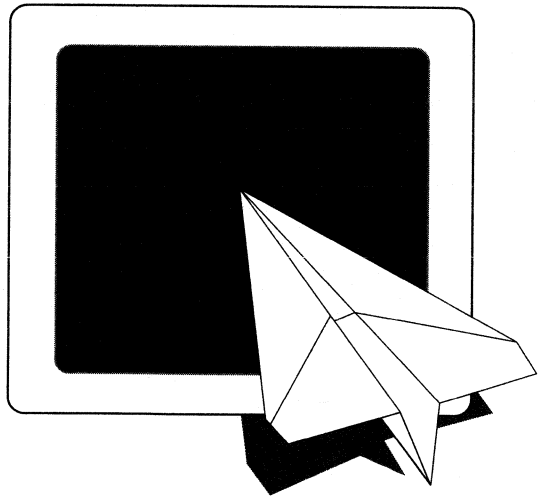


XC3000 Design Demonstration Board

XC4000 Design Demonstration Board

FPGA Design Demonstration Board

XPP/Serial Config. PROM Programmer

XChecker

Index

XACT Hardware and Peripherals Guide

XACT Hardware and Peripherals Guide — April, 1994 (0401132 01) *Printed in U.S.A.*

Preface

About This Manual

This manual describes the Xilinx hardware and associated software interfaces for the XACT Development System. The hardware consists of demonstration boards, which allow you to verify the design; an XChecker/download cable, which allows you to download and read back configuration data of designs; a PROM programmer, which allows you to create PROMs; the XPP PROM Programmer software interface; and the XChecker software interface.

Before using this manual, you should be familiar with the operations that are common to all Xilinx's software tools: how to bring up the system, select a tool for use, specify operations, and manage design data. These topics are covered in the *XACT Reference Guide*.

Manual Contents

This manual covers the following topics:

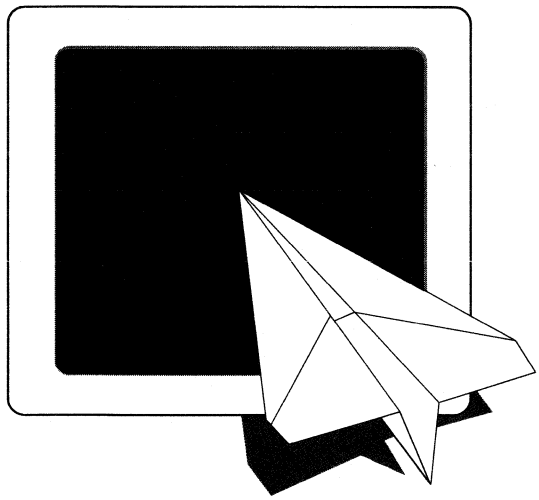
- Chapter 1, "XC3000 Demonstration Board," describes the function and operation of the XC3000 Demonstration Board.
- Chapter 2, "XC4000 Demonstration Board," describes the function and operation of the XC4000 Demonstration Board.
- Chapter 3, "FPGA Demonstration Board," describes the function and operation of the FPGA Demonstration Board, which combines the functionality of the XC3000 and XC4000 Demonstration Boards.
- Chapter 4, "XChecker Universal Download/Readback Cable and Logic Probe," describes how to use the cable and software interface to download, read back, verify configuration data, and probe internal logic states of designs.
- Chapter 5, "XPP/Serial Configuration PROM Programmer," describes how to use the HW112 PROM programmer and the XPP software interface.

Conventions

The following conventions are used in this manual's syntactical statements:

`Courier font regular`	System messages or program files appear in regular Courier font.
`Courier font bold`	Literal commands that you must enter in syntax statements are in bold Courier font.
italic font	Variables that you replace in syntax statements are in italic font.
`[ ]`	Square brackets denote optional items or parameters. However, in bus specifications, such as bus [7:0], they are required.
`{ }`	Braces enclose a list of items from which you must choose one or more.
. . .	A vertical ellipsis indicates material that has been omitted.
...	A horizontal ellipsis indicates that the preceding can be repeated one or more times.
|	A vertical bar separates items in a list of choices.
↵	This symbol denotes a carriage return.

Table of Contents



XC3000 Design Demonstration Board

XACT Hardware and Peripherals Guide

XACT Hardware and Peripherals Guide — April, 1994 (0401132 01) *Printed in U.S.A.*

XC3000 Design Demonstration Board

This Product is Compatible with the Families Indicated.

☑ XC2000	☑ XC3100	☑ XC3100A	☐ XC4000H
☑ XC2000L	☑ XC3000A	☐ XC4000	☐ XC7200
☑ XC3000	☑ XC3000L	☐ XC4000A	☐ XC7300

The XC3000 Demonstration Board is a simple tool for experimenting with designs for Xilinx FPGAs. Designers can use these boards to become familiar with the LCA devices as well as the XACT Development System. The board has all of the components necessary to load designs into LCA devices.

NOTE *Xilinx has three demonstration boards that allow you to test designs. Refer to the table below for the chapter that contains details about the demonstration board you are using.*

Board	Chapter
XC3000	XC3000 Design Demonstration Board
XC4000	XC4000 Design Demonstration Board
FPGA	FPGA Design Demonstration Board

You can use XACT Development System software to create designs for the LCA device on the demo board. The designs are then loaded in the LCA device via the XChecker or Download cable supplied with the XACT system. However, you should start by using the sample design rolldice provided with the XACT Development System software.

In addition, the demo board can be customized to accept any of the XC17xx Serial Configuration PROMs, which can be programmed with LCA designs for downloading to the demo board LCA device. This allows you to load LCA designs without having a system available. Refer to the section "Using the Demo Board with a PROM" at the end of this chapter for details about this option.

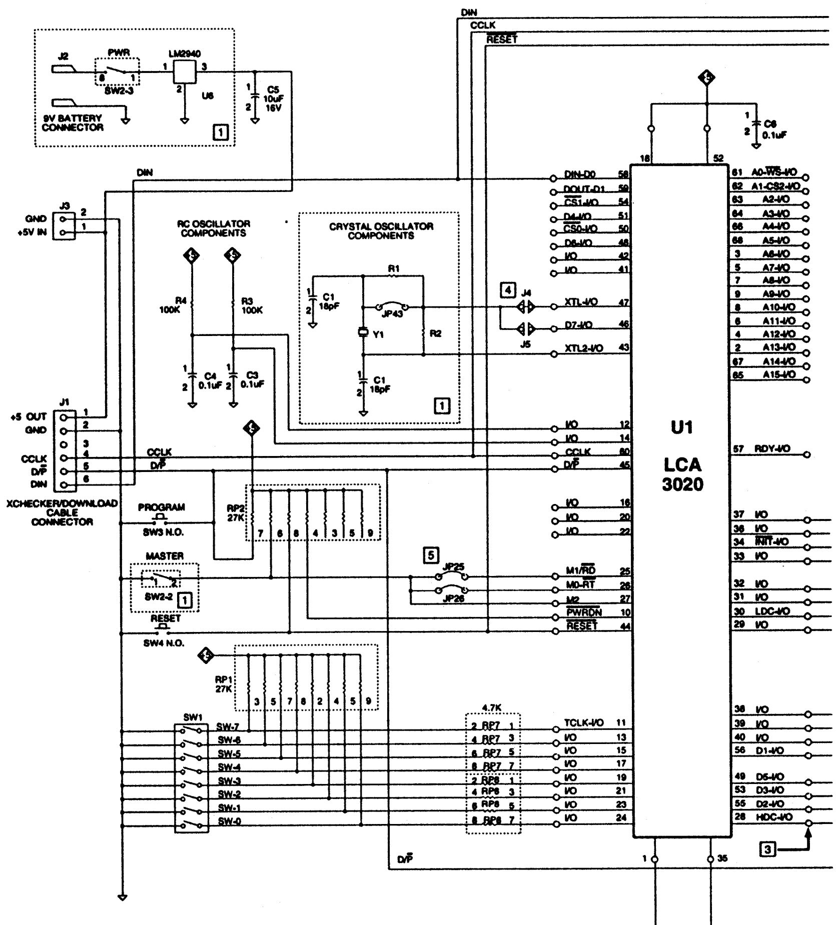
System Requirements

Following are the minimal requirements for using a Xilinx Demonstration Board.

- +5 V, 200 mA power supply
- PC or workstation

- XACT Design Editor
- Xilinx XChecker or parallel Download cable

NO TAG shows the complete schematic of the demo board, including PROM upgrade options.



X2591a

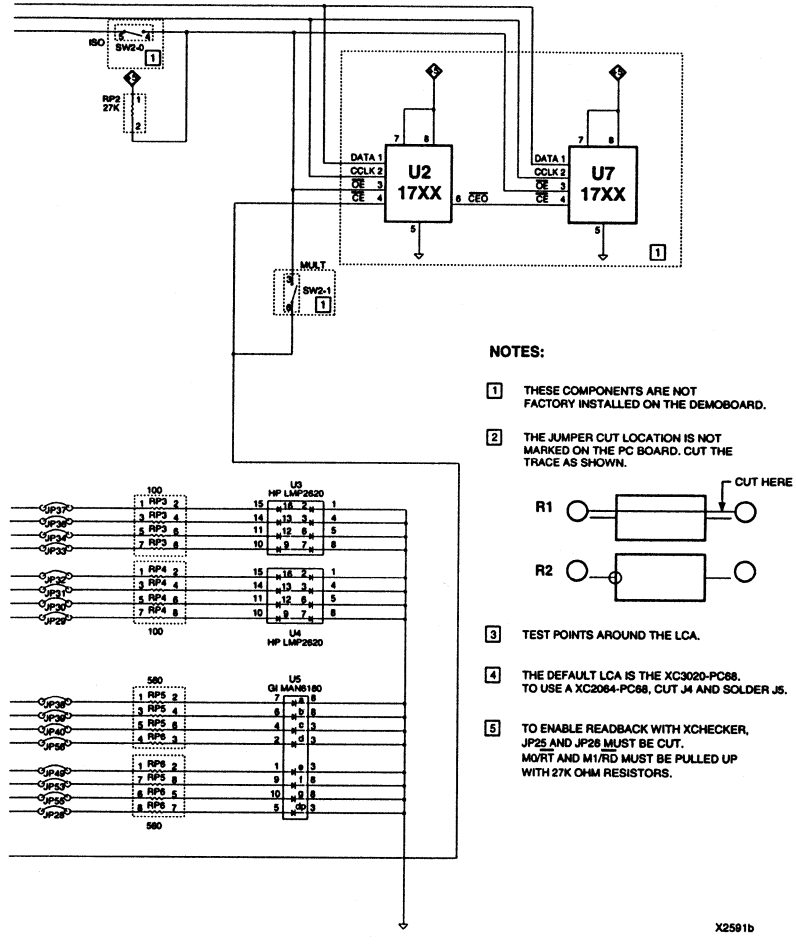


Figure 1-1 XC4000 Demonstration Board Schematic

Demonstration Board Description

The XC3000 demonstration board includes the following components, as shown in Figure 1-2.

- One LCA socket (68-pin PLCC)
- Eight LED indicators
- One 7-segment display
- One 8-position DIP switch
- Two momentary switches (one for programming, one for reset)
- RC relaxation oscillator
- XChecker or Download cable connector

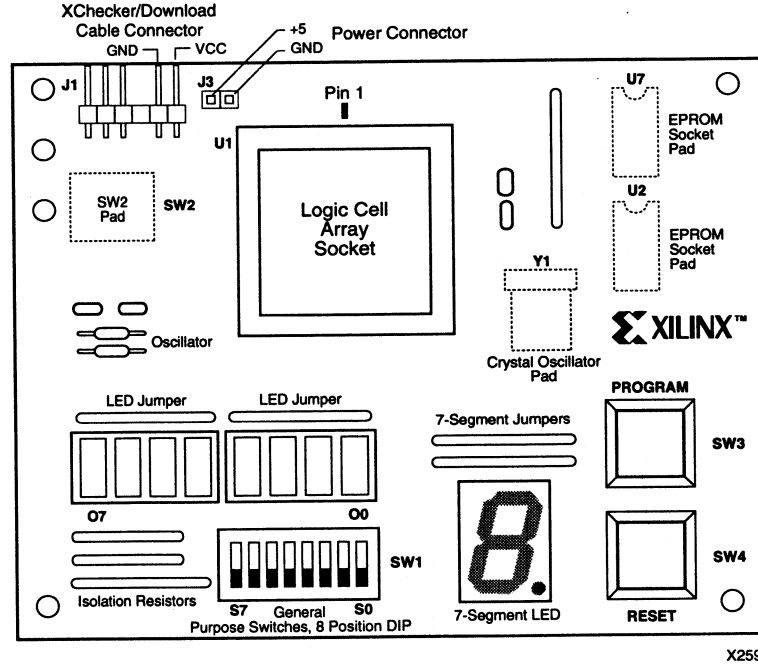


Figure 1-2 Demonstration Board Layout

The demo board components listed above are described in detail next.

XChecker and Download Cable and Power Connectors

The Download cable connector is a 5-pin male connector that mates with J1 on the board. There is also a loose-lead connector that can be used with J1. The connections for the loose-lead connector wires are shown below in Table 1-1 and Figure 1-3.

Table 1-1 XChecker/Download Cable Connections

J1 Pin	Cable Lead	Wire
1	V_{CC}	Red Wire
2	GND	Black Wire
3	Alignment key	
4	CCLK	Yellow Wire
5	D/$\overline{P}$	Blue Wire
6	DIN	Green Wire

The power connector J3 is a 2-pin male connector. Pin 1 connects to V_{CC} (+5 V) and pin 2 connects to ground. See Figure 1-3. The board requires a +5 V power supply rated at 200 mA because of the LED indicators. User designs not utilizing the demo board LED indicators may require only a few milliamperes.

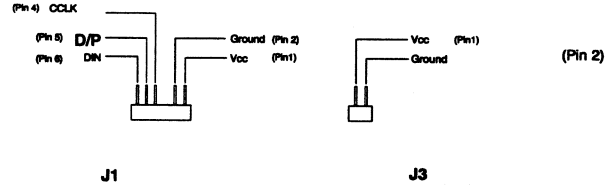


Figure 1-3 Download and Power Connections to the Demo Board

RESET Pushbutton (SW4)

The RESET pushbutton is connected to the $\overline{\text{RESET}}$ input of the LCA. Depressing the pushbutton while the LCA device is operating causes a global asynchronous reset of all IOB and CLB storage elements in the LCA device.

PROGRAM Pushbutton (SW3)

The PROGRAM pushbutton is connected to the DONE/$\overline{\text{PROG}}$ (D/$\overline{P}$) pin of the FPGA. Pressing this pushbutton pulls the D/P pin to a Low logic level, causing the LCA device to enter a configuration state.

DIP Switch

An 8-position DIP switch is connected to eight of the general-purpose I/O pins on the LCA device to control the logic level of the signals to the pins. Any DIP switch in the On position causes a logic Low to appear at the corresponding I/O pin on the LCA device. Pull-up resistors are connected to each switch. Any DIP switch in the Off position causes a logic High to appear at the corresponding I/O pin.

It is not recommended that I/O pins that are connected to these DIP-switch positions be configured as outputs because a closed switch position (Off) ties the LCA output to ground via a resistor. The LCA I/O lines connected to the DIP switch positions are shown in Table 1-2.

Table 1-2 Input Pin Connections

Switch	Pin
SW1-7	11
SW1-6	13
SW1-5	15
SW1-4	17
SW1-3	19
SW1-2	21
SW1-1	23
SW1-0	24

Each of the eight switch positions of the DIP switch is effectively isolated from the LCA I/O pins by isolation resistors RP7 and RP8.

LEDs

There are eight LEDs connected to the LCA I/O pins, as shown in Table 1-3. When On, the LEDs indicate that the signal to the corresponding LCA pin is a logic High.

Table 1-3 LED I/O Pin Connections

LED	Pin
O7	37
O6	36
O5	34
O4	33

LED	Pin
O3	32
O2	31
O1	30
O0	29

Each of the eight LEDs can be isolated from the LCA I/O pins by cutting one of the traces in the jumper area above the LEDs. This isolates the corresponding LCA I/O pin from the LED and its current-limiting resistor. Each jumper trace is labeled with the corresponding LCA I/O pin number.

7-segment Display

The 7-segment LED display is connected to eight I/O pins on the LCA as shown in Table 1-4.

Table 1-4 7-Segment Display Pin Connections

Segment	Pin
a	38
b	39
c	40
d	56
e	49
f	53
g	55
Decimal point	28

Each LED segment is turned On by driving the corresponding LCA pin High (Logic 1). The decimal point also serves as a programming indicator, as the LCA drives pin 28 High when the LCA is in the configuration state; this pin is HDC (High During Configuration). The rolldice sample design included with the XACT Development System contains a 7-segment decoder circuit.

Each of the eight LED segments can be isolated from the LCA I/O pins by cutting one of the traces in the jumper area above the 7-segment display. This isolates the corresponding LCA I/O pin from both the LED segment and its current-limiting resistor. Each jumper trace is labeled with a corresponding LCA I/O pin number.

RC Oscillator

The demo board includes an RC circuit relaxation oscillator when used with the GOSC macro. When the LCA device is configured with CMOS thresholds, as in the sample design file rolldice, the oscillator generates a clock of approximately 100 Hz. The RC oscillator is connected to LCA pins 12 and 14.

Crystal Oscillator (Optional)

The LCA device has on-chip circuitry to drive a crystal oscillator, and the demo board has provisions for a crystal-oscillator circuit (Figure 1-4). Refer to Figure 1-4 for recommended crystal-oscillator component values. If your crystal oscillator configuration requires resistor R1, cut the trace shorting R1 as shown in note 2 of the schematic (Figure 1-1).

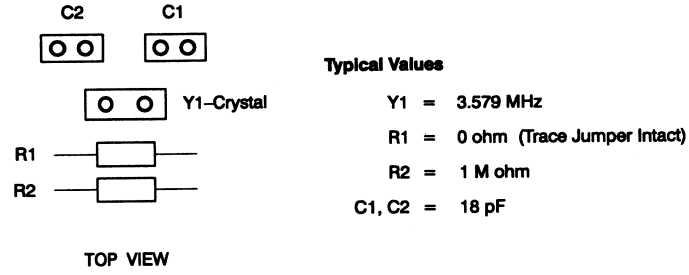


Figure 1-4 Crystal Oscillator Option Components

If using an XC2064 device instead of an XC3020 device and using the crystal oscillator, cut the trace at J4 on the underside of the board. Solder the jumper at J5.

Demonstration Board Operation

The operation of the XC3000 Demonstration Board is explained in next three sections, as follows:

- Creating a Design for Download
- Using the Demo Board with the Download or XChecker Cable
- Using the Demo Board with a PROM

Normally you would use the demo board with the Download or XChecker cable, loading an LCA design such as rolldice from your system. These procedures are described in the Using the Demo Board with the Download or XChecker Cable section.

To use the board as a stand-alone aid, you need to upgrade the board to accept PROMs. The upgrade steps and PROM procedures are described in the Using the Demo Board with a PROM section.

Creating a Design for Download

In the \XACT\examples\core\rolldice directory (for PCs), or $XACT/examples/core/rolldice directory (for workstations) there is a sample design called rolldice.bit. Alternatively, you can create your own design using the following steps:

1. Place and route the design.

   Produce a routed design (*design*.lca) by using a design-entry tool and ADI, or using XDE for manual implementation.

2. Generate a bitstream and set configuration options.

   Generate a configuration bitstream for the design (*design*.bit) with the appropriate configuration options. Do this by using MakeBits. For more information about MakeBits, see "The MakeBits Program" in the *XACT Reference Guide*.

3. Create a PROM file (optional).

   This step is optional since the XChecker software can use the *design*.bit file as input. A PROM file is generated by using MakePROM. For information on PROM files, see "The MakePROM Program" in the *XACT Reference Guide*.

Using the Demo Board with the Download or XChecker Cable

Using the rolldice .bit design found in $XACT/examples/core/rolldice (for workstations) and \XACT\examples\core\rolldice (for PCs) or another design which has been processed to a BIT or hexadecimal

(MCS, TEK, or EXO) format, it can be downloaded to the demo board using XChecker or the Download cable.

For a description of this design, see the Sample Design section in this chapter. Perform the following steps to download this design to the demo board.

1. With the power off, attach a +5 V, 200 mA power supply to the power connector J3 on the demo board. Connect J3 pin 1 to +5 V and J3 pin 2 to Ground.
2. Attach the XChecker cable to a serial port on your PC or workstation. For the Download cable, attach it to a parallel port on the PC. See the XChecker Universal Download/Readback Cable and Logic Probe chapter in this manual for information on cable connections.
3. Connect the XChecker or Download cable to the demo board, as shown in Figure 1-5.

   When using the XChecker cable, only one of the two-keyed connectors is needed.

   The XChecker and Download cables draw power from the target system through the V_{CC} and GND wires. Therefore, power to the XChecker or Download cable, as well as to the target FPGA, must be stable. You must not connect the XChecker or Download pins to any signals before connecting V_{CC} and Ground to the demo board.

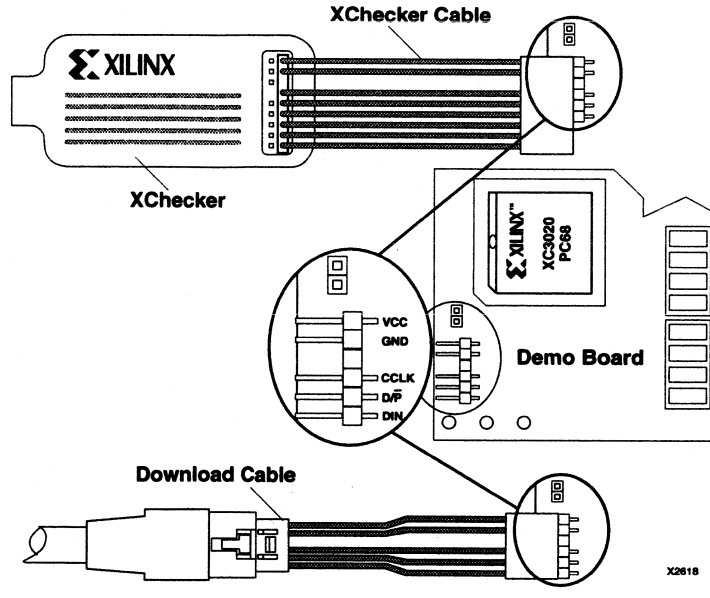


Figure 1-5 Connecting XChecker/Download Cable to Demo Board

4. Turn the +5 V power supply on.
5. From within XDM, select XChecker from the Verify menu. The XChecker software can also be started from the operating system prompt by entering the following.

   xchecker *design_name*

6. The XChecker software identifies the type of cable attached to the computer. Press the ↵ key to download the design to the demo board. If the XChecker program cannot communicate with the cable, or issues an error message, check the power supply and cable connections.
7. XChecker indicates that the LCA device is loading. Once the LCA device has finished configuration, XChecker displays the message "DONE went High." At this point, the LCA device can operate with the downloaded design.

NOTE *Demo boards are set up for use with the Download or XChecker cable as supplied. If you are using a demo board that has been upgraded to use PROMs, make sure that SW2-2 of the optional PROM switch is Off (Slave mode).*

Using the Demo Board with a PROM

Although primarily intended for configuration with the XChecker or the Download cable, the LCA device on the demo board can be configured from a PROM. You should, however, already be familiar with use of a cable and the sample design before upgrading the demo board for use with a PROM. If you do upgrade for the PROM option, the demo board does remain compatible for use with the download cable.

To upgrade the demo board for use with a PROM, you need the following items.

- Two 8-pin DIP sockets (U2 and U7 on the schematic, Figure 1-1)
- One 4-position DIP switch (SW2 on the schematic)
- One or more XC17xx PROMs and a PROM programmer

The PROM sockets and the DIP switch need to be soldered into their respective positions on the demo board. Also, one or more LCA designs must be programmed into the PROM or PROMs you are using.

PROM DIP Switch

A 4-position DIP switch contains two PROM bank-select positions and a master-slave mode position. This switch is not required for operation with the XChecker or the Download cable.

Table 1-5 SW2 Switch Functions

Switch Position	Purpose
SW-3 (PWR)	Reserved
SW-2 (Master)	Master (On)/Slave (Off) Select
SW-1 (MULT)	Single Configuration (On)/Multiple Configuration (Off) Select
SW-0 (ISO)	Ties PROM Reset to LCA Reset (On)

Position SW-2 of the PROM switch determines the mode (Master or Slave) for the LCA device the next time it is configured, either at power on or when the program switch is pressed.

If there is no PROM switch, or when position SW-2 is off, the LCA device is configured in a Slave mode, required for operation with the XChecker or the Download cable. When the PROM switch is installed and position SW-2 is On, the LCA device is configured in a Master mode, and attempts to configure itself from the XC17XX PROM.

NOTE *Jumper JP26 connects M0/$\overline{\text{RTRIG}}$ and M1/$\overline{\text{RDATA}}$. This trace must be cut to allow Readback with the demo board and the XChecker cable.*

Position SW-1 of the PROM switch controls the configuration mode. When open, it allows multiple configuration. When closed, only a single configuration is allowed. It must be closed when you are using XC1736A PROMs.

Position SW-0 of the PROM switch times or isolates the PROM reset and the LCA reset. When position SW-0 is open, you can reset the LCA device without resetting the PROM address or bit counters. This allows you to load multiple configurations.

The XC1736A SCP does not support multiple LCA configuration patterns. This is because the polarity of the RESET/$\overline{OE}$ pin is fixed and not programmable, as with the XC1765 and the larger XC17XX PROMs. For more than one LCA configuration pattern, a SCP must be programmed with the RESET/OE pin polarity at an active Low logic level. This prevents the SCP address bit counter from being reset when the RESET (SW4) pushbutton is depressed.

Creating an LCA Configuration PROM

There are four basic functions necessary if you want to use a PROM to configure the demo board LCA device.

- Creating the LCA Design
- Converting the Design to a Bitstream
- Creating a PROM Hex File
- Programming the PROM

Creating the LCA Design

To ensure a successful introduction to the process of creating a demo board PROM, consider one of the sample designs provided with the XACT Development System for your first PROM design. If you are not using one of the sample LCA designs provided with the XACT Development System, create your own design using the EditLCA program in XDE. The EditLCA program commands are described in the XDE chapter of this manual.

Converting the Design to a Bitstream

Once the LCA design file has been created, use the MakeBits program to compile the design file (LCA) to an LCA configuration bitstream file (BIT). MakeBits program commands are described in "The MakeBits Program" in this manual.

Creating a PROM Hex File

The MakePROM program is used to create a hex file for a PROM. For details about the MakePROM program, see "The MakePROM Program" in this manual.

Programming the PROM

Now that you have created the LCA design, converted the design file to a bitstream file, and created a PROM hex file in the proper format, you have a PROM-ready design file on your system. You need to connect a PROM programmer to you system so you can program PROMs.

Configuring the Demo Board LCA with a PROM

Be sure that your demo board has been upgraded to support a PROM, as described above.

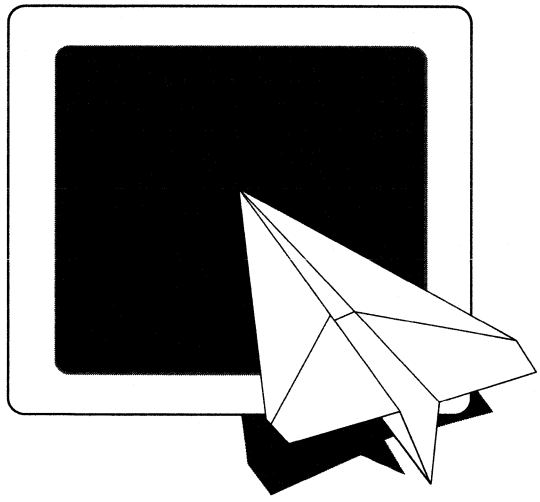
To set up the demo board, first be sure power is Off. Install a PROM in PROM socket . Set position 2 of the 4-position PROM switch (switch #2) to On, the Master mode, and disconnect the XChecker or Download cable from the demo board.

To configure the LCA device, turn the +5 V power supply on. The LCA device will configure itself in Master-serial mode from the PROM(s). If more than one LCA configuration pattern was programmed into the PROM(s), pressing the PROGRAM pushbutton (SW3) loads the next pattern to the LCA device.

NOTE *The Download cable does not usually cause signal contention with the PROM-mode circuitry on the demo board. However, since the Download cable is connected to the PC parallel-printer port, data sent to the printer port could cause the Download cable to disrupt PROM-based operation of the demo board, so it is wise to disconnect it.*

Sample Design

The sample LCA design is located in the \XACT\examples\core\rolldice directory for PCs. For workstations, the sample LCA design is located in the $XACT/examples/core/rolldice subdirectory in the Xilinx software directory. The sample design's name is rolldice. This design simulates the random rolling of two six-sided dice. The result of each die throw is shown on the 7-segment display. Switch SW1-7 (pin 11 on the LCA) in the on position stops the roll and displays dice values. In the off position, the dice are "rolled."



XC4000 Design Demonstration Board

XACT Hardware and Peripherals Guide

XACT Hardware and Peripherals Guide — April, 1994 (0401132 01) *Printed in U.S.A.*

XC4000 Design Demonstration Board

This Product is Compatible with the Families Indicated.

☐ XC2000	☐ XC3100	☐ XC3100A	☑ XC4000H
☐ XC2000L	☐ XC3000A	☑ XC4000	☐ XC7200
☐ XC3000	☐ XC3000L	☑ XC4000A	☐ XC7300

The XC4000 Demo Board is a stand-alone board based on the XC4000 Logic Cell Array (LCA). The XC4000 Demo Board allows you to become familiar with the XC4000 Logic Cell Arrays and the XACT Development System.

NOTE *Xilinx has three demonstration boards that allow you to test designs. Refer to the table below for the chapter that contains details about the demonstration board you are using.*

Board	Chapter
XC3000	XC3000 Design Demonstration Board
XC4000	XC4000 Design Demonstration Board
FPGA	FPGA Design Demonstration Board

The XC4000 Demo Board contains eight LEDs, two 7-segment displays, two octal DIP switches, and three momentary switches. Each of these components is hard wired to specific input and output pins, jumper posts, or headers. These features are useful in prototype demonstrations.

The XC4000 Demo Board has been designed to allow easy access to I/O and configuration control lines by providing standard .025″ square header posts for each of the 84 LCA pins. Three 8-pin serial PROM sockets are provided. These sockets support user-supplied serial PROMs. The wiring to these sockets is cascaded to allow for large or multiple designs. All LCA configuration modes can be selected. The XC4000 Demo Board is powered by a +5 V, 200 mA power supply or through a power regulator sourced by a 7 to 27 V power supply. Space is provided for an optional power regulator and crystal oscillator.

Designs are created for the XC4000 Demo Board using the XACT Development System software. A sample design is provided with the XC4000 Demo Board. LCA designs can be downloaded into the XC4000 Demo Board via the XChecker cable or loaded from a serial PROM.

System Requirements

The following items are required to operate the XC4000 Demo Board.

- +5 V, 200 mA power supply
- XC40XX-PC84 Logic Cell Array
- XACT Development System software
- PC or workstation with a standard DB-9 or DB-25 RS-232 serial port to connect to the XChecker cable
- Minimum of 640 kB of base memory
- XChecker cable provided with XACT

Demonstration Board Description

The XC4000 Demo Board is a single-board system for experimenting and developing prototypes with Logic Cell Array designs. The XC4000 Demo Board is assembled with the following features.

- Socket for the Logic Cell Array (XC40XX PC84)
- Eight general-purpose LED indicators
- Two 7-segment displays
- General-purpose octal-DIP switch
- One spare pushbutton
- Program and Reset pushbuttons
- Octal-DIP switch used for mode selection and configuration
- Connectors for the XChecker cable
- Three 8-pin serial PROM sockets
- Eighty-four .025″ square header pins attached to each LCA pin
- Options for using either a +5 V or a variable power supply (using an optional regulator)
- Space for an optional +5 V regulator
- Space for an optional crystal oscillator

Demonstration Board Components

Figure 2-1 shows the component positions on the XC4000 Demo Board. Component part descriptions follow. For additional information, a schematic of the XC4000 Demo Board is shown in Figure 2-2.

