

Programming Xilinx XC9500 CPLDs on GENRAD Testers

Preface

Table of Contents

Introduction

Creating SVF Files

Creating Compiled Test Files



The Xilinx logo shown above is a registered trademark of Xilinx, Inc.

XILINX, XACT, XC2064, XC3090, XC4005, XC5210, XC-DS501, FPGA Architect, FPGA Foundry, NeoCAD, NeoCAD EPIC, NeoCAD PRISM, NeoROUTE, Plus Logic, Plustran, P+, Timing Wizard, and TRACE are registered trademarks of Xilinx, Inc.



The shadow X shown above is a trademark of Xilinx, Inc.

All XC-prefix product designations, XACTstep, XACTstep Advanced, XACTstep Foundry, XACT-Floorplanner, XACT-Performance, XAPP, XAM, X-BLOX, X-BLOX plus, XChecker, XDM, XDS, XEPLD, XPP, XSI, BITA, Configurable Logic Cell, CLC, Dual Block, FastCLK, FastCONNECT, FastFLASH, FastMap, HardWire, LCA, LogiBLOX, Logic Cell, LogiCORE, LogicProfessor, MicroVia, PLUSASM, PowerGuide, PowerMaze, Select-RAM, SMARTswitch, TrueMap, UIM, VectorMaze, VersaBlock, VersaRing, XABEL, Xilinx Foundation Series, and ZERO+ are trademarks of Xilinx, Inc. The Programmable Logic Company and The Programmable Gate Array Company are service marks of Xilinx, Inc.

IBM is a registered trademark and PC/AT, PC/XT, PS/2 and Micro Channel are trademarks of International Business Machines Corporation. DASH, Data I/O and FutureNet are registered trademarks and ABEL, ABEL-HDL and ABEL-PLA are trademarks of Data I/O Corporation. SimuCad and Silos are registered trademarks and P-Silos and P/C-Silos are trademarks of SimuCad Corporation. Microsoft is a registered trademark and MS-DOS is a trademark of Microsoft Corporation. Centronics is a registered trademark of Centronics Data Computer Corporation. PALASM is a registered trademark of Advanced Micro Devices, Inc. UNIX is a trademark of AT&T Technologies, Inc. CUPL, PROLINK, and MAKEPRG are trademarks of Logical Devices, Inc. Apollo and AEGIS are registered trademarks of Hewlett-Packard Corporation. Mentor and IDEA are registered trademarks and NETED, Design Architect, System Architect, QuickSim, QuickSim II, and EXPAND are trademarks of Mentor Graphics, Inc. Sun is a registered trademark of Sun Microsystems, Inc. SCHEMA II+ and SCHEMA III are trademarks of Omaton Corporation. OrCAD is a registered trademark of OrCAD Systems Corporation. Viewlogic, Viewsim, and Viewdraw are registered trademarks of Viewlogic Systems, Inc. CASE Technology is a trademark of CASE Technology, a division of the Teradyne Electronic Design Automation Group. DECstation is a trademark of Digital Equipment Corporation. Synopsys is a registered trademark of Synopsys, Inc. Verilog is a registered trademark of Cadence Design Systems, Inc. FLEXlm is a trademark of Globetrotter, Inc. DynaText is a registered trademark of Inso Corporation.

Xilinx, Inc. does not assume any liability arising out of the application or use of any product described or shown herein; nor does it convey any license under its patents, copyrights, or maskwork rights or any rights of others. Xilinx, Inc. reserves the right to make changes, at any time, in order to improve reliability, function or design and to supply the best product possible. Xilinx, Inc. will not assume responsibility for the use of any circuitry described herein other than circuitry entirely embodied in its products. Xilinx, Inc. devices and products are protected under one or more of the following U.S. Patents: 4,642,487; 4,695,740; 4,706,216; 4,713,557; 4,746,822; 4,750,155; 4,758,985; 4,820,937; 4,821,233; 4,835,418; 4,853,626; 4,855,619; 4,855,669; 4,902,910; 4,940,909; 4,967,107; 5,012,135; 5,023,606; 5,028,821; 5,047,710; 5,068,603; 5,140,193; 5,148,390; 5,155,432; 5,166,858; 5,224,056; 5,243,238; 5,245,277; 5,267,187; 5,291,079; 5,295,090; 5,302,866; 5,319,252; 5,319,254; 5,321,704; 5,329,174; 5,329,181; 5,331,220; 5,331,226; 5,332,929; 5,337,255; 5,343,406; 5,349,248; 5,349,249; 5,349,250; 5,349,691; 5,357,153; 5,360,747; 5,361,229; 5,362,999; 5,365,125; 5,367,207; 5,386,154; 5,394,104; 5,399,924; 5,399,925; 5,410,189; 5,410,194; 5,414,377; 5,422,833; 5,426,378; 5,426,379; 5,430,687; 5,432,719; 5,448,181; 5,448,493;

