Appendix B

BitGen and PROMGen Switches and Options

Using BitGen

BitGen produces a bitstream for Xilinx device configuration. After the design has been completely routed, it is necessary to configure the device so that it can execute the desired function. The Xilinx bitstream necessary to configure the device is generated with BitGen. BitGen takes a fully routed NCD (Circuit Description) file as its input and produces a configuration bitstream—a binary file with a .bit extension.

The BIT file contains all of the configuration information from the NCD file defining the internal logic and interconnections of the FPGA, plus device-specific information from other files associated with the target device. The binary data in the BIT file can then be downloaded into the FPGA memory cells, or it can be used to create a PROM file (see Figure B-1).

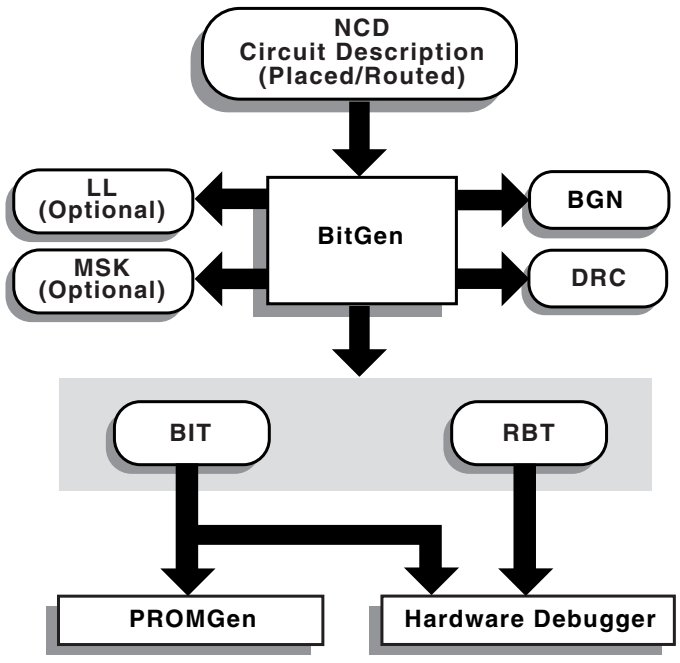

Figure B-1: **BitGen**

BitGen Syntax

The following syntax creates a bitstream from your NCD file.

```
bitgen [options] infile[.ncd] [outfile] [pcf_file]
```

options is one or more of the options listed in the "BitGen Options" on page 487.

Infile is the name of the NCD design for which you want to create the bitstream. You can specify only one design file, and it must be the first file specified on the command line.

You do not have to use an extension. If you do not, **.ncd** is assumed. If you do use an extension, it must be **.ncd**.

Outfile is the name of the output file. If you do not specify an output file name, BitGen creates one in the same directory as the input file. If you specify -l on the command line, the extension is .ll (see -l command line option). If you specify -m (see -m command line option), the extension is .msk. If you specify -b, the extension is .rbt. Otherwise the extension is .bit. If you do not specify an extension, BitGen appends one according to the aforementioned rules. If you do include an extension, it must also conform to the rules.

Pcf_file is the name of a physical constraints (PCF) file. BitGen uses this file to determine which nets in the design are critical for tiedown, which is not available for Virtex families. BitGen automatically reads the .pcf file by default. If the physical constraints file is the second file specified on the command line, it must have a .pcf extension. If it is the third file specified, the extension is optional; .pcf is assumed. If a .pcf file name is specified, it must exist, otherwise the input design name with a .pcf extension is read if that file exists.

A report file containing all BitGen's output is automatically created under the same directory as the output file. The report file has the same root name as the output file with a .bgn extension.

BitGen Files

This section describes input files that BitGen requires and output files that BitGen generates.

Input Files

Input to BitGen consists of the following files.

- NCD file—a physical description of the design mapped, placed and routed in the target device. The NCD file must be fully routed.
- PCF—an optional user-modifiable ASCII Physical Constraints File. If you specify a PCF file on the BitGen command line, BitGen uses this file to determine which nets in the design are critical for tiedown (not used for Virtex families).

