_5	J3, pin 30	W7
xhdr_d<15>	5	IO_L22N_5	J3, pin 32	AB7
xhdr_dir	4	IO_L94N_4/VF_4	J3, pin 1	U12
xhdr2_ack	1	IO_L21P_1	J2, pin 21	D16
xhdr2_clk	4	IO_L96N_4/GCLK1S	J2, pin 11	AA12
xhdr2_d<0>	1	IO_L22N_1	J2, pin 2	A16
xhdr2_d<1>	1	IO_L22P_1	J2, pin 4	B16
xhdr2_d<2>	1	IO_L24N_1	J2, pin 6	F14
xhdr2_d<3>	1	IO_L24P_1	J2, pin 8	E15
xhdr2_d<4>	2	IO_L19N_2	J2, pin 10	F19
xhdr2_d<5>	2	IO_L19P_2	J2, pin 12	F20
xhdr2_d<6>	2	IO_L21N_2	J2, pin 14	F21
xhdr2_d<7>	2	IO_L21P_2/VF_2	J2, pin 16	F22
xhdr2_d<8>	2	IO_L22N_2	J2, pin 18	G18
xhdr2_d<9>	2	IO_L22P_2	J2, pin 20	H18
xhdr2_d<10>	2	IO_L24N_2	J2, pin 22	G19
xhdr2_d<11>	2	IO_L24P_2	J2, pin 24	G20
xhdr2_d<12>	3	IO_L19N_3	J2, pin 26	T18
xhdr2_d<13>	3	IO_L21N_3/VF_3	J2, pin 28	U20
xhdr2_d<14>	3	IO_L22N_3	J2, pin 30	U22
xhdr2_d<15>	3	IO_L22P_3	J2, pin 32	U21
xhdr2_dir	1	IO_L21N_1/VF_1	J2, pin 1	C16
xpa_scl	5	IO_L24N_5	P4, pin 63	U9
xpa_sda	5	IO_L22P_5	P4, pin 64	AA7
xprom_td0		TDI		D3
xsw_usr<0>	5	IO_L92N_5	21160 bus	Y10
xsw_usr<1>	5	IO_L92P_5	21160 bus	W10
(Sheet 9 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xsw_usr<2>	4	IO_L93N_4	21160 bus	AA13
xsw_usr<3>	4	IO_L92P_4	21160 bus	Y13
xuser_clk	1	IO_L96P_1/GCLK2S		F13
xusr_led_1	5	IO_L93N_5	21160 bus	AB10
xusr_led_2	5	IO_L93P_5	21160 bus	AA10
xv2_cclk		CCLK	Peripheral bus	Y19
xv2_din	4	IO_L02N_4/D0	Peripheral bus	V18
xv2_dout	4	IO_L01N_4/DOUT	Peripheral bus	AB19
xv2_haclk	5	IO_L95P_5/GCLK4P	21160 bus	V11
xv2_HBG_l	4	IO_L06P_4	21160 bus	AB17
xv2_hbgen_l	5	IO_L95N_5/GCLK5S	21160 bus	W11
xv2_init_b	4	IO_L01P_4/INIT_B	21160 bus	AA19
xv2_l0ack	0	IO_L92P_0	PCI bus	C10
xv2_l0clk	0	IO_L96N_0/GCLK5P	P4, pin 7	B11
xv2_l0clk1	0	IO_L93P_0		A10
xv2_l0dat<0>	0	IO_L02N_0	P4, pin 11	C4
xv2_l0dat<1>	0	IO_L01P_0	P4, pin 12	A4
xv2_l0dat<2>	0	IO_L02P_0	P4, pin 13	C5
xv2_l0dat<3>	0	IO_L04P_0	P4, pin 14	C6
xv2_l0dat<4>	0	IO_L06P_0	P4, pin 15	E8
xv2_l0dat<5>	0	IO_L06N_0	P4, pin 16	E7
xv2_l0dat<6>	0	IO_L04N_0/VF_0	P4, pin 17	D6
xv2_l0dat<7>	0	IO_L05N_0	P4, pin 18	B6
xv2_l0fsync	0	IO_L93N_0	P4, pin 10	B10
xv2_l1ack	1	IO_L91N_1	P4, pin 22	E13
xv2_l1clk1	0	IO_L94N_0/VF_0	P4, pin 21	E11
xv2_l1clk2	0	IO_L95N_0/GCLK7P		D11
(Sheet 10 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xv2_l1dat<0>	0	IO_L21N_0	P4, pin 25	D7
xv2_l1dat<1>	0	IO_L21P_0/VF_0	P4, pin 26	C7
xv2_l1dat<2>	0	IO_L22N_0	P4, pin 27	B7
xv2_l1dat<3>	0	IO_L22P_0	P4, pin 28	A7
xv2_l1dat<4>	0	IO_L24N_0	P4, pin 29	D8
xv2_l1dat<5>	0	IO_L24P_0	P4, pin 30	C8
xv2_l1dat<6>	0	IO_L05P_0	P4, pin 31	A6
xv2_l1dat<7>	0	IO_L01N_0	P4, pin 32	B4
xv2_l1fsync	0	IO_L91N_0/VF_0	P4, pin 24	E10
xv2_l2ack	1	IO_L49N_1	P4, pin 36	C15
xv2_l2clk	1	IO_L95N_1/GCLK1P	P4, pin 35	E12
xv2_l2dat<0>	0	IO_L49N_0	P4, pin 39	B8
xv2_l2dat<1>	0	IO_L49P_0	P4, pin 40	A8
xv2_l2dat<2>	0	IO_L51N_0	P4, pin 41	E9
xv2_l2dat<3>	0	IO_L51P_0/VF_0	P4, pin 42	F9
xv2_l2dat<4>	0	IO_L52N_0	P4, pin 43	D9
xv2_l2dat<5>	0	IO_L52P_0	P4, pin 44	C9
xv2_l2dat<6>	0	IO_L54N_0	P4, pin 45	B9
xv2_l2dat<7>	0	IO_L54P_0	P4, pin 46	A9
xv2_l2fsync	1	IO_L06P_1	P4, pin 38	E17
xv2_l3ack	1	IO_L04N_1	P4, pin 50	C17
xv2_l3clk	1	IO_L96N_1/GCLK3P	P4, pin 49	F12
xv2_l3dat<0>	1	IO_L51N_1/VF_1	P4, pin 53	A15
xv2_l3dat<1>	1	IO_L51P_1	P4, pin 54	B15
xv2_l3dat<2>	1	IO_L52N_1	P4, pin 55	C14
xv2_l3dat<3>	1	IO_L52P_1	P4, pin 56	D14
xv2_l3dat<4>	1	IO_L54N_1	P4, pin 57	A14
(Sheet 11 of 12)				