5,450,021; 5,450,022; 5,453,706; 5,466,117; 5,469,003; 5,475,253; 5,477,414; 5,481,206; 5,483,478; 5,486,707; 5,486,776; 5,488,316; 5,489,858; 5,489,866; 5,491,353; 5,495,196; 5,498,979; 5,498,989; 5,499,192; 5,500,608; 5,500,609; 5,502,000; 5,502,440; RE 34,363, RE 34,444, and RE 34,808. Other U.S. and foreign patents pending. Xilinx, Inc. does not represent that devices shown or products described herein are free from patent infringement or from any other third party right. Xilinx, Inc. assumes no obligation to correct any errors contained herein or to advise any user of this text of any correction if such be made. Xilinx, Inc. will not assume any liability for the accuracy or correctness of any engineering or software support or assistance provided to a user.

Xilinx products are not intended for use in life support appliances, devices, or systems. Use of a Xilinx product in such applications without the written consent of the appropriate Xilinx officer is prohibited.

Copyright 1991-1997 Xilinx, Inc. All Rights Reserved.

April 10, 1997

Preface

About This Manual

This manual describes how to program Xilinx XC9500 CPLDs on GenRad testers.

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.

Manual Contents

This manual covers the following topics.

- Chapter 1, "Introduction," lays out the basic procedure for programming an XC9500 CPLD in a GenRad test environment.
- Chapter 2, "Creating SVF Files," discusses how to create an SVF files on PCs, and on Sun and HP workstations.
- Chapter 3, "Creating Compiled Test Files," discusses how to use **svf2dts** to create compiled test files for use in the GenRad test environment.

Conventions

In this manual the following conventions are used for syntax clarification and command line entries.

- *Courier font* indicates messages, prompts, and program files that the system displays, as shown in the following example.

```
speed grade: -100
```

- **Courier bold** indicates literal commands that you must enter in a syntax statement.

```
rpt_del_net=
```

- *Italic font* indicates variables in a syntax statement. See also, other conventions used on the following page.

```
xdelay design
```

- Square brackets “[]” indicate an optional entry or parameter. However, in bus specifications, such as bus [7:0], they are required.

```
xdelay [option] design
```

- Braces “{ }” enclose a list of items from which you choose one or more.

```
xnfprep designname ignore_rlocs={true|false}
```

- A vertical bar “|” separates items in a list of choices.

```
symbol editor [bus|pins]
```

Other conventions used in this manual include the following.

- *Italic font* indicates references to manuals, as shown in the following example.

See the *Development System Reference Guide* for more information.

- *Italic font* indicates emphasis in body text.

If a wire is drawn so that it overlaps the pin of a symbol, the two nets *are not* connected.

- A vertical ellipsis indicates repetitive material that has been omitted.

```
IOB #1: Name = QOUT'  
IOB #2: Name = CLKIN'  
.  
.  
.
```

- A horizontal ellipsis “...” indicates that the preceding can be repeated one or more times.

```
allow block blockname loc1 loc2 ... locn ;
```

Contents

Programming Xilinx XC9500 CPLDs on GENRAD Testers

	About This Manual	v
	Manual Contents	v
Chapter 1	Introduction	
Chapter 2	Creating SVF Files	
	Creating an SVF File Using EZTag	2-1
	On a Workstation	2-1
	On a PC	2-3
	Software Restrictions	2-5
Chapter 3	Creating GenRad Test Files	
	Using xsvf2dts	3-1
	Procedure	3-1
	TRST Optional Pin	3-2

Chapter 1

Introduction

This document describes the procedures necessary to program Xilinx XC9500 CPLD designs in a GenRad test environment. The procedures described in this document lay out the necessary steps you need to perform; they are:

- Creating an SVF File Using EZTag
- Generating a GenRad ISP Program

EZTag will translate JEDEC files into Serial Vector Format (SVF) files. The **svf2dts** program translates SVF files to GenRad digital test source. This allows you to take SVF files created in EZTag and translate them for use in a GenRad test environment.