Output Files

Output from BitGen consists of the following files.

- BIT file—a binary file with a .bit extension. The BIT file contains all of the configuration information from the NCD file defining the internal logic and interconnections of the FPGA, plus device-specific information from other files associated with the target device. The binary data in the BIT file can then be downloaded into the FPGA memory cells, or it can be used to create a PROM file (see "Using PROMGen" on page 491).
- RBT file—an optional "rawbits" file with an .rbt extension. The rawbits file is ASCII ones and zeros representing the data in the bitstream file. If you enter a -b option on the BitGen command line, an RBT file is produced in addition to the binary BIT file (see "-b (Create Rawbits File)" on page 487).
- LL file—an optional ASCII logic allocation file with a .ll extension. The logic allocation file indicates the bitstream position of latches, flip-flops, and IOB inputs and outputs. A .ll file is produced if you enter a -l option on the BitGen command line (see "-l (Create a Logic Allocation File)" on page 491).

- MSK file—an optional mask file with an .msk extension. This file is used to compare relevant bit locations for executing a readback of configuration data contained in an operating FPGA. A MSK file is produced if you enter a -m option on the BitGen command line (see "-m (Generate a Mask File)" on page 491).
- BGN file—a report file containing information about the BitGen run.
- DRC file—a Design Rule Check (DRC) file for the design. A DRC runs and the DRC file is produced unless you enter a -d option on the BitGen command line (see "-d (Do Not Run DRC)" on page 487).

BitGen Options

Following is a description of command line options and how they affect BitGen behavior.

-b (Create Rawbits File)

Create a "rawbits" (*file_name*.rbt) file. The rawbits file consists of ASCII ones and zeros representing the data in the bitstream file.

If you are using a microprocessor to configure a single FPGA, you can include the rawbits file in the source code as a text file to represent the configuration data. The sequence of characters in the rawbits file is the same as the sequence of bits written into the FPGA.

-d (Do Not Run DRC)

Do not run DRC (Design Rule Check). Without the -d option, BitGen runs a DRC and saves the DRC results in two output files: the BitGen report file (*file_name*.bgn) and the DRC file (*file_name*.drc). If you enter the -d option, no DRC information appears in the report file and no DRC file is produced.

Running DRC before a bitstream is produced detects any errors that could cause the FPGA to malfunction. If DRC does not detect any errors, BitGen produces a bitstream file (unless you use the -j option described in the "-j (No BIT File)" on page 491).

-f (Execute Commands File)

-f *command_file*

The -f option executes the command line arguments in the specified *command_file*.

-g (Set Configuration)

-g *option*:*setting*

The -g option specifies the startup timing and other bitstream options for Xilinx FPGAs. The settings for the -g option depend on the design's architecture. These options have the following syntax.

Compress

Enable bitstream compression using multiple frame writes (MFW).

Readback

This allows the user to perform Readback by the creating the necessary bitstream (**.rbb** file).

CRC

Virtex-II allows the user to enable or disable the CRC checking. If CRC checking is disabled, a CBC (Constant Bit Check) is used instead.

Settings: Enable, Disable

Default: Enable

DebugBitstream

This option creates a modified bitstream which loads each frame individually, and places an LOUT write after each, for debugging purposes. This option should be used only in Master or Slave Serial downloads.

Settings: Yes, No

Default: No

ConfigRate

Virtex-II devices use an internal oscillator to generate CCLK when configuring in Master SelectMAP or Master Serial modes. This option sets the CCLK rate in MHz.

Settings: 4,5,6,7,8,10,13,15,20,26,30,34,41,45,51,55,60,130

Default: 4

StartupClk

The last few cycles of configuration is called the startup sequence. The startup sequence can be clocked by CCLK signal, a User clock (connected to the STARTUP block), or TCK (the JTAG clock).

Settings: CCLK, UserClk, JTAGClk

Default: CCLK

PowerdownStatus

This options allows the user to choose whether the DONE pin is used as the PowerDown pin after configuration.

Settings: Enable, Disable

Default: Enable

DCMShutdown

If the DCMShutdown option is enabled, the DCM resets if the SHUTDOWN and AGHIGH commands are performed.