RMPM Signal Name	Bank	Virtex Pin Name (FG456)	Board Location	Pad Location
xv2_l3dat<5>	1	IO_L54P_1	P4, pin 58	B14
xv2_l3dat<6>	1	IO_L05P_1	P4, pin 59	B17
xv2_l3dat<7>	1	IO_L06N_1	P4, pin 60	E16
xv2_l3fsync	1	IO_L02P_1	P4, pin 52	D18
xv2_m0		M0	Peripheral bus	AB2
xv2_m1		M1	Peripheral bus	W3
xv2_m2		M2	Peripheral bus	AB3
xv2_reset_l	4	IO_L95N_4/GCLK3S	21160 bus	W12
xv2_td0		TDO	21160 bus	D20
xv2_tdmrd0	2	IO_L52P_2	P4, pin 1	K18
xv2_tdmrfs0	2	IO_L51P_2/VF_2	P4, pin 2	J22
xv2_tdmtd0	2	IO_L52N_2	P4, pin 3	K17
xv2_tdmtrc0	2	IO_L51N_2	P4, pin 4	J21
(Sheet 12 of 12)				

Appendix A

Example Programs for the Remora-PMC+

This appendix provides a user level description for the Remora-PMC+ I/O FPGA example configuration shipped with the Remora-PMC+, including slave bus interface, I/O control logic, loopback link port, link port to SDRAM, and SDRAM to link port. It also includes descriptions of the Virtex-II addresses used by the SharcFIN ASIC.