The **svf2dts** program runs under DOS window. EZTag runs under Windows 95 or Windows NT on a PC and in a SUN workstation environment.

Note: The installation procedure for **svf2dts** is found in the README file on the disk. This is an ASCII text file and can be viewed from the DOS editor (`edit readme.txt`) or the Windows Notepad.

Chapter 2 describes the procedure for creating an SVF file from EZTag from both PC and workstation environments. Chapter 3 describes how to produce the **.dts** file you will need for GenRad tester programming.



Figure 1-1 Program Flow

Hardware Considerations

This software and methodology applies to the following Genrad testers running GenRad software release 3.2 or greater.

- GR2287L
- GR2286i
- GR2287i
- GR2283i
- GR2284i
- GR2281i
- GR2280i

Chapter 2

Creating SVF Files

Creating an SVF File Using EZTag

This procedure describes how to create an SVF file; it assumes that the Xilinx XACT-CPLD version 6.0.1 (or newer) or XABEL-CPLD version 6.1.1 (or newer) is being used, which includes the XC9500 fitter and the EZTag software.

Note: XABEL-CPLD runs on PCs only.

On a Workstation

If you have a **Sun or HP workstation**, from the XACT command line use the following procedure:

1. Fit the design and create a JEDEC output file.
2. Invoke the EZTag software

```
eztag -svf
```

The following message appears:

```
Xilinx (R) EZTAG XC9500-CPLD-6.0.3-JTAG Boundary-Scan  
Download
```

```
Copyright (C) Xilinx Inc. 1991-1996. All Rights  
Reserved.
```

```
-----  
SVF GENERATION MODE.  
EZTAG ?
```

3. Set up the device types and assign design names.

On a **Sun or HP workstation**, type the following command at the EZTAG ? prompt:

```
part deviceType1:designName1
    deviceType2:designName2
    deviceTypeN:designNameN <CR>
```

where **deviceType** is the name of the BSDL file for that device and **designName** is the name of the design to translate into SVF. Multiple *deviceType:designName* pairs are separated by spaces. For example:

```
part xc95108:abc12 xc95216:wxyz
```

This command defines the JTAG device chain, from one to any number of devices. The parts specified in the **part** command should be arranged in order beginning with the first device to receive TDI and ending with the last device to output TDO.

Note: For any non-XC9500 part in the JTAG chain make certain that the BSDL file for the specified part is available along the XACT path and is called *deviceType.bsd*. The design name in this case can be any arbitrary name.

4. Execute an EZTag operation.
 - **erase designName** — generates an SVF file for the bit sequence needed to erase the specified part.
 - **verify designName [-j jedecFileName]** — generates an SVF file that specifies the bit sequence to read back the device contents and compare it against the contents of the specified JEDEC file.
 - **program [-b] designName [-j jedecFileName]** — generates an SVF file that specifies the bit sequence to program the specified part from a JEDEC file named *designName.jed* (or alternately, the JEDEC file name specified after the “-j”). The program command option adds the following functionality:
 - b — Used to specify programming of a blank part. This is typically specified for parts delivered blank from the factory.
5. Exit EZTag

On a **Sun or HP workstation** you will exit EZTag by entering the following command:

```
quit
```

Note: The SVF file will be named *designName.svf*, and will be created in the current working directory (the directory in which EZTAG is being run). Consecutive operations on the same *designName* file will overwrite the SVF file each time. The SVF file contains all data and commands necessary to perform the specified function.

On a PC

If you are using a PC with Windows 3.1 or 95, use the following procedure.

1. Fit the design and create a JEDEC output file.
2. Double-click on the EZTag icon. The EZTag-JTAG Download Software will appear.

