

Encrypting Bitfiles For the PCI GS

EDT PCI GS boards having revision numbers greater than 40 include a small battery to maintain power on the Xilinx field-programmable gate array (FPGA) XC2VP50 or XC2VP70 internal memory. This memory is intended to store the key used to encrypt an FPGA configuration file (*bitfile*). Any board having the appropriate key stored in this memory can then run the bitfile encrypted using that key; without it, the encrypted bitfile will not run. The boards can also, of course, run unencrypted bitfiles as before.

This document explains how to use the Xilinx tools `bitgen` and `impact` (ISE 9.1 service pack 3) to encrypt a bitfile, store the associated keyfile on the PCI GS, load the encrypted bitfile on the PCI GS, and ensure that the keyfile has been retained so that the encrypted bitfile can be loaded and run again after the board has been power-cycled.

When encrypting a bitfile, `bitgen` generates six keys, only three of which are used to encrypt a given bitfile. (You can choose the keys, and choose which ones are used, and in which order; or you can allow `bitgen` to randomly generate keys and randomly determine which ones are used, and in which order.) All six keys are saved in the keyfile, which also specifies the order in which the keys are used. A different set of three keys out of these same six can be used to encrypt another bitfile, a different set for yet another, and so on, with the order changing too in each case. In this manner, one keyfile stored on a PCI GS can be used to encrypt a number of different bitfiles, each uniquely.

Xilinx uses the triple DES algorithm to encrypt and decrypt the bitfiles.

To make use of this feature requires four main steps:

1. Make an encrypted bitfile and its accompanying keyfile using the Xilinx tools.
2. Also using Xilinx tools, program the key that was used to encrypt the bitfile into the PCI GS.
3. Load the encrypted bitfile onto the PCI GS as before.
4. Test to make sure that the key has been stored on the PCI GS.

These steps are detailed below.

NOTE In the instructions below, placeholders for values you must supply appear in italics, as in this example in which you must supply a filename: `bitload bitfile_name`

Make the encrypted bitfile.

This procedure assumes that you have already done the programming necessary to create your bitfile, and that you've tested it.

1. Open the Xilinx ISE Tools, if necessary.
2. In the left pane of the **Generate Programming File Properties** window, click on **Encryption Options**.
3. Check the **Encrypt Bitstream** checkbox.
4. In the resulting dialog, assign the keys and the order if you like, or leave the fields blank to allow `bitgen` to do so randomly.
5. Click **Apply** to have your changes take effect.

6. Click **OK** to dismiss the dialog.

This produces an encrypted FPGA configuration file with the file extension `.bit`, and a keyfile containing the key used to encrypt it, having the file extension `.nky`.

7. To avoid overwriting your key file and possibly losing the key in case of a system malfunction, copy the `.nky` file to a new filename with the same extension. Once programmed into the board, the keyfile cannot be read back or removed, only overwritten.

If subsequent `bitgen` runs are required for this bitfile, you'll need to specify the keyfile name unless you wish to store a new keyfile on the board.

Program the key into the PCI GS.

To program the key into the Xilinx XC2VP50 or XC2VP70 internal memory on the PCI GS, you'll need a Xilinx JTAG cable and the Xilinx `impact` application.

1. Connect one end of the JTAG cable to the JTAG interface on the back edge of the PCI GS, connecting the TDO signal to the pin labeled **TDO V2P**. (Due to the connector in the way, part of the label may not be visible. The correct pin is the bottommost one — the bottom of the board is the edge that holds the PCI Bus connector.)
2. Connect the USB or parallel port connector at the other end of the cable to a computer on which the `impact` application is installed.
3. Invoke `impact`.
4. In the resulting dialog, click **Configure Devices Using Boundary Scan (JTAG)**.
5. In the **Assign New Configuration File** dialog, select the keyfile you created earlier.
6. Click **Open**.
7. Click **OK** again to dismiss the dialog.
8. Right-click on the icon representing the XC2VP50 or XC2VP70.
9. Select **Program...**
10. Click **OK**.

The selected keyfile is now written to the internal memory of the Xilinx XC2VP50 or XC2VP70 on the attached PCI GS board.

Load the encrypted bitfile on the PCI GS.

Using the EDT application `bitload`, load the encrypted bitfile onto the PCI GS as you would an unencrypted bitfile. (Supply the `-u unit_number` option if the board you're loading is not the default unit 0.)

```
bitload bitfile_name
```

`bitload` prints a message specifying whether the bitfile loaded or the operation failed.

NOTE Without the correct key already stored in the board, an encrypted bitfile will fail to load.

Test to ensure the key has been retained.

1. Power-cycle the host computer.
2. Run `bitload` again to reload the encrypted bitfile.

If the key has been successfully retained in internal memory, the encrypted bitfile loads again; otherwise, it fails to load.

Battery Life and Replacement

The current drain required to maintain the memory is tiny, the greater part of the leakage due to the materials used in the board. We expect the battery life, therefore, to be essentially equal to the battery's storage life. Xilinx claims the battery's storage life is fifteen years, although the battery manufacturer claims only ten years. The warmer the environment in which the battery is stored, the shorter its life.

Whether the board is powered on or not, no current drains from the battery except for leakage.

If the battery fails, the key is lost; bitfiles encrypted with it will no longer load on that board, although unencrypted bitfiles will, of course, load and run as usual.

Because the battery socket is less reliable than the battery itself, a socket is not used. Instead, batteries are attached to the board using a soldered connection. To replace the battery:

1. Separate the main and mezzanine boards.
2. Use a soldering iron to remove the old battery.
3. Solder a new battery into the old location.

Alternatively, return the board to EDT for battery replacement. After replacing the battery, the key must be reprogrammed.