Note *The FPGA source code for these examples is available in the Remora FPGA Developer's Kit. If you would like to purchase the Developer's Kit, or for more information, please contact BittWare's Sales department.*

A.1 Programming Information for the SharcFIN ASIC

A.1.1 Overview of the SharcFIN

The SharcFIN ASIC maps into PCI and the ADSP-21160 cluster bus. Although the Remora-PMC+ has no DSPs, the Virtex-II FPGA resides on the ADSP-21160 cluster bus and uses the space normally occupied by DSP 1 on a multiprocessor BittWare ADSP-21160 board. The SharcFIN uses PCI Base Address Registers (BARs) to map its various parts onto the PCI bus. As described in "Accessing Multiprocessor Memory Space" on page 62, BAR2 from the PCI allows access to the ADSP-21160 multiprocessor memory space (MMS), which in this case maps to the Virtex-II. For details about the SharcFIN, see Chapter 5 in this manual and the *SFIN-160 SharcFIN ASIC User's Guide* by BittWare.

A.1.2 Register Descriptions

Registers for the Virtex-II are located at 0×00100000 on the processor bus (0×100000 32-bit offset from BAR2 from PCI bus). The Virtex-II registers from BAR2 are listed in Table A–1, and specific examples for each register are described in the sections that follow.

Table A–1 Memory Map for the Virtex-II Registers, Chip Base Address 0×00100000

Location	Size	Description
0×00100000	32-bit	Miscellaneous register
0×00100001	32-bit	32-bit Status register
0×00100004	64-bit	Link Port to SDRAM Control register (address)
0×00100005	64-bit	Link port to SDRAM Control register (contents)
0×00100006 and 0×00100007	64-bit	Link port to SDRAM Control register
0×00100008 and 0×00100009	64-bit	Front panel primary I/O connector (J1) Read/ Write register
0×0010000A and 0×0010000B	64-bit	Front panel primary I/O connector (J1) Drive Enable register
0×0010000C	32-bit	User I/O connector (J2) Read/Write register
0×0010000D	32-bit	User I/O connector (J2) Drive Enable register
0×0010000E	32-bit	User I/O connector (J3) Read/Write register
0×0010000F	32-bit	User I/O connector (J3) Drive Enable register

Note *When permorming reads, BAR2 should be accessed by byte or word accesses. When performing writes, BAR2 should be accessed by word access only, as byte writes will corrupt the rest of the word.*

A.2 Software Examples

The following sections describe software examples that enhance the Remora-PMC+'s user I/O capabilities:

- slave bus interface between the PCI bus and local bus
- control logic for the I/O connectors
- link port to SDRAM control
- SDRAM to link port control
- loopback link port test

A.2.1 Slave Bus Interface

The slave bus interface allows you to read and write a block of registers from the PCI bus via the local bus. Table A–2 below and Table A–3 on page 119describe the most commonly used signals..

Table A–2 Contents of 0×00100000 32-bit Miscellaneous Register

Bit	Description	Type	Condition
0	Drives HBGEn	R/W	• A 1 in this bit allows the Virtex to handle HBR requests from host (ha_hbg_l = xv2_hbg_l) • A 0 in this bit allows the host to run logic to control HBG (ha_hbg_l = ha_hbr_1)
3:1	Unused		
4	UserLED(0)	R/W	Drives UserLED(0) when B5 is low
5	Source for UserLED(0)	R/W	Selects B4 (low) or UserSw(0) (high) as a source for UserLED(0)
6	UserLED(1)	R/W	Drives UserLED(1) when B7 is low
7	Source for UserLED(1)	R/W	Selects B6 (low) or UserSw(1) (high) as a source for UserLED(1)
8	xPA_SCL Enable*	R/W	• Writing 1 to this bit disables the driver on xPA_SCL • Writing 0 to this bit enables the driver on xPA_SCL and drives it to 0
9	xPA_SDA Enable*	R/W	• Writing 1 to this bit disables the driver on xPA_SDA • Writing 0 to this bit enables the driver on xPA_SDA and drives it to 0

* A read of B8 and B9 returns the status of the driver, not signals xPA_SCL and xPA_SDA. See Table A–3, "Contents of 0×00100001 32-bit Status Register," on page 119 to determine the values of these signals.