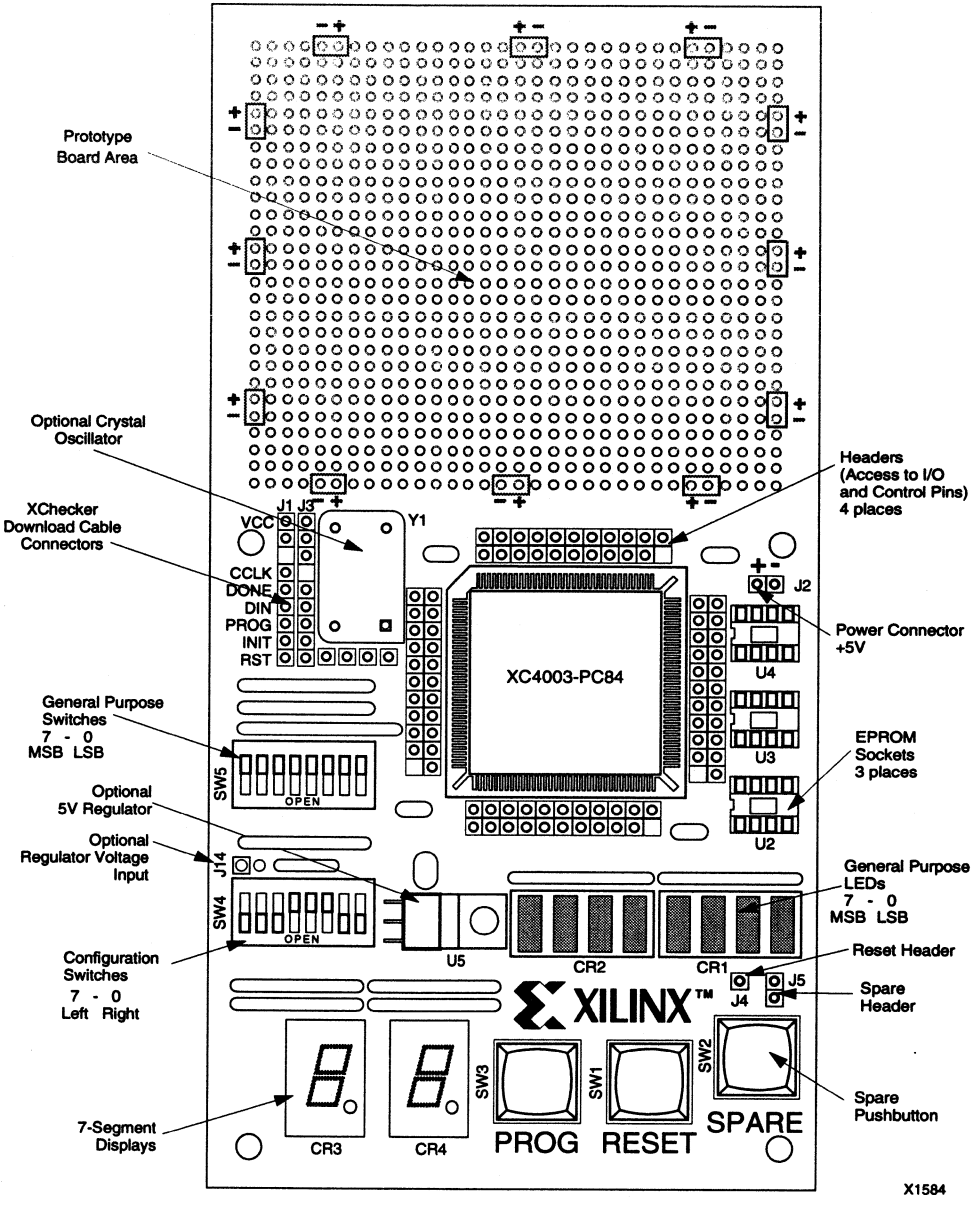
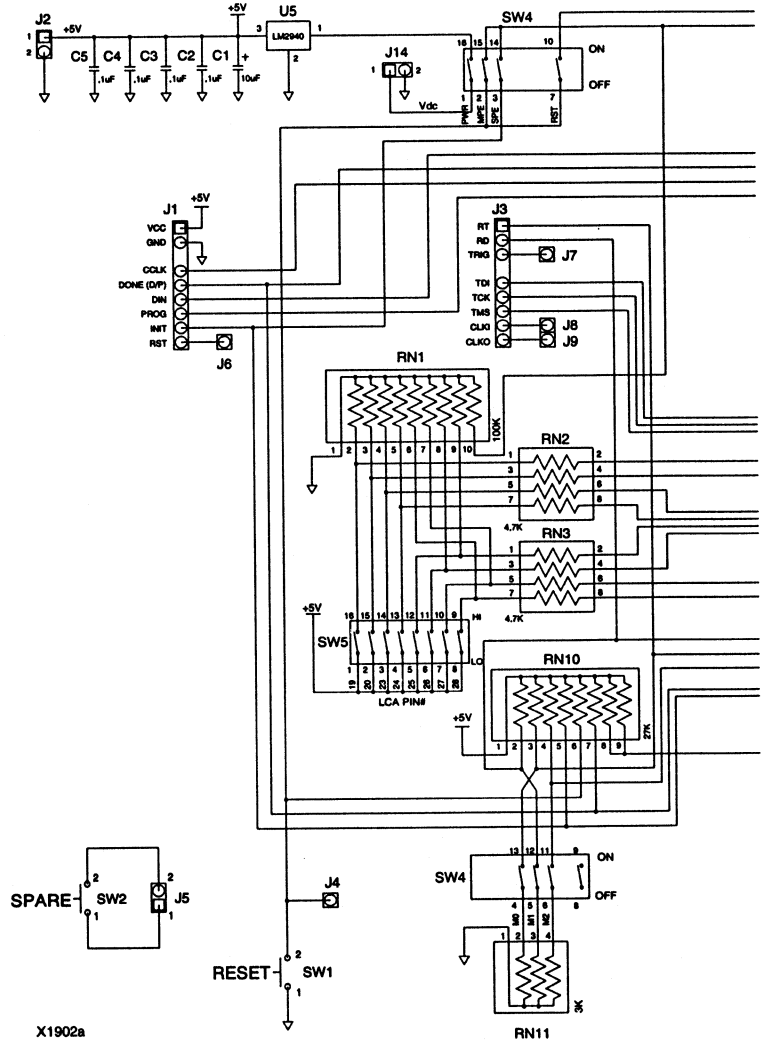


Figure 2-1 Demonstration Board Layout


J2
+5V
C5
C4
C3
C2
C1
.1uF
10uF
U5
LM2940
SW4
ON
OFF
J14
Vdc
PWR
MAPE
SPE
RST
J1
VCC
GND
CCLK
DONE (D/P)
DIN
PROG
INT
RST
J6
J3
RT
RD
TRIG
J7
TDI
TCK
TMS
CLKI
CLKO
J8
J9
RN1
100K
RN2
4.7K
RN3
SW5
HI
LO
LCA PIN#
RN10
27K
SPARE
SW2
J5
J4
RESET
SW1
X1902a
RN11
3K

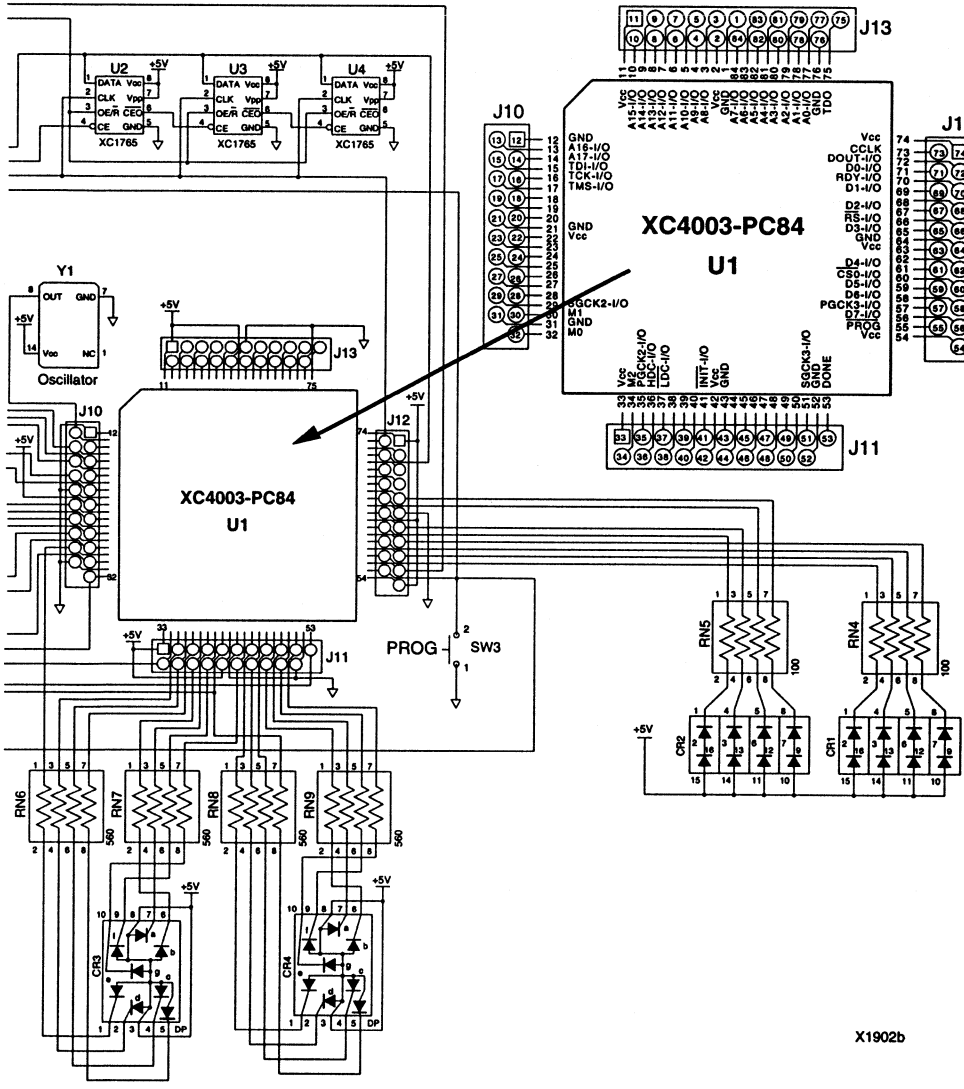


Figure 2-2 Demonstration Board Schematic

Five Volt Power Connector (J2)

+5 V and Ground are connected to the XC4000 Demo Board at connector J2 as shown in Table 2-1. Figure 2-3 shows the power pins.

Table 2-1 J2 Pin Connections

J2 Pin	Connection
1	V_{CC} (+5 V)
2	Ground

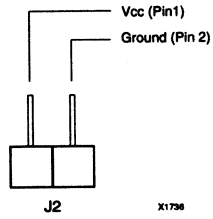


Figure 2-3 Power Pins

The XC4000 Demo Board requires a +5 V supply rated at 200 mA. The power connector J2 is a 2 pin male connector. Pin 1 connects to V_{CC} (+5 V) and pin 2 connects to ground. The 200 mA is specified because of the LED indicators. If your design does not use the on-board LEDs, you might need only a few milliamperes. The required current depends on the number of I/O lines employed.

Alternate Power Connector (J14)

The optional regulator voltage input and Ground are connected to the XC4000 Demo Board at location J14.

A +9 V battery can be connected to location J14; you can also use any DC power supply, depending on the type of +5 V regulator used. At higher voltage levels, you should consider power dissipation limits of the +5 V regulator (U5). The alternate power location J14 provides two holes to connect alternate power. See Figure 2-4. Hole 1 (the square pad) connects to the positive terminal of the optional regulator voltage input, and hole 2 (the round pad) connects to Ground. Hole 1 of J14 is routed through SW4 to U5, where a +5 V voltage regulator can be installed.

XChecker Connector (J1, J3)

XChecker can be thought of as a logic probe specially designed to work with Xilinx XC2000, XC3000, and XC4000 families of FPGAs. In this chapter the XChecker cable discussion is limited to the connection and downloading of configuration data to the XC4000 Demo Board. See Figure 2-4

NOTE *The XChecker cable draws its power from the target system through the V_{CC} and GND wires. Therefore, power to XChecker, as well as to the target FPGA, must be stable. You must not connect the XChecker pins to any signals before connecting V_{CC} and Ground to the cable.*

XChecker provides the following connectors.

- Header connectors — Two standard 9 pin headers that fit onto .025″ square pins. These header connectors are keyed to assure proper orientation in the cable assembly.
- Flying lead — These leads are terminated in standard female connectors that fit onto a .025″ square pin.

Six signal pins, one Ground and one V_{CC} pin are needed for downloading a configuration bitstream. Table 2-2 defines the connections and operation of the XChecker cable, as it connects to J1.

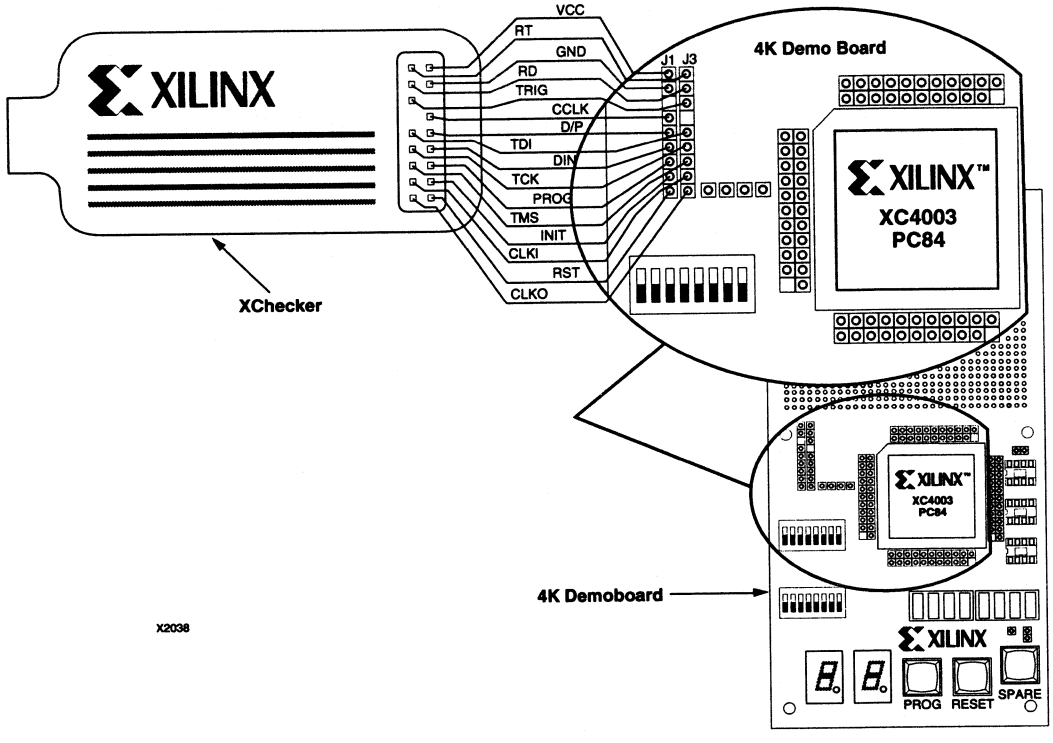


Figure 2-4 XChecker Connections

Table 2-2 J1 Pin Definitions

Name	Description	Pin #	Function
VCC	Power	J1–1	This pin supplies V_{CC} (+5 V) to the XChecker cable. Connects to target system's V_{CC}.
GND	Ground	J1–2	Supplies ground reference to the XChecker cable. Connects to target system's Ground.
CCLK	Configuration Clock	J1–4	During configuration this pin provides CCLK. Connects to the target LCA's CCLK pin.
DONE	Done	J1–5	This pin is used to indicate configuration is complete. Connects to the target LCA's DONE pin
DIN	Data In	J1–6	During configuration, this pin transmits configuration data to the target LCA and is tri-stated at all other times. Connects to the target LCA's DIN pin.
PROG	$\overline{\text{Program}}$	J1–7	For XC4000 parts, this pin provides the reprogram pulse. Connects to $\overline{\text{PROG}}$ pin in XC4000 parts.
INIT	Initialize	J1–8	A logic 0 at this pin during configuration indicates a CRC error has occurred. Connects to the target LCA's INIT pin.
RST	Reset	J1–9	After configuration this pin can be driven Low to Reset the target LCAs internal latches and flip-flops. Not connected to LCA on the demo board; connects to the target LCA's Reset pin.

RESET (SW1) Pushbutton

This momentary switch is connected to MPE (SW4–6) and RST (SW4–1). If the (SW4–1) is on and Reset (SW1) is pressed, an active Low Reset signal is applied to pin 56 of the LCA. To cause a Global Set-Reset (GSR) Reset on this demo board you must, in your design, attach pin 56 to an inverter and then the GSR pin on the Startup symbol.

The inverter must be included between the pad and the Startup symbol because GSR is active High and the XC4000 Demo Board Reset (SW1) is normally pulled High. Since GSR is connected to general purpose I/O, Reset only becomes active after configuration. An active High GSR causes each register device to clear to its default state. GSR does not clear the LCA's configuration memory.

If MPE (SW4–6) is closed and Reset (SW1) is pressed, an active Low Reset signal is applied to the OE/$\overline{R}$ pin of the user's serial PROM. While not asserted, Reset (SW1) is pulled up by a 27K ohm resistor. Header pin J4 provides access to the Reset signal line.

PROG (SW3) Pushbutton

This momentary switch is attached to the $\overline{\text{Program}}$ pin (55) of the LCA. This is an active Low signal. When asserted, the LCA is interrupted and clears its configuration memory. When $\overline{\text{Program}}$ goes High, the LCA finishes the current clear cycle. It then executes another complete clear cycle before it releases INIT and goes into a wait state. It remains in this wait state until INIT is released. Both $\overline{\text{Program}}$ and INIT are pulled up through resistors on the XC4000 Demo Board.

General-Purpose Octal DIP Switch (SW5)

An octal DIP switch is connected directly to eight general-purpose I/O lines of the LCA. They control the logic level to the I/O lines. Any DIP switch position that is HI causes a logic High to appear at the corresponding LCA I/O pin. Pull-down resistors are connected to each switch position so that any DIP switch position that is LO causes a logic Low to appear at the LCA I/O pin. The LCA I/O lines connected to the DIP switch positions are shown in Table 2-3. The leftmost DIP switch is the MSB, referred to as SW5-7. The rightmost DIP switch is the LSB, referred to as SW5-0.

Pins that are connected to these DIP switch positions are intended to be inputs. It is possible for these pins to be driven by another external signal or be configured as outputs if you connect these alternate signals to the header of the corresponding I/O pin. Each of the eight switch positions of the DIP switch (SW5) are isolated from the LCA I/O pins by 4.7 kΩ resistor networks RN2 and RN3.

Table 2-3 Input Pin Connections

Switch	LCA Pin #
SW5–7	19
SW5–6	20
SW5–5	23

Switch	LCA Pin #
SW5–4	24
SW5–3	25
SW5–2	26
SW5–1	27
SW5–0	28

SPARE Pushbutton (SW2)

The XC4000 Demo Board has an extra momentary switch that can be used as an input. The header pins at J5 connect to SW2.

LED Indicators (CR2, CR1)

Eight LEDs are connected to the LCA I/O pins as shown in Table 2-4. The LEDs are configured so the LCA output sinks current. When the LCA output is Low, the LED is turned on. When developing a prototype design, an inverter can be placed on the I/O net to cause the LED to light when a High is present. The leftmost LED above the Program switch is the MSB and is referred to as O7. The rightmost LED is the LSB and is referred to as O0.

Table 2-4 LED I/O Pin Connections

LED	LCA Pin #
O7	61
O6	62
O5	65
O4	66
O3	57
O2	58
O1	59
O0	60

7-Segment Displays (CR3, CR4)

The two 7-segment LED displays are connected to two sets of eight I/O pins on the LCA as shown in Table 2-5.

Table 2-5 7-Segment Display Pin Connections

Segment	LCA Pin # (CR3)	LCA Pin # (CR4)
a	39	49
b	38	48
c	36	47
d	35	46
e	29	45
f	40	50
g	44	51
Decimal Point	37	41

Each LED segment is turned on by driving the corresponding LCA pin Low with Logic 0. The decimal point on CR4, pin 41, also serves as a programming data error indicator. While the LCA drives the INIT pin (41) Low when the LCA is in the internal clearing state, it should be High during configuration. If the decimal point of CR4 is on (INIT Low), there is a programming error. The decimal point on CR3 is tied to the LCA's LDC (Low During Configuration) pin and is on when the LCA is waiting to be configured.

Configuration Switches (SW4)

The leftmost DIP switch is referred to as SW4-7. The rightmost DIP switch is referred to as SW4-0.

PWR Power (SW4-7)

This switch is turned on to connect the alternate power supply to the optional +5 V regulator (U5). The alternate and +5 V power supply should not be used at the same time.

MPE Multiple Program Enable (SW4-6)

This switch is turned on to allow manual reset of the serial PROMs. At power-up, in master-serial mode, the first bitstream is loaded in to the LCA. The RESET switch (SW1) is connected to the OE/$\overline{\text{Reset}}$ of the PROM when MPE is on. If you press the RESET switch (SW1), the serial PROMs are reset. If you then press the PROG switch (SW3), the LCA is loaded with the first bitstream again. If you press the PROG switch (SW3) without the RESET switch (SW1), then the LCA loads a configuration bitstream continuing from the address last read in the serial PROM.

SPE Single Program Enable (SW4-5)

This switch is turned on if you want to load the first configuration bitstream from the serial PROMs each time the Program switch is pressed. When this switch is on, the INIT pin (41) resets the PROMs when the PROG switch (SW3) is pressed. This causes the first configuration bitstream to be reloaded. This switch is turned off when you have different designs loaded in the user serial PROMs and would like to load them consecutively upon pressing the PROG switch (SW3).

NOTE

For proper operation, MPE and SPE must not be On at the same time. MPE and SPE are only used in conjunction with the serial PROMs. Serial PROMs must be configured as OE/$\overline{\text{Reset}}$ to allow MPE and SPE to work properly.

M0, M1, M2 Mode Pins (SW4-4,3,2)

When using the XChecker cable to download, these switches must be off. This places the device in Serial Slave mode; a requirement for downloading. To use the serial PROMs, these switches must be on to place the LCA in the Master Serial mode.

RST Reset (SW4-1)

When this switch is on, the RESET switch (SW1) is routed to pin 56 of the LCA. The internal Global Set-Reset (GSR) requires an active High signal. Thus, to cause a GSR Reset you must, in your design, attach pin 56 to an inverter and then the GSR pin on the Startup symbol. To use the RESET switch (SW1) as a selective Reset or optional momentary Low signal, you simply route pin 56 to the applicable logic. If it is OFF, pressing the RESET switch has no effect on the LCA.

Switch SW4-0 is not used.

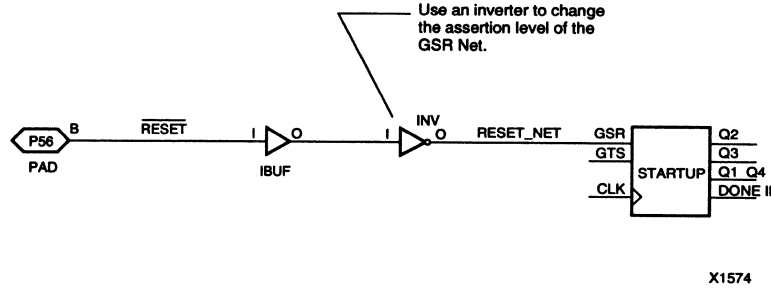


Figure 2-5 Global Reset Usage

Crystal Oscillator Option (Y1)

A standard 4 pin, +5 V crystal oscillator can be added to the XC4000 Demo Board. When power is applied, the oscillator's output is present

at pin 13 of the LCA. To use this oscillator signal, you must configure pin 13 as an input. This oscillator signal, as with any input, is valid and active as soon as the device is configured. This will be at least two clock cycles before the I/Os are released. See the Start–up Timing diagram in the "XC4000 Logic Cell Array Families" section in *The Programmable Logic Data Book.*

Downloading a Design

The following sections describe how to download a demo design using XChecker and loading a configuration PROM. Follow the steps to assure proper operation. A number of demonstration designs have been supplied with the Xilinx XC4000 Demo Board and can be found in the $XACT\examples/core/lite4kA (for workstations) and \XACT\examples\core\lite4kA (for PCs) directory. Please read the text files that accompany these designs.

Downloading with XChecker

If you just want to download a demo design, change to the XACT/examples/core/lite4kA directory and refer to the section Connecting XChecker To The Target System section in the XChecker chapter in this manual. You can also view or edit the demo designs supplied with the XC4000 Demo Board.

IMPORTANT

Make backups before making changes to any demo design files.

1. Place and route the design.

   Produce a routed design (*design*.lca) using a design entry tool and PPR or XDE for manual implementation.

   For XC4000 designs, if you want a global Reset signal you must include the Startup symbol in your design and select the location of the RESET pin. To cause a GSR you must, in your design, attach pin 56 to an inverter and then the GSR pin on the Startup symbol. GSR is active High so an inverter must be included between the pad and the Startup symbol.

2. Generate a bitstream and set configuration options.

   Generate a configuration bitstream for the design (design.bit) with the appropriate configuration options. This is accomplished by using the MakeBits program.

3. Create a PROM File (optional)

   Generate a PROM file (design.mcs, design.tek, or design.exo). This step is optional, as XChecker can use the design.bit file as input. A PROM file is generated by using the MakePROM program.

4. Connect XChecker to the target system.

   When using XChecker to download, only one of the two-keyed connectors are needed. See Figure 2-4.

   The XChecker cable draws its power from the target system through the V_{CC} and GND wires. Therefore, power to XChecker, as well as to the target FPGA, must be stable. You must not connect the XChecker pins to any signals before connecting V_{CC} and Ground to the XC4000 Demo Board.

   When using XChecker to download, only one of the two-keyed connectors are needed.

   Connect the Keyed connector to J1 on the XC4000 Demo Board to the XChecker cable. See Figure 2-4.

5. Set Mode Switches

   When using the XChecker cable, the M0, M1, and M2 switches must be off. This places the device in the Serial Slave mode and ready for downloading.

Starting XChecker

From within XDM (version 2.3 or greater), select XChecker from the Verify menu. You can edit the xchecker.pro file if you desire. You can also invoke XChecker from the operating system prompt.

`xchecker` *design name*

When you start XChecker without specifying any options, XChecker selects the port to which the cable is connected and sets the baud rate to the maximum allowed by the platform.

XChecker indicates that the LCA design is loading. When it is complete, it is indicated that the Done pin went High. At this point, the loaded bit file will function as designed.

Loading with a Configuration PROM

If you already have a design burned in a PROM, skip to "Setting Configuration for a PROM." You can also view or edit the demo designs supplied with the XC4000 Demo Board.

IMPORTANT

Make backups before making changes to any demo design files.

1. Place and route the design.

   Produce a routed design (*design*.lca) using a design entry tool and PPR or XDE for manual implementation.

2. Generate a bitstream and set configuration options.

   Use the MakeBits program to generate a configuration bitstream for the design (design.bit) with the appropriate configuration options.

3. Create a PROM file.

   Use the MakePROM program to generate a PROM file (design.mcs, design.tek or design.exo). See the MakePROM documentation in the *XACT Reference Guide* to create a PROM file. Then, follow the instructions for burning a PROM in the XPP/Serial PROM Programmer chapter later in this manual.

NOTE *The XC1765 PROMs must be programmed with the reset polarity set for active Low.*

4. Place the PROM on the XC4000 Demo Board.

   After you have a PROM that has a configuration bitstream burned in to it, place it into the XC4000 Demo Board with power off. Place it into the first available PROM socket starting with U2. If the design fills more than one PROM, place the first one into U2 and then any subsequent PROM(s) in the next available sockets (U3, U4).

5. Set Mode Switches.

   When using the serial PROM(s), the M0, M1, and M2 switches must be ON. This causes the device to be in the Active Master Bit Serial mode. Set the MPE, SPE, and RST switches to the desired positions.

6. Load the LCA.

   After the PROM has been inserted into the socket and the configuration switches have been set, apply power to the XC4000 Demo Board. The LCA configures and upon DONE going High the design logic becomes active.

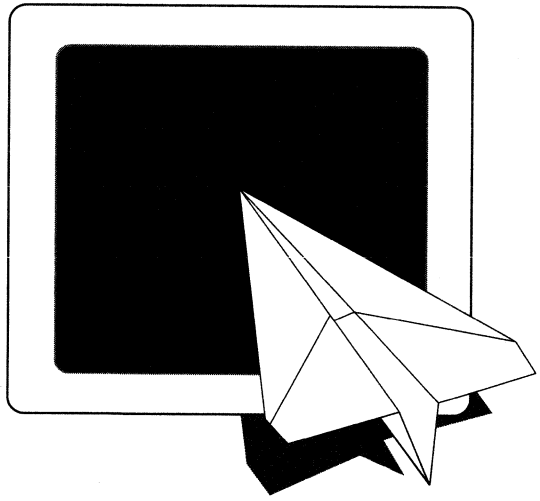
XC4000 Demonstration Designs

The demo design lite4ka.bit incorporates the XC4000s ability to build ROM out of function generators. The ROM is an eight bit 128 word deep ROM. This ROM file was built using MemGen and was filled with eight 16 word patterns. The ROM is addressed by an 8-bit binary counter. By setting the Demo board DIP switches (SW5) as shown in Table 2-6 you can select three different modes. Switch SW5–0 controls

the clock speed. The ROM data can display a single or multiple 16-word patterns. If switch SW5–1 is on, single patterns are displayed; if it is off, multiple patterns are displayed. If the single pattern mode is selected, then switches SW5–5 through SW5–7 are used to decode one of eight patterns. The counter mode is selected by switch SW5–2. This displays the current address or BCD count when SW5-2 is on. If switch SW5–2 is off, then the ROM data is displayed. SW5-0 chooses clock speed. On is 15 Hz, and off is 7.5 Hz.

Table 2-6 Lite4ka.bit Design Switch Configuration

Switch	Mode	On	Off
SW5–0	Clock Speed	Fast	Slow
SW5–1	Pattern Output	Multiple	Single
SW5–2	Counter Output	Display Counter Output	Display ROM Output
SW5–3	n/c		
SW5–4	n/c		
SW5–5	Pattern Decode 0		
SW5–6	Pattern Decode 1		
SW5–7	Pattern Decode 2		



FPGA Design Demonstration Board

XACT Hardware and Peripherals Guide

XACT Hardware and Peripherals Guide — April, 1994 (0401132 01) *Printed in U.S.A.*

FPGA Design Demonstration Board

This Product is Compatible with the Families Indicated.

☑ XC2000	☑ XC3100	☑ XC3100A	☑ XC4000H
☑ XC2000L	☑ XC3000A	☑ XC4000	☐ XC7200
☑ XC3000	☑ XC3000L	☑ XC4000A	☐ XC7300

The FPGA Demo Board is a stand-alone board for experimenting and developing prototypes with FPGAs using Xilinx FPGA architecture. The FPGA Demo Board allows you to become familiar with all the Xilinx FPGA device families and the XACT Development System.

