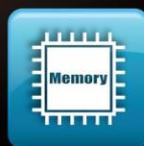
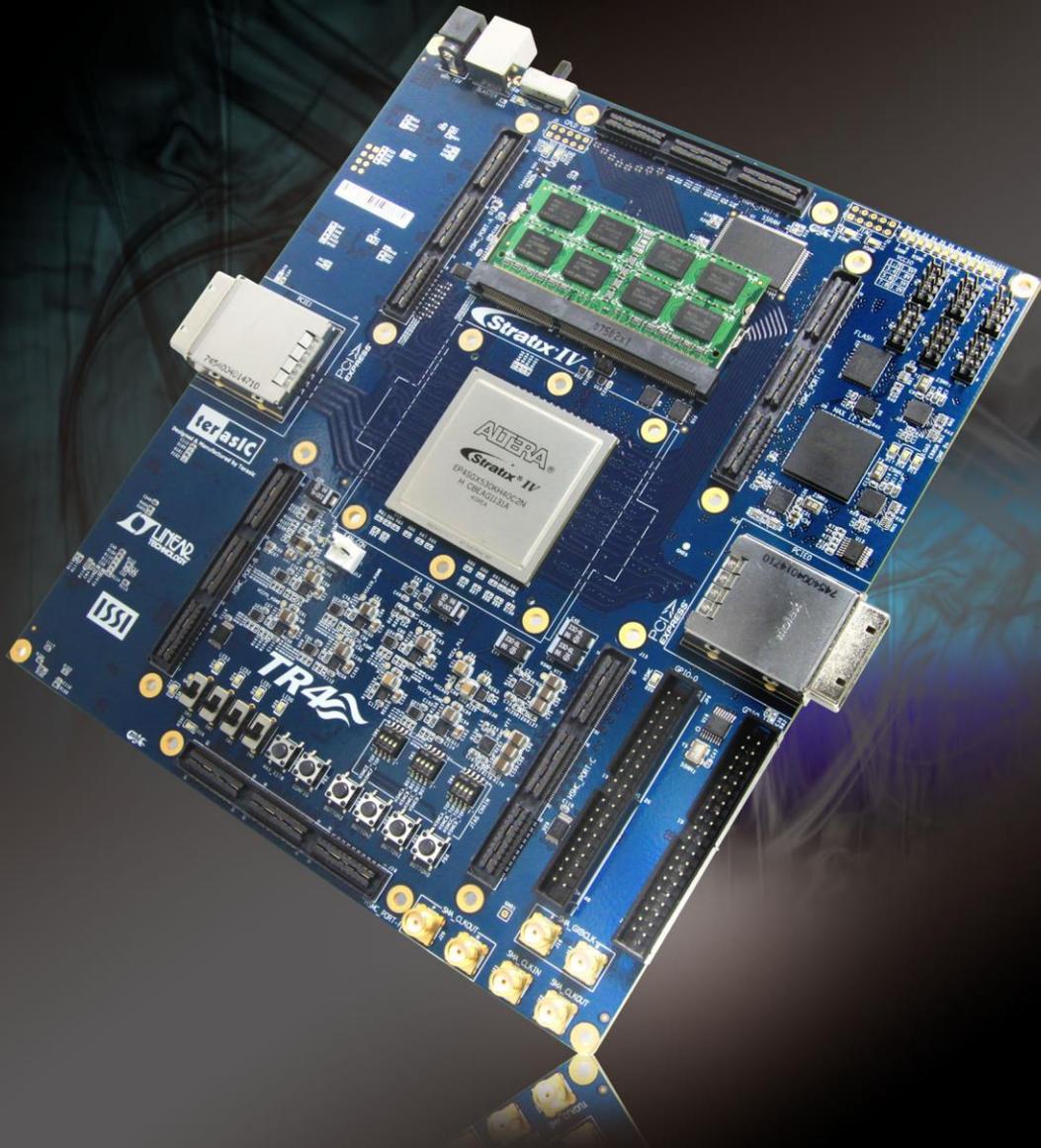


TR4

FPGA Development Kit

Using the Design Security Feature in TR4



Using the Design Security Feature in TR4

FPGA Development Kit

1. Introduction

This application note describes how user can use the design security features in TR4 FPGA development kit to protect user's designs against unauthorized copying, reverse engineering, and tampering of configuration files. This application note provide the hardware and software requirements for TR4 board and provides configuration flow for implementing a secure design.