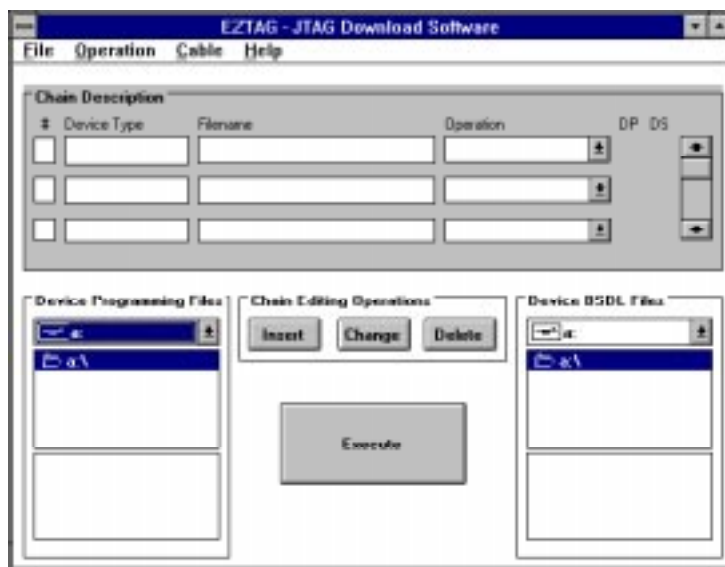


Figure 2-1 EZTag - JTAG Download Software

3. Assign device types for each device in the chain. For each device:
 - a) Under **Device Programming Files** find the disk and directory containing the files you want to use.
 - b) Double-click on the file name. The **Device Type** and **File name** should appear in the **Chain Description**.

- c) Click once on the **Operation** down arrow to select the type of operation you want to select. Select **Program** (this is the default).
- 4. Specify the BSDL files for these parts as follows:
 - a) Under **Device BSDL Files**, find the disk and directory containing the files you want to use.
 - b) Double-click on the *filename* of the device. Type "OTHER." The selected *filename* should appear in the chain description.
 - c) The operation will be set to BYPASS and cannot be changed.

Note: For any non-XC9500 part in the JTAG chain make certain that the BSDL file for the specified part is available along the XACT path and is called *deviceType.bsd*. The design name in this case can be any arbitrary name.

- 5. Highlight one of the devices by clicking on it once with the mouse, then select one of the following operations:

Operations > Erase

generates an SVF file for the bit sequence needed to erase the specified part.

Operations > Verify

generates an SVF file that specifies the bit sequence to read back the device contents and compare it against the contents of the specified JEDEC file.

Operations > Program

generates an SVF file that specifies the bit sequence to program the specified part from a JEDEC file named *designName.jed*.

- 6. On a **PC** you will exit with:

File > Exit

Note: The SVF file will be named *designName.svf*, and will be created in the current working directory (the directory in which EZTAG is being run). Consecutive operations on the same *designName* file will overwrite the SVF file each time. The SVF file contains all data and commands necessary to perform the specified function.

Software Restrictions

EZTAG can generate SVF files only for devices for which XC9500 JEDEC files can be created.

The current software can only generate SVF files for operations on one part at a time. If there are several parts to be programmed, additional program commands must be executed — one for each part, creating multiple SVF files. In each SVF file, one device will be programmed while the others are held in bypass mode.

Chapter 3

Creating GenRad Test Files

Using svf2dts

Use the SVF file that you generated above as input to the **svf2dts** Conversion Utility. This tool takes SVF files and creates executable digital test source (DTS) files.

The **svf2dts** Conversion Utility runs in the DOS environment. It will also run under Windows. The program uses no graphics and only 128 kilobytes of memory, but requires at least 24 megabytes of hard disk for output files.

Procedure

1. The program will create the **.dts** and **.rpt** files in the same directory as the **svf2dts.exe** executable. Place the **.svf** file for the part you want to test in the same directory.
2. To run the program, execute from the same directory:

```
svf2dts filename.svf
```

The program will ask for the following information:

- a) A base name to use on output files.

This feature allows you to use a different name for the **.dts** and **.rpt** files than you had on the **.svf** files. This allows easier integration into the ATG process.

- b) The reference designator of the device to be tested.

This assists in properly identifying a failed device.

- c) The total number of pins for the device, and the pin numbers associated with required signals. The required pin numbers are for TMS, TCK, TDI, TDO, and TRST (if applicable).

If you used an alpha-numeric for the signal names above, you must edit the DTS file to add or change all other pins on the device to be alpha-numeric. This will maintain consistency across library modules.

- d) The fail bit to be used if the test fails.

The default is 191. If there are more than one device being programmed, use different fail bits for each device to allow routines to report correctly which device is failing.

Fail bits 193 and 194 are reserved and should not be used. They are used in status verification and will be cleared.

After you have entered the fail bit the program will start execution. Run time for this program will vary from 4 to 15 minutes depending on the size of the **.svf** file and the speed of your computer.

TRST Optional Pin

The signal TRST is optional. If it is not available and a TRST record is found, the record will be ignored and a warning will be put into the **.rpt** file.