NOTE *Xilinx has three demonstration boards that allow you to test designs. Refer to the table below for the chapter that contains details about the demonstration board you are using.*

Board	Chapter Name
XC3000	XC3000 Design Demonstration Board
XC4000	XC4000 Design Demonstration Board
FPGA	FPGA Design Demonstration Board

The FPGA Demo Board comes with an XC3020APC68 and an XC4003APC84 part. The demo board can be configured either with the XChecker (slave serial mode) or the onboard 17XXX (master serial mode). It has the following features:

- One socket for an XC2000/XC3000 device
- One socket for an XC4000 device
- One 17XXX socket for each FPGA
- An XChecker/Download cable header for each FPGA
- Daisy-chain configuration with the XC4000 device at the head of the chain

- 8-DIP Switches to set up the XC4000 and XC2000/XC3000 FPGAs

XC2000/XC3000 (SW1)	XC4000 (SW2)
1 – INP	1 – PWR (power)
2 – MPE	2 – MPE (multiple configurations)
3 – SPE	3 – SPE (single configurations)
4 – M0	4 – M0
5 – M1	5 – M1
6 – M2	6 – M2
7 – MCLK	7 – RST
8 – DOUT	8 – INIT

- 16 I/O lines are connected between the two FPGAs
- The XC2000/XC3000 has an external relaxation oscillator
- The XC4000 OSC4 library symbol, using pin 19 of the XC4003A, can drive the XC3000 TCLKIN on pin 11 of the XC3020A
- The XC4000 OSC4, using pin 13, can drive the XC2000/XC3000 alternate clock buffer (BCLKIN) on pin 43
- 8-DIP switches to set logic input levels; switch outputs drive both FPGAs; closing switches drive signals to logic 1's
- Program, Reset, and Spare Push Button switches are common to both FPGAs

- XC2000/XC3000 displays use eight LED bars in one row and one 7-segment LED (in Figure 3-1)
- XC4000 displays use eight LED bars in one row and two 7-segment LEDs (in Figure 3-1)

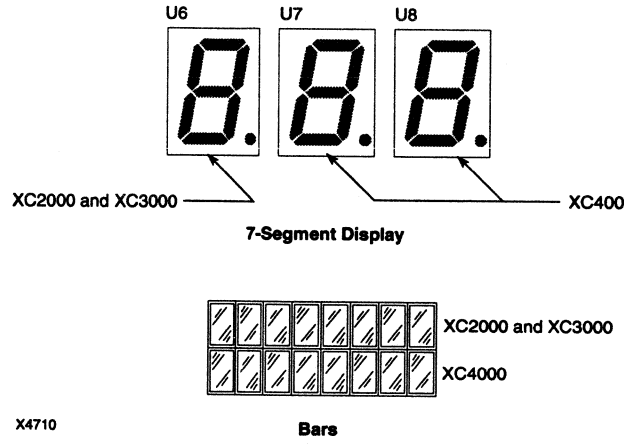
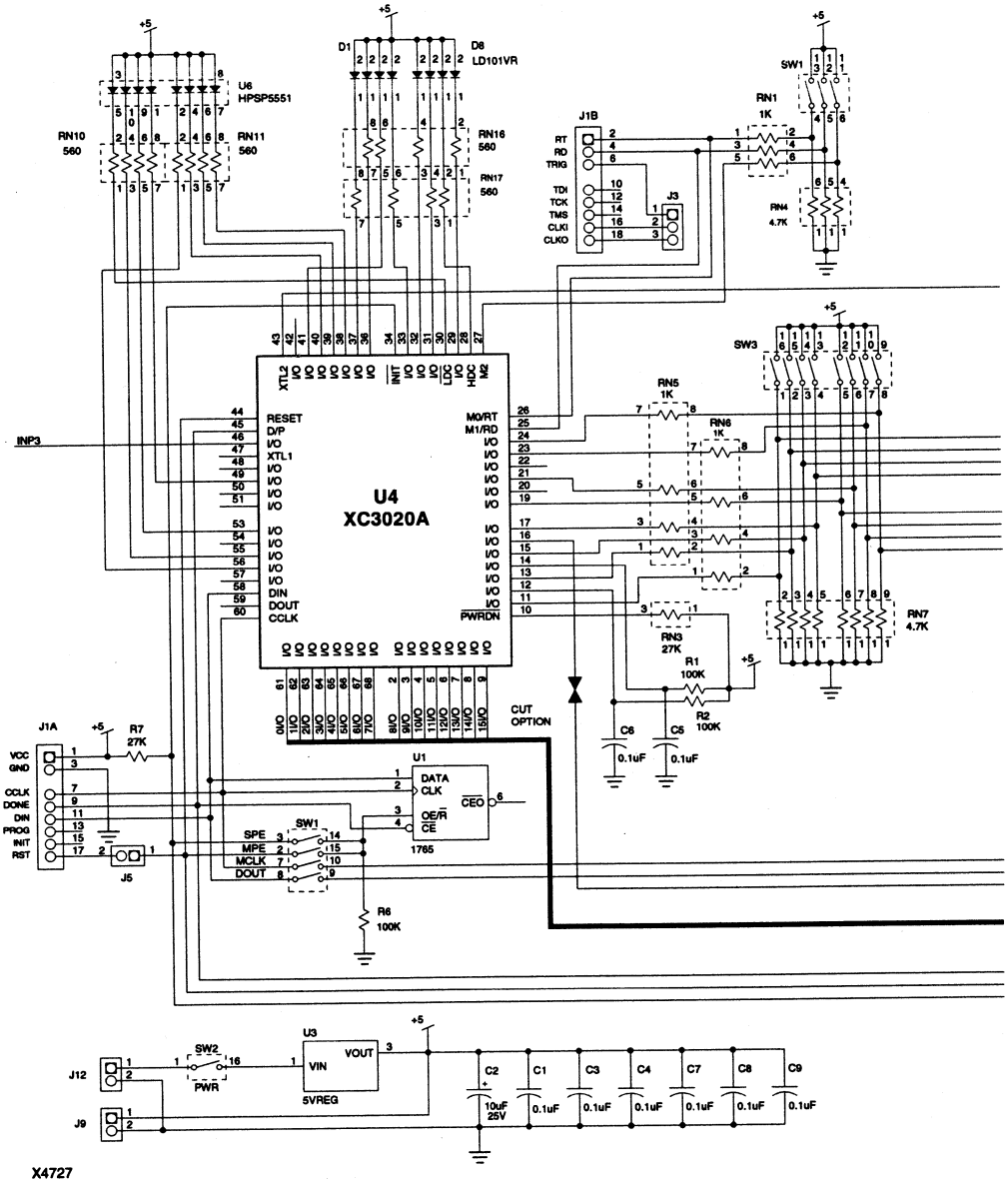


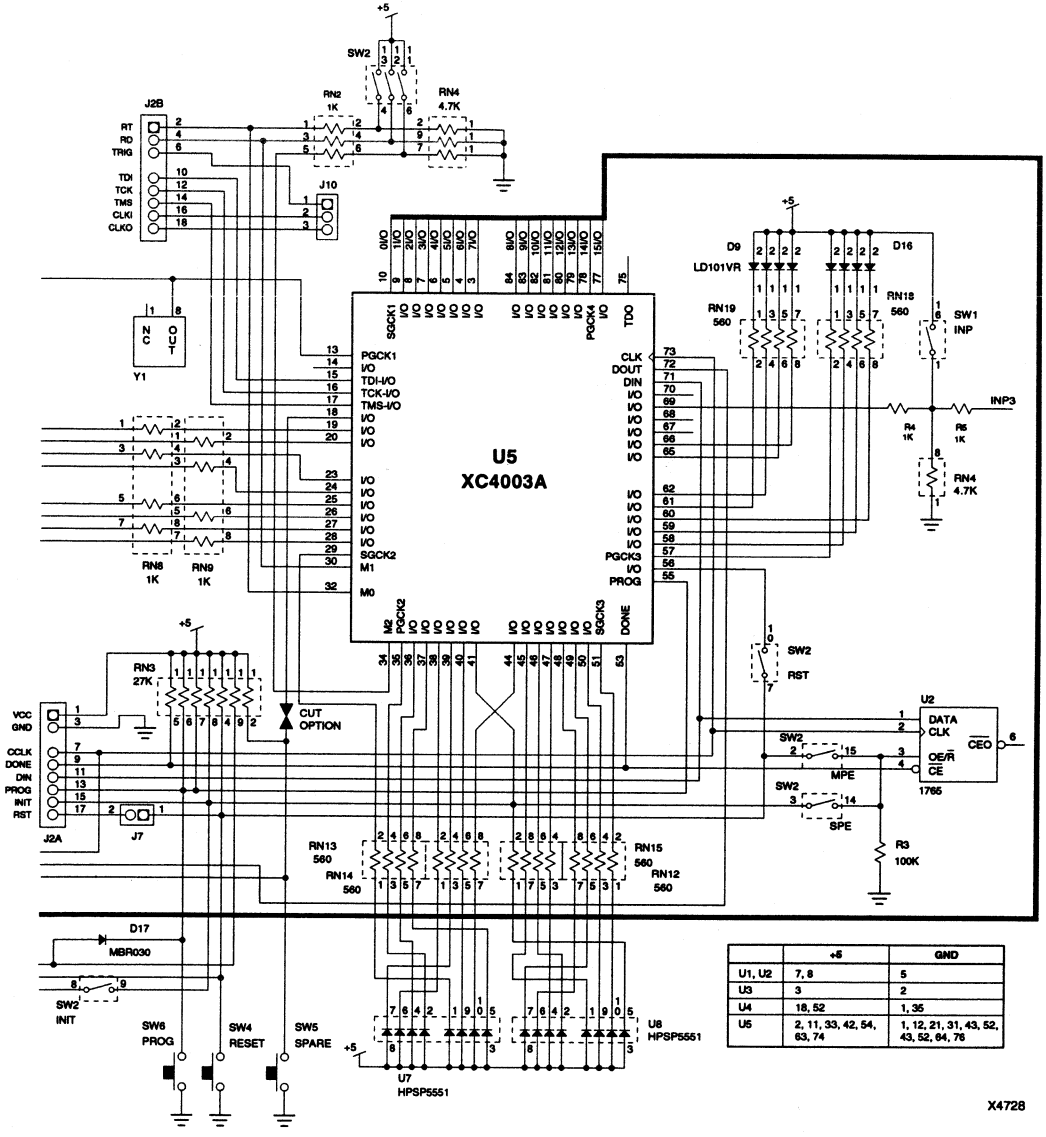
Figure 3-1 FPGA Demo Board Displays

- Space for an optional +5V regulator for battery operation
- Space for an optional crystal oscillator
- Headers for FPGA probe points
- Prototype area on PC board

FPGA Demo Board Components

Figure 3-2 shows the schematic of the FPGA Demo Board. Figure 3-3 shows the component layout of the FPGA Demo Board. Descriptions of the important board components and their board reference designator follow. General components are listed first, then the XC4003A components, followed by the XC3020A components.


U4
XC3020A
RESET
DIN
DOUT
CCLK
PWRDN
M0/RT
M1/RD
M2
INIT
LDC
HDC
INP3
CUT OPTION
D1
D8
LD101VR
U6
HPSP5551
RN10
560
RN11
560
RN16
560
RN17
560
J1B
RT
RD
TRIG
TDI
TCK
TMS
CLKI
CLKO
J3
SW1
RN1
1K
RN4
4.7K
SW3
RN5
1K
RN6
1K
RN7
4.7K
RN3
27K
R1
100K
R2
100K
C6
0.1uF
C5
0.1uF
J1A
VCC
GND
CCLK
DONE
DIN
PROG
INIT
RST
R7
27K
J5
U1
DATA
CLK
CEO
OE/R
OE
1765
SPE
MPE
MCLK
DOUT
R6
100K
J12
SW2
PWR
U3
VIN
VOUT
5VREG
C2
10uF
25V
C1
C3
C4
C7
C8
C9
0.1uF
X4727



	+5	GND
U1, U2	7, 8	5
U3	3	2
U4	18, 52	1, 35
U5	2, 11, 33, 42, 54, 63, 74	1, 12, 21, 31, 43, 52, 43, 52, 64, 76

Figure 3-2 FPGA Demonstration Board Schematic

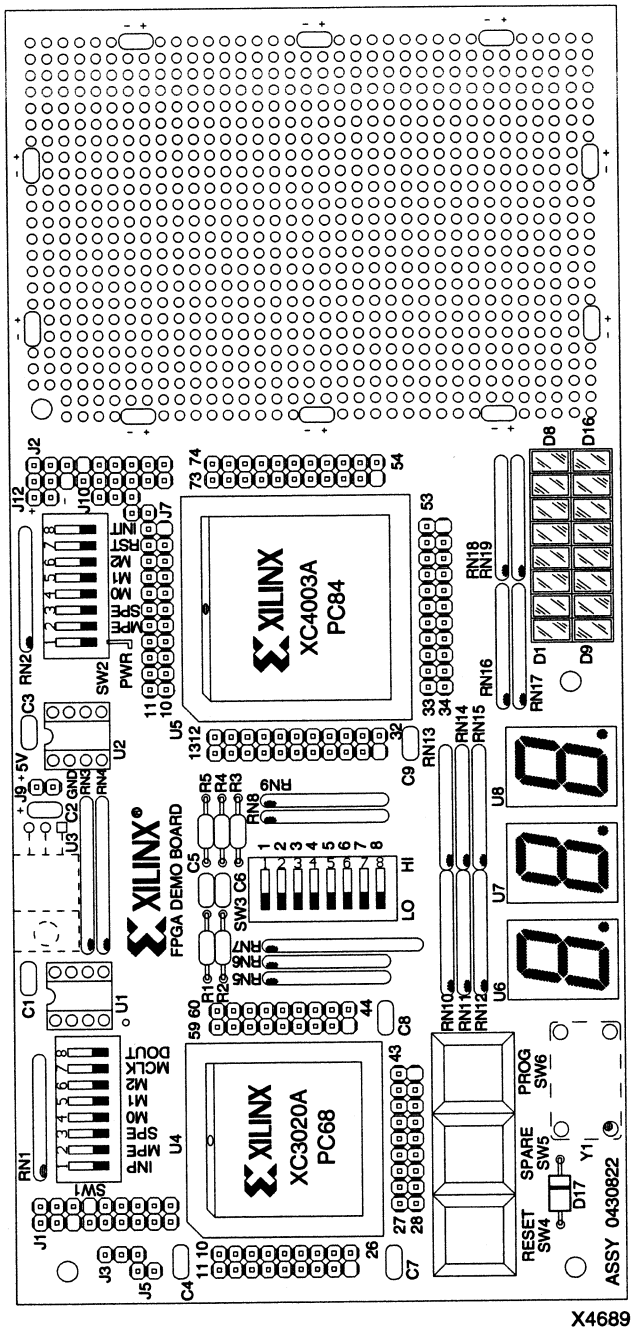


Figure 3-3 FPGA Demo Board Component Layout

FPGA Board General Components

This section describes the general and common components that are on the FPGA Demonstration Board.

+5V Power Connector (J9)

A regulated +5 Volts and Ground are connected to the FPGA Demo Board through connector J9. Pin 1 (square pad) is +5V and pin 2 is Ground. The power supply should provide at least 250mA of current to drive the LED displays.

Unregulated Power Input (J12)

The Unregulated Power Input provides a way to power the FPGA Demo Board from an unregulated source, such as a 9V battery or an AC adapter. The input should be 7VDC – 12VDC @250ma, typically. You must consider the power dissipation requirements of the voltage regulator U3 if the voltage input is greater than 9V.

The Unregulated Power Input J12 provides two holes to connect the unregulated power source. The hole with the square pad, marked with a '+' is the positive input. The other hole, marked with a '–' is circuit Ground. The positive input is connected through the power on–off switch SW2-1 to U3-1, which is the optional +5V regulator. U3 must be installed to use this input.

+5V Regulator Option (U3)

A three terminal +5V regulator, such as the LM2940CT (Figure Figure 3-4) can be installed that allows powering the demo board from an unregulated power supply, such as a 9V battery. Pin 1 (square pad) is Vin, pin 2 is Ground, and pin 3 is +5V out.

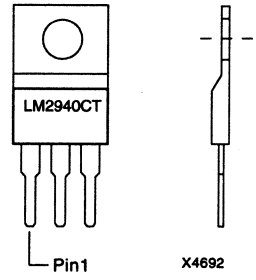


Figure 3-4 LM2940CT +5V Regulator

RESET Pushbutton (SW4)

The RESET pushbutton, when pressed, can apply an active low Reset signal to the FPGAs and configuration PROMs, depending on how the Reset signal routing is configured. RESET is normally pulled high through a 27K ohm resistor.

SPARE Pushbutton (SW5)

The SPARE pushbutton applies an active low signal to the XC3020A on pin 16, and to the XC4003A on pin 18. These pins can be isolated from the switch by using the trace-cut options on the solder side of the board. The trace-cut options appear as point-to-point triangles; the trace-cut option for the XC3020A is under its socket and the trace-cut option for the XC4003A is under R3. The SPARE signal is pulled high through a 27K ohm resistor.

PROGram Pushbutton (SW6)

The PROG pushbutton applies an active low signal to the DONE/PROGRAM input on the XC3020A FPGA socket at pin 45 and to the PROGRAM input on the XC4003A FPGA socket at pin 55. The PROG signal is normally pulled high through a 13.5K ohm resistor.

Eight General-Purpose Input Switches (SW3)

Eight switches are connected to eight general-purpose inputs on both the XC3020A and the XC4003A FPGAs. These switches provide logic input to the FPGAs. An FPGA Input pin is set to a logic '1' when a switch is on, and a logic '0' when a switch is off. See Figure 3-5.

The FPGA pins connected to this switch are intended for use as inputs; however, each FPGA pin has a 1k ohm resistor that isolates it from the switch so it is possible to define them as outputs. It is also possible to drive them from an external source by connecting that signal to the FPGA Probe Point Header. Table 3-1 lists the FPGA pin connections.

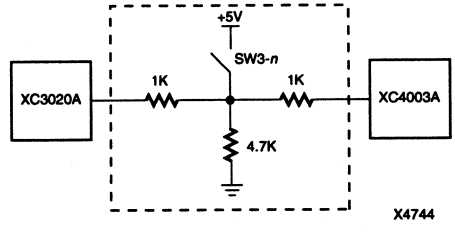


Figure 3-5 FPGA Demo Board General Purpose Switch

Table 3-1 FPGA Pin Connections

Switch	XC3020A	XC40003A
SW3-1	11	19
SW3-2	13	20
SW3-3	15	23
SW3-4	17	24
SW3-5	19	25
SW3-6	21	26
SW3-7	23	27
SW3-8	24	28

7-Segment Displays (U6, U7, U8)

Three 7-segment displays are included with the left-most display (U6) connected to the XC3020A FPGA, and the right two displays (U7 and U8) connected to the XC4003A. .

Each LED segment is turned on by driving the corresponding FPGA pin 'LOW' with a logic '0.' The Decimal Point on U8 is connected to the INIT pin of the XC4003A (pin 41), and serves as a programming error indicator. The Decimal Point should be on while the FPGA is in its internal clearing state, then it should remain off during configuration. If the Decimal Point comes back on, there has been a programming error.

The Decimal Points on U6 and U7 are tied to the LDC (LOW During Configuration) pins of the XC3020A and XC4003A, respectively. The Decimal Points are on while the FPGAs are waiting to be configured.

Table 3-2 shows the I/O pin definitions. Figure 3-6 shows the 7-segment display.

Table 3-2 7-Segment I/O Connections

Display Segment	XC3020A	XC40003A	XC40003A
	U6	U7	U8
a	38	39	49
b	39	38	48
c	40	36	47
d	56	35	46
e	49	29	45
f	53	40	50
g	55	44	51
decimal point	30	37	41

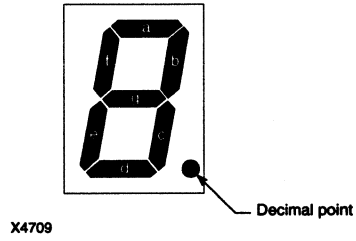


Figure 3-6 7-Segment Display

LED Indicators (D1-D8, D9-D16)

Eight LEDs are connected to the I/O pins of each FPGA. D1-D8 connect to the XC3020A and D9-D16 connect to the XC4003A. An LED can be turned on by driving its corresponding FPGA pin low with a logic '0.' Table 3-3 shows the FPGA connections.

Table 3-3 LED Indicators for XC3020A and XC4003A

LED	XC3020A Pin	LED	XC4003A Pin
D1	37	D9	61
D2	36	D10	62
D3	41	D11	65
D4	33	D12	66
D5	32	D13	57
D6	31	D14	58
D7	28	D15	59
D8	29	D16	60

I/O Line Connections

The XC3020A and XC4003A parts have 16 I/O lines connected together as shown in Table 3-4.

Table 3-4 I/O Line Connections for the XC3020A and XC4003A

I/O Line	XC3020A Pin	XC4003A Pin
I/O 0	61	10
I/O 1	62	9
I/O 2	63	8
I/O 3	64	7
I/O 4	65	6
I/O 5	66	5
I/O 6	67	4
I/O 7	68	3
I/O 8	2	84

I/O Line	XC3020A Pin	XC4003A Pin
I/O 9	3	83
I/O 10	4	82
I/O 11	5	81
I/O 12	6	80
I/O 13	7	79
I/O 14	8	78
I/O 15	9	77

Optional Crystal Oscillator (Y1)

A standard 4-pin crystal oscillator can be added to the FPGA Demo Board. The oscillator output drives the XC3020A XTL2 input at pin 43 and the XC4003A PGCK1 input at pin 13.

Prototype Area

The Prototype area is a 0.1–inch grid of holes where additional circuitry can be added to the demo board. A +5V bus (component side) and a Ground bus (solder side) are available on the perimeter of this area. There are also locations for filter capacitors.

XC4003A Components

This section describes the XC4003A components on the FPGA Demo Board.

XC40003A FPGA and Socket (U5)

The XC4003A FPGA occupies socket U5 on the demo board.

XC4003A Probe Points

All pins of the XC4003A are connected to the headers that surround the FPGA socket. These pins provide convenient points for probing signals or to make wirewrap connections to other circuitry, such as on the prototype area. Pin numbering increases inside row to outside, counterclockwise. See the corners of each header for the starting number of that header.

XC4003A Configuration Switches (SW2)

The following explanations describe each of the SW2 switches.

PWR — Power (SW2-1)

This switch turns the Unregulated Power Input on or off to the +5V regulator U3.

MPE — Multiple Program Enable (SW2-2)

With MPE turned on and SPE turned off, the Configuration PROM (U2) is reset by the RESET pushbutton (SW4). Configuration mode must be set to Master-Serial. After a Reset or power-up, the first bitstream stored in the serial PROM is loaded into the XC4003A. Pressing RESET, resets the serial PROM address pointer. Pressing PROG (SW6), loads the XC4003A with the first bitstream again. If you press PROG without pressing RESET, the XC4003A is loaded with the next bitstream that is stored in the serial PROM. The size of the serial PROM limits the number of bitstreams that can be sequentially loaded.

SPE — Single Program Enable (SW2-3)

With SPE turned on and MPE turned off, the Configuration PROM (U2) is reset by the XC4003A's INIT output, which is driven low whenever PROG (SW6) is pressed. The first bitstream stored in the serial PROM is loaded into the XC4003A.

NOTE *MPE and SPE must not be on at the same time. MPE and SPE are only used in conjunction with the serial PROMs. The serial PROMs must be configured as OE/$\overline{\text{Reset}}$ to allow MPE and SPE to function properly.*

M0, M1, M2 — Mode Pins (SW2-4,5,6)

These three switches must be on to configure the XC4003A using the XChecker/Download Cable. When these switches are on, the FPGA is in Serial Slave mode. To configure the XC4003A from the onboard serial PROM, these three switches must be off, placing the FPGA in Master Slave mode.

RST — Reset (SW2-7)

When this switch is on, it connects the RESET pushbutton (SW4) to XC4003A pin 56.

INIT — Initialize (SW2-8)

When this switch is on, it connects the XC3020A INIT pin to the XC4003A INIT pin. This connection is used to configure FPGAs in a daisy chain with the XC4003A at the head of the chain.

NOTE *INIT should only be used to configure FPGAs in a daisy chain.*

XChecker/Download Cable Connector (J2)

Table 3-5 provides a detailed description of the XChecker/Download Cable connector J2.

Table 3-5 XChecker/Download Cable Connector (J2)

Pin	Name	Function	Pin	Name	Function
J2–1*	VCC	Supplies +5V to XChecker cable.	J2–2	RT	Read Trigger allows XChecker to trigger a read-back of the XC4003A. Connects to XC4003A pin 32.
J2–3*	GND	Supplies ground reference to XChecker cable.	J2–4	RD	Used by XChecker for Readback Data. Connects to XC4003A pin 30.
J2–5	N.C.		J2–6	TRIG	An XChecker input that allows an external event to trigger readback of the XC4003A or outputting a burst of clocks to the XC4003A. Connects to tie-point J10-1.
J2–7*	CCLK	During configuration or readback, this pin provides the Clock. Connects to XC4003A input pin 73.	J2–8	N.C.	
J2–9*	DONE	Indicates when configuration is complete. Connects to XC4003A output pin 53.	J2–10**	TDI	Used for boundary-scan data input to the XC4003A. Connects to XC4003A pin 15.
J2–11*	DIN	During configuration, this pin provides configuration data. Connects to XC4003A DIN input pin 71.	J2–12**	TCK	Boundary scan Clock input to the XC4003A. Connects to XC4003A pin 16.

Pin	Name	Function	Pin	Name	Function
J2–13	PROG	This pin provides the program pulse that causes the FPGA to configure. Connects to XC4003A PROG input pin 55.	J2–14**	TMS	Boundary scan Mode input to the XC4003A. Connects to XC4003A pin 17.
J2–15	INIT	If a CRC error occurs during configuration, this pin goes low. Connects to XC4003A INIT pin 41.	J2–16	CLKI	A system clock input to XChecker to be controlled and output on CLKO. Connects to tie point J10-2.
J2–17	RST	Connects to jumper J7. If connected, allows XChecker to provide a Reset input (same as pressing the Reset button).	J2–18	CLKO	A system clock output controlled by XChecker. Used to single-step or burst clocks to the XC4003A. Connects to tie point J10-3.

Pins marked with an asterisk (*) indicate pins that the Download Cable supports. With the Download Cable connected, J2-9 provides both the DONE and PROG functions. Since the XC4003A requires a Program input that is separate from DONE, the PROG button must be pressed before configuring the XC4003A.

Pins marked with a double asterisk (**) indicate Boundary Scan operations that are not supported with XChecker.

Jumper J7 and Tiepoints J10 (1-3)

Jumper J7 allows the XChecker signal RST on J2-17 to drive the reset line on the demo board. Tiepoint pins are provided for jumpering the following XChecker signals into the circuit. Tiepoint J10-1 is connected to TRIG on J2-6; Tiepoint J10-2 is connected to CLK1 on J2-16; and, Tiepoint J10-3 is connected to CLK0 on J2-18. See Table 3-5 for more details on the XChecker/Download Cable connections.

Serial PROM Socket (U2)

This Serial PROM is used to configure the XC4003A or the XC4003A and XC3020A connected in a daisy chain. The configuration mode must be the Master Serial mode to configure from the Serial PROM.

XC3020A Components

This section describes the XC3020A components on the FPGA Demo Board.

XC3020A FPGA and Socket (U4)

The XC3020A FPGA occupies socket U4 on the demo board.

XC3020A Probe Points

All pins of the XC3020A FPGA are connected to the headers that surround the FPGA socket. These pins provide convenient points for probing signals or making wirewrap connections to other circuitry, such as the prototype area. Pin numbering increases inside row to outside, counterclockwise. See the corners of each header for the starting number of that header. Refer to Table 3-4 for information.

The XC3020A I/O pins 2-9 and 61-68 are connected to XC4003A pins 3-10 and 77-84, respectively. The XC3020A pins share the XC4003A Probe Points header.

XC3020A Configuration Switches (SW1)

The following explanations describe each of the SW1 switches.

INP — Input Switch (SW1-1)

This is an extra switch, which has been connected to provide an extra logic input to the XC3020A pin 46 and the XC4003A pin 69. The FPGA input pins are set to a logic '1' when the switch is on and a logic '0' when the switch is off.

The FPGA pins connected to this switch are intended for use as inputs; however, they have a 1K ohm resistor that isolates them from the switch, so it is possible to define them as outputs. It is also possible to drive them from an external source by connecting that signal to the FPGA Probe Point Header. See Figure 3-7 for details.

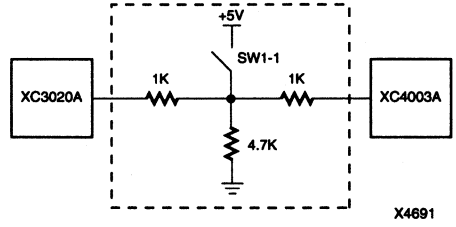


Figure 3-7 Configuration Switch SW1

MPE — Multiple Program Enable (SW1-2)

When MPE is on and SPE is off, the Configuration PROM (U1) is reset by the RESET pushbutton (SW4). Configuration must be set to the Master Serial mode. After a Reset or power-up, the first bitstream

stored in the serial PROM is loaded into the XC3020A FPGA. If RESET is pressed, the serial PROM address pointer is reset. If PROG (SW6) is pressed, the XC3020A is loaded with the first bitstream, again. If PROG is pressed without RESET, then the XC3020A is loaded with the next bitstream stored in the serial PROM. The number of bitstreams that can be sequentially loaded is limited by the size of the serial PROM.

SPE — Single Program Enable (SW1-3)

When SPE is on and MPE is off, the Configuration PROM (U1) is reset by the XC3020A's INIT output, which is driven low whenever PROG (SW6) is pressed. The first bitstream stored in the serial PROM is loaded into the XC3020A FPGA.

NOTE *MPE and SPE must not be on at the same time. MPE and SPE are only used in conjunction with the serial PROMs. The serial PROMs must be configured as OE/RESET to allow MPE and SPE to function properly.*

M0, M1, M2 — Mode Pins (SW1-4,5,6)

To configure the XC3020A using the XChecker/Download Cable, these switches must be on, placing the FPGA in serial slave mode. To configure from the onboard serial PROM, these switches must be off to place the FPGA in master slave mode.

MCLK — Master Clock (SW1-7)

When this switch is on, it connects the XC4003A Configuration Clock (pin 73) to the Configuration Clock on the XC3020A (pin 60). This connection is used to configure FPGAs in a daisy chain with the XC4003A at the head.

DOUT — Data Out (SW1-8)

When this switch is on, it connects the XC4003A Data Out line (pin 72) to the Data In line of the XC3020A. This configuration is used to configure FPGAs in a daisy chain with the XC4003A at the head.

NOTE *MCLK and DOUT should only be used to configure the FPGAs in a daisy chain.*

XChecker/Download Cable Connecter (J1)

Table 3-6 describes the pins of the XChecker/Download cable J1 connector.

Table 3-6 XChecker/Download Cable Connector J1

Pin	Name	Function	Pin	Name	Function
J1-1*	VCC	Supplies +5V to the XChecker cable.	J1-2	RT	Read Trigger allows XChecker to trigger a readback of the XC3020A. Connects to XC3020A pin 26.
J1-3*	GND	Supplies ground reference to XChecker cable.	J1-4	RD	Used by XChecker for Readback Data. Connects to XC3020A pin 25.
J1-5*	N.C.		J1-6	TRIG	An XChecker input that allows an external event to trigger readback of the XC3020A or outputting a burst of clocks to the XC3020A. Connects to tiepoint J3-1.
J1-7*	CCLK	During configuration or readback, this pin provides the Clock. Connects to XC3020A input pin 50.	J1-8	N.C.	
J1-9*	D/P	Used to start Configuration and indicate completion. Connects to XC3020A DONE/PROGRAM pin 45.	J1-10	N.C.	
J1-11*	DIN	During Configuration, this pin provides configuration data. Connects to XC3020A DIN input pin 58.	J1-12	N.C.	
J1-13	N.C.		J1-14	N.C.	
J1-15	N.C.		J1-16	CLKI	System clock input to XChecker to be controlled and output on CLKO. Connects to tiepoint J3-2.
J1-17	RST	Connects to jumper J5. If connected, allows XChecker to provide a Reset input (same as pressing Reset button).	J1-18	CLKO	System clock output controlled by XChecker. Used to single-step or burst clocks to the XC3020A. Connects to tiepoint J3-3.

Pins with an asterisk (*) are supported by the Download cable.

Jumper J5 allows the XChecker signal RST on J1-17 to drive the reset line on the demo board. Tiepoint pins are provided for jumpering the following XChecker signals into your circuit. Tiepoint J3-1 is

connected to TRIG on J1-6; Tiepoint J3-2 is connected to CLK1 on J1-16; and, Tiepoint J3-3 is connected to CLK0 on J1-18. See Table 3-6 for more information on the XChecker/Download Cable connections.

Serial PROM Socket (U1)

This Serial PROM is used for configuring the XC3020A. You must use the Master Serial mode to configure from the Serial PROM.

Relaxation Oscillator Components (R1 C5, R2 C6)

R1, C5 and R2, C6 are two RC networks connected to the XC3020A at pins 12 and 14. These RC networks are for use in a relaxation oscillator such as, the circuit is shown in Figure 3-8.

With the components provided, R1 = R2 = 100k ohms and C5 = C6 = 0.1uF, the oscillator generates an output frequency of approximately 100Hz.

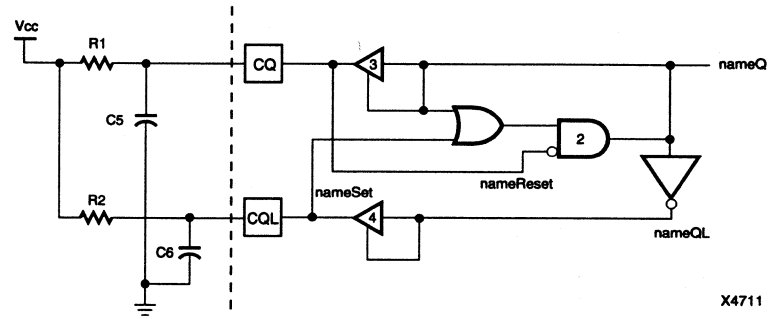


Figure 3-8 Relaxation Oscillator Schematic

Figure 3-9 shows the RC network.

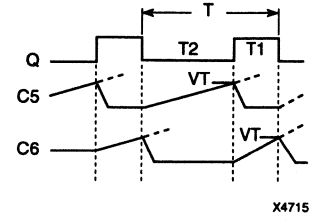


Figure 3-9 Network Calculation Formula

The formula for calculating the RC network is as follows:
T = T1 + T2 = N ((R1C5) + (R2C6))

Where:
N = approximately 0.35 for TTl threshold
= approximately 0.75 for CMOS threshold
when each capacitor is allowed by the LCA to discharge during the opposite timing phase.

Mode Switch Settings

This section describes the SW1 and SW2 switch settings for configuring the XC3020A and XC4003A:

- From the XChecker/Download Cable
- From the Serial PROM (single program)
- From the Serial PROM (multiple program)
- In a daisy chain

Table 3-7 Configuring the XC3020A from the XChecker/Download Cable

Switch	Name	Position	Switch	Name	Position
SW1-1	INP	X	SW2-1	PWR	X
SW1-2	MPE	OFF	SW2-2	MPE	X
SW1-3	SPE	OFF	SW2-3	SPE	X
SW1-4	M0	ON	SW2-4	M0	X
SW1-5	M1	ON	SW2-5	M1	X
SW1-6	M2	ON	SW2-6	M2	X
SW1-7	MCLK	OFF	SW2-7	RST	X
SW1-8	DOUT	OFF	SW2-8	INIT	OFF

X indicates Don't Care.

Table 3-8 Configuring the XC4003A from the XChecker/Download Cable

Switch	Name	Position	Switch	Name	Position
SW1-1	INP	X	SW2-1	PWR	X
SW1-2	MPE	X	SW2-2	MPE	OFF
SW1-3	SPE	X	SW2-3	SPE	OFF
SW1-4	M0	X	SW2-4	M0	ON
SW1-5	M1	X	SW2-5	M1	ON
SW1-6	M2	X	SW2-6	M2	ON
SW1-7	MCLK	OFF	SW2-7	RST	X
SW1-8	DOUT	OFF	SW2-8	INIT	OFF

X indicates Don't Care.