Table A–3 Contents of 0×00100001 32-bit Status Register

Bit	Description	Type	Value
0	Status of connector J1 pin 35	R/O	0 = low; 1 = high
1	Status of connector J1 pin 68	R/O	0 = low; 1 = high
2	Status of connector J1 pin 1	R/O	0 = low; 1 = high
3	Status of connector J1 pin 34	R/O	0 = low; 1 = high
4	Status of UserSw(3)	R/O	0 = low and indicates UserSw(3) is ON; 1 = high
5	Status of UserSw(2)	R/O	0 = low and indicates UserSw(2) is ON; 1 = high
6	Status of UserSw(1)	R/O	0 = low and indicates UserSw(1) is ON; 1 = high
7	Status of UserSw(0)	R/O	0 = low and indicates UserSw(0) is ON; 1 = high
8	Value of xPA_SCL	R/O	0 = low; 1 = high
9	Value of xPA_SDA	R/O	0 = low; 1 = high
15:10	Unused		
16	Empty flag for Link0 to SDRAM FIFO	R/O	0 = low; 1 = empty*
17	Full flag for Link0 to SDRAM FIFO	R/O	0 = low; 1 = full*
18	Empty flag for SDRAM to Link1	R/O	0 = low; 1 = empty*
19	Full flag for SDRAM to Link1	R/O	0 = low; 1 = full*
20	Empty flag for Link2 to Link3 FIFO	R/O	0 = low; 1 = empty*
21	Full flag for the Link2 to Link3 FIFO	R/O	0 = low; 1 = full*
22	Empty flag for the Link3 to Link2 FIFO	R/O	0 = low; 1 = empty*
23	Full flag for the Link3 to Link2 FIFO	R/O	0 = low; 1 = full*
31:28	Virtex-II chip Rev ID	R/O	"0 ."
27:24	Virtex-II chip Build ID	R/O	"0"

* B16 – B23 FIFO flags are active high, therefore a value of 1 in any of these bits means that the flag description is true.

A.2.2 Control Logic for I/O Connectors

Control logic for the 68-pin front panel primary I/O connector J1, and 32-pin user I/O connectors J2 and J3 allows the board to be used as a simple I/O card without any FPGA programming. In addition, the control logic gives you a baseline to modify if you need more full-functioned I/O capabilities. Table A–4 on page 121 lists the contents of the Read/Write register for primary connector J1, and the description applies to all the bits in the register. Table A–5 on page 122 describes the J1 connector Drive Enable register. User I/O connectors J2 and J3 have identical Read/Write and Drive Enable registers, and are listed together in Table A–5 on page 122 and Table A–6 on page 123.

Note *The values for I/O connector J1 pins 1, 34, 35 and 68 can be read in the status register, see Table A–3. To prevent potential damage to the board these pins can not be driven in the control logic example.*

Table A–4 0×00100008 (bits 31:0) and 0×00100009 (bits 63:32) 64-bit Front Panel Primary I/O Connector J1 Read/Write Register (Default Value = 0×0)

Bit	J1 Pin	Bit	J1 Pin	Type	Description
0	36	32	2	R/W	• A write to each bit writes to the output register • A read of each bit returns the value of the corresponding connector pin
1	37	33	3	R/W	
2	38	34	4	R/W	
3	39	35	5	R/W	
4	40	36	6	R/W	
5	41	37	7	R/W	
6	42	38	8	R/W	
7	43	39	9	R/W	
8	44	40	10	R/W	
9	45	41	11	R/W	
10	46	42	12	R/W	
11	47	43	13	R/W	
12	48	44	14	R/W	
13	49	45	15	R/W	
14	50	46	16	R/W	
15	51	47	17	R/W	
16	52	48	18	R/W	
17	53	49	19	R/W	
18	54	50	20	R/W	
19	55	51	21	R/W	
20	56	52	22	R/W	
21	57	53	23	R/W	
22	58	54	24	R/W	
23	59	55	25	R/W	
24	60	56	26	R/W	
25	61	57	27	R/W	
26	62	58	28	R/W	
27	63	59	29	R/W	
28	64	60	30	R/W	
29	65	61	31	R/W	
30	66	62	32	R/W	
31	67	63	33	R/W	