Settings: Enable, Disable

Default: Enable

CclkPin

This option selects an internal pullup on the CCLK pin.

Settings: Pullnone, Pullup

Default: Pullup

DonePin

This option selects an internal pullup on the DONE pin.

Settings: Pullnone, Pullup

Default: Pullup

M0Pin

This option selects an internal pullup or pulldown on the M0 (Mode 0) pin.

Settings: Pullnone, Pullup, Pulldown

Default: Pullup

M1Pin

This option selects an internal pullup or pulldown on the M1 (Mode 1) pin.

Settings: Pullnone, Pullup, Pulldown

Default: Pullup

M2Pin

This option selects an internal pullup or pulldown on the M2 (Mode 2) pin.

Settings: Pullnone, Pullup, Pulldown

Default: Pullup

ProgPin

This options selects an internal pullup on the PROGRAM pin.

Settings: Pullnone, Pullup

Default: Pullup

TckPin

This option selects an internal pullup or pulldown on the TCK (JTAG Clock) pin.

Settings: Pullnone, Pullup, Pulldown

Default: Pullup

TdiPin

This option selects an internal pullup or pulldown on the TDI (JTAG Input) pin.

Settings: Pullnone, Pullup, Pulldown

Default: Pullup

TdoPin

This option selects an internal pullup or pulldown on the TDO (JTAG Output) pin.

Settings: Pullnone, Pullup, Pulldown

Default: Pullnone

TmsPin

This option selects an internal pullup or pulldown on the TMS (JTAG Mode Select) pin.

Settings: Pullnone, Pullup, Pulldown

Default: Pullup

UnusedPin

This option selects an internal pullup or pulldown on all unused I/Os.

Settings: Pullnone, Pullup, Pulldown

Default: Pulldown

GWE_cycle

Selects the startup phase that asserts the internal write enable to flip-flops, LUT RAMs, shift registers, and BRAMs. Before the startup phase both BRAM writing and reading are disabled. The Done setting asserts GWE when the DoneIn signal is high. DoneIn is either the value of the DONE pin or a delayed version if DonePipe=Yes. The Keep setting is used to keep the current value of the GWE signal.

Settings: 1, 2, 3, 4, 5, 6, Done, Keep

Default: 6

GTS_cycle

Selects the startup phase that releases the internal 3-state control to the I/O buffers. The Done setting releases GTSA when the DoneIn signal is high. DoneIn is either the value of the DONE pin or a delayed version if DonePipe=Yes. The Keep setting is used to keep the current value of the GTS signal.

Settings: 1, 2, 3, 4, 5, 6, Done, Keep

Default: 5

LCK_cycle

Selects the startup phase to wait until DCM locks are asserted.

Settings: 0, 1, 2, 3, 4, 5, 6, NoWait

Default: NoWait

MATCH_cycle

Selects the startup phase to wait until DCI locks are asserted.

Settings: 0, 1, 2, 3, 4, 5, 6, NoWait

Default: NoWait

DONE_cycle

Selects the startup phase that activates the FPGA DONE signal. DONE is delayed when DonePipe=Yes.

Settings: 1, 2, 3, 4, 5, 6

Default: 4

Persist

This option is needed for Readback and Partial Reconfiguration using the configuration pins. If Persist=Yes, all the configuration pins used retain their function. Which configuration pins are persisted is determined by the mode pin settings. If a serial mode is chosen, the persisted pins would be $\overline{\text{INIT}}$, DOUT, and DIN. If a SelectMAP mode is chosen, the persisted pins would be $\overline{\text{INIT}}$, BUSY, D0-D7, $\overline{\text{CS}}$, and $\overline{\text{WRITE}}$.

Settings: Yes, No

Default: No

DriveDone

This option actively drives the DONE pin high as opposed to an open-drain driver. Take care when setting DriveDone=Yes in daisy chain applications.

Settings: Yes, No

Default: No

DonePipe

This option is intended for use with FPGAs being set up in a high-speed daisy chain configuration. When set to Yes, the FPGA waits on the DONE pin, and waits for the first StartupClk edge before moving to the Done state.