When configuring both the XC3020A and XC4003A using the XChecker/Download Cable, configure the XC4003A first. If the order is reversed, the XC3020A's configuration is lost when configuring the XC4003A because the $\overline{\text{PROG}}$ signal is connected directly to the XC4003A $\overline{\text{PROG}}$ input and through a diode to the XC3020A DONE/$\overline{\text{PROG}}$ input.

Table 3-9 Configuring the XC3020A from the Serial PROM (Single Program)

Switch	Name	Position	Switch	Name	Position
SW1-1	INP	X	SW2-1	PWR	X
SW1-2	MPE	OFF	SW2-2	MPE	X
SW1-3	SPE	ON	SW2-3	SPE	X
SW1-4	M0	OFF	SW2-4	M0	X
SW1-5	M1	OFF	SW2-5	M1	X
SW1-6	M2	OFF	SW2-6	M2	X
SW1-7	MCLK	OFF	SW2-7	RST	X
SW1-8	DOUT	OFF	SW2-8	INIT	OFF

X indicates Don't Care.

Table 3-10 Configuring the XC4003A from the Serial PROM (Single Program)

Switch	Name	Position	Switch	Name	Position
SW1-1	INP	X	SW2-1	PWR	X
SW1-2	MPE	X	SW2-2	MPE	OFF
SW1-3	SPE	X	SW2-3	SPE	ON
SW1-4	M0	X	SW2-4	M0	OFF
SW1-5	M1	X	SW2-5	M1	OFF
SW1-6	M2	X	SW2-6	M2	OFF
SW1-7	MCLK	OFF	SW2-7	RST	X
SW1-8	DOUT	OFF	SW2-8	INIT	OFF

X indicates Don't Care.

Table 3-11 Configuring the XC3020A from the Serial PROM (Multiple Program)

Switch	Name	Position	Switch	Name	Position
SW1-1	INP	X	SW2-1	PWR	X
SW1-2	MPE	ON	SW2-2	MPE	X
SW1-3	SPE	OFF	SW2-3	SPE	X
SW1-4	M0	OFF	SW2-4	M0	X
SW1-5	M1	OFF	SW2-5	M1	X
SW1-6	M2	OFF	SW2-6	M2	X
SW1-7	MCLK	OFF	SW2-7	RST	X
SW1-8	DOUT	OFF	SW2-8	INIT	OFF

X indicates Don't Care.

Table 3-12 Configuring the XC4003A from the Serial PROM (Multiple Program)

Switch	Name	Position	Switch	Name	Position
SW1-1	INP	X	SW2-1	PWR	X
SW1-2	MPE	X	SW2-2	MPE	ON
SW1-3	SPE	X	SW2-3	SPE	OFF
SW1-4	M0	X	SW2-4	M0	OFF
SW1-5	M1	X	SW2-5	M1	OFF
SW1-6	M2	X	SW2-6	M2	OFF
SW1-7	MCLK	OFF	SW2-7	RST	X
SW1-8	DOUT	OFF	SW2-8	INIT	OFF

X indicates Don't Care.

Table 3-13 Configuring the XC3020A and XC4003A in a Daisy Chain from the XChecker/Download Cable

Switch	Name	Position	Switch	Name	Position
SW1-1	INP	X	SW2-1	PWR	X
SW1-2	MPE	OFF	SW2-2	MPE	OFF
SW1-3	SPE	OFF	SW2-3	SPE	OFF
SW1-4	M0	ON	SW2-4	M0	ON
SW1-5	M1	ON	SW2-5	M1	ON
SW1-6	M2	ON	SW2-6	M2	ON
SW1-7	MCLK	ON	SW2-7	RST	X
SW1-8	DOUT	ON	SW2-8	INIT	ON

X indicates Don't Care.

Table 3-14 Configuring the XC3020A and XC4003A in a Daisy Chain from the Serial PROM (Single Program)

Switch	Name	Position	Switch	Name	Position
SW1-1	INP	X	SW2-1	PWR	X
SW1-2	MPE	OFF	SW2-2	MPE	OFF
SW1-3	SPE	OFF	SW2-3	SPE	ON
SW1-4	M0	ON	SW2-4	M0	OFF
SW1-5	M1	ON	SW2-5	M1	OFF
SW1-6	M2	ON	SW2-6	M2	OFF
SW1-7	MCLK	ON	SW2-7	RST	X
SW1-8	DOUT	ON	SW2-8	INIT	ON

X indicates Don't Care.

Table 3-15 Configuring the XC3020A and XC4003A in a Daisy Chain from the Serial PROM (Multiple Program)

Switch	Name	Position	Switch	Name	Position
SW1-1	INP	X	SW2-1	PWR	X
SW1-2	MPE	OFF	SW2-2	MPE	ON
SW1-3	SPE	OFF	SW2-3	SPE	OFF
SW1-4	M0	ON	SW2-4	M0	OFF
SW1-5	M1	ON	SW2-5	M1	OFF
SW1-6	M2	ON	SW2-6	M2	OFF
SW1-7	MCLK	ON	SW2-7	RST	X
SW1-8	DOUT	ON	SW2-8	INIT	ON

X indicates Don't Care.

FPGA Demonstration Board Operation

NOTE

The information in this section of the chapter is applicable to both the XC3020A and the XC4003A devices. However, for clarity only references to the XC4003A device exist.

The operation of the Demonstration board is discussed in three sections.

- Downloading with XChecker
- Loading with a Configuration PROM
- FPGA Demonstration Designs

Downloading with XChecker

You must follow the recommended design flow to assure proper operation. A demonstration design has been supplied with the Xilinx FPGA Demo Board in the XACT\examples\core\litefpga directory. Please read the text files that accompany these designs to acquaint yourself with the information.

If you just want to download a demo design, change to the XACT\examples\core\litefpga directory and refer to the XChecker chapter in this manual for more information. You can also view or edit the demo designs supplied with the FPGA Demo Board.

IMPORTANT

Make backups before making changes to any demo design files.

1. Produce a routed design (*design*.lca) using a design entry tool and the appropriate place and route tool or XDE for manual implementation.

   For XC4000 designs, if you want a global Reset signal you must include the Startup symbol in your design and select the location of the RESET pin. To cause a GSR you must, in your design, attach pin 56 to an inverter and then the GSR pin on the Startup symbol. GSR is active High so an inverter must be included between the pad and the Startup symbol.

2. Generate a bitstream for the design (*design*.bit) with the appropriate configuration options using the MakeBits program.
3. Create a PROM File (optional).

4. Generate a PROM file (*design*.mcs, *design*.tek, or *design*.exo) using the MakePROM program. This step is optional, as XChecker can use the *design*.bit file as input.
5. Connect XChecker to the target system.

   The XChecker cable draws its power from the target system through the V_{CC} and GND wires. Therefore, power to XChecker, as well as to the target FPGA, must be stable. You must not connect the XChecker pins to any signals before connecting V_{CC} and Ground to the FPGA Demo Board.

   When using XChecker to download, only one of the two-keyed connectors are needed.
6. Connect the Keyed connector to J1 (for the XC3020A) and J2 (for the XC4003A) on the FPGA Demo Board to the XChecker cable.
7. Set the Mode Switches.

   When using the XChecker cable, the M0, M1, and M2 switches must be on. This setting causes the device to be in the Serial Slave mode. Refer to Table 3-13 for the switch settings necessary to configure a daisy chain.

Starting XChecker

From within XDM (version 2.3 or greater), select XChecker from the Verify menu. You can edit the xchecker.pro file if you desire. You can also start XChecker from the operating system prompt.

xchecker *design name*

When you start XChecker with no options, XChecker selects the port where the cable is found and sets the baud rate to the maximum allowed by the platform.

XChecker indicates that the FPGA design is loading. When it is complete, it is indicated that the DONE pin went High. At this point, the loaded bit file will function as designed.

Loading with a Configuration PROM

If you already have a design burned in a PROM, skip to "Setting Configuration for a PROM." You can also view or edit the demo designs supplied with the FPGA Demo Board.

IMPORTANT

Make backups before making changes to any demo design files.

1. Place and route the design.

   Produce a routed design (*design*.lca) using a design entry tool and the appropriate place and route tool or XDE for manual implementation.

2. Generate a configuration bitstream for the design (*design*.bit) with the appropriate configuration options using the MakeBits program.

3. Create a PROM file.

   Generate a PROM file (design.mcs, design.tek or design.exo) using the MakePROM program. See the MakePROM documentation in the *XACT Reference Guide* to create a PROM file and then follow the instructions for burning a PROM in the XPP serial configuration PROM programmer documentation.

NOTE *The XC17XXX PROMs must be programmed with the reset polarity set for active Low.*

4. Place the PROM on the FPGA Demo Board.

   After you have a PROM that has a configuration bitstream burned in to it, place it into the FPGA Demo Board with power off. Use socket U2 for XC4003A devices and for XC4003A and XC3020A devices in a daisy chain with the XC4003A at the head of the chain. Use socket U1 for XC3020A devices.

5. Set Mode Switches.

   When using the serial PROM(s), the M0, M1, and M2 switches must be off. This setting causes the device to be in the Active Master Serial mode. Set the MPE, SPE, and RST switches to the desired positions. Refer to Table 3-14 and Table 3-15 for switch settings required to configure a daisy chain.

6. Load the FPGA.

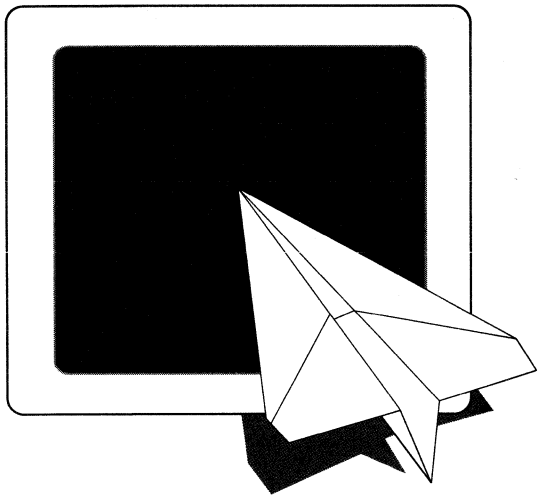
   After the PROM has been inserted into the socket and the configuration switches have been set, apply power to the FPGA Demo Board. The FPGA configures and upon DONE going High the design logic becomes active.

Demonstration Designs

The example design in $XACT/examples/core/litefpga (workstations) and \XACT\examples\core\litefpga (PCs) incorporates the ability of the XC4003A to build ROM out of function generators. The ROM macros store a sequence of patterns that are displayed on the 7-segment displays and the LED bar graphs of the FPGA Demo board.

The litefpga.mcs design file is a daisy chain of an XC4003A design and an XC3020A design. When you are ready to download the litefpga.mcs design, set up the FPGA Demo board as shown in Table 3-13.

The design schematics are available by calling the Xilinx Technical Support Hotline.



XPP/Serial Config.
PROM Programmer

XACT Hardware and Peripherals Guide

XACT Hardware and Peripherals Guide — April, 1994 (0401132 01) *Printed in U.S.A.*

XPP/Serial Configuration PROM Programmer

This Program is Compatible with the Families Indicated.

☑ XC2000	☑ XC3100	☑ XC3100A	☑ XC4000H
☑ XC2000L	☑ XC3000A	☑ XC4000	☐ XC7200
☑ XC3000	☑ XC3000L	☑ XC4000A	☐ XC7300

When using Xilinx Serial Configuration PROMs to configure XC2000-, XC3000-, and XC4000-family LCAs, you can program them with the HW112 Serial Configuration PROM programmer, or one of the third-party PROM programmers listed in *The Programmable Logic Data Book*. The HW112 connects to any of the serial ports on a PC, or a Sun, DEC, or Apollo workstation.

The XPP program controls the HW112 programmer via the keyboard on PCs, and the mouse and/or keyboard on workstations. Since the XPP software is not node-locked, you can install the software and programming unit on multiple PCs or workstations.

The HW112 supports the following PROMs in 8-pin DIP packages:

- XC1718D, XC1718L
- XC1736A, XC1736D, AM1736
- XC1765, XC1765D, XC1765L, AM1765
- XC17128
- XC17256D

To configure these PROMs in 20-pin PLCC packages, you need to use the HW113-PC20 adapter, which fits on top of the HW112. The HW113-SO8 adapter is used with small outline 8-pin (SO8) packages.

XPP supports the programming of multiple device types in a single session on PC platforms. Large configuration patterns of daisy-chained patterns can be programmed over different device types.

Programming Flow

To configure a PROM, you must first compile the LCA design into a BIT file by using the MakeBits program within the XACT Development System. When configuring a single LCA, you can use the BIT file as input to the PROM programmer. For a daisy-chain of LCAs,

you must create a combined hex file using the MakePROM program. The XPP programming software then downloads the data to the programming unit and programs the device. (Figure 4-1.)

The MakePROM utility currently supports three hex file formats listed below; you can use any of these formats with XPP to program serial configuration PROMs.

- Intel MCS-86 Hexadecimal Object (MCS extension)
- Motorola EXORMAX (EXO extension)
- Tektronix Hexadecimal (TEK extension)

Additional information about the MakeBits and MakePROM programs is available in this manual.

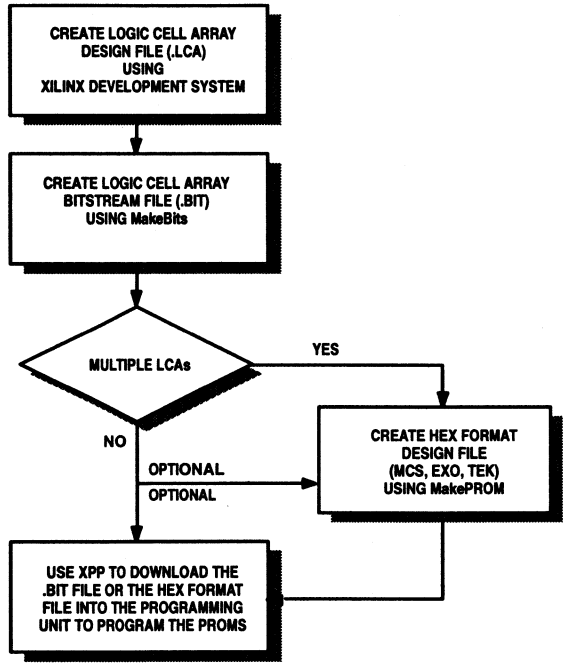


Figure 4-1 Overview of the Programming Process

Programmer Setup

NOTE *Before you set up the programmer, make sure that you have the XPP software on your system. If it is not already installed, do so by following the installation instructions included with the XPP program.*

Figure 4-2 and Figure 4-3 depict the front and rear panels of the HW112 programmer. Follow the instructions below to set up the HW112.

1. Turn off the rocker switch on the rear panel of the programmer.
2. Connect an RS232 cable (not supplied) between the PC/workstation serial port and the programmer serial port.

   Figure 4-4 illustrates the proper cable connections.
3. Connect the AC adapter to the power connector input and AC line source.

   or
4. Connect the HW112 power connector to a 9 V, 1 A regulated supply with a 5mm O.D. x 2.1mm I.D. female power plug (not supplied).

IMPORTANT *Be sure to check the power supply. The HW112 requires a 9 Vdc, 1 A supply. The unit could be damaged if it is connected to any other power source.*

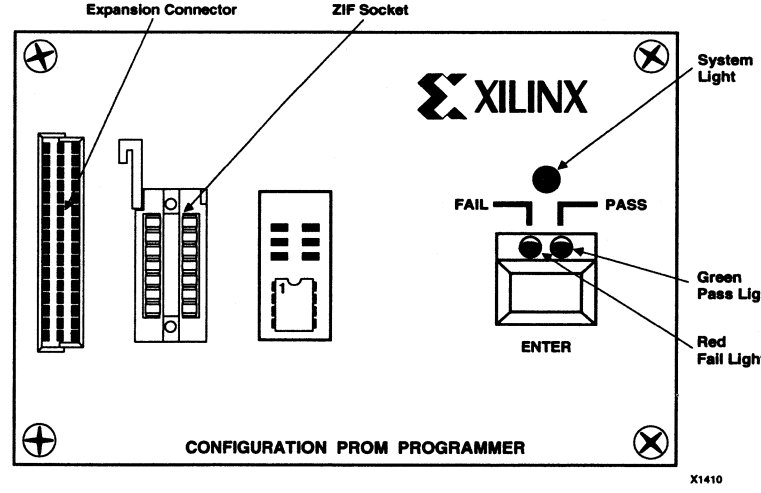


Figure 4-2 Top Panel of Programmer

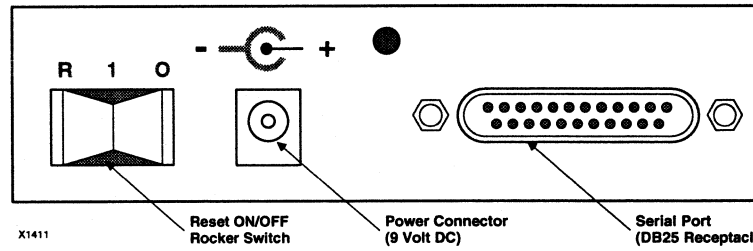


Figure 4-3 Rear Panel of Programmer

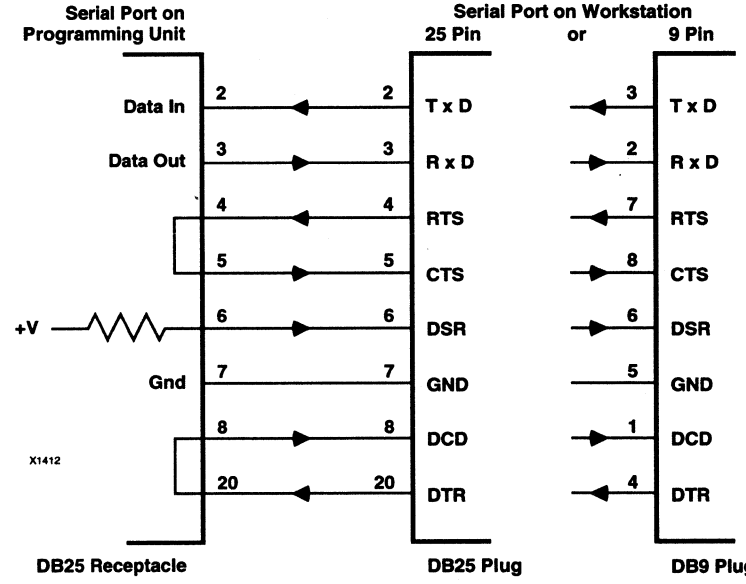


Figure 4-4 Serial Cable Connection

Turn the programmer power switch to On before running the XPP software. After the power is turned on, the self-test firmware takes a few seconds to test the hardware. The red System light flashes during the power-on self-test, and then remains lit. A flashing red Fail light on the Enter button indicates a hardware problem. Refer to Table 4-1 to determine the nature of the problem (contact your local sales representative for information about replacing a programmer).

Table 4-1 Serial PROM Programmer Troubleshooting

Fail Indicator Flashing	Problem
1 Time	Microcontroller RAM Failure
2 Times	Bad PROM Checksum
3 Times	SRAM
4 Times	D/A Converter

XPP Setup

Environment Variables

Before you run XPP, you must set its environment variables. For PCs and compatibles, you need to set three variables: MACHINE, PATH, and XACT. For workstations you only need to set the PATH and XACT environment variables.

MACHINE (PCs and Compatibles Only)

The MACHINE environment variable indicates the type of machine you are using. Specify NEC for NEC 9801 computers, IBMPC for PCs and other compatibles. For example, enter "Set MACHINE=IBMPC" in the autoexec.bat file. The default is IBMPC.

PATH

The operating system uses the PATH environment variable as a search path for the XPP program. Make sure that the XACT directory in which XPP resides is in the search path.

XACT

The XACT environment variable points to the directory where the program-dependent files are found. For example, on the PC, enter "Set XACT=*drive*:\XACT" in the autoexec.bat file.

All XPP files, except log and text files, are searched in a prescribed path, starting with the current directory, and then through the XACT environment and its subdirectories: files, data, and designs.

XPP Configuration

To start XPP, enter "xpp" at the operating system prompt, or select XPP from the Verify menu within the Xilinx Design Manager (XDM). When running XPP for the first time, you must configure the software to specify the serial port, baud rate, sound setting, device count, and

device name. Follow the instructions on the screen to configure the XPP settings described below.

Port Name

Workstation Users — The options for the serial port are /dev/ttyd0, /dev/sio2, and /dev/ttya. Consult your system administrator for the serial device name assigned to your system. Table 4-2 lists the default port settings for supported platforms.

PC Users — The options for the serial port are COM1 and COM2. The default setting is COM1.

Table 4-2 Valid System Port Names

Machine	Serial Port
DEC 3100	/dev/ttyd0
Sun	/dev/ttya
Apollo	/dev/sio1
IBM PC	com1

Baud Rate

XPP operates at standard baud rates from 300 to 19200. The default baud rate setting is 9600.

Sound

Errors during programming cause a beep or other sound alert from the host when the sound setting is on (default). To turn the sound from off to on, or on to off, enter "y" at the prompt when changing the settings interactively.

Device Count

The device-count setting determines the default number of devices to be programmed or compared. You can override this setting during programming or comparing. The default device-count setting is one.

Device Name

XPP requires that the exact device name be specified in order to program a PROM or perform other XPP operations. One or more device names may be specified; the full name of the PROM(s) must be used. For example, if only an XC1736D device is to be programmed in a session, you would enter "XC1736D" when prompted for a new device list.

If more than one device type is to be programmed (for example, XC1765, XC17128, and XC1736D), you would enter the following string when prompted for a new device list.

```
XC1765 XC17128 XC1736D
```

The default setting is one device type, XC1736D.

NOTE *Error messages that appear during configuration are explained in the XPP Error Messages section at the end of this chapter.*

XPP saves the configuration settings in a profile file named xpp.pro located in the current directory. These profile settings are used each time you run the XPP program. After the first run of XPP, the initialization process described above is bypassed as long as the xpp.pro file is present in your search path.

If you want to change the settings, run XPP with the –s option. You are given the choice of saving the new settings in the xpp.pro file or using the new settings temporarily.

Using XPP (PC Users)

You can use the PC version of XPP in two ways: Interactive mode and Batch mode. Interactive mode is for engineering development, while Batch mode is used to support manufacturing needs.

When you execute XPP, it reads the previously created configuration profile and tries to establish communication with the HW112 programmer. If XPP cannot establish communication with the HW112 programmer, it aborts with an error message.

Once communication is established, XPP displays the configured device type, the bootstrap firmware version number, and then waits for you to press the Enter key before displaying the Main Programming Menu.

Command Syntax

xpp -a -h -s -b *filename*.**bat -d** *device* **-polarity** *hi, lo data file*

–a Use ANSI Video Interface

This option signals XPP to use the ANSI video interface, and is primarily used on NEC 9801 computers, because they depend on an ansi.sys driver to handle all screen I/O. Most MS-DOS computers do not require this option.

–h Display XPP Help Information

The –h option displays information about XPP, including program execution, options, files used and created by XPP, and environment variables.

–s Enter Setup Mode

Using the –s option, you can change the configuration information in the xpp.pro file, which XPP reads upon execution.

–b Execute Batch File

This option executes the XPP-created batch file named *filename*.bat. XPP batch files are created using option 9 (Create a Batch File) in the Main Programming Menu.

–d Set the Device Type

Using the –d option, you can specify the supported devices when invoking XPP. The device type specified by this option overrides the device type stored in the xpp.pro.

–polarity low/high Set RESET Line Polarity (XC1718D, XC1718L, XC1736D, XC1765, XC1765D, XC1765L, XC17128 and XC17256D)

The –polarity option applies only to those serial configuration PROMs with programmable RESET polarity. It specifies the PROM's RESET/$\overline{OE}$ line as an active Low or active High reset. The RESET/$\overline{OE}$ are the same pin, but different signals. The –p option only allows for changing the RESET signal, not the $\overline{OE}$ signal. The default is active High reset.

datafile LCA Design Data File

You also need to specify the design data file containing the LCA design that is to be processed. The design data file can use formats from Intel (MCS), Tektronix (TEK), Motorola (EXO), or Xilinx (BIT or RBT). If you do not specify the data file, XPP searches in this order: BIT, MCS, TEK, EXO.

Function Keys

The three function keys (F1, F9, and F10) supported in the PC version of XPP are described below. They are only supported from the Main Programming Menu.

F1

When you are in the Main Programming Menu, F1 displays help for any highlighted menu selection, or any submenu appearing on the screen. Press the Escape key to exit the help function and return to your previous location.

F9

By pressing F9 while the Main Programming Menu is displayed, you can change the screen color through the following submenu:

```
SCREEN COLOR CONFIGURATION:

1) Monochrome
2) Color Palette 1
3) Color Palette 2
4) Color Palette 3

Select palette code (1-4) :
```

Type a number from 1 to 4 to select the desired palette; then press the Enter key.

F10

Pressing the F10 key opens a DOS shell, so you can enter or execute DOS commands without leaving XPP. While in the DOS shell, enter "exit" at the prompt to return to XPP. You can only use the F10 key while in the Main Programming Menu.

Interactive Mode

Using Interactive Mode, you can operate XPP from the Main Programming Menu, shown in Figure 4-5. Use the Up and Down arrow keys to scroll through the menu items; the current menu item appears highlighted. Press the Enter key to select the highlighted menu item. Press the Escape key to cancel any operations. To exit the program from the main menu, enter an upper or lowercase "x."

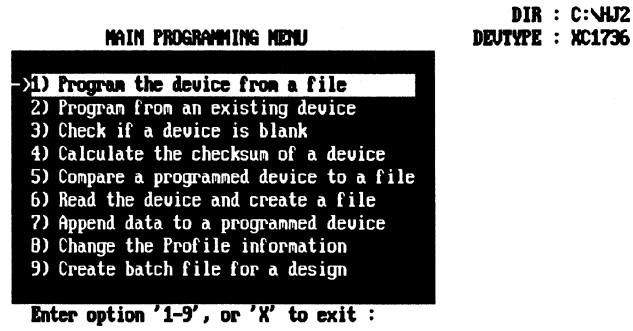


Figure 4-5 Main Programming Menu

Option 1 Program a Device from a File

Use this option to program serial PROMs with the contents of an existing BIT file or hex file. Perform the following steps after selecting this option from the Main Programming menu.

1. Enter the full name of the BIT file or the hex file to be used. Include the filename extension (BIT, MCS, and so forth).

NOTE *You can type "?" and press the ↵ key to see the list of data files in the current directory.*

2. Press the ↵ key to use the default repetition count, or enter the desired repetition number.
3. Insert a serial PROM into the Zero Insertion Force (ZIF) socket on the programmer.

NOTE *Make sure the device-type you are using matches the device-type list specified in the xpp.pro file. Use option 8 in the Main Programming menu to change the device type and remember that XC17XX devices are one time programmable. Multiple device-type programming is only supported by option 1.*

4. Press either the ↵ key on the keyboard or the Enter button on the programmer to start the programming process.

Notes

- The following submenu, which allows you to set up Low or High enable on the RESET line, appears whenever you try to program a device with programmable RESET polarity.

```
RESET POLARITY MODE:

1) high enable <- default
2) low enable

Select polarity mode (1-2):
Press "Enter" to use default mode
```

  Although large LCA configuration files can be programmed into multiple device types, devices that have the reset polarity mode changed to an active Low logic level cannot be cascaded with XC1718, XC1736, or XC1736A devices. If "mixing" XC1718, XC1736, or XC1736A PROMs with other Xilinx serial PROMs, the reset polarity mode of the larger devices must be unchanged.

- If programming is successful, XPP indicates so by issuing a message on-screen, and the green Pass light on the programmer lights. If programming more than one PROM, XPP prompts for the next device, which must be of the same type.

  If several copies of a single PROM design are to be made, XPP prompts you for the next device, which must be of the same type.

  If several copies of a multiple PROM design are to be made, XPP programs all PROMs of the same type at one time. For example, if an LCA configuration pattern was to be programmed into an XC1765 and XC17128 and ten copies were to be made, XPP would program ten XC1765 devices before prompting for an XC17128.

- If programming is unsuccessful, XPP displays an error message on-screen and the Fail light on the programmer flashes red. To correct this problem, try programming a different PROM by repeating steps 3 and 4.

- XPP permits you to ignore the condition of the device and continue programming.

- XPP checks if the device is blank before actually programming it; if it is not, a prompt appears. You can either continue with the same PROM, or try a different one.

- The checksum of the configuration data is printed on-screen immediately after XPP reads the source file.

- If a data file is large, such as an XC4013 configuration bitstream, it can require several PROMs to hold the complete design. These PROMs may or may not be of the same type. When one device fills, XPP prompts for the next device. XPP refers to the content of each PROM as a pattern, such as "pattern 1" and "pattern 2" for data that requires two PROM devices. The programming unit can only program one pattern at a time; therefore, all devices used to store the first pattern must be programmed before programming devices that will store the next pattern.

Option 2 Program from an Existing Device

This option reads the data from a programmed serial PROM and programs the same data into a blank serial PROM, making a duplicate. Perform the following steps after selecting this option from the main menu.

1. Insert the programmed serial PROM into the ZIF socket and press the Enter key.
2. Press the ↵ key again to accept the default repetition count (specified in the xpp.pro file), or enter a different repetition number.
3. Remove the original serial PROM from the ZIF socket.
4. Insert the a blank serial PROM into the ZIF socket. Make sure the device is of the type defined in the xpp.pro file. Use option 8 from the Main Programming Menu if you need to change the device type.
5. Press the ↵ key on the keyboard or the Enter button on the programmer when you are ready to program the device.

Notes

- If programming is successful, XPP displays a message, and the green Pass light on the programmer lights. If programming more than one PROM, XPP prompts for the next device. At that time, remove the PROM from the socket and then repeat steps 4 and 5 above.
- If programming is unsuccessful because the PROM has already been programmed or because it is faulty, XPP displays an error message and the red Fail light on the programmer flashes. To correct this problem, try programming a different PROM by removing the bad one from the socket and repeating steps 4 and 5 above.
- XPP permits you to ignore the condition of the device and continue programming.
- XPP verifies that the device is blank before actually programming it.

- The checksum of the configuration data is displayed immediately after XPP reads the source file.

Option 3 Check if a Device is Blank

Use this option to determine that the PROM device is blank. Perform the following steps after selecting this option from the main menu.

1. Press the ↵ key to accept the default repetition count, or enter a different repetition number.
2. Insert the serial PROM to be tested into the ZIF socket.
3. Press either the ↵ key on the keyboard or the Enter button on the programmer to start the programming process.
4. XPP displays the result of the check on-screen.
5. If the repetition count is set to more than 1, remove the tested device and repeat the four preceding steps for each additional device to be tested.

Option 4 Calculate the Checksum of a Device

Use this option to calculate the checksum of a device, not the checksum of a file. Perform the following steps after selecting this option from the main menu.

1. Press the ↵ key to use the default repetition count, or enter a different repetition number.
2. Insert the serial PROM to be checked into the ZIF socket.
3. Press the ↵ key to start calculating the checksum.
4. XPP displays the checksum result on-screen.
5. If default repetition count is set to more than 1, remove the tested device and repeat the four preceding steps for each additional device to be tested.

Notes

- The checksum calculation technique is consistent throughout XPP, but might not be compatible with the checksum technique used by the XPROM program (used with an earlier PROM programmer supplied by Xilinx) or software from other vendors.

Option 5 Compare a Programmed Device to a File

Use this option to determine the integrity of the data within a device. It compares the data from a BIT file or a hex file to the corresponding

programmed data from a PROM. If the data file is large enough to require more than one PROM, XPP compares one device at a time to the source contents. XPP refers to the contents as a "pattern." Perform the following steps after selecting this option from the main menu.

1. Enter the name of the BIT file or the hex file to be used. Include the filename extension (.BIT, .MCS, and so forth). If this data file is large, XPP displays the number of patterns produced.

NOTE *You can type "?" followed by Enter to see the list of data files in the current directory.*

2. Press the ↵ key to accept the default repetition count or enter a different repetition number.
3. XPP displays the pattern number to be compared. Select the appropriate device then insert it into the ZIF socket.
4. At this time, XPP prompts you to press the Enter key before it starts comparing the device to the data file. Press either the ↵ key on the keyboard or the Enter button on the programmer to start the comparison.

Notes

- XPP indicates that the data from the source file and the data from the PROM matches.
- If the data in both sources do not match, XPP displays the number of bits that are different.
- If applicable, XPP also prompts for the next device. Remove the present device and repeat steps 3 and 4.
- The programmer compares the file data to the PROM data until the data is exhausted or the number of mismatches reaches 255 at which time the process halts and you are returned to the Main Programming Menu.
- To save the entire data file, including the different bits, use option 6.

Option 6 Read the Device and Create a File

This option reads a serial PROM and saves its contents as a text file. Perform the following steps after selecting this option from the main menu.

1. Insert the programmed serial PROM into the ZIF socket on the programmer.
2. Select one of the following output formats from the submenu that appears.

```
1   ASCII one's and zero's.
2   ASCII HEX characters.
3   Compare against a file.
```

   If you select the third option, XPP prompts for the name of the BIT or hex file to be compared to the device. Enter the comparison file name, including the extension (for example, prom1.mcs, lca2.bit) and press the ↵ key.