Table A–5 0×0010000A (bits 31:0) and 0×0010000B (bits 63:32) 64-bit Front Panel Primary I/O Connector J1 Drive Enable Register (Default Value = 0×0)

Bit	J1 Pin	Bit	J1 Pin	Type	Description
0	36	32	2	R/W	• A read of each bit returns the output enable register • A write to this location writes to the output enable register: 0 = disables the corresponding driver 1 = enables the corresponding driver and outputs the corresponding bit in the output register
1	37	33	3	R/W	
2	38	34	4	R/W	
3	39	35	5	R/W	
4	40	36	6	R/W	
5	41	37	7	R/W	
6	42	38	8	R/W	
7	43	39	9	R/W	
8	44	40	10	R/W	
9	45	41	11	R/W	
10	46	42	12	R/W	
11	47	43	13	R/W	
12	48	44	14	R/W	
13	49	45	15	R/W	
14	50	46	16	R/W	
15	51	47	17	R/W	
16	52	48	18	R/W	
17	53	49	19	R/W	
18	54	50	20	R/W	
19	55	51	21	R/W	
20	56	52	22	R/W	
21	57	53	23	R/W	
22	58	54	24	R/W	
23	59	55	25	R/W	
24	60	56	26	R/W	
25	61	57	27	R/W	
26	62	58	28	R/W	
27	63	59	29	R/W	
28	64	60	30	R/W	
29	65	61	31	R/W	
30	66	62	32	R/W	
31	67	63	33	R/W	

Table A–6 Contents of Read/Write Registers 0×0010000C 32-bit Connector J2 and 0×0010000E 32-bit Connector J3 (Default Value = 0×0)

Bit	J2/J3 Pin	R/W	Description
0	2	R/W	• A write to each bit (bits 15:0) writes to the output register • A read of each bit (bits 15:0) returns the value on the connector pin
1	4	R/W	
2	6	R/W	
3	8	R/W	
4	10	R/W	
5	12	R/W	
6	14	R/W	
7	16	R/W	
8	18	R/W	
9	20	R/W	
10	22	R/W	
11	24	R/W	
12	26	R/W	
13	28	R/W	
14	30	R/W	
15	32	R/W	
16	21	R/W	Pin 21 ACK signal and FLAG0 to the SharcFIN
17	1	R/W	Pin 1 DIR signal* • When high, the 16 data pins (2, 4, 6,...32) act as outputs from the Virtex • When low, the 16 data pins (2, 4, 6,... 32) act as inputs to the Virtex
18	11	R/W	Pin 11 CLK signal
19:32	Unused		

* The functionality of DIR is overridden on a per-pin basis by a 1 in the drive enable register.

A.2.3 Link 0 to SDRAM

In this example, data received by link port 0 (Link 0) is written to SDRAM at a self-incrementing location set by the PCI-accessible Link to SDRAM Control register (0×00100004).

Table A–7 64-bit SDRAM to Link Control Register

Address	Description
0×00100004	Current address
0×00100005	Unused

A.2.4 SDRAM to Link 1

This example allows you to transfer data from SDRAM to link port 1 (Link 1). The PCI-accessible SDRAM to Link Control register (0×00100006 and 0×00100007) sets the starting point and number of 64-bit words that should be transferred respectively. When 0×00100007 is non-zero, this example will read data from SDRAM and transfer it to Link 1. Each word that is transferred increments the address (0×00100006) and decrements the counter (0×00100007).

Table A–8 64-bit Link to SDRAM Control register

Address	Description
0×00100006	Current address
0×00100007	Number of 64-bit words to be transferred

A.2.5 Loopback Link Ports

This example uses the turns link ports 2 and 3 into a loopback port. Data received by Link 2 is stored in the FIFO memory and retransmitted on Link 3. Similarly, data received by Link 3 is stored in FIFO memory and retransmitted on Link 2. Note that there are no registers associated with this example.