Settings: Yes, No

Default: No

Security

This options selects the level of bitstream security. Selecting Level 1 disables Readback, and selecting Level 2 disables Readback and reconfiguration.

Settings: Level1, Level2, None

Default: None

UserID

The user can enter up to an 8-digit hexadecimal code (32-bit value) in the UserID register. You can use the register to identify implementation or design revisions.

Settings: <any 8-digit hex string>

Default: 0xFFFFFFFF

-h or -help (Command Usage)

-h *architecture*

Displays a usage message for BitGen. The usage message displays all available options for BitGen operating on the specified *architecture*.

-j (No BIT File)

Do not create a bitstream file (.bit file). This option is generally used when you want to generate a report without producing a bitstream. For example, if you wanted to run DRC without producing a bitstream file, you would use the -j option.

Note: The .msk or .rbt files might still be created.

-l (Create a Logic Allocation File)

This option creates an ASCII logic allocation file (*design*.**ll**) for the selected design. The logic allocation file indicates the bitstream position of latches, flip-flops, and IOB inputs and outputs.

In some applications, you may want to observe the contents of the FPGA internal registers at different times. The file created by the -l option helps you identify which bits in the current bitstream represent outputs of flip-flops and latches. Bits are referenced by frame and bit number within the frame.

The Hardware Debugger uses the **design.ll** file to locate signal values inside a readback bitstream.

-m (Generate a Mask File)

Creates a mask file. This file is used to compare relevant bit locations for executing a readback of configuration data contained in an operating FPGA.

-w (Overwrite Existing Output File)

Enables you to overwrite an existing BIT, LL, MSK, or RBT output file.

B

Using PROMGen

The PROMGen program is compatible with the following families.

- Virtex/Virtex-E/Virtex-II

PROMGen formats a BitGen-generated configuration bitstream (BIT) file into a PROM format file (Figure B-2).

The PROM file contains configuration data for the FPGA device. PROMGen converts a BIT file into one of three PROM formats: MCS-86 (Intel), EXORMAX (Motorola), or TEKHEX (Tektronix). It can also generate a Hex file format.

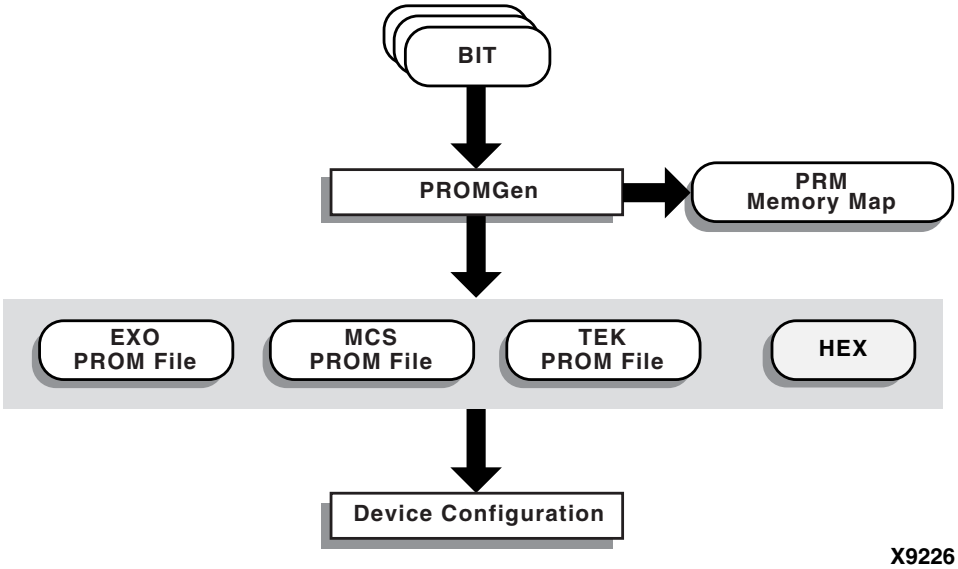

Figure B-2: **PROMGen**

There are two functionally equivalent versions of PROMGen. There is a stand-alone version you can access from an operating system prompt. You can also access an interactive version, called the PROM File Formatter, from inside the Design Manager for Alliance or the Project Manager in Foundation. This chapter describes the stand-alone version; the interactive version is described in the *PROM File Formatter Guide*.