3. At the next prompt, enter the output file name and press the ↵ key. You do not need to enter the extension because it defaults to TXT.
4. Insert the Serial Configuration PROM into the ZIF socket.
5. Press the ↵ key on the keyboard or press the Enter button on the programmer to start the read or output process.

Notes

- If you selected the third option in step 2, XPP writes the differences between the two files to the output file. The following notations are used to describe the differences.

  "/" The device bit is a zero, but the source file was a one.
  "." The device bit is a one, but the source file was a zero.

Option 7 Append Data to a Programmed Device

This option allows you to concatenate additional configuration data to a previously programmed PROM. This adds a second design file to a serial PROM that already contains a design. This is usually done with a PROM that programs a daisy-chain of LCAs. Perform the following steps after selecting this option from the main menu.

1. Enter the name of the design file that already exists in the PROM.
2. Enter the name of the design file including the filename extension (BIT, MCS, and so forth) to be added to this PROM. If the add-on design file is too large for the existing space, an additional device is required to accommodate the rest of the design file. XPP indicates the number of patterns generated before programming starts.
3. Press the ↵ key on the keyboard to accept the default repetition count or enter a different repetition number.

4. Insert the previously programmed PROM into the ZIF socket. Make sure that the device type specified in the xpp.pro file matches the inserted device. Either press the ↵ key on the keyboard or the Enter button on the programmer to start the concatenating process. If the device types do not match, press the Escape key to abort and use option 8 from the menu to change the profile.

Notes

- If programming is successful, XPP displays the status and the green Pass light on the programmer lights.
- If programming is unsuccessful because the PROM has already been programmed and you did not enter the name of the file it already contains, or because the PROM is faulty, XPP displays an error message and the red Fail light on the programmer flashes. To correct this problem, make sure you correctly specified the name of the file already in the PROM, or try programming a different PROM by removing it from the socket and repeating steps 3 and 4.
- When appropriate, XPP prompts for the next device. Remove the PROM from the socket and repeat step 4.
- To concatenate the additional data at the right location, you must specify the existing data file in the programmed PROMs. With this information, XPP calculates the correct starting point for additional data. Only the last partially programmed device is prompted for insertion if the originally programmed files used multiple PROMs.

Option 8 Change the Profile Information

Use this option to change the following settings in the present profile: serial port, baud rate, sound on/off, repetition count, and device list. After making changes, you can update the profile by entering "y" at the prompt; enter "n" or press the Escape key to cancel.

XPP uses the most recent data even if you do not update the profile. If you change the baud rate, XPP automatically resets the programming unit and then sets the new baud rate. Press the ↵ key once more to return to the main menu.

Option 9 Creating a Batch File for a Design

Use this option to create a batch file, which simplifies the PROM programming process for manufacturing. After creating this batch file, you can execute it from the DOS prompt. Perform the following steps after selecting this option from the main menu.

1. Enter the name of the design file to program into the PROM (for example, design1.mcs).

2. Enter the name you want to give the batch file (for example, design1.bat).

Notes

- The program creates the batch file with the necessary executing instructions.
- The settings in the current directory xpp.pro file determine the repetition count, baud rate, communication port, and device list.

  Make sure the xpp.pro file contains the correct settings before running XPP in batch mode. If more than one device type is to be programmed during the batch mode session, be sure that the device list contains the proper device names before starting.
- Batch files that XPP creates are meant to be executed as created. Editing of these files is not recommended.

Batch File Mode

Using Batch mode, you can configure XPP so that the Main Programming menu is bypassed and PROMs are programmed with a preset design file. You must have the following files in the current directory to operate XPP in the batch mode.

- The BAT file that was created using option 9
- The xpp.pro file with the desired settings
- The design file (DSF) specified in the BAT file

  At the beginning of execution, XPP loads the input file into the programmer and then prompts for a device to be programmed. Perform the following steps to continue this process.

  1. Press the ↵ key to use the default repetition count, or enter the desired repetition count.
  2. Insert a blank serial PROM into the ZIF socket. Make sure the device is the same type as the type defined in the xpp.pro file. Use option 8 if you need to change the device type.
  3. Press either the ↵ key on the keyboard or the Enter button on the programmer to start programming the device.

Notes

- If programming is successful, XPP displays the status and the green Pass light on the programmer lights.

- If programming is unsuccessful because the PROM is faulty or has already been programmed, XPP displays an error message and the red Fail light on the programmer flashes.
- XPP permits you to ignore the condition of the device and continue programming.
- When the repetition count is reached, XPP terminates and automatically exits to DOS.
- Pressing the Escape key when there is a pause during execution, such as, when XPP issues a prompt, also terminates the program and exits to DOS.

Using XPP (Workstation Users)

The workstation version of XPP provides two user interfaces: command-line entries and Interactive Mode. Typically, you use the command-line entries for production, as only a limited number of functions are available from the command line. You use the Interactive Mode for prototyping.

When you execute XPP, it reads the previously created configuration profile (xpp.pro) and tries to establish communication with the programming unit. If XPP cannot establish communication with the programming unit, it aborts with an error message. The error needs to be resolved before you can continue.

Also, because XPP checks xpp.pro for configuration data, you might need to change some of the settings by running XPP with the –setup option to change the port specification, baud rate, device number, and so forth.

When you start XPP and there are no hardware or configuration problems and communication is established, the PROM Programmer displays the configured device type and the bootstrap firmware version number.

Command-line Syntax

The syntax for using XPP from the command line is as follows:

xpp *[command-line parameters] command <parameters>]*

Command *Parameter*

numdev is a widely-used command *parameter* that allows you to select the number of devices, from 1 to 1000, to be programmed with the same design file. If you do not specify a value, XPP reads the value from the xpp.pro file. If there is no such value in the file, XPP sets the value to 1.

Command-line Parameters

There are three parameters you can specify on the command line, as described below:

–help

This parameter displays information about XPP, including program execution, options, environment variables; and files used and created by XPP.

–dev *name*

Use this parameter to specify a PROM device to use. Valid PROMs are XC1718D, XC1718L, XC1736A, XC1736D, 1736AMD, XC1765, XC1765D, XC1765L, 1765AMD, XC17128, and XC17256D.

–setup

This parameter puts you in the interactive mode for the Setup command.

Commands

The following commands are available from the command line.

program

This command allows you to program the device from a file.

```
program [-high|-low] design [numdev]
```

where:

–high programs RESET polarity of the device to high. This is the default.

–low programs RESET polarity of the device to low.

design is the name of the design file to program into the device. The design file can be one of the following formats:

- .bit Xilinx bit file format generated by the MakeBits program
- .mcs Intel PROM file format
- .tek Tektronix PROM file format
- .exo Motorola PROM file format
- .rbt Xilinx ASCII bit file format

copy

This command allows you to program the device from an existing programmed device. The RESET polarity is the same as the existing device that XPP read.

```
copy [numdev]
```

check

This command allows you to verify the status of a device. If the device is not blank, the command displays the checksum of the device.

```
check [numdev]
```

checksum

This command reads a programmed device, calculates and displays its checksum.

```
checksum [numdev]
```

compare

This command reads the specified design file and compares the data to a programmed device. You can specify a file to output the compared data or display the number of different bytes on the screen.

```
compare [-out name] design [numdev]
```

where:

–out ***name*** is the specified output file that has the recorded differences between the design file and the programmed device. The output file contains the ASCII bits that were read back from the device. The difference between the device and the design file are marked by two special characters:

- '/' device data is 1 and design data is 0
- '.' device data is 0 and design data is 1

If you do not specify the –out parameter, XPP displays the number of bytes that are different between the design file and the device.

design is the name of the design file to program into the device. The design file can be one of the following formats:

- .bit Xilinx bit file format generated by the MakeBits program
- .mcs Intel PROM file format
- .tek Tektronix PROM file format
- .exo Motorola PROM file format

- .rbt Xilinx ASCII bit file format

read

This command reads the device data into a specified file. The data can be formatted in different number bases in the file.

read [-bin|-hex|-dec] [-out *name*] [numdev]

where:

–bin formats the data in binary characters (base 2). This is the default.

–hex formats the data in hexadecimal characters (base 16).

–dec formats the data in decimal characters (base 10).

–out *name* is the specified output file to record the data read back from a device. The default is to write the data in binary characters. If you do not use the –out option, XPP displays the data on the screen.

append

This command appends new data to the specified device. Append adds the data from another design file to the end of the current programming in a device.

append *design1 design2* [numdev]

where:

design1 is the name of the design file already programmed into the device. XPP reads the original design file to calculate where the current programming ends in the device. The design file must be in the current directory, or the file name must contain the full path to the file.

design2 is the name of the design file to be appended to the device. The design file must be in the current directory, or the file name must contain the full path to the file.

setup

The command allows you set and modify the programmer options. If you use this command with no parameters, XPP prompts for all required paramenters.

```
setup [-dev devices
       -port name
       -count numdev
       -baud rate
       -sound mode]
```

where:

–dev *devices* specifies the devices that are used. Enter a list of device names with at least one space separating the names.

–port *name* specifies the name of the serial port that connects to the HW112 PROM Programmer. Use one of the following port names:

- Apollo — /dev/sio (Default), /dev/sio1, /dev/sio2
- RS6000 — /dev/tty0 (Default), /dev/tty1
- TCP/DEC Alpha — /dev/tty00 (Default), /dev/tty01
- Sun — /dev/ttya (Default), /dev/ttyb

–count *numdev* specifies the number of devices to be used as the default number [numdev] for other commands. Enter a positive integer value from 1 to 1000. If you do not specify a value, XPP reads the value from the xpp.pro file; and if there is no value in the xpp.pro file, the value is set to 1.

–baud *rate* specifies the serial communication baud rate. The HW112 PROM Programmer supports two baud rates, 9600 and 19200. If you do not specify a baud rate, XPP reads the value from the xpp.pro file. If no such value exists in the xpp.pro file, the default is 9600 baud.

–sound *mode* specifies whether a beep sounds when XPP encounters an error during programming. Enter one of two modes:

- on — enables beep sound
- off — disables beep sound

Examples

This section contains three examples that use the command-line syntax.

Example 1

To program five XC1736D devices with low reset polarity, using MCS file *xxx*.mcs, enter this string:

```
xpp -dev xc1736d program -low xxx 5
```

Example 2

To copy 15 XC1736D devices, enter this string:

```
xpp -dev xc1736d copy 15
```

Example 3

To specify a new port (on a Sun workstation) and baud rate and change to the Interactive mode, enter this string:

```
xpp -port /dev/ttya -baud 19200
```

Interactive Mode

The Interactive Mode of XPP prompts you for commands and programmer settings. You start the Interactive Mode at the system prompt by typing XPP without any options. The Interactive Mode allows you to view help information for the available commands and settings. The commands available in this mode are the same as for the command line. Refer to the previous command descriptions for details.

Using the Interactive Mode, you can operate XPP from the Main Programming Menu, shown in Figure 4-6.

In the Interactive Mode, you can pre-set the port and device settings at the command line. You can also direct XPP to start immediately with the Setup command at the beginning of the interactive session.

The Menu

In the Interactive Mode, the menu is always displayed so you can select the appropriate command.

At the XPP? prompt enter the number of the option you want to use, from 1 to 9 or 'X'.

```
------------------------------------------------------------
                MAIN PROGRAMMING MENU

   1) program  - program the device from a file
   2) copy     - program from an existing device
   3) check    - check status of device
   4) checksum - calculate the checksum of a device
   5) compare  - compare the device to a design file
   6) read     - read the device data into a file
   7) append   - append new data to the device
   8) setup    - set the programmer options
   X) exit     - quit XPP
------------------------------------------------------------
Enter option '1-9' or 'X' to exit or type a command
For information on a specific  command, type 'help <command>'.

XPP ?
```

Figure 4-6 Main Programming Menu

Syntax

The syntax for starting XPP in the Interactive Mode is as follows:

```
XPP
```

The syntax for using the online help is as follows:

```
help topic
```

For example to get additional information about the program command, type:

```
help program
```

Interactive Commands

This section describes the Interactive commands available. You enter these commands at the XPP ? prompt.

baud 9600| 19200

This command allows you to change the baud rate.

count

This command sets the new number of device repetition to use as the default value when programming the devices.

design *design_name*

This command allows you to specify the default design name to use when programming.

device *device_types*

This command allows you to specify a list of devices to program. You can specify mixed device types in the same string.

help *command*

This comand displays help messages about the specified command. When you substitute the command variable with the keyword index, you get a list of commands for which help is available. For example:

```
help index
```

This command string generates a list of all XPP commands, as follows:

```
APPEND      BATCH     BAUD      CHECK     CHECKSUM
COMMANDS    COMPARE   COPY      HELP      MAIN=MENU
PATH        PORT      PROGRAM   QUIT      READ
RESET       SAVE      SETUP     XPP=TUTOR
```

path dirs

This command sets up the searched path. XPP uses the searched path to search for a design file. XPP scans for files, in order, in all the specified directories.

port *portname*

This command allows you to specify the communication port to which the PROM programmer is connected.

setup [–dev name] [–port name] [–count #] [baud 9600|19200] [–sound on|off]

This command allows you to define important communication parameters.

sound [–on|–off]

This command allows you to set the beep sound either on or off when XPP detects a programming error.

reset

This commands allows you to reset the HW112 programmer..

append *#devices designname*

This command allows you to append a new design into the devices.

check *#devices*

This command allows you determine whether the devices are blank. If the device is not blank, XPP calculates its checksum.

compare [–n # –out name]

This command allows you to compare a device to a design.

copy #devices

This command allows you to copy devices from a master device.

read [–bh –bin –dec –hex –rbt –n # –out name]

This command reads the contents of a device and allows you to save that information to a file in the specified format.

program [–high –low] *design_name #devices*

This command allows you to program the devices from a design.

Searching a Design File

When you specify a design file without a path or type, XPP scans the search path for a matching file. To specify a search path, set up the XACT environment as follows:

```
setenv XACT dir1:dir2:dir3
```

XPP scans for file types in the following order:

- *name*.bit
- *name*.mcs
- *name*.tek
- *name*.exo

Next, XPP scans for the complete file name, in the following order:

- current directory
- subdirectory 'data'
- subdirectory 'designs'
- subdirectory 'msg'
- next directory in the search path

Search path is disabled as the default. XPP only scans the current directory and its subdirectories. To turn on the searching scheme, enter the following command at the XPP prompt:

```
set use_path on
```

XPP Profile

The xpp.pro file stores all parameters set up in the current XPP session. These stored parameters are then loaded for the next session. XPP searches xpp.pro file using the same search path as a design file. To set up the HW112 PROM Programmer the first time, follow these steps:

1. Connect the HW112 Programmer to a serial port on your workstation and turn the power on.

NOTE *You can connect the HW112 to your workstation while your system is on.*

When powered on, the DS-112 runs a self-diagnostic. The FAIL LED blinks if there is an error.

2. At the system prompt, type the following command:

```
xpp setup
```

3. Answer the questions to the setup command to establish a profile for the port name, baud rate, beep sound, and the default number of devices to be used.

   XPP saves this information in the xpp.pro file for subsequent sessions.

Error Messages and Recovery Techniques

Bad Device in socket.

The device in the socket is bad or the wrong type. Try another device or check the device type.

Batch file *filename* **is not located.**

The specified batch file could not be found.

Bit file *filename* **empty.**

There is no data in the bit file.

Bit file *filename* **is too small.**

The bit file is corrupted.

Can't open bit file *filename.*

The bit file could not be found. Check the search path.

Can't open file *filename.*

The file could not be found. Check the search path.

Cannot create output file *filename.*

The output file was not created. Check file name and file permissions.

Cannot open log file *filename.*

The log was not created. Check file name and file permissions.

CEO didn't go low - Wrong device.

Current device is the wrong type or is bad.

Checksum Error.

Data transmission problem. Check communication cable.

Checksum error *Expected x Actual y.*

Data transmission problem. Check communication cable.

Communication failure on port *name.*

Communication problem. Make sure the communication port name is valid and the hardware is properly connected. If using a network, make sure that you are the only one using the programmer.

Communication line is broken.

Communication problem. Make sure the hardware is properly connected. If using a network, make sure that you are the only one using the programmer.

Datafile *filename* **is corrupted.**

Data file is corrupted. Regenerate the data file.

Device *name*.**dsf is not located.**

The device-specific file (DSF) was not found. It must be located in the search path.

Device failed during Verification.

The device was programmed, but readback verification failed. The device may be bad.

Failed to communicate with port *name.*

Communication failure. Make sure the serial port name is valid and functional. If using a network, make sure that you are the only one using the programmer.

Failed to establish communication.

Communication failure. Make sure the serial port name is valid and functional. If using a network, make sure that you are the only one using the programmer.

Failed to re-establish communication with the programmer.

Communication failure. Make sure the serial port name is valid and functional. If using a network, make sure that you are the only one using the programmer.

Failed to set to baudrate *rate.*

Communication failure. Make sure baud rate is correct.

File *filename* **doesn't exist and it cannot be created.**

The file cannot be created. Check file name and file permission.

File *filename* **is empty.**

The specified file is empty.

File *filename* **is not located.**

File is not found. Check file name and search path.

File *filename* **is too large to process.**

The data file specified is too large to load in memory.

Hardware D/A Converter failed.

Programmer hardware problem. Turn programmer off and try again.

Help file *filename* **is not accessible.**

Help file is not accessible. Check file permission.

Help file 'xpp.hlp' is not located.

Help file was not found in the search path. xpp.hlp is installed in install_dir/files.

Host software timed out.

Turn off the programmer and try again. If you are connected to a network, make sure that you are the only one using the programmer.

Initialization failure. Profile information can be changed by running 'xpp -s'.

Internal communication failure. Make sure the serial port name is valid and functional and the baud rate is correct.

Invalid Programmer command.

Turn off the programmer and try again. If using a network, make sure that you are the only one using the programmer.

Invalid baud rate *rate.*

The specified baud rate is not supported by the system. Restart XPP with the –s option to change the baud rate.

Invalid bit file format.

The BIT file is corrupted. Regenerate the bit file from the LCA file.

Invalid data in file.

Hex file is corrupted. Regenerate the data file from the BIT file.

Invalid data record in file *filename*, **line** *number.*

Hex file is corrupted. Regenerate the data file from the BIT file.

Invalid function on command line *name.*

The command line only accepts these functions: program, append, check, compare, copy, and read.

Invalid header for data packet.

Communication problem. If using a network, make sure that you are the only one using the programmer.

Invalid host ACK for data packet.

Communication problem. If using a network, make sure that you are the only one using the programmer.

Not enough memory in the programmer.

The device-specific file (DSF) is too big. It may be corrupted. Reinstall the software and try again.

PROM CRC test failed.

Programmer hardware problem. Turn off the programmer and try again. If using a network, make sure that you are the only one using the programmer.

Received *number x hex number* **when expecting** *number* **from programmer.**

Communication problem. If using a network, make sure that you are the only one using the programmer.

Received NAK: *number* **when expecting** *number* **from programmer**

Communication problem. If using a network, make sure that you are the only one using the programmer.

Received only *number* **bytes of data. Expected** *number.*

Communication problem. If using a network, make sure that you are the only one using the programmer.

Time out on SOH receiving.

Communication problem. If using a network, make sure that you are the only one using the programmer.

Timeout on command *name.*

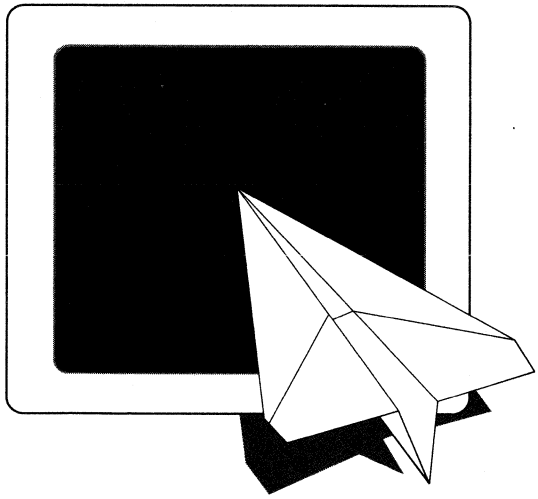
Communication problem. If using a network, make sure that you are the only one using the programmer.

Unexpected EOF in file *filename*, **line** *number.*

Unknown PROM file format in *filename*; hex file is corrupted.

Unexpected token *number* **in file** *filename*, **line** *number.*

Unknown PROM file format in *filename*; hex file is corrupted.



XACT Hardware and Peripherals Guide

XChecker

XACT Hardware and Peripherals Guide — April, 1994 (0401132 01) *Printed in U.S.A.*

XChecker Universal Download/Readback Cable and Logic Probe

This Program is Compatible with the Families Indicated.

☑ XC2000	☑ XC3100	☑ XC3100A	☑ XC4000H
☑ XC2000L	☑ XC3000A	☑ XC4000	☐ XC7200
☑ XC3000	☑ XC3000L	☑ XC4000A	☐ XC7300

The XChecker™ cable and software are designed to work with the Xilinx XC2000, XC3000, XC3100, and XC4000 families of FPGAs. They are used to download, read back, and verify configuration data, as well as to probe internal logic states of XC2000, XC3000, and XC4000 designs.

The XChecker cable and software support the following capabilities.

- XChecker allows you to download a design to the LCA on the target system.
- The 3 V Adapter allows user target systems containing XC2000L and XC3000L low voltage LCA devices to interface with XChecker.
- After configuring an FPGA, XChecker can verify its configuration by comparing it to the original design.
- You can probe an LCA's internal logic with XChecker to debug your designs. Probing is the execution of a readback of all the configuration data and extracting the internal logic states of desired signals from it.

XChecker software supports all previous versions of parallel and serial download cables, for download. Replacing previous Xilinx cable/download packages with the new XChecker cable/software package has no impact on the download connections to the target system LCA.

XChecker Hardware

XChecker hardware includes a DB-9/DB-25 adapter, the cable assembly with internal logic, a supplied test fixture, a set of headers to connect the cable to your target system, and a 3 V low voltage adapter. Using XChecker requires a minimum of 512 kB of base memory for PC

compatibles and a standard DB-9 or DB-25 RS-232 serial port. If you have a different serial port connection, you need to provide an appropriate adapter. See Figure 5-1 and Figure 5-2.

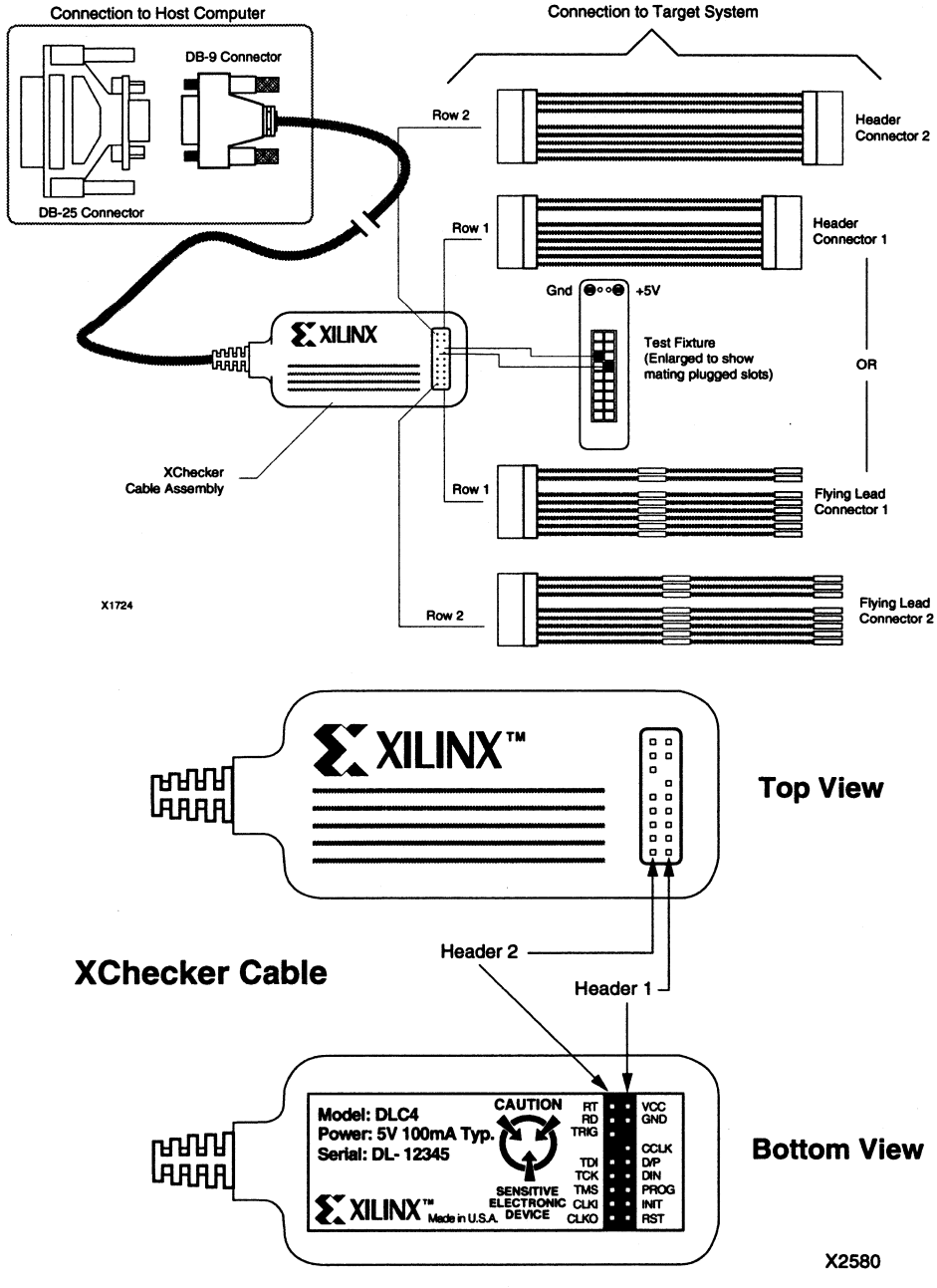


Figure 5-1 XChecker Hardware and Accessories

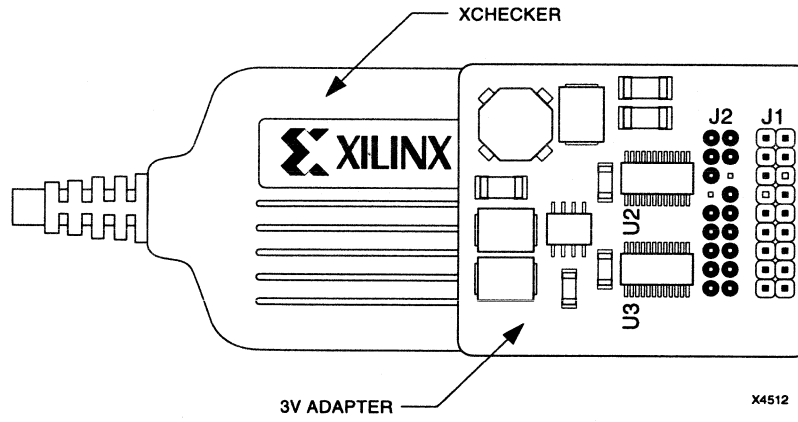


Figure 5-2 3 V Adapter

The cable assembly houses internal circuitry consisting of a Xilinx FPGA, a static RAM, and an oscillator circuit. The internal Xilinx FPGA functions as an interface between the XChecker software and the target FPGA. The static RAM stores the configuration data for download and readback. The oscillator circuit provides a system clock and allows download and readback of configuration data.

The 3 V Adapter accepts Vcc supply voltages from the target system from 2.9 V to 5.25 V. The 3 V Adapter contains a voltage 'step-up' circuit that generates the 5 V supply voltage needed by XChecker.

Since the 3 V adapter can accept input voltages up to 5.25 V, there is no need to remove the adapter when moving the XChecker cable between low voltage systems and higher 5 V systems. Except for the voltage conversion, the 3 V Adapter is completely transparent to the XChecker hardware or the target system.

For downloading configuration data, the XChecker cable can be used with a single FPGA, or several connected in a daisy chain. When used to read back data or as a logic probe, the XChecker cable can only be used with one device at a time.

The XChecker cable transmits configuration data to all target LCAs at 921 kHz. Readback of configuration data from XC2000 and XC3000 devices is also at 921 kHz. However, with XC4000 devices, readback can be performed at three different clock rates: 921 kHz, ~2.75 MHz, and ~5.5 MHz.

Communication between the host system and the XChecker cable is dependent on host system capability. Table 5-1 lists the valid baud rates.

Table 5-1 Valid Baud Rates

	Baud Rate			
Platform	**9600**	**19200**	**38400**	**115.2K**
IBM PC	X	X	X	X
NEC PC	X			
APOLLO	X	X	X	
DEC3100	X	X	X	
SUN	X	X	X	

Using Previous Download Cables with XChecker Software

Although you are encouraged to use the XChecker cable with the XChecker software to replace previous download cables and download programs, you can use XChecker software with previous download cables.

The software supports all previous download cables, parallel and serial. However, these previous cables can only be used to download a configuration bitstream, they cannot be used for readback.

If you do use XChecker software with a previous download cable, keep the following points in mind.

- Previous versions of the download cable were made to download XC3000 and XC2000 designs, not XC4000 designs. The basic limitation of the previous cables is that they do not have a PROG pin to initiate a re-program in XC4000 devices. They also do not have an INIT pin to check for Cyclical Redundancy Check (CRC) errors during configuration.

NOTE *To use a parallel download cable to download designs to the XC4000 family of devices, you must manually toggle low the PROG pin. PROG is active when it is low.*

- Previous download cables do not support readback or verification.
- For the PC, the download cable is a parallel cable, requiring connection to the parallel port. (The XChecker cable is serial.)

There are only two situations when you might prefer using previous download cables instead of the new XChecker cable.

You might have circuit boards with header connectors keyed to match the previous cable headers. However, you could use the XChecker cable with its flying lead connectors. Simply match the labeled flying leads to the equivalent signals on your system.

You may have circuit boards where power consumption is a critical factor. (The XChecker cable requires about 100 mA; the parallel cable used with PCs draws less power from the target LCA board.) In such cases, you may use either the XChecker software or the Download program to download the bitstream.

Preparations for Using the XChecker Cable and Software

A sample LCA design named XROMs is provided with the XChecker software, and is loaded in your \XACT\DESIGNS\CORE\XCHECKER directory when you install the XChecker software. You can use the XROMs design and the XC4000 Demonstration Board as examples. A schematic representation of XROMs is shown in Figure 5-3.

The files that you need are xroms.lca, xroms.ll, xroms.rpt, xroms.bit, and xroms.xnf. They are also installed in the \XACT\DESIGNS\CORE\XCHECKER directory when you install the XChecker program.

These files were compiled for an XC4003PC84 device. If you are using a different device, you need to compile the file xroms.xnf for the part you are using.

In this sample design, the 500 kHz clock is taken from of the OSC4 block and made available as the output signal TO_CLKI. The clock is then returned to the design via the input signal FROM_CLKO and fed to the 4-bit counter (CB4RE). The counter simply increments the addresses to the ROM cells which have been initialized to predetermined values from the schematic editor. (See Table 5-3).