Design security is an important consideration for digital designers, especially in today's highly competitive commercial and military environments. The TR4 board feathers design security play an important role in larger and more critical system components. Altera Stratix® IV GX FPGA (EP4SGX230C2/EP4SGX530C2) address these concerns with the ability to decrypt a configuration bitstream using the 256-bit Advanced Encryption Standard (AES) algorithm, an industry standard encryption algorithm. For more information about design security features, please refer Altera application note "[AN556: Using the Design Security Features in Altera FPGAs](#)".

2. Hardware and Software Requirements

This section provides the hardware and software requirements for the TR4 board design with security feature. When using this feature, user must make sure that specified resistances on TR4 board connection rightly, to select "[FPP with design security feature and/or decompression enabled](#)" Configuration Scheme for FPGA. A volatile or non-volatile key is stored in the FPGAs. The key is programmed before the FPGAs is configured and enters user mode.

2.1 Hardware Requirements

2.1.1 Resistance connection

TR4 factory default Configuration Scheme is "[Fast passive parallel](#)", So before using design security features, user must make sure that the resistance R451, R452, R453, R454 one the TR4 board connection rightly, to select "[FPP with design security feature and/or decompression enabled](#)" Configuration Scheme for FPGA. Figure 2-1 shown the location of these resistances and connection of factory default and design security respectively. For more information about design security features, please refer Altera application note "[AN556: Using the Design Security Features in Altera FPGAs](#)", and "[Chapter 10: Configuration, Design Security, and Remote System Upgrades in Stratix IV Devices](#)" of Altera stratix4_handbook.

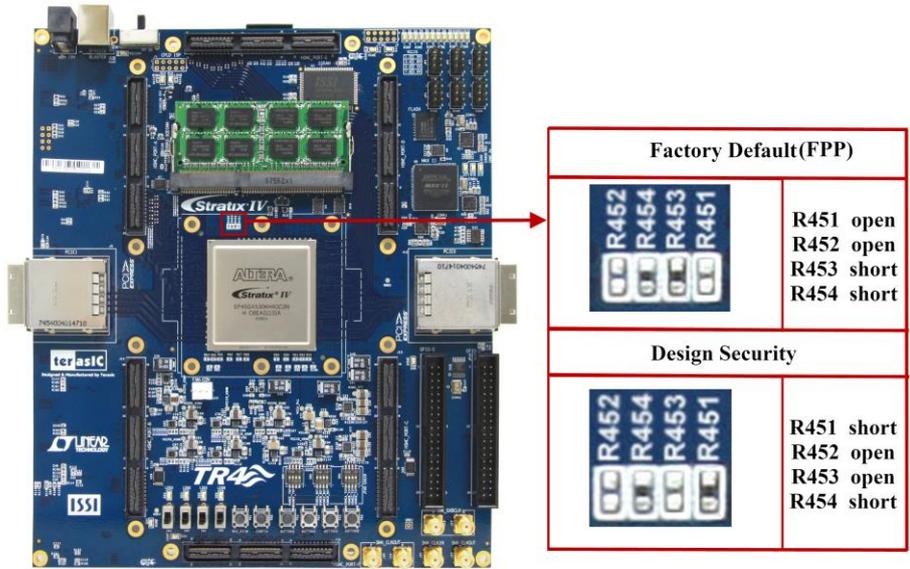


Figure 2-1 connection of factory default and design security

2.1.2 Battery Assemble

FPGAs offer both volatile and non-volatile key storage. The volatile key storage requires battery backup to allow the key to be updated, while the non-volatile key storage allows only one key to be programmed but does not require a battery. If user want to apply design security with volatile key , the user should first install a battery holder on the bottom TR4 board, since factory default don't assemble it and also, don't provide battery holder. Figure 2-2 shows the TR4 board with battery and without battery respectively. The battery holder's part number is *1060TR* produced by *Keystone Electronics Corporation* , The battery is 3V.