You can also use PROMGen to concatenate bitstream files to daisy-chain FPGAs.

Note: If the destination PROM is one of the Xilinx Serial PROMs, you are using a Xilinx PROM Programmer, and the FPGAs are not being daisy-chained, it is not necessary to make a PROM file. See the *Hardware User Guide* for more information about daisy-chained designs

PROMGen Syntax

Use the following syntax to start PROMGen from the operating system prompt:

```
promgen [options]
```

Options can be any number of the options listed in"PROMGen Options" on page 493. Separate multiple options with spaces.

PROMGen Files

This section describes the PROMGen input and output files.

Input Files

The input to PROMGEN consists of BIT files— one or more bitstream files. BIT files contain configuration data for an FPGA design.

Output Files

Output from PROMGEN consists of the following files.

- PROM files—The file or files containing the PROM configuration information. Depending on the PROM file format used by the PROM programmer, you can output a TEK, MCS, or EXO file. If you are using a microprocessor to configure your devices, you can output a HEX file, containing a hexadecimal representation of the bitstream.
- PRM file—The PRM file is a PROM image file. It contains a memory map of the output PROM file. The file has a **.prm** extension.

Bit Swapping in PROM Files

PROMGen produces a PROM file in which the bits within a byte are swapped compared to the bits in the input BIT file. Bit swapping (also called "bit mirroring") reverses the bits within each byte, as shown in Figure B-3.

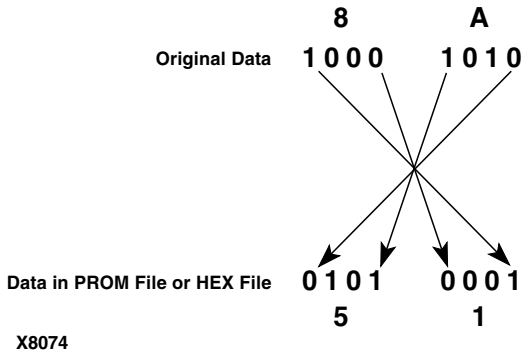

Figure B-3: **Bit Swapping**

In a bitstream contained in a BIT file, the Least Significant Bit (LSB) is always on the left side of a byte. But when a PROM programmer or a microprocessor reads a data byte, it identifies the LSB on the right side of the byte. In order for the PROM programmer or microprocessor to read the bitstream correctly, the bits in each byte must first be swapped so they are read in the correct order.

In this release of the Xilinx Development System, the bits are swapped for all of the PROM formats: MCS, EXO, and TEK. For a HEX file output, bit swapping is on by default, but it can be turned off by entering a -b PROMGen option that is available only for HEX file format.

PROMGen Options

This section describes the options that are available for the PROMGen command.

-b (Disable Bit Swapping—HEX Format Only)

This option only applies if the -p option specifies a HEX file for the output of PROMGen. By default (no -b option), bits in the HEX file are swapped compared to bits in the input BIT files. If you enter a -b option, the bits are not swapped. Bit swapping is described in "Bit Swapping in PROM Files" on page 493.

-c (Checksum)

```
promgen -c
```

The -c option generates a checksum value appearing in the .prm file. This value should match the checksum in the prom programmer. Use this option to verify that correct data was programmed into the prom.

-d (Load Downward)

```
promgen -d hexaddress0 filename filename...
```

This option loads one or more BIT files from the starting address in a downward direction. Specifying several files after this option causes the files to be concatenated in a daisy chain. You can specify multiple -d options to load files at different addresses. You must specify this option immediately before the input bitstream file.

The multiple file syntax is as follows:

```
promgen -d hexaddress0 filename filename...
```

The multiple **-d** options syntax is as follows:

```
promgen -d hexaddress1 filename -d hexaddress2 filename...
```

-f (Execute Commands File)

```
-f command_file
```

The -f option executes the command line arguments in the specified *command_file*.

-help (Command Help)

This option displays help that describes the PROMGen options.