NOTE *Special XC4000 symbols are used (Startup and Readback); note their connections. These special symbols are found in the schematic editor library. You need to include these symbols in your designs.*

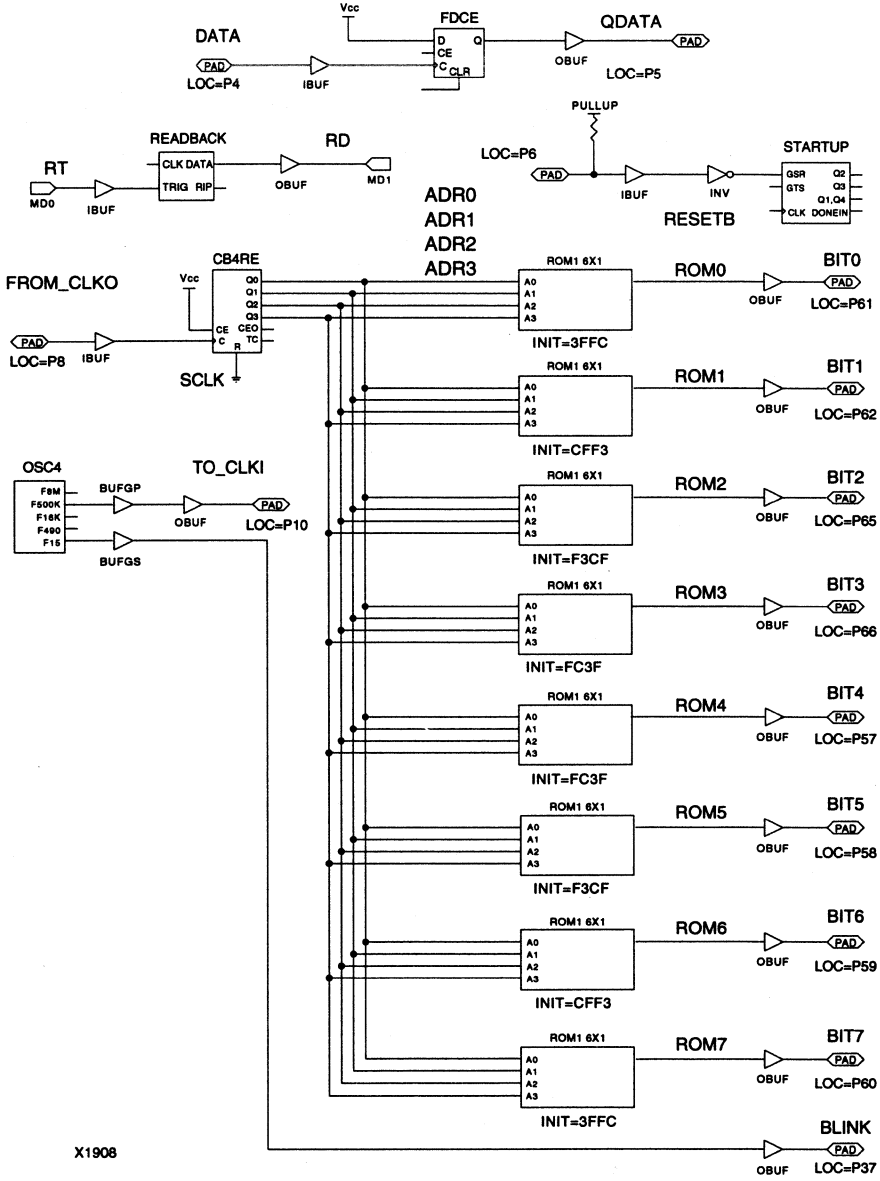


Figure 5-3 XROMs Sample LCA Design

Table 5-2 XROMs Sample Design Pin Locations for XC4003APC84 LCA

Pin Location	Pin Name
P61	: BIT0
P62	: BIT1
P65	: BIT2
P66	: BIT3
P57	: BIT4
P58	: BIT5
P59	: BIT6
P60	: BIT7
P37	: BLINK
P4	: DATA
P8	: FROM_CLKO
P5	: QDATA
P6	: RESETB
P10	: TO_CLKI

Table 5-3 Contents of the Sample Design ROM Cells

Address	ROM contents 76543210
0:	01111110
1:	01111110
2:	10111101
3:	10111101
4:	11011011
5:	11011011
6:	11100111
7:	11100111
8:	11100111
9:	11100111
10:	11011011
11:	11011011
12:	10111101
13:	10111101
14:	01111110
15:	01111110

If you do not want to use the sample design, follow these steps before using XChecker:

1. Create a design.
2. Generate a bitstream and set configuration options.

You must also generate a logic allocation file (*design*.ll) for probing. See Figure 5-4 for an illustration of the design flow of XChecker.

XChecker software can use a *design*.bit or a *design*.rbt file as input, so generating a PROM file (*design*.mcs, *design*.tek, or *design*.exo) is optional. Use MakePROM to generate a PROM file. For more information about PROM files, read the MakePROM chapter in the *XACT Reference Guide, Volume 2*.

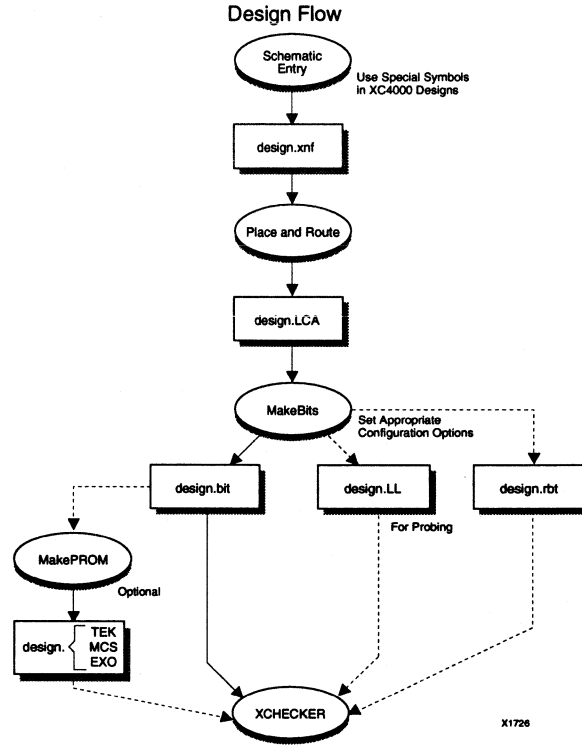


Figure 5-4 XChecker Design Flow

Creating a Downloadable Design

If you have already created a configuration bitstream, please see the section on Using XChecker.

Use any Xilinx-supported schematic editor to enter the design. For more information, refer to the appropriate interface user guide.

No special symbols or signals are required for XC2000 and XC3000 designs or for XC4000 designs intended for download only. XC4000 schematic symbols are required for the readback, verification, and logic probing of XC4000 designs. These symbols are provided with your schematic editor interface.

Include the Readback symbol in the schematic with the appropriate connections, as shown in Figure 5-5. This assigns external pins to execute the readback functions RTRIG and RDATA.

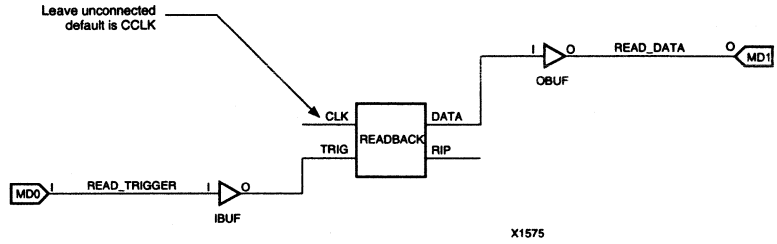


Figure 5-5 XC4000 Readback Symbol

NOTE *If you want RTRIG and RDATA to correspond to the mode pins M0 and M1, use the special schematic symbols MD0 and MD1, as shown in Figure 5-5.*

Also include the Startup symbol to select the location of the RESET pin and to set the polarity of the RESET signal to Low. The default polarity of the RESET pin for XC4000 parts is active High, and XChecker software expects the RESET signal to be active Low. See Figure 5-6.

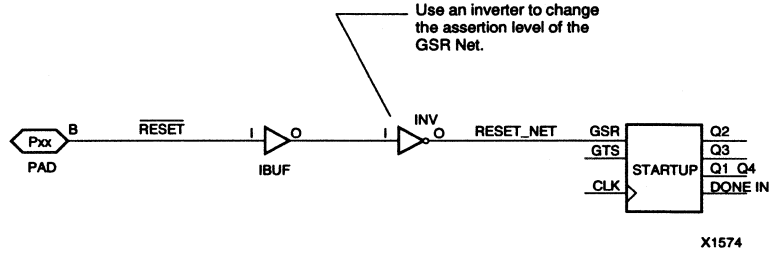


Figure 5-6 XC4000 Startup Symbol

NOTE *You do not need to use the special pads MD0 and MD1 to connect to the Readback symbol. You may use normal IPAD and OPAD primitives if you want the signals RTRIG and RDATA to be assigned to locations other than M0 and M1.*

Notice the inverter in the Startup symbol implements the Active-Low Reset asserted by XChecker, on the RST wire.

Generating a Bitstream

To download, use XMake to create a configuration bitstream.

XC4000 designs or XC3000 designs used for probing require the use of specific options for MakeBits. For this reason, specify the –m option; this option runs all appropriate programs and stops beforeMakeBits.

Use MakeBits to generate a configuration bitstream for the design (*design*.bit) with appropriate configuration options.

For XC4000 designs, you need to enable readback of the device and enable a pull–up resistor for the DONE pin. Do this by using MakeBits from the command line or from the Xilinx Design Editor (XDE).

From the command line, use MakeBits with the following options.

MakeBits -f readcapture:enable donepin:pullup *design*.**lca**

From XDE, select MakeBits from the Programs menu and select the following configuration options.

ReadCapture	**Enable**
DonePin	**PullUp**

To probe designs, you must create a logic allocation file (*design*.ll). The design.ll file provides bit locations of the values of RAM, I/O, latches, and flip–flops. To create a logic allocation file, select Config→Makebits→MakeLL from XDE. You can also specify the –l option from the command line. For more information about MakeBits, read the MakeBits chapter in the *XACT Reference Guide, Volume 2*.

Connecting the XChecker Cable

There are three simple steps for connecting the cable:

- Connect the cable to your host system RS-232 serial port
- Perform cable self-check with the text fixture and the Diagnostic command
- Connect the cable to your target system.

Connecting the Cable to Your Host System

The XChecker cable connects to your system RS-232 serial port. A DB-9/DB-25 adapter is provided, and it accommodates most serial ports.

Connecting the 3 V Adapter

When connecting the 3 V Adapter, it is necessary to use normal ESD precautions. This adapter is static sensitive and can be damaged by ESD energy.

Ensure that you are adequately grounded before connecting or using the 3 V Adapter. Refer to Figure 5-2 (Top View of the XChecker Cable) and place the adapter on top of the XChecker case aligning the 18-pin female socket (J2) with the 18-pin male connector on XChecker. The Xilinx logo on the XChecker case and the adapter silkscreen must be oriented the same way. Gripping the adapter board by the edges, gently press down until the adapter makes a solid connection to XChecker.

Performing Cable Self-Check

Use the test fixture and the Diagnostics command to perform a self-test. The test fixture is a small printed circuit card with a keyed header connector that fits onto the cable assembly. It is only used to validate proper operation of the cable by executing the Diagnostic command. See the Diagnostics command in the Valid Commands in the Interactive Mode section for more information.

The test fixture has connectors for +5 V and Ground. It is necessary to connect +5 V and Ground to these connectors when executing a diagnostic check since XChecker draws power from your target system, not from the host system.

Verifying 3 V Adapter Operation

The Diagnostic Board that is shipped with the XChecker cable can also test the 3 V Adapter. To verify correct operation, follow these steps.

1. Plug the adapter onto XChecker, they plug the Diagnostics Board onto the male connector (J1) on the adapter.
2. Attach a power supply to the Diagnostics Board Vcc and Gnd terminals and set the voltage to about 3.3 V.
3. Run the XChecker software and type the following command at the prompt:

```
>diag
```

   This command runs the diagnostics test. The test results should be identical to running the same diagnostics test without the 3 V Adapter connected to XChecker.

4. Attach the 18-signal flying wire or header block to the 18-pin male connector (J1) on top of the 3 V Adapter.

   XChecker operation is unchanged.

5. Attach XChecker and the 3 V Adapter to your target system.

Using the 3 V Adapter with XC2000L and XC3000L LCA Devices

To use the 3 V Adapter, follow these steps:

1. Attach the 18-signal flying wire or header block to the 18-pin male connector (J1) on top of the 3 V Adapter.

   XChecker operation is unchanged.

2. Attach XChecker and the 3 V Adapter to your target system.

The configuration of the XC2000L and XC3000L devices is the same as for the XC2000, XC3000, XC3000A, and XC3100 family devices.

The readback clock speed of the XC3000L devices has been slowed because of lower Vcc supply voltage. If the supply voltage of the target system is lower than 3 V, you might see error messages like the following when reading back or verifying a configured XC3000L device.

```
XCHECKER? verify
Design design_name has 128 probeable signals.
Readback 1847 bytes of configuration.
Verifying datafile design_name...MISMATCHED
Total of 405 bits mismatched.
```

Connection to Your Target System

You need appropriate pins on the target system for connecting the target system board to the header connection on the cable. These connectors must be standard 0.025″ square male pins that have dedicated traces to the target LCA control pins. You can connect to these pins with a header connector or a flying lead connector. (Both types are provided.)

IMPORTANT

The XChecker cable draws its power from the target system through V_{CC} and GND. Therefore, power to XChecker, as well as to the target FPGA, must be stable. Do not connect any signals before connecting V_{CC} and Ground. See the "Troubleshooting Guide" section.

Header Connector

Two standard 9-pin (8 signals, 1 key) header connectors that fit 0.025″ square male pins are provided. The pins are in the same order as the entries in Table 5-4 and Table 5-5. These header connectors are keyed to assure proper orientation to the cable assembly.

Flying Lead Connectors

Two flying lead header connectors with eight standard individual female connectors on one end that fit onto 0.025″ square male pins are provided. Each lead is labeled to identify the proper pin connection.

Cable Connections

Connections between the cable assembly and the target system use 16 leads. The cable has 14 signal connections, plus V_{CC} and Ground. Not all of the signal pins are required for every function. Figure 5-7,

Figure 5-8, Figure 5-9, and Figure 5-10 show the necessary connections for specific applications.

Once installed properly, the connectors provide power to the cable, transmit your system clock signal, allow download and readback of configuration data, and provide for logic probe sampling of configuration-related pins.

Each pin has a 100 Ω series resistor. You must provide an external pull-up resistor (approximately 10–50 kΩ) where indicated. Table 5-4 and Table 5-5 list all pins and appropriate connections.

Table 5-4 XChecker Cable Connections and Definitions

Name	Function	Connections		
		XC2000	XC3000	XC4000
VCC	Power — Supplies V_{cc} (5 V, 100 mA, typically) to the cable.	To target system V_{CC}		
GND	Ground — Supplies ground reference to the cable.	To target system Ground		
CCLK	Configuration Clock* — Provides configuration clock to the target system during configuration and readback.	To target system Configuration Clock		
D/P	Done/$\overline{\text{Program}}$* — Signals end of configuration for all families, and provides a re-program pulse for XC2000/XC3000 parts.	Connect to the D/$\overline{\text{P}}$ pin with a 10–50 kΩ pull-up resistor.		Connect to the target system DONE pin with a 10–50 kΩ pull-up resistor.
DIN	Data In — Provides configuration data to the target system during configuration and is 3-state at all other times.	Connect to the target system D_{IN}.		
PROG	$\overline{\text{Program}}$* — Provides a reprogram pulse for XC4000 parts.	Unconnected.		Connect to target system $\overline{\text{PROG}}$ with a 10–50 kΩ pull-up resistor.

Name	Function	Connections		
		XC2000	XC3000	XC4000
INIT	Initialize* — A status pin indicating the start of configuration for XC3000/XC4000 parts. For XC4000 devices, a logic zero on this pin during configuration indicates that a CRC error has occurred.	Unconnected.	Connect to the target system INIT with a 10–50 kΩ pull-up resistor.	
RST	$\overline{\text{Reset}}$* — After configuration, this pin can drive Low to reset the target LCA internal latches and flip-flops.	Connects to the target LCA $\overline{\text{RESET}}$ pin with a 10–50 kΩ pull-up resistor.		User-programmable connection; requires a 10–50 kΩ pull-up resistor.
RT	Read Trigger* — Initiates a read back by causing a Low-to-High transition at the target LCA RTRIG pin.	Connect to M0/RTRIG with a 10–50 kΩ pull-up resistor.		User-programmable connection; requires a 10–50 kΩ pull-up resistor.
RD	Read Data* — Read-back data from the target LCA is read at this pin.	Connect to M1/RDATA through a 10–50 kΩ pull-up resistor (must be slave serial configuration mode).		User-programmable connection; requires a 10–50 kΩ pull-up resistor.
TRIG	System Trigger* — A Low-to-High transition on this pin signals the XChecker electronics to initiate a readback.	Connect to the target system read back trigger.		
TDI TCK TMS	Reserved*	Do not connect. These pins are reserved.		

Name	Function	Connections XC2000 XC3000 XC4000
CLKI	Clock Input — Transmits your system clock to the XChecker electronics. This clock must be from 120 kHz to 10 MHz. Connecting this pin to the clock of the target system allows synchronizing the triggering of readback to the target system clock.	Connect to source of target system clock (for synchronous readback and probing). See Figure 5-10.
CLKO	Clock Output — The target system clock is on this pin. The clock can come from either the CLKI pin, or it be internally generated by the XChecker cable.	Connect to destination of target system clock (for synchronous readback and probing). See Figure 5-10.

*These signals can be sampled and their logic states displayed. See the Status command in the section titled "Valid Commands in Interactive Mode."

NOTE *XChecker does not drive the configuration mode pins (M0, M1, M2) during configuration. Logic levels for these pins must be specified by the user externally.*

Table 5-5 Operation Mode Connections

Header	Pin Name	Download Only	Download, Readback, & Synchronous Logic Probe	Download, Readback, & Asynchronous Logic Probe
1	VCC	X	X	X
	GND	X	X	X
	CCLK	X	X	X
	D/P	X	X	X
	DIN	X	X	X
	PROG	X*	X*	X*
	INIT	X**	X**	X**
	RST	X	X	X
2	RT	N/C	X	X
	RD	N/C	X	X
	TRIG	N/C	Opt	Opt
	TDI	N/C	N/C	N/C
	TCK	N/C	N/C	N/C
	TMS	N/C	N/C	N/C
	CLKI	N/C	X	N/C
	CLKO	N/C	X	N/C

Individual pin descriptions and corresponding connections to the target circuit board are listed in Table 5-4.

X = Connect as indicated in Table 5-4
N/C = Leave unconnected
* = Connect only to XC4000 devices
** = Connect only to XC3000/XC4000 devices
Opt = Optional. XC4000 devices can have the system trigger connected directly to the target LCA RTRIG pin to latch the state of the device instead of waiting for the XChecker software to initiate the readback.

Connecting for Download

Connect the XChecker cable to the host system and your target FPGA device, as shown in Figure 5-7. This only allows you to download the design. You cannot verify the design.

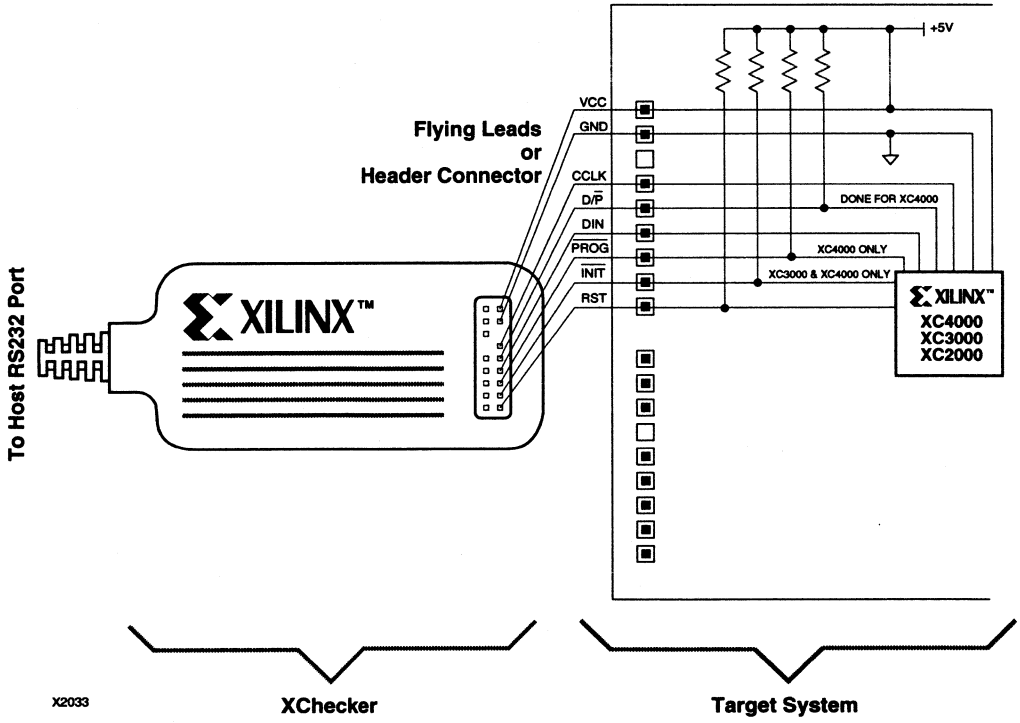


Figure 5-7 Downloading Configuration Data

If you are using the sample design, XROMs, you need a jumper between the device pins TO_CLKI and FROM_CLKO to provide a system clock. You can determine the location of these pins from the report file xroms.rpt. For XROMs, these pin locations are also listed in Table 5-2.

Connecting for Verification

You can connect XChecker for download and verification or only verification of a configured LCA. To download and verify an LCA, connect XChecker as shown in Figure 5-8. To verify a previously configured LCA, connect XChecker as shown in Figure 5-9.

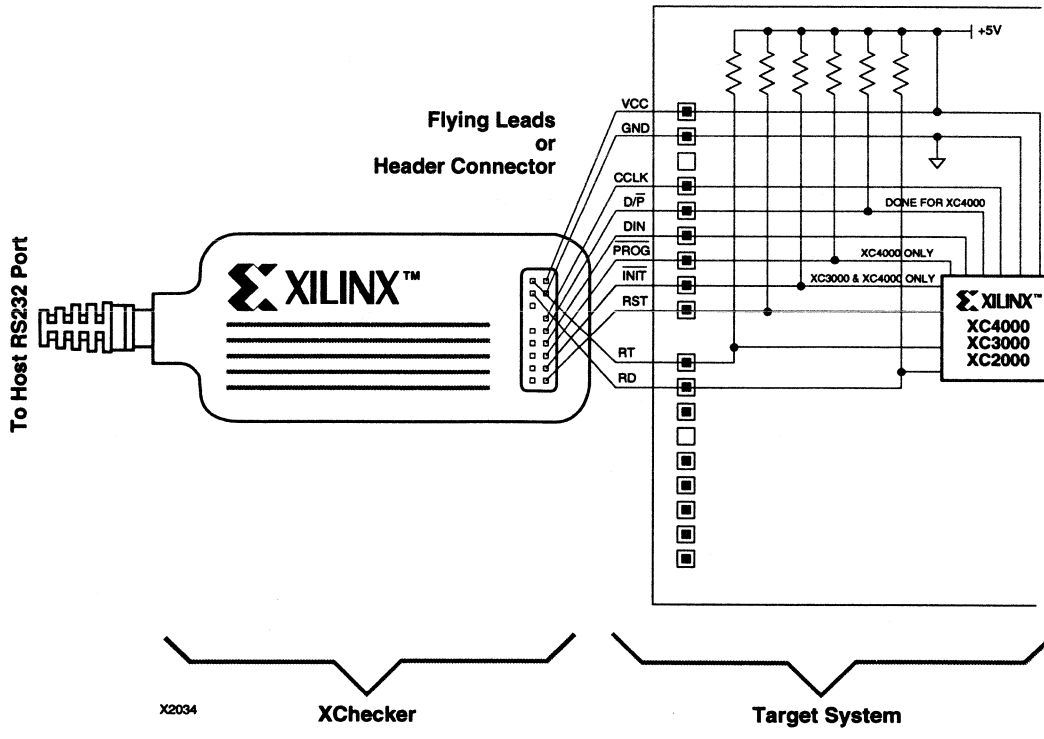


Figure 5-8 XChecker Cable Connections for Downloading and Verification

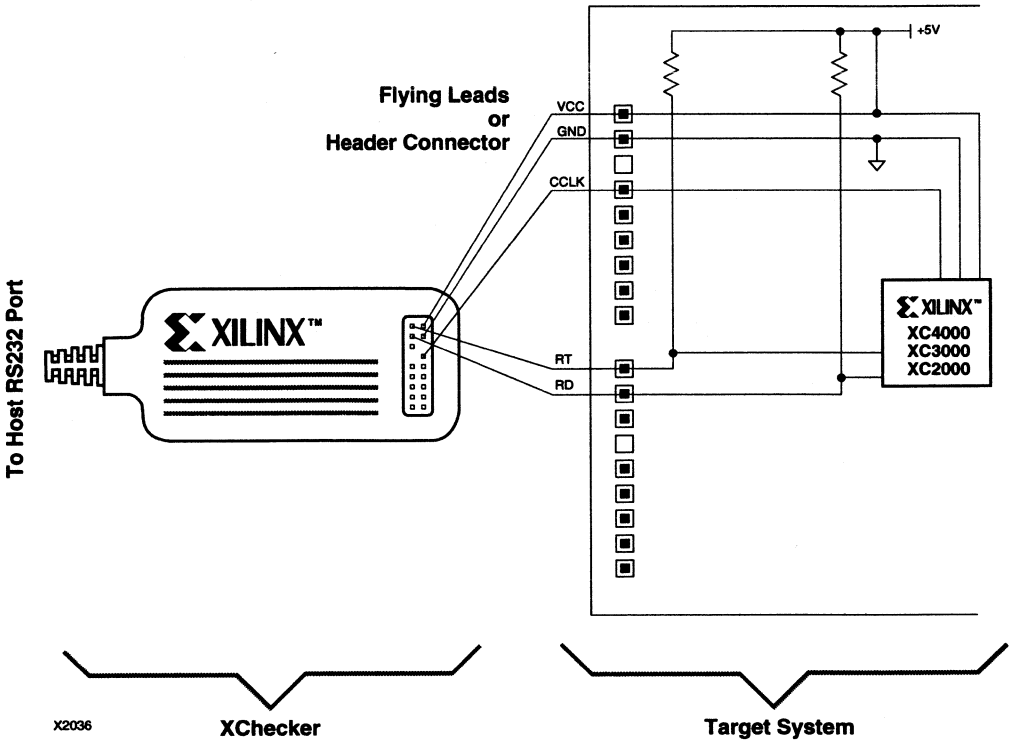


Figure 5-9 XChecker Cable Connections for Verification Only

Pay attention to the following guidelines.

- If you are using the sample design, remember to place a jumper between the signals TO_CLKI and FROM_CLKO to provide a system clock. (Since verification of the configuration bitstream is intended, it makes no difference whether you connect the clock from the target system to XChecker.)
- Remember to connect XChecker RT and RD pins to the LCA (programmable) RTRIG and $\overline{\text{RDATA}}$ pins. If you used the primitives MD0 and MD1 to place these signals, you might determine the location of these pins using the pinout tables in the Databook (pins M0 and M1). If you used regular IPAD/OPAD primitives, consult the design.rpt file.

Connecting for Synchronous Probing

Connect XChecker to the target LCA, as shown in Figure 5-10. This allows you to download, probe, and verify your design.

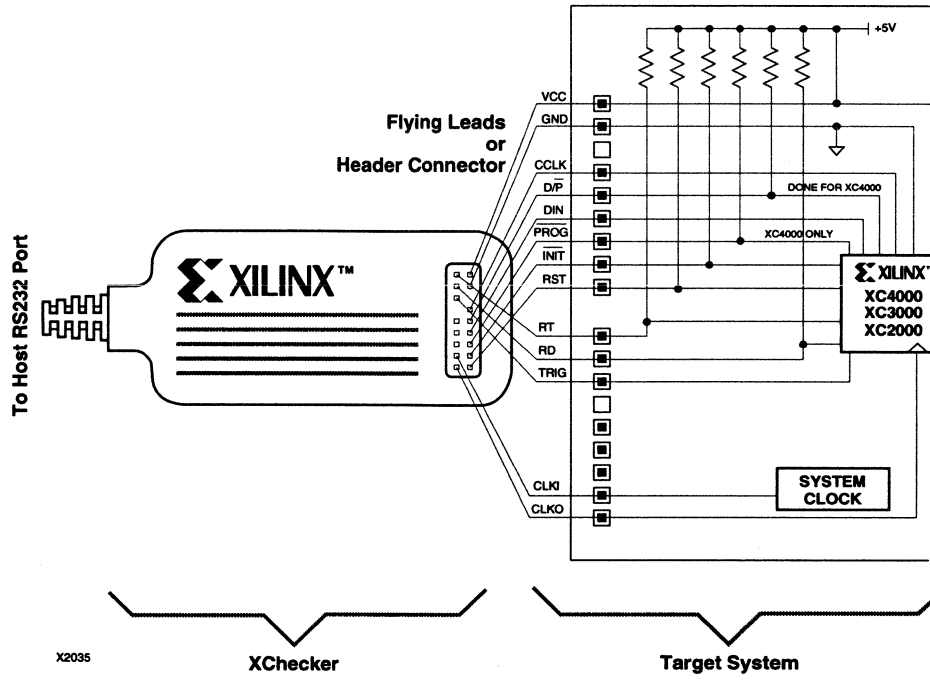


Figure 5-10 XChecker Cable Connections for Synchronous Probing

NOTE

XC3000 design internal logic probing must be synchronous. This means that XChecker interrupts your LCA system clock, executes readback of the LCA internal logic, and reapplies your target LCA system clock.

Pay attention to the following guidelines.

- Be sure to connect XChecker CLKO and CLKI pins, as shown in Figure 5-10, from the clock source to the XChecker CLKI and from the CLKO to your target LCA. This allows XChecker to control the target LCA system clock. If you are using the sample design, consult Table 5-2 (or the file XROMS.RPT) and connect a jumper between the signals FROM_CLKO and TO_CLKI to the XChecker pins, CLKO and CLKI, respectively.
- To use an external trigger such as, the terminal count of a counter or some other condition in your target board to initiate a readback, make sure to connect this signal to the XChecker TRIG pin. You can also use an internal trigger (pressing a key on the keyboard) to initiate a readback.

- Make sure to connect the XChecker RT and RD pins to the LCA RTRIG and RDATA pins, respectively. If you used the symbols MD0 and MD1, consult the specific part pinout tables. If you used normal PADs, consult your *design*.RPT file.

Connecting for Asynchronous Probing

Connect XChecker to the target LCA, as shown in Figure 5-11. This allows the target system to run while a readback is executed. (The sample design can be used in these examples.) Pay particular attention to the clock and trigger connections.

- Be sure to provide a system clock. If you are using the sample design, provide a jumper between the signals TO_CLKI and FROM_CLKO to leave the system running. There is no need to provide the system clock to XChecker, as readback is executed independently from the system clock.
- Connect an external trigger to the XChecker TRIG pin. If you are using the XROMs sample design, connect the signal BIT3 (pin P66 if you are using the sample design bitstream for an XC4003PC84) directly to the TRIG pin of XChecker.

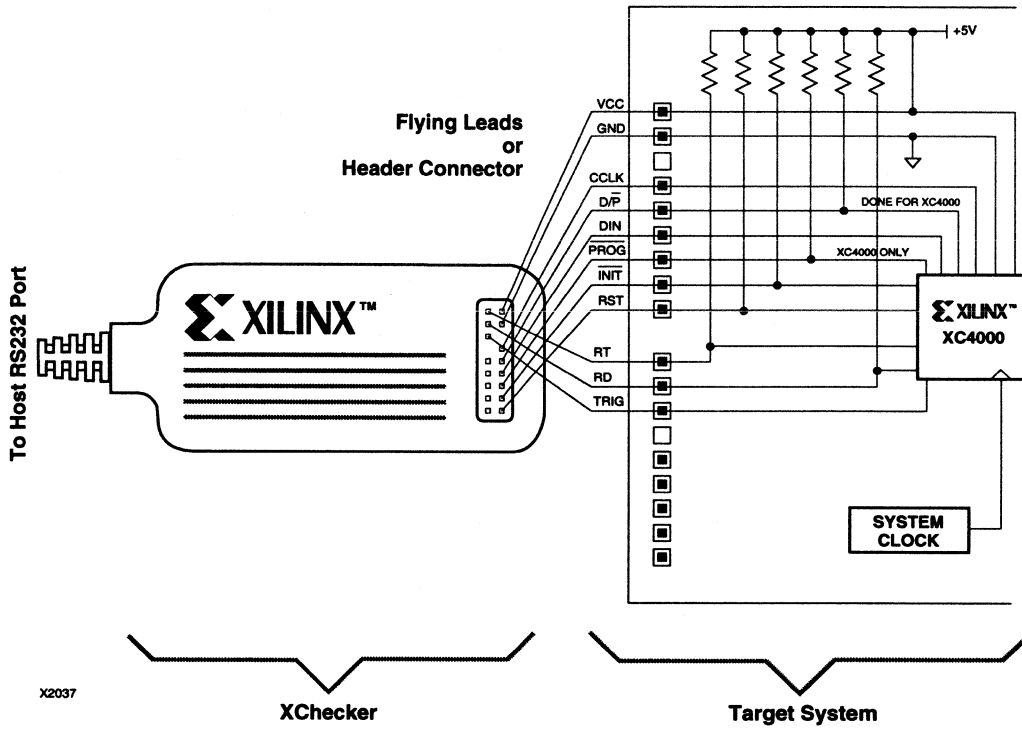


Figure 5-11 XChecker Cable Connections for Asynchronous Probing

NOTE *Connecting the system trigger the XChecker TRIG pin will insert one extra clock before the readback is initiated.*

Using the XChecker Software

This section describes the files and commands used by the XChecker software.

XChecker Files

You need to become familiar with the files described below that are used by the XChecker software. Most of these files are described in the MakeBits and MakePROM chapters of the *XACT Reference Guide, Volume 2.*

design.bit

The *design*.bit file contains the configuration information for the target FPGA design (.bit file). This file is generated by using MakeBits and

must be located in the same directory that was used to start the XChecker software.

design.ll

The *design*.ll file is the logic allocation file containing the bitstream positions of flip-flops, latches, RAM and CLB outputs, IOB inputs, and IOB registered outputs. This file must be located in the same directory that was used to start the XChecker software.

If you intend to probe the target FPGA internal logic states, refer to the MakeLL command in the MakeBits chapter of the *XACT Reference Guide, Volume 2*.

design.rbt

The *design*.rbt file is the ASCII equivalent of the *design*.bit file. Raw BIT files can be used to configure a target LCA.

xchecker.pro

The xchecker.pro file contains the default values for all XChecker options: part, design, baud, and port. These option values are updated at the end of every XChecker session. For XChecker to recognize an xchecker.pro file, it must be located in the same directory that was used to start the XChecker software.

parttype.ll

The part type files are generic logic allocation files for all of the part types that might be targets such as, 4005pc84.ll. They are provided by the XChecker software and are needed for verification and probing of the bitstream.

batch_file.cmd

The batch files are text files used to execute commands in the batch mode, and the extension ".cmd" is required.

design.exo, .mcs, .tek

These design files contain the configuration information for each design. Extensions for these files are ".exo, .mcs, and .tek." These files are optional since the XChecker software can take a BIT file as input. They are created by using MakePROM.