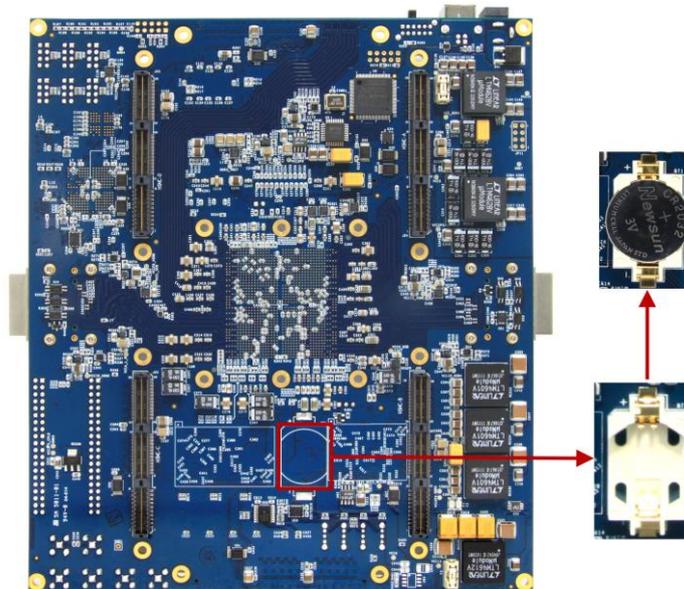


Figure 2-2 Install battery holder and battery

2.2 Software Requirements

TR4 board features Stratix IV FPGA, user must use the Quartus® II software version 9.0 or later, To enable the design security feature of 40-nm FPGAs, To enable the design security feature, user can obtain a license file from Altera Technical Support.

3. Steps for Implementing a Secure Configuration Demonstration

This section will describe how to design a volatile key to cipher the FPGA, Figure 3-1 shows the PFL interface during Flash Programming. The section will first describe how to Generate .ekp File and Encrypt Configuration File. And then show how to configure Parallel Flash Loader (PFL) in programming a parallel flash device.

Note: The project in 3.1 and 3.2 are both using the Quartus II software version 11.0.

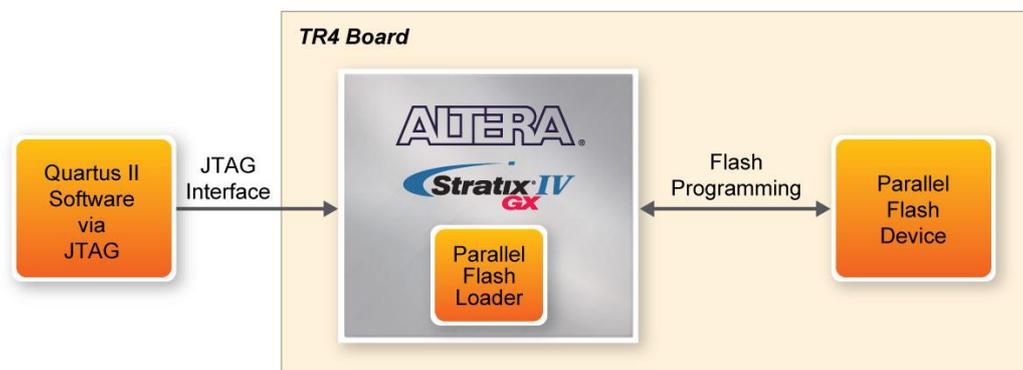


Figure 3-1 PFL Interface During Flash Programming

3.1 Generate the .ekp File and Encrypt the Configuration File

To generate a .ekp file and encrypt your configuration file using Quartus II Software, follow these steps:

1. Obtain a license file to enable the design security feature from Altera Technical Support.
2. Start the Quartus II software.
3. On the **Tools** menu, click **License Setup**. The Options dialog box displays the License Setup options.
4. In the License file field, enter the location and name of the license file, or browse to and select the license file.
5. Click **OK**.
6. Compile the design you want to encrypt with one of the following options:
 - a. On the **Processing** menu, click **Start Compilation**
 - b. On the **Processing** menu, point to **Start** and click **Start Assembler**.An unencrypted SRAM Object File (.sof) is generated.
7. On the **File** menu, click **Convert