-l option (Disable Length Count)

```
promgen -l
```

The -l option disables the length counter in the FPGA bitstream. It is valid only for 4000EX, 4000XL, 4000XLA, 4000XV, and SpartanXL Devices. Use this option when chaining together bitstreams exceeding the 24 bit limit imposed by the length counter.

-n (Add BIT FIles)

```
-n file1[.bit] file2[.bit]...
```

This option loads one or more BIT files up or down from the next available address following the previous load. The first -n option *must* follow a -u or -d option because -n does not establish a direction. Files specified with this option are not daisy-chained to previous files. Files are loaded in the direction established by the nearest prior -u, -d, or -n option.

The following syntax shows how to specify multiple files. When you specify multiple files, PROMGen daisy-chains the files.

```
promgen -d hexaddress file0 -n file1 file2...
```

The following syntax when using multiple -n options prevents the files from being daisy-chained:

```
promgen -d hexaddress file0 -n file1 -n file2...
```

-o (Output File Name)

```
-o file1[.ext] file2[.ext]...
```

This option specifies the output file name of a PROM if it is different from the default. If you do not specify an output file name, the PROM file has the same name as the first BIT file loaded.

ext is the extension for the applicable PROM format.

Multiple file names may be specified to split the information into multiple files. If only one name is supplied for split PROM files (by you or by default), the output PROM files are named *file*_#.ext, where *file* is the base name, # is 0, 1, etc., and *ext* is the extension for the applicable PROM format.

```
promgen -d hexaddress file0 -o filename
```

-p (PROM Format)

```
-p {mcs | exo | tek | hex}
```

This option sets the PROM format to one of the following: MCS (Intel MCS86), EXO (Motorola EXORMAX), TEK (Tektronix TEKHEX). The option may also produce a HEX file, which is a hexadecimal representation of the configuration bitstream used for microprocessor downloads. If specified, the -p option must precede any -u, -d, or -n options. The default format is MCS.

-r (Load PROM File)

```
-r promfile
```

This option reads an existing PROM file as input instead of a BIT file. All of the PROMGen output options may be used, so the -r option can be used for splitting an existing PROM file into multiple PROM files or for converting an existing PROM file to another format.

-s (PROM Size)

```
-s promsize1 promsize2...
```

This option sets the PROM size in kilobytes. The PROM size must be a power of 2. The default value is 64 kilobytes. The -s option must precede any -u, -d, or -n options.

Multiple *promsize* entries for the -s option indicates the PROM will be split into multiple PROM files.

Note: PROMGen PROM sizes are specified in bytes. *The Programmable Logic Data Book* specifies PROM sizes in bits for Xilinx serial PROMs (see -x option).

-u (Load Upward)

```
-u hexaddress0 filename1 filename2...
```

This option loads one or more BIT files from the starting address in an upward direction. When you specify several files after this option, PROMGen concatenates the files in a daisy chain. You can load files at different addresses by specifying multiple -u options.

This option must be specified immediately before the input bitstream file.

-x (Specify Xilinx PROM)

```
-x xilinx_prom1 xilinx_prom2...
```

The -x option specifies one or more Xilinx serial PROMs for which the PROM files are targeted. Use this option instead of the -s option if you know the Xilinx PROMs to use.

Multiple xilinx_*prom* entries for the -x option indicates the PROM will be split into multiple PROM files.

Examples

To load the file test.bit up from address 0x0000 in MCS format, enter the following information at the command line.

```
promgen -u 0 test
```

To daisy-chain the files test1.bit and test2.bit up from address 0x0000 and the files test3.bit and test4.bit from address 0x4000 while using a 32K PROM and the Motorola EXORmax format, enter the following information at the command line.

```
promgen -s 32 -p exo -u 00 test1 test2 -u 4000 test3 test4
```

To load the file test.bit into the PROM programmer in a downward direction starting at address 0x400, using a Xilinx XC1718D PROM, enter the following information at the command line.

```
promgen -x xc1718d -d 0x400 test
```

To specify a PROM file name that is different from the default file name enter the following information at the command line.

```
promgen options filename -o newfilename
```