Invoking XChecker

XChecker can be invoked in any of three ways, as follows:

- Command line entry from the system shell. Only download and readback are supported. There is no capability to interactively probe the internal logic.
- Command line entry from XDM.
- Interactive Commands from the system shell. This mode offers additional commands for download and readback and also provides for probing the internal logic states of the target LCA device.

Downloading

A design can be downloaded after connecting the XChecker cable to the host system and target FPGA. To download the XROMs sample design, enter the following at the operating system prompt.

```
xchecker xroms
```

NOTE *The device you are downloading with the XChecker cable must be in serial slave mode.*

Since no options were specified, the XChecker software selects the port where the cable is connected and set the baud rate to the maximum allowed by the platform. The communication port, as well as the baud rate, can be modified by changing the appropriate settings in the xchecker.pro file.

1. To download in an interactive mode, enter the following command at the system prompt.

```
xchecker
```

You will see the following message on the screen.

```
Xilinx (R) XChecker 1.00 -Download/Readback LCA via
download cable
Copyright (C) Xilinx, Inc. 1991. All rights reserved.

Cable ID type is 'SERIAL-Readback'
Cable is connected to 'com1'
Baud rate is 115200
```

2. Then, enter this command string:

load *design name*

The following message appears on the screen.

```
About to download 'xroms.bit'.
Press ENTER key to continue or Q to quit:
```

3. After pressing Enter, the following message displays:

```
Total of 11875 bytes transmitted.
DONE signal went high.
Transmitting time = 1.81 secs
```

XChecker can also be accessed within the Xilinx Design Manager (XDM). To do so, select XChecker from the Verify menu, or enter "xchecker" at the command line.

The total bytes transmitted, the time required, the baud rate, and the port used varies depending on your selection of part type and platform.

Note that connecting the XChecker cable to download a bitstream only does not allow verification of the bitstream after configuration. You cannot use the –v option.

Verifying

After an FPGA has been properly configured, you can verify its configuration and compare it to your original design.

In most applications, verification is not needed, but this feature can be helpful with designs that experience extremely unstable or noisy V_{cc} conditions, or if your design standards require that stored data be read back periodically. Refer to the *XACT User Guide* for more information about how Readback and Verification are implemented.

To download and verify the XROMs sample design, enter the following at the operating system prompt.

```
xchecker -v xroms
```

Specifying the –v option causes the XChecker cable to download the file design.bit to the target LCA and initiate a readback immediately after the configuration data has been downloaded. See the waveforms in Figure 5-12.

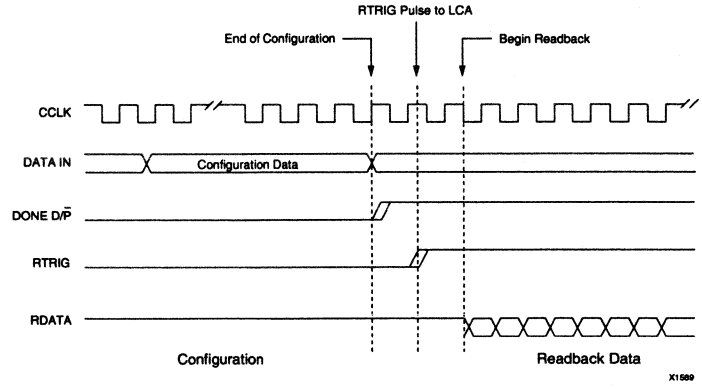


Figure 5-12 Waveforms for Verification After Download; XChecker –v Design

To execute a readback after the device has been in operation, you must use interactive commands. The following sequence illustrates the process.

```
OS_PROMPT> xchecker
```

This invokes the interactive mode, and the prompt "XCHECKER ?" appears.

```
XCHECKER ? load design_name
```

The load command downloads *design*.bit to the target LCA. After the target LCA is in operation and a readback is desired, you can execute a verify command.

```
XCHECKER ? verify design_name
```

This initiates a readback, and the data is compared to the file *design*.bit. See Figure 5-13.

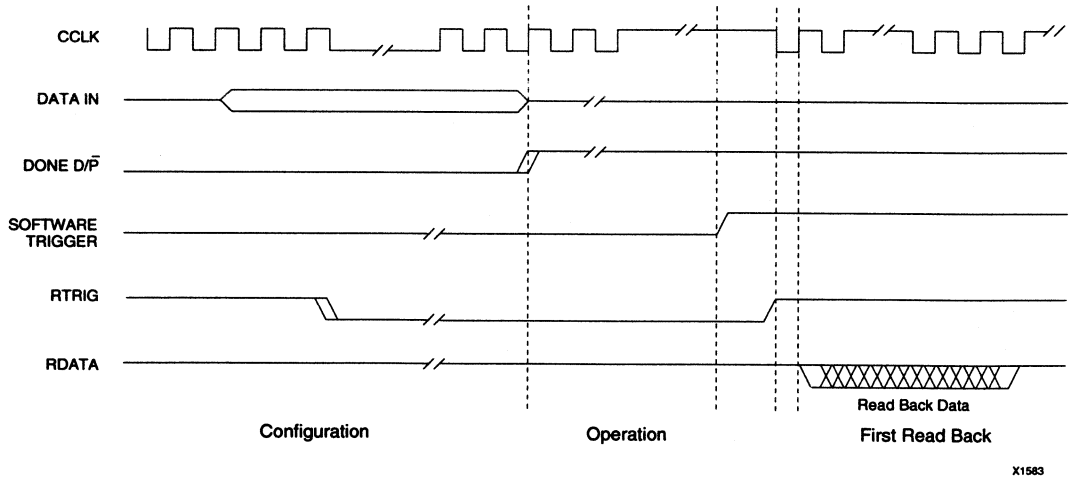


Figure 5-13 Waveforms for Verification Some Time After Download

Probing

Probing the LCA internal logic is done by executing a readback of all the configuration data and extracting the internal logic states from it. You might want to probe the internal logic to debug your designs using XChecker as an in-circuit emulator.

XC2000 and XC3000 Designs

You need to stop the LCA system clock while Readback is in progress to obtain a static "snapshot" of the internal logic states. XChecker controls the LCA system clock if appropriate clock connections have been made. See Figure 5-10.

Example 1

Suppose you want to determine the eight consecutive values of a design, s_0, s_1, s_2, and s_3, after an interrupt to a microprocessor in the board occurs. The interrupt signal will be IR, connected to the XChecker TRIG pin.

1. Enter the xchecker command at the operating system prompt.

```
> xchecker
```

   This invokes the interactive mode and the prompt "XCHECKER ?" appears.

2. Load the design.

```
XCHECKER ? load design
```

Entering "load design" downloads *design*.bit to the target LCA. After the target LCA is in operation, probing internal flip-flops and latches is possible. For instance, the following would group s_0 through s_3 into a signal named s_bus.

```
XCHECKER ? group s_bus s_0 s_1 s_2 s_3
```

3. Set the display mode for s_bus to hexadecimal.

```
XCHECKER ? pick -hex s_bus
```

4. Apply the internal system clock to the target LCA, after a trigger has been acknowledged, at a rate of 2.75 MHz.

```
XCHECKER ? clock -int -speed 3
```

5. Set the readback trigger (IR) to be extermal and detected as a Low-to-High transition at the cable TRIG pin.

```
XCHECKER ? trigger -auto -clock 32 16
-timeout 3
```

After a readback command is executed, and upon detecting a trigger, the XChecker system generates 32 system clocks and then initiates a readback. Then, if several snapshots are read back by the readback command, 16 clocks are inserted between each readback. If a trigger is not received within three seconds after the readback command is executed, the XChecker system times out and does not execute a readback.

6. Use the readback command as follows:

```
XChecker ? readback 8
```

The XChecker system issues 32 system clocks on the CLKO pin (per the trigger command) and then reads back eight consecutive bitstreams (snapshots) after acknowledging the trigger and inserting eight internally generated system clocks between each snapshot (as described above). See the waveforms in Figure 5-14.

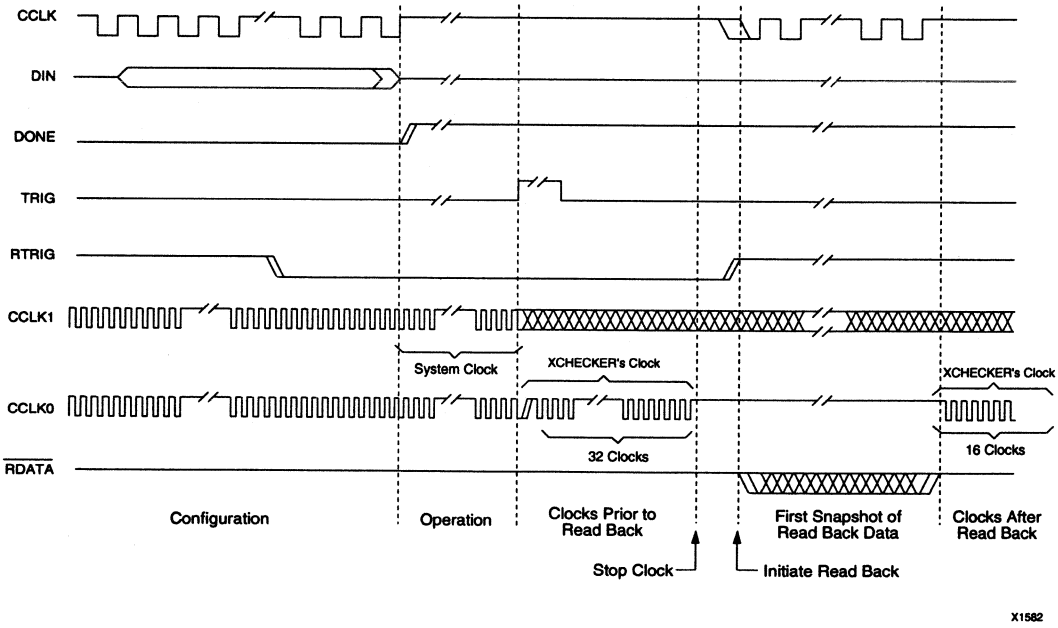


Figure 5-14 Waveforms for Collecting 1-of-16 Snapshots Using trigger –auto –clock 32 16 –timeout 3

7. Display the node values.

```
XChecker ? show -sn 8 8
```

The show command displays the node values on the screen, starting with snapshot 8 and ending with snapshot 16. The node values are displayed on the screen in the following format.

```
     S
     B
     u
     s
8:   A
9:   B
10:  C
11:  D
12:  E
13:  F
14:  0
15:  1
16:  2
```

Example 2: Single Step Analysis

Suppose you want to debug a design containing a state machine. You have the input variables A, B, C, and D, and the outputs signals LOAD and READ. You want to stop the state machine, initialize it to the initial state, and single step it, executing a readback in each state.

1. Enter the xchecker command without options at the system shell. to enter the interactive mode.

```
xchecker
```

In the interactive mode, all XChecker functions are available.

2. Load the design.

```
XCHECKER ? load 3kstate
```

3. Add the desired signals to the display list. The default format is binary.

```
XCHECKER ? pick A B C D LOAD READ
```

4. Now, setup the trigger. You want to trigger using your keyboard (the manual option).

```
XCHECKER ? trig -man
```

You need to determine the clock source to your system. You can choose between the clock already in your board (–ext), which has been fed to XChecker, or one generated by XChecker (–int). For this example, the XChecker clock is used at speed 1 (921 kHz).

5. Set the clock source.

```
XCHECKER ? clock -int -speed 1
```

6. To single step the design, executing one readback between clock sequences, use this sequence.

```
clock -stop        #stop the clock
reset              #initialize state machine
readback           #read one snapshot
clock 1            #send one clock
readback -a        #append to previous snapshot
clock 10           #send 10 clocks
readback -a        #append to previous snapshot
clock -resume      #resume clock
```

7. To show the collected data for this sequence, use the Show command.

```
XCHECKER ? show -vdata
```

The collected data appears as follows.

```
          LR
          OE
          AA
----ABCDDD
  0:000010
  1:010110
  2:011001
```

XC4000 Designs

You can read back an XC4000 LCA in two ways.

- Synchronous Probing

  You can stop the system clock and execute a readback (as done with the XC3000 and XC2000 devices). This method is synchronous probing, as each readback is synchronized to the system clock.

- Asynchronous Probing

  You can keep the system clock running while the readback is in progress. The XC4000 devices latch their internal logic states in special latches when a readback is requested. Therefore, the system clock need not be stopped to collect coherent data. This method is asynchronous probing, as readback is independent of the system clock.

Synchronous Probing

In the following examples you pass control of the system clock to XChecker to synchronize readback.

You can use any combination of clock (target or XChecker) and trigger (manual, none, or auto). In the examples below, the focus is not on the clock but on the use of the trigger command. Use the Clock command to select either the internal XChecker clock or the external target system clock.

Using the Keyboard to Initiate a Readback (trig –manual)

1. First, enter the following command at the system shell.

```
xchecker
```

2. Then download the XROMs design.

```
XCHECKER ? load xroms
```

3. Now, group the signals ROM 0–7 into a more convenient name such as "pattern."

```
XCHECKER ? group pattern ROM7 ROM6 ROM5 ROM4
ROM3 ROM2 ROM1 ROM0
```

4. Prepare to display the new signal "pattern" in binary format (default).

```
XCHECKER ? pick pattern
```

5. Next, setup the trigger for the subsequent readback. This causes XChecker to reset the LCA before readback and inserts one clock pulse between each subsequent readback.

```
XCHECKER ? trig -manual -reset -clock 0 1
```

6. To readback 16 snapshots enter the following command:

```
XCHECKER ? readback 16
```

   XChecker waits for an Enter key signal before starting a readback. After the readback is completed, display the signal "pattern" on the screen with the Show command. The default display format is a vertical listing of signal data.

7. Use the horizontal option (–hdata) to display the signal horizontally.

```
XCHECKER ? show -vdata
```

   The pattern in Figure 5-15 corresponds to the contents of the 16 ROM cells beginning at address 0.

```
p
a
t
t
e
r
n
0:   01111110
1:   01111110
2:   10111101
3:   10111101
4:   11011011
5:   11011011
6:   11100111
7:   11100111
8:   11100111
9:   11100111
10:  11011011
11:  11011011
12:  10111101
```

```
13:  10111101
14:  01111110
15:  01111110
```

Figure 5-15 16 Consecutive Snapshots Beginning at ROM Address 0

Using an External Trigger to Initiate a Readback (trig –auto)

To use an external trigger, you need a toggle switch or other means to provide a Low-to-High transition at the XChecker TRIG pin.

For the following examples, connect the signal BIT3 (pin P66 if you are using the sample design bitstream for an XC4003PC84) directly to the TRIG pin of XChecker. The signal BIT3 has only one Low-to-High transition when the Address (ADR0-3) increments from 9 to 10 (hex 9 to hex A), as shown in Figure 5-16. Therefore, because XChecker inserts one clock after the TRIG occurs, using BIT3 as a trigger should always produce the contents of the ROM Address 11 to be read back in the signals BIT 0–7.

NOTE *XChecker inserts one clock after acknowledging a TRIG before a readback is initiated.*

```
76543210
0:  01111110
1:  01111110
2:  10111101
3:  10111101
4:  11011011
5:  11011011
6:  11100111
7:  11100111
8:  11100111
9:  11100111
10: 11011011<— Low-to-High in signal BIT3 at address 10
11: 11011011<— Signals at address 11 read
12: 10111101
13: 10111101
14: 01111110
15: 01111110
```

Figure 5-16 Only One Low-to-High Transition in Signal BIT3

1. Enter the xchecker command at the system shell.

```
xchecker
```

2. Then, download the XROMs design.

```
XCHECKER ? load xroms
```

3. Group the signals ROM 0-7 into a more convenient name (pattern).

```
XCHECKER ? group pattern ROM7 ROM6 ROM5 ROM4
ROM3 ROM2 ROM1 ROM0
```

4. Prepare to display the new signal "pattern" in binary format (default).

```
XCHECKER ? pick pattern
```

5. Next, setup the trigger for the subsequent readback. An external trigger on the TRIG pin will initiate a readback.

```
XCHECKER ? trig -auto
```

There are no clocks issued before or after the snapshot is collected, as the clock parameter was not used with the "trig" command.

6. Use Readback to readback one snapshot.

```
XCHECKER ? readback
```

XChecker waits for the Low-to-High transition of BIT3 and initiates a readback.

7. Use Show to display the signals "address" and "pattern" on the screen.

```
XCHECKER ? show -hdata
```

```
a    p
d    a
d    t
r    t
e    e
s    r
s    n
0:   101111011011
```

The first four bits correspond to "address" 11 (hex B) and the last eight to the contents of the ROM cells at that address (pattern).

Initiating a Readback Without a Trigger (trig –none)

1. Enter the xchecker command at the system shell:

```
xchecker
```

2. Then, download the XROMs design.

```
XCHECKER ? load xroms
```

3. Group the signals ROM 0-7 into the name "pattern."

```
XCHECKER ? group pattern ROM7 ROM6 ROM5 ROM4
ROM3 ROM2 ROM1 ROM0
```

4. Prepare to display the new signal "pattern" in binary format (the default).

```
XCHECKER ? pick pattern
```

5. Next, setup the trigger for the subsequent readback.

```
XCHECKER ? trig -none -clock 8 1 -reset
```

With trig –none, an external trigger is not expected for a readback. XChecker resets the target device and inserts eight clocks before the first readback and one after that, if the following command is used.

6. To readback 16 snapshots, use Readback.

```
XCHECKER ? readback 16
```

The readback sequence starts immediately since trigger –none was selected.

7. Display the signal pattern on the screen, use the Show command.

```
XCHECKER ? show -vdata
```

```
p
a
t
t
e
r
n
0:   11100111
1:   11100111
2:   11011011
3:   11011011
4:   10111101
5:   10111101
6:   01111110
7:   01111110
8:   01111110
9:   01111110
10:  10111101
11:  10111101
12:  11011011
13:  11011011
14:  11100111
15:  11100111
```

Figure 5-17 16 Consecutive Snapshots Beginning at ROM Address 8

Since the readback command issued 8 clocks before the first snapshot, the pattern in Figure 5-17 corresponds to the contents of the eight ROM cells beginning at address 8 and wrapping around to address 0.

Asynchronous Probing

The following example uses the signal BIT3 as an external trigger with the –auto option to illustrate an XC4000 feature. The XC4000 devices latch the internal logic state in special latches upon receiving a Low-to-High transition at the RTRIG pin. Thus, when using XC4000 devices, it is not always necessary to connect the CLKI and CLKO pins to the target system.

Observe that the signal BIT3 experiences only one Low-to-High transition when the address (ADR 0-3) increments from 9 to 10 (hex 9 to hex A). Using BIT3 as a trigger always produces the contents of the ROM address 11 (hex B) to be read back in the signals BIT 0-7.

1. Enter the xchecker command at the system shell:

```
xchecker
```

2. Then, download the XROMs design.

```
XCHECKER ? load xroms
```

3. Now, group the signals ROM outputs (ROM 0-7) into a more convenient name, "pattern," and the address signals (ADR 0-3) into "address."

```
XCHECKER ? group pattern ROM7 ROM6 ROM5 ROM4
ROM3 ROM2 ROM1 ROM0

XCHECKER ? group address ADR3 ADR2 ADR1 ADR0
```

4. Prepare to display the new signals "pattern" and "address" in binary format (the default).

```
XCHECKER ? pick address pattern
```

5. Next, setup the trigger for the subsequent readback. An external trigger will initiate a readback.

```
XCHECKER ? trig -auto
```

6. To capture one snapshot, use Readback.

```
XCHECKER ? readback
```

7. The readback occurs on the next Low-to-High transition of BIT3. To display the signals "address" and "pattern" on the screen, use the Show command.

```
XCHECKER ? show -hdata

a    p
d    a
d    t
r    t
e    e
s    r
s    n
0:   101111011011
```

The first four bits correspond to "address" (hex B) and the last eight to the contents of the ROM cells at that address "pattern."

How to Use XChecker to Probe RAM Bits in an XC4000 Part

Probing internal RAM bits is different than probing other signals since internal RAM bits are not shown in the design.ll file.

There are two ways you can probe the value of internal RAM bits.

- You can probe the output net of the RAM cell and address the RAM bit in question.

- You can probe the individual RAM bit(s) by specifying the "bit name" at the probe command.

Of the two methods, the first one probably involves less work. The second method, however, provides the capability to probe any RAM bit at any time, without having to address it.

Example of Probing Individual RAM Bits

Probing individual RAM bits requires you to use EditLCA (XDE) to determine the CLB location of the RAM cell. The CLB location is then used to identify the RAM bit.

Suppose that you have a design that implements a RAM using the hard macro RM16X4H (available from your schematic library). Suppose that the output nets of the RAM are called *my_ram 0* through *my_ram 3*. As an example you may want to probe the bits at address 7. PPR implements this RAM macro using two CLBs. To find which CLBs PPR used to implement the RAM, do the following.

1. Enter EditLCA.

   You enter EditLCA by invoking XDE, selecting the design, and selecting EditLCA from the Programs menu. (Refer to the XDE chapter for more information.)

2. Find the output nets of the RAM.

   In EditLCA, you can do this using the following sequence of mouse selections.

   Screen → Findnet

   At the prompt, enter the names of the output nets. Use the wildcard symbol (*) to represent all the outputs.

3. Determine the CLB position of the RAM.

   The mouse points to the CLB where the output net is located, and the location of the CLB is displayed at the command line. An example of a CLB location is CLB_R2C3.

4. Use the CLB location to identify the desired bits. The following notation identifies RAM bits in a CLB.

   CLB_LOCATION.M F_or_G *bit_number*

   An example of CLB_LOCATION is CLB_R2C3. The character "M" is mandatory to indicate that a memory bit follows. F_or_G are the characters "F" or "G," depending on the function generator used to implement the RAM. In a CLB implementing two 16x1 RAMs, use "F" for nets sourced by the X pin and "G" for nets sourced by the Y pin. Omit this character in CLBs

implementing 32x1 RAMs. The *bit_number* is a number from 0 to 15.

For example, suppose that in your 16x4 RAM, the output nets are sourced as follows.

my_ram0 sourced by CLB_R2C3.X

my_ram1 sourced by CLB_R2C3.Y

my_ram2 sourced by CLB_C3R3.X

my_ram3 sourced by CLB_C3R3.Y

The bits at address 7 are identified as below.

BC.MF7 for bit 0 at address 7

BC.MG7 for bit 1 at address 7

BC.MF7 for bit 2 at address 7

BC.MG7 for bit 3 at address 7

In another example design, where CLB_CxRy is part of a 32 x m(0-31) RAM, the notation for bit n is as shown below.

CLB_CxRy.Mn

where n is any number from 0 to 31.

5. Use the pick command to add the desired bits to the "pick set."

```
pick -add BC.MF7 BC.MG7
```

Displaying Readback Data in the Viewlogic Viewwave Environment

Readback data from the target system can be displayed in the Viewlogic Viewwave environment. This is accomplished with the XChecker Export command in the interactive mode. The following sequence is an example of a typical display session.

1. Download a bitstream file:

```
load design
```

2. Select the signals to probe with this command string:

```
probe -add tog1 tog2 tog3
```

3. Select signals for display with this command string:

```
pick -add tog1 tog2 tog3
```

4. Read back three snapshots using this command string:

```
readback 3
```

5. Write the results to a GEN file:

```
export -v design
```

Invoke Workview, and from the Workview screen, enter the following sequence of mouse clicks on the menu, using the left mouse button.

Window → Open → Viewwave → Generic

At the prompt, enter the design name.

At the prompt, double-click the middle mouse button to select the entire screen.

The Viewwave screen displays the snapshots of collected Readback data at equally spaced intervals. There is no time reference, because the XChecker software reports sets of readback data, not real-time information.

Valid Options from the Command Line

Valid XChecker command line options are described below. The data files are configuration bitstream files in BIT or RBT format, or in any one of the supported PROM formats mentioned above: TEK, EXO, or MCS. When you do not specify any options or data files, the XChecker system defaults to the interactive mode. If you do not specify any options but you do specify a data file, a default set of options specified in the xchecker.pro file is used to set the port.

Command line syntax is as below.

`xchecker options` *datafile*

NOTE

In these descriptions and examples, possible options are shown in brackets. Variables that you can name or specify are shown in italic typeface. All options can be abbreviated to the minimum number of distinctive characters in the option name.

Commands, options, and signal names are not case-sensitive.

– batch Batch Mode Operation

Syntax –batch *bat_file*.cmd

Abbreviation b

The Batch option executes commands in batch mode. The indicated *bat_file* must have a ".cmd" extension and contain valid XChecker commands, including interactive commands. You can add comments to files by using the # symbol, either on the command line or on a new line.

– h The Help Option

Syntax –help

Abbreviation h

The Help option displays command line usage information.

– pa Specify Part Type

Syntax –part *parttype*

Abbreviation pa

The Part Type option defines the part to be used such as., 3090pc84, 3020APC68, or 4005PC156. This option is only required when configuration information is stored in a PROM file (MCS, TEK, or EXO) and readback, verification, or probing arre to be executed.

– po Specify Port Name

Syntax –port *portname*

Abbreviation po

This option identifies the port used. If it is not specified, the default is AUTO, which searches for a cable connected to any port, parallel or serial. Valid ports for supported platforms are listed in Table 5-6 .

Table 5-6 Valid Ports for the XChecker Cable

IBM PC	com1	com2	lpt1	lpt2
NEC	com 1			
Apollo	/dev/sio1	/dev/sio2		
DEC3100	/dev/ttyd0	/dev/ttyd1		
Sun	/dev/ttya	/dev/ttyb		

– v Verify Download and Readback

Syntax –v

Abbreviation v

The verification option executes a download and readback of the current LCA design for verification. Configuration data is read back from the LCA and compared with the original bitstream. If you do not specify the Verify option, readback is not executed after configuration.

Valid Commands in the Interactive Mode

Interactive commands are accessed by entering "xchecker" at the system prompt. Valid commands are listed next.

NOTE

In these descriptions and examples, possible options are shown in brackets. Variables that you can name or specify are shown in italic typeface. You can abbreviate the commands using the least number of distinctive characters, as with the command line options, but you must use at least two characters. Also, you can repeat the previous command by entering either an equal sign (=) or an exclamation point (!).

Batch Execute in Batch Mode

Syntax batch *bat_file*.cmd

Abbreviation bat

The Batch command executes commands in a batch mode. The indicated *bat_file* must have a ".cmd" extension and contain valid XChecker commands. As an example, to execute the file *bat_file*.cmd, you could use either of the following methods.

XCHECKER ? batch *bat_file*

XCHECKER ? < *bat_file*

The "#" sign can be used to insert comments in a batch file. Any text after a "#" sign will be ignored.

Baud Specify Baud Rate

Syntax baud *baud_rate*

Abbreviation bau

This command specifies a new communication baud rate. Upon initialization, the fastest baud rate for your host system is automatically selected. Valid baud rate values are listed in Table 5-7 below.

Table 5-7 Vaild Baud Rates

	Baud Rate			
Platform	**9600**	**19200**	**38400**	**115200**
IBM PC	X	X	X	X
NEC PC	X			
Apollo	X	X	X	
DEC3100	X	X	X	
Sun	X	X	X	

Clock Specify Clock Source

Syntax [–intl–ext] [–speed *1/3/5/11*] [–stopl–resume] *nclocks*

Abbreviation cl

The Clock command specifies the source of the system clock for the target LCA. This clock is made available at the CLKO pin and is used for single-stepping the target LCA during a hardware debug session. The following are the available options and parameters.

– int Internal Clock

The internal XChecker clock is to be used during single stepping.

– ext External Clock

The external system clock is to be used during single stepping. Use this option if you need a clock speed other than the ones provided with XChecker. Clock frequencies must be in a range from 120 kHz to 10 MHz and are applied at the CLKI pin.

– speed *x* Clock Speed

Using the Speed option specifies the speed of the clock inside XChecker by selecting the –int option. You can select this clock to appear on CLKO. The rates, indicated by variables, are listed below.

1 ~0.921MHz
3 ~2.75MHz
5 ~5.50MHz
11 ~11.0MHz

The Speed option specifies the system clock rate during single stepping as well as the readback clock rate. XChecker limits XC4000 readback to 5.5 MHz. If option 11 is selected for an internal burst clock, XChecker will burst clocks at 11 MHz on the CLKO pin between readback snapshots and slow to 5.5 MHz during actual snapshots.

When performing Readback with XC2000/XC3000 devices, only option variable 1 is valid (0.921 MHz).

– stop Stop the Clock

This option stops the clock on the CLKO pin: the XChecker system basically parks the CLKO pin High. This option is used in combination with the Resume option to single step the target LCA system clock, assuming the proper connections have been made.

– resume Reapply the Clock

This reapplies the clock on CLKO. By selecting the –ext option, the XChecker system basically becomes a transparent buffer between the CLKI and CLKO pins, with a delay of approximately 100 ns across it. This option is used in combination with the Stop option to single step the system clock, given that the proper connections have been made.

nclocks Specify the Number of Clocks the System Issues

The variable *nclocks* specifies the number of clocks the system issues to single step the target LCA. Valid values for *nclocks* range from 0 to 32767. This variable must be specified without any options, and is only valid if a Clock command with a Stop option has previously been issued.

The following is the sequence for single stepping an LCA design.

`clock -stop`	Stop the system clock provided on the CLKO pin
`clock -speed 3`	Set the system clock (CLKO pin) to 2.75 MHz
`clock 32`	Send 32 clocks from the CLKO pin
`clock 8`	Send eight more clocks from the CLKO pin
`clock -resume`	Reapply the free-running system clock on the CLKO pin

NOTE *The XChecker system introduces approximately a 100 ns delay between the CLKI and CLKO pins. Also, when single-stepping a target device during debug, the system clock is applied in bursts of up to 127 clocks. To apply more than 127 clocks, the XChecker system pauses between bursts.*

Browse Scan Down Data Display

Syntax browse

Abbreviation br

Browse is used only on the PC for scanning data displayed when you use the Show command. The screen shows 24 lines of data and the prompt "more." Enter "y" to see another 24 lines of data, or "n" to quit Browse.

Diagnostics Perform Cable Check

Syntax diag *num*

Abbreviation dia

The Diagnostic command performs a check of the XChecker cable hardware. To execute this command, you need to place the test fixture on the XChecker cable, and connect the fixture to V_{CC} and Ground. The Diagnostics include tests to verify that data can be transmitted and received at the different baud rates and tests to assure that the internal LCA and RAM are operational.

The variable *num* represents the number of times to repeat the diagnostic tests (default is 1).

NOTE *This command requires about one minute to execute.*

Exit Terminate Session

Syntax exit

Abbreviation exi

The Exit command terminates the current XChecker session and returns to the system shell. You will be asked whether to save the profile information. All current program options are saved to the xchecker.pro file.

Export Save Readback Data

Syntax export [–v] *save_file*

Abbreviation exp

This command saves the current readback data to the file *save_file*. The file *save_file* is written in one of two formats: a binary file or a Viewlogic generic (GEN) file. The binary format is useful for saving readback data to be displayed in later XChecker sessions. The GEN file is used to display readback data in Viewwave format. See also the Import command.

– v Save as Viewlogic Generic File

Save readback data to a Viewlogic GEN file. Viewlogic users can take advantage of this feature by displaying the readback data in the Viewwave environment.

Group Define/Name a Signal Group

Syntax group *group_name signal_list*

Abbreviation gr

The Group command defines a group name for signal_list. The variables *group_name* and *signal_list* specify the name "group_name" for the list of signals in "signal_list." An asterisk (*) can be a wildcard. The signal names are displayed in 24-line segments. Enter a "y" to scroll forward to the next 24 lines. Enter an "n" to exit Group. Consider the following examples.

```
group address a0 a1 a2 a3
```

The command above causes the signals a0, a1, a2, and a3 to be displayed as one single signal "address" by the Pick and Show commands.

```
group data d*
```

The command above causes all signals (in the file design.ll) beginning with the character "d" to be grouped in the name data.

Help Online Help

Syntax help *topic*

Abbreviation he

The Help command displays online help for the topic requested in 24-line segments. Enter "y" to scroll forward to the next 24 lines. Enter "n" to exit Help.

Import Retrieve Data

Syntax import *save_file*

Abbreviation im

The Import command retrieves the binary information stored by the export command from the file *save_file*. This is useful for displaying data from previous XChecker sessions. Refer to the Export command.

Load Download Design to LCA

Syntax load [–v] *design*

Abbreviation loa

This command downloads the specified design to your LCA, or chain of LCAs. With the –v option, the design is readback after download to verify the current configuration data. The variable *design* specifies the file design.bit or design.rbt (or design.tek, .exo, or .mcs) as the file containing the configuration information.

– v Verify Download and Readback

The verification option executes a download and readback of the current LCA design for verification. Configuration data is read back from the LCA and compared with the original bitstream. If you do not specify the Verify option, readback is not executed after configuration.

NOTE *This option is not appropriate for daisy-chained LCAs, since only the first device is read back.*

Log Send Screen Display to File

Syntax log –out *filename string*

Abbreviation log

The Log command sends the screen output to the file *filename*. It is useful for capturing the output of a Readback or a Show command.

– out Name File for Screen Display

Using log –out *filename string* closes any previous log file, opens a new one, and places the string at the beginning of the file.

The variable *string* is used to insert user comments into the log file, which normally only captures what is displayed on the screen.

List List Matching File Names

Syntax ls [–s] *spec*

Abbreviation ls

List directory entries of matching filenames. Use the asterisk (*) as a wildcard.

– s Search

This option expands the search for matching files in the directory indicated by the XACT environment variable.

Part Specify Part Type

Syntax part *parttype*

Abbreviation pa

The Part command specifies the LCA part type in use. Specifying the part type is only necessary if a PROM file is used as input and probing is desired. The XChecker program searches for a *parttype*.ll file in the current directory and then in the directory indicated by the XACT environment variable.

Pick Specify Signal and Display Format

Syntax pick [–binl–octl–hexl–dec] [–addl–deletel–all] *signal_list* [–clear]

Abbreviation pi

The Pick command selects the list of signals to be displayed by the show command (called the display set), as well as the format for their display. The signals defined by *signal_list* are a sub-set of the signals in the *design*.ll file; or they could be defined by the Probe command. The command can also be used to modify previous display settings. Using Pick without any options lists the signal names that have been selected. The list appears in 24-line segments. You can scroll forward by entering a "y". To exit Pick, enter an "n". Valid options are listed next.

– bin Binary

This option displays signals in binary format

– oct Octal

This option displays signals in octal format

– hex Hexadecimal

This option displays signals in hexadecimal format

– dec Decimal

This option displays signals in decimal format

– addl – delete Add/Delete Signal(s) from Display Set

Use –addl –delete [signal_listl–all] to add or delete a signal or list of signals to/from the display set. Use –all to add or remove all signals to/from the display set.

– clear Clear

This is equivalent to –delete –all

Examples

pick -bin -add ***address data***

The example above causes the signals named *address* and *data* to be added to the display set and displayed in binary.

pick -hex ***bus***

The example above causes the group of signals named *bus* to be added to the display set and displayed in hexadecimal. (See the Group command.)

```
pick -delete -all
```

The example above causes all signals to be removed from the display set.

Port Specify Download/Readback Port

Syntax port *portname*

Abbreviation po

Specifies the download/readback port. Table 5-8 lists the valid entries; the ports listed in bold face are the defaults. If the port is defined as Auto, all ports are scanned to search for a cable.

Table 5-8 Valid Ports for the XChecker Cable

IBM PC	com1	com2	lpt1*	lpt2*
NEC	**com 1**			
Apollo	**/dev/sio1**	/dev/sio2		
DEC3100	**/dev/ttyd0**	/dev/ttyd1		
Sun	**/dev/ttya**	/dev/ttyb		

* To be used with the parallel download cable only (no Readback).

Probe Define Signals to be Probed

Syntax probe [–add|–delete] *signal_list* [–all] [–clear]

Abbreviation pr

This command adds or removes signals (including RAM bits) to be probed (the probe set). These signals must be valid signal names found in the file *design*.ll. (See "XChecker Files" earlier in this chapter.) Using Probe without any options displays the signals that have been selected. The names are displayed in 24-line segments. To scroll forward, enter a "y". To exit Probe, enter an "n". The following are valid options for Probe.

– add| – delete Add/Delete Signals

Add or delete the specified signals. Add is the default.

– all Specify All Signals

Use the All option to specify all the signals contained in the file.

– clear Clear

Clear is equivalent to –delete –all.

Examples

```
probe -add -all
```

The probe command with the options specified above adds all the signals in the *design*.ll file to the probe set.

```
probe -add address mux
probe -delete add
```

These commands with the options specified above add the signals "address" and "mux" and removes the signal "add" from the probe set.

NOTE

The signals indicated in the probe command need to be valid node names, found in the design.ll file. Notice that these node names may differ from the original schematic names in that a hierarchical prefix may be added by the software. The complete hierarchical name needs to be included in the probe command.

Quit Terminate Session

Syntax quit

Abbreviation qu

Quit terminates the current XChecker session and returns to the system shell. You are asked whether to save the profile information. Current program options are saved to the xchecker.pro file.

Readback Read Back Data Snapshots

Syntax readback [–a] [–c] *#snapshot*

Abbreviation rea

The Readback command reads back data snapshots from your LCA for either verification of the bitstream or verification of logic states. The default is to read the contents of flip-flops only. To view the snapshots, you need to use the Show command after the Readback command. There are two options for readback.

– a Append

The Append options lets you add additional snapshots. If you executed a readback of five snapshots and then used the Append option to specify five more, executing a Show command displays ten shapshots.

– c Read Back Configuration Bitstream

Readback the configuration bitstream. The difference between this option and the verify command is that, in this case, the read back bitstream is not compared to the original configuration data.

Snapshot Show One Set of Readback Data

The *snapshot* variable indicates the maximum number of snapshots to be read back and collected. It can be used with or without the –c option. A snapshot corresponds to one complete set of readback data, and only signals defined in the Probe command are included in the snapshot.

Reset Reset Target LCA/Cable

Syntax reset [–cable]

Abbreviation res

The Reset command resets the internal Logic of the target LCA or resets the XChecker cable. The default is to reset the target LCA. There is a Cable option which initiates a reprogram of the XChecker cable internal LCA. This is useful in the event of power glitches that could affect proper cable operation. Also, when you have had the system off and turn it back on, using the Reset command with the Cable option reinitializes the cable, setting the proper baud rate, etc.

NOTE *You need to specify the location of the RESET pin for XC4000 parts and connect the XChecker cable to that pin. See Figure 5-6 and Table 5-4.*

Save Save Option Settings

Syntax save

Abbreviation sa

The Save command saves four interactive command results in the xchecker.pro file: Baud rate (Baud command), Design name (Load command), Device type (Parttype command) and Port name (Port command).

At initialization, the xchecker.pro file is read to set up the defaults for the current session. This file must be located in the current directory or in the XACT environment search path. The profile is updated at the end of every session. There is no xchecker.pro file until you have exited XChecker at least once.

SettingsDisplay Settings

Syntax settings

Abbreviation se

The Settings command provides a listing of the following information: the port, the baud rate, the type of cable, the design name, the part type and package type, the source of the clock, status of the hardware trigger. The number of clocks for first and subsequent snapshots, the number of signals defined in the Probe list, and the number of signals defined in the Display list are also listed.

Show Display Readback Mode

Syntax show [–signal] *xy* [–snapshot] *nm* [–tabular| –vdata] –text| –hdata]

Abbreviation sh

The Show command displays the data read back in the format specified in the Pick command. The data is displayed in groups of 24 lines. To scroll forward, enter a "y". To exit Show, enter an "n". There are several options available for the Show command.

– signal Select Signals to Display

Selects the number of signals to be displayed, beginning with the "x" signal and ending with the "y" signal. This option is useful for selecting a set of signals for display without changing the display set.

– snapshot Select Snapshots to Display

Defines the number of the first snapshot to show (n) and the total number of snapshots to show (m).

NOTE *One snapshot corresponds to one complete set of readback data from the target LCA.*

– vdata Vertical Display

Signal data is displayed vertically from top to bottom. This is the default mode.

– hdata Horizontal Display

Signal data is displayed horizontally from left to right.

Examples

```
display -bin BUS DATA S0 S1 S2 S3
show -vdata -snap 51 4
```

These commands with the options shown above cause the signal groups BUS and DATA and the signals S0, S1, S2, and S3 to display in binary tabular (vertical) format, starting with snapshot 51 and ending with snapshot 54, as shown in Table 5-9 below.

Table 5-9 Snapshot Examples

Snapshot	BUS	DATA	S0	S1	S2	S3
51	0000001	01001	0	1	0	1
52	1000101	10101	1	1	1	0
53	1010110	11101	0	0	0	1
54	101011	11111	0	0	0	1

Status Show Logic Levels of XChecker Pins

Syntax status

Abbreviation st

The Status command displays the logic level of these XChecker Pins: RST, INIT, D/$\overline{P}$, $\overline{PROG}$, CCLK, TRIG, RT, and RD. The TRIG pin status is registered, so the Status command display shows TRIG High if at any time a trigger has been acknowledged.

Sys Temporarily Exit to Operating System

Syntax sys

Abbreviation none

The Sys command allows you to temporarily exit to the operating system prompt. Enter "exit" to return to XChecker.

Trigger Select Trigger for Readback

Syntax trigger [–auto|–manual|–none] [–clock] *xy* [–timeout] [–reset]

Abbreviation tr

The Trigger command selects the triggering event to be used for the next readback. A Trigger command should always be executed before a Readback command. When a Readback command is executed, XChecker waits for a trigger to initiate a readback. Once a trigger is acknowledged by the XChecker cable, the number of clocks specified by the Clock option is issued and a readback is initiated.

NOTE *With XC4000 devices, the system trigger can be connected directly to the target LCA TRIG input of the readback symbol pin to latch the state of the device instantly instead of waiting for the XChecker software to initiate readback.*

The following triggers and options are described below.

– auto Auto Trigger

Indicates that the trigger is provided by a Low-to-High transition at the XChecker TRIG pin.

– manual Manual Trigger

Indicates that the trigger is provided by pressing the ↵ key on the host system keyboard.

– none Readback Immediately

Initiates a readback immediately after a readback command is issued, without expecting a trigger, or stopping the system clock.

– clock Pause Between Readbacks

Indicates that, after acknowledging a trigger, the XChecker software issues "x" clocks before the first readback is initiated and "y" clocks between each snapshot. This option is functional only if the CLKO pin is connected to the target LCA.

– timeout Cancel If

Indicates that readback is cancelled if a trigger is not received within *t* seconds (*t* must be an integer between 0 and 32767).

– reset Reset LCA

Resets the target LCA before starting a readback by sending a pulse on the RESET line.

Examples

```
trigger -auto -clock 2 5
readback 10
```

The XChecker system waits for a Low-to-High transition at the TRIG pin, stops the system clock (parks CLKO High), and then issues two system clocks on the CLKO pin. Then it initiates a readback by sending a Low-to-High pulse on the RT pin. After the first snapshot is read back, five system clocks are issued, and the whole process repeats nine more times. The free-running system clock is reapplied after readback is finished.

```
trigger -manual
readback 1
```

The XChecker system stops the system clock and initiates a readback immediately after the Enter key is pressed, collecting one snapshot. The system clock is reapplied after the readback is finished.

```
trigger -none
readback 2
```

The XChecker system initiates two consecutive readbacks immediately after the Readback command is issued.

Verify Verify Target LCA Bitstream

Syntax verify *filename*.bit

Abbreviation ve

This command compares the current bitstream in the target LCA to the one in filename BIT to verify correctness. This command differs from the load –v command in that the configuration bitstream is not downloaded, only read back and compared.

Troubleshooting Guide

This section is a simple guide to understanding and addressing the more common issues encountered when configuring LCAs with the XChecker cable. They are likely to fall into the following basic groups.

- Communication — This section describes several issues involving the integrity of the bitstream and the configuration clock transmitted to target LCAs.
- Improper Connections — This section involves assigning configuration pins to invalid signals or voltage levels.
- Improper or Unstable V_{CC} — This section describes some causes of incorrect configuration sequences and incorrect handshaking signals from the LCA.

Communication

Observing the following guidelines should minimize communication difficulties.

Do not attach extension cables to the target LCA side of the XChecker cable; this can compromise configuration data integrity.

Make sure that any noise or ringing in the configuration clock (CCLK) is not above 20% of V_{CC} for a logic zero or below 70% of V_{CC} for a logic one.

Attach the configuration leads firmly to the target board using the headers provided or equivalent leads.

Use the verify feature to assure integrity of the configuration data. This can be done from the command line with the –v option or in the interactive mode by specifying the verify command.

With XC4000 devices, use the status command to monitor the INIT pin. If CRC is enabled, a logic zero at the INIT pin signals the occurrence of a corrupted frame of data. CRC can only be enabled for XC4000 devices during the execution of the MakeBits program.

Use the test fixture with the diag command in the interactive mode to assure continuity of the leads and verify proper operation of the cable.

Improper Connections

Always make sure that XChecker leads are connected properly for the mode of operation desired. (Refer to Table 5-4 and Table 5-5.)

NOTE *Connecting the XChecker leads to erroneous signals of conflicting voltage levels can result in permanent damage to XChecker internal hardware.*

Use the interactive status command to display the logic state of the leads. This helps determine connectivity between the target system and the XChecker cable.

Use the test fixture with the Diagnostic command in the interactive mode to assure continuity of the leads and verify proper operation of the cable.

On workstations, the port you choose to connect the XChecker cable to must be read and writable. XChecker, when used on the workstation, might issue a message stating that the cable is not connected to port ttyx. When you see this message, follow the check list below:

- The board must have the power on, since XChecker uses power from the board.
- Check the device driver using the following command string:

  `ls -l /dev/ttya /dev/ttyb`

The result should be the following :

crw–rw–rw– 1 root 12, 0 *month date time* /dev/ttya

crw–rw–rw– 1 root 12, 1 *month date time* /dev/ttyb

- Reconnect the XChecker cable to another valid port.
- Read the /etc/ttyab file. There should be two lines, as follows:

  ttya "/usr/etc/getty std.9600" unknown off local secure

  ttyb "/usr/etc/getty std.9600" unknown off local secure

If a port is used to connect to a modem or to a remote login, you cannot use that port. It should be on instead of off. Consult your System Adminstrator if you see different information in the /etc/ttyab file.

Improper or Unstable V_{CC}

Never connect the control signals to XChecker before V_{CC} and Ground. The following sequence is recommended:

- Turn off power to the target system
- Connect V_{CC}, Ground, and then the signal leads
- Turn on power to the target system.

IMPORTANT

As with any CMOS device, the input/output pins of the internal LCA should always be at a lower or equal potential than the rail voltage to avoid internal damage.

Make sure V_{CC} rises to a stable level within 10ms. Stable V_{CC} should be between 4.75 V and 5.25 V.

In the event of power glitches, use the interactive Reset command with the Cable option (–c) to reconfigure the XChecker internal LCA and use the interactive load command to reconfigure the target LCA.

Use the test fixture with the Diagnostics command in the interactive mode to assure continuity of the leads and verify proper operation of the cable.

Warning Messages

Warning 001 **Current design does not have ReadCapture option set. Snapshot readback is incorrect.**

To verify the design or probe the internal logic, set the following configuration options in MakeBits (from XDE).

For the XC4000 parts

SyncToDone: **No**
ReadCapture: **Enable**
ReadClk: **CCLK**
DoneActive: **C1 C2 C3 C4**
OutputsActive: **C2 C3 C4**

For the XC3000 and XC2000 parts

Read:**CMD**

Refer to the MakeBits chapter in the *XACT Reference Guide, Volume 2* for details.

Warning 002 **Current design does not have Readback clock set to CCLK. Snapshot readback is incorrect.**

To verify the design or probe the internal logic, set the following configuration options in MakeBits (from XDE).

For the XC4000 parts

SyncToDone: **No**
ReadCapture: **Enable**
ReadClk: **CCLK**
DoneActive: **C1 C2 C3 C4**
OutputsActive: **C2 C3 C4**

For the XC3000 and XC2000 parts

Read:**CMD**

Refer to the MakeBits chapter in the *XACT Reference Guide, Volume 2* for more details.

Warning 003 **Current design does not have DONE set to be active before or at IOB enable. Device may not be configured correctly.**

For XC4000 parts, you need to set the following configuration options in MakeBits (from XDE).

For the XC4000 parts

DoneActive: **C1 C2 C3 C4**

Refer to the MakeBits chapter in the *XACT Reference Guide, Volume 2* for details.

For the XC3000 and XC2000 parts

Read:**CMD**

Refer to the MakeBits chapter in the *XACT Reference Guide, Volume 2* for more details.

Warning 004 **Current design may not have one of the following options set up correctly. Please verify them with XDE.**

For the XC4000 parts

SyncToDone: **No**
ReadCapture: **Enable**
ReadClk: **CCLK**
DoneActive: **C1 C2 C3 C4**
OutputsActive: **C2 C3 C4**

To verify the design or probe the internal logic, you need to set the following configuration options in MakeBits (from XDE).

For the XC3000 and XC2000 parts

Read:**CMD**

Refer to the MakeBits chapter in the *XACT Reference Guide, Volume 2* for more details.

Warning 005 **LL file** *filename.ll* **is not found. Design verification will not work correctly.**

The file *filename*.ll must be found in the current directory or in the directory XACT\DATA. (For workstations, the XACT directory is referred to as the "Xilinx_dir" in the installation notes.) Check the read permissions for your directory and for the file *filename*.ll. Check your XACT environment variable to make sure that it points to the XACT directory.

Error Messages and Recovery Techniques

Error 001 **Command file** *batfile.cmd* **is not found.**

Make sure that the command file you specified is in the current directory or the environment search path. Make sure that the command file has the extension .cmd

Error 002 **Internal Error – Command table syntax error Cmd=valid_command.**

This is an internal program error that normally should not occur. Try entering the command sequence again. If the error persists, try reinstalling your XChecker software. If the error reappears, call Xilinx Technical Support. Be prepared to duplicate the error and reference specific files or examples.

Error 003 **Diagnostic is not supported on this cable.**

The diagnostics command, as well as other readback and verification commands are only supported for the XChecker cable.

Error 004 **Not enough memory.**

XChecker requires a minimum of 512 kB free base RAM. If you are using a PC, make sure no other program or TSR (Terminate and Stay Resident application, i.e., mouse drivers, network drivers, window drivers etc.) is running simultaneously with XChecker. To free memory, type EXIT at the DOS command line.

Error 010 **Cannot open output file** *filename.*

Check available disk space. Current directory or file must have write permission.

Error 011 **Cannot create output file** *filename.*

Check available disk space. Current directory or file must have write permission.

Error 012 **Cannot open input file** *filename.*

Make sure that *filename* exists in you working directory or in the environment search path. Current directory or file must have write permission.

Error 013 **File** *filename* **is not found.**

The file *filename* does not exits in the current directory or search path. Check your environment search path to make sure that it contains the directory where *filename* is.

Error 021 **Help file** *XChecker.hlp* **is not accessible.**

Make sure that the XACT environment variable points to the XACT directory in the PC (the XACT directory in the PC is equivalent to the "Installation Directory" on the workstations). Also make sure that xchecker.hlp is in the directory XACT\MSG. If you cannot find xchecker.hlp in the XACT\MSG directory, you must reinstall the XChecker software.

Error 022 **No help for command** *command entered.*

Help is not available for the specified command. Refer to the "Valid Commands in the Interactive Mode" section in this chapter for help.

Error 023 **Cannot save configuration to** *filename.pro.*

Check available disk space. Current directory or file must have write permission.

Error 024 **Invalid command at line** *line number.*

Check the file xchecker.pro in your current directory for illegal commands. Delete the xchecker.pro file. XChecker will create a new profile when you exit.

Error 030 **Ambiguous command.**

Enter the minimum unique characters that identify the command or enter complete commands with no abbreviations.

Error 031 **Invalid command.**

The command you entered is illegal. Refer to the "Valid Commands in the Interactive Mode" section in this chapter for help.

Error 032 **Invalid number of arguments.**

Refer to the "Valid Commands in the Interactive Mode" section in this chapter for help.

Error 033 **Invalid option** *selected option.*

Refer to the "Valid Options from the Command Line" and "Valid Commands in the Interactive Mode" sections in this chapter for help.

Error 034 **Invalid value given to** *parameter.*

Refer to the "Valid Options from the Command Line" and "Valid Commands in the Interactive Mode" sections in this chapter for help.

Error 035 **Value is required for** *command entered.*

Refer to the "Valid Commands in the Interactive Mode" section in this chapter for help.

Error 050 *System Error Messages*

System error codes are usually a string of messages generated by your operating system.

Error 051 **System file error code.**

System error codes are usually a string of messages generated by your operating system.

Error 101 **Cable is not initialized.**

Try the command "reset –cable" or cycle power to XChecker and then "reset –cable." See the "Improper or Unstable Vcc" section in this chapter.

Error 102 **Cable is not located.**

No cable has been recognized at any port. Make sure there is power to your board and to XChecker. If using the Test Fixture, you must connect Vcc and Ground to it. XChecker draws power from your target system, not from your host computer. Also make sure the RS-232 connector is firmly attached.

For the PC, remember that your serial port needs to be set to the following IRQ lines and I/O addresses:

COM1 IRQ4 at address 03f8

COM2 IRQ3 at address 02f8

Error 103 **Invalid port name.**

Refer to the "XChecker Hardware" section in this chapter for help.

Error 104 **Invalid baud specified.**

Refer to the "XChecker Hardware" section in this chapter for help.

Error 105 **Cable is not reset.**

Try to cycle power to the cable. Try the command "reset –cable."

Error 107 **Communication line is broken.**

Try using the "reset –cable" command. Also make sure there is power to your board and to XChecker. Check all power and port connections.

Error 108 **Communication checksum error.**

Check for noise induced into your target system or from your target system into the XChecker connections. Do not use cable extensions. The XChecker cable length is tested to produce minimal noise levels. Remember that a logic High must be 80-100% of Vcc and a Logic Low must be 0-25% of Vcc.

Error 109 **Cable has no power.**

Make sure there is power from your target system to XChecker. Remember that XChecker draws power from an external source, not from the host computer.

Error 110 **Communication time-out.**

XChecker has not received an expected signal, for example, a system trigger to initiate readback or data coming from readback. Make sure that the selected options for trigger and readback are what you intended. Check all connections. For XC4000 devices, make sure that you used the Readback symbol in your schematics and check the

location of RTRIG and RDATA. For XC3000 and XC2000 devices, check the location of the signals RTRIG and RDATA in your pinouts. Consult your APR/PPR report files for signal locations.

Error 111 **DONE did not go high as expected.**

The design has not been configured properly. Make sure that you compiled your design for the correct part type and package. Make sure that the target LCA has been set-up for Serial Slave mode. Check all connections. Remember that connections to the LCA differ slightly across the different families. Check the bitstream options in MakeBits to assure that a pull-up has been selected for the DONE pin.

Error 112 **Cannot reset DONE signal.**

XChecker cannot set DONE signal to Low. Make sure DONE is pulled-up, not tied High. Check the corresponding package pinouts.

Error 113 **DONE went high earlier than expected.**

Make sure that you compiled your design for the correct part type and package. Check for noise in the CCLK line. Refer to the "Troubleshooting Guide" section.

Error 114 **DONE went high later than expected.**

Make sure that you compiled your design for the correct part type and package. Check for noise in the CCLK line. Refer to the "Troubleshooting Guide" section.

Error 120 **Cannot communicate to the cable.**

Try using the "reset –cable" command. Also make sure there is power to your board and to XChecker. Check all connections. Make sure the RS-232 connector is firmly attached.

Error 121 **Cable datafile** *filename* **is empty.**

Try using the "reset –cable" command. Make sure that the XACT environment variable points to the XACT directory on the PC. (On workstations, the XACT directory is equivalent to the "Xilinx_Directory" referenced in the installation notes).

Error 122 **Cable datafile** *filename* **has invalid format.**

Make sure that the XACT environment variable points to the XACT directory on the PC (On the workstations the XACT directory is equivalent to the "Xilinx_Directory" referenced in the installation notes).

Error 122 **Can't open cable datafile** *filename.*

Make sure that the XACT environment variable points to the XACT directory on the PC (On the workstations the XACT directory is equivalent to the "Xilinx_Directory" referenced in the installation notes).

Error 123 **No XChecker cable is connected to the port** *portname.*

Make sure there is power to your board and to XChecker. If using the Test Fixture, you must connect Vcc and Ground to it. Remember that XChecker draws power from your target system, not from your host computer. Also make sure the RS-232 connector is firmly attached.

For the PC, remember that your serial port (for use with XChecker and serial cables) needs to be set to the following IRQ lines and I/O addresses:

COM1 IRQ4 at address 03f8

COM2 IRQ3 at address 02f8

Error 124 **No XChecker cable is connected to the system.**

Make sure there is power to your board and to XChecker. If using the supplied test fixture, you must connect Vcc and Ground to it. Remember that XChecker draws power from your target system, not from your host computer. Also make sure the RS-232 connector is firmly attached.

For the PC, remember that your serial port (for use with XChecker and serial cables) needs to be set to the following IRQ lines and I/O addresses:

COM1 IRQ4 at address 03f8

COM2 IRQ3 at address 02f8

Error 125 **Fail reading cable status.**

Try using the "reset –cable" command. Also make sure there is power to your board and to XChecker. Check all connections.

Error 126 **Unsupported command for this cable.**

See the "Valid Commands in the Interactive Mode" for specific commands valid with the XChecker cable. If you are using the previous parallel or serial download cables, you can only use the Load command to download.

Error 128 **Read only** *number of bits received.*

Check all connections. Check for noise induced into your target system or from your target system into the XChecker connections. Do not use cable extensions. The XChecker cable length is tested to produce minimal noise levels. Remember that a logic High must be 80-100% of Vcc and a Logic Low must be 0-25% of Vcc.

Error 130 **Invalid baud rate. Current baud rate is** *baud rate.*

See Table 5-1 for valid baud rates for your computer.

Error 131 **Missing baud rate. Current baud rate is** *baud rate.*

See the "Valid Commands in the Interactive Mode" section in this manual for correct command usage.

Error 132 **Block** *number*: **Communication Error.**

Check for noisy connections. Check power stability. You may want to try the "reset –c" command to reinitialize the cable and try to read back again.

Error 133 **Block %d: Communication Timeout.**

XChecker has not received an expected block of data. Check all connections. Check for power stability and for contention with the control signals that can cause a Readback Abort.

Error 134 **Cannot communicate with port** *port name.*

Check this manual for supported ports. See the Port command. When using the XChecker cable with a PC the serial port needs to be set to the following IRQ lines and I/O addresses:

COM1 IRQ4 at address 03f8

COM2 IRQ3 at address 02f8

Error 135 **Invalid port name** *port name.*

Check this manual for supported ports. See the Port command.

Error 140 **Datafile** *filename* **has invalid format.**

XChecker supports the following formats: Xilinx .bit, Xilinx .rbt, Intel .mcs, Tektronix .tek and Motorola .exo.

Error 141 **Datafile** *filename* **is empty.**

The specified datafile is either empty or contains invalid data. XChecker supports the following formats: Xilinx .bit, Intel .mcs, Tektronix .tek and Motorola .exo.

Error 142 **Datafile** *filename* **is not found.**

The file *filename* does not exits in the current directory or search path. Check your environment search path to make sure that it contains the directory where *filename* is.

Error 143 **Can't open datafile** *filename.*

The file *filename* does not exits in the current directory or search path. Check your environment search path to make sure that it contains the directory where *filename* is.

Error 150 **Only** *number* **bytes read.**

Check all connections. Do not use cable extensions. The XChecker cable length is tested to produce minimal noise levels.

Error 180 **Cannot create export file** *filename.*

Check available disk space. Current directory or file must have write permission.

Error 181 **Cannot open export file** *filename.*

Check available disk space and file accessibility such as, write privileges to the current directory).

Error 182 **Missing design name to export.**

You must specify a name to create an export file. Check for correct command usage.

Error 183 **Missing design name to import.**

You must specify a name to open an import file. Check for correct command usage.

Error 190 **Missing .ll filename.**

You must specify a file name for the .ll file.

Error 191 **.ll file** *filename.ll* **is not found.**

The file *filename.ll* must be found in the current directory or in the directory XACT\DATA (for workstations the XACT directory is referred to as the "Xilinx_dir" in the installation notes). Check read permissions to your directory and to the file *filename*.ll. Check your XACT environment variable to make sure that it points to the XACT directory.

Error 192 **Cannot open .ll file** *filename.ll.*

The file *filename*.ll must be found in the current directory or in the directory XACT\DATA (for workstations the XACT directory is referred to as the "Xilinx_dir" in the installation notes). Check read permissions to your directory and to the file *filename*.ll. Check your XACT environment variable to make sure that it points to the XACT directory.

Error 202 **Invalid number of clocks.**

The valid number of clocks range from 1 to 32767. See the clock command in this manual for correct usage.

Error 210 **Not enough memory for loading design datafile.**

XChecker requires a minimum of 512 kB free base RAM. If you are using a PC, make sure no other program or TSR (Terminate and Stay Resident application such as, mouse drivers, network drivers, window drivers) is running simultaneous with XChecker. To free memory, type "exit" at the DOS command line.

Error 211 **Not enough memory for bitmap result.**

XChecker requires a minimum of 512 kB free base RAM. If you are using a PC, make sure no other program or TSR (Terminate and Stay Resident application such as, mouse drivers, network drivers, window drivers) is running simultaneous with XChecker. To free memory, type "exit" at the DOS command line.

Error 220 **No part type is defined.**

The part type specified in your design or by you (using the Part command) is invalid. Check *The Programmable Logic Data Book* for valid part types and packages.

Error 240 **Data is corrupted. Bad checksum in file** *PROM file*, **line** *linenumber*.

The input (hexadecimal) file is corrupted. Try regenerating the input file using MakePROM. Remember that supported hexadecimal formats are: Intel .mcs, Motorola .exo, and Tektronix .tek.

Error 241 **Invalid data character in file** *PROM file*, **line** *linenumber.*

The input (hexadecimal) file is corrupted. Try regenerating the input file using MakePROM. Remember that supported hexadecimal formats are: Intel .mcs, Motorola .exo, and Tektronix .tek.

Error 242 **Invalid datafile format in file** *PROM file*, **line** *linenumber.*

XChecker supports the following formats: Xilinx .bit, Intel .mcs, Tektronix .tek and Motorola .exo.

Error 243 **Internal Error** *errorname* **in file** *filename*, **line** *linenumber.*

An illegal character has been detected in *filename*. This could be caused by an illegally formatted PROM file (supported PROM formats are .tek, .exo, and .mcs) or simply a corrupted data or command file.

Error 250 **Readback configuration contains all** *0/1.*

Check all connections. Particularly, check the connections to the XChecker RT and RD pins. See Table 5-4 for pin functions and proper connections. Check MakeBits options: in XC3000 and XC2000 devices make sure that you have selected readback upon command from MakeBits (Read option must be Cmd); in XC4000 devices make sure that ReadCapture is Enabled. Check power to the target LCA.

Error 251 **No data in readback buffer.**

You need to execute a readback command before you can show any data.

Error 260 **Signal name** *name* **is not defined.**

Execute a Probe command (without arguments) to display the available signal names. You can also check the design.ll file for this information. Make sure to include the complete net names, with their hierarchical prefixes. Remember that valid signal names are: user-defined net names connected to flip-flops latches and memory outputs; user-defined group names and IOB names.

Error 262 **Signal name** *name* **is not user defined.**

Execute a Probe command (without arguments) to display the available signal names. You can also check the design.ll file for this information. Make sure to include the complete net names, with their hierarchical prefixes. Remember that valid signal names are: user-defined net names connected to flip-flops latches and memory outputs, user-defined group names, and IOB names.

Error 263 **DONE signal went High early.**

Make sure that you compiled your design for the correct part type and package. Check for noise in the CCLK line. Refer to the "Trouble Shooting Guide" section.

Error 264 **DONE signal did not go High.**

The design has not been configured properly. Make sure that you compiled your design for the correct part type and package. Make sure that the target LCA has been set up for Serial Slave mode. Check all connections. Remember that connections to the LCA differ slightly across the different families. Check the bitstream options in MakeBits to assure that a pull-up has been selected for the DONE pin.

Error 265 **INIT signal is low.**

The design has not been configured properly. Make sure that you compiled your design for the correct part type and package. Make sure that the target LCA has been set-up for Serial Slave mode. Check all connections. Remember that connections to the LCA differ slightly across the different families. Check for noisy signals that may corrupt the data or CCLK going to the target LCA.

Error 266 **Probe list is empty.**

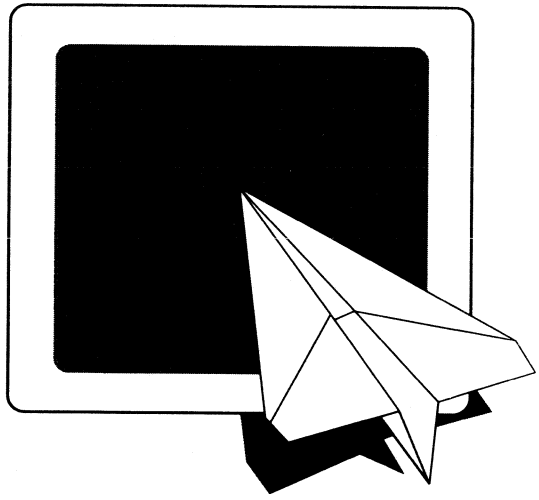
If you have executed a Probe -clear command, you need to execute a Probe command to add signals to the probe list. Check the file design.ll to make sure that it contains your design's net names.

Error 267 **Display list is empty.**

You need to execute a Pick command before you can show the readback data. Check the Pick and Show commands in this manual.

Error 268 **Group list is empty.**

See the "Valid Commands in the Interactive Mode" section in this chapter for proper use of the Group command.



XACT Hardware and Peripherals Guide

Index

XACT Hardware and Peripherals Guide — April, 1994 (0401132 01) *Printed in U.S.A.*

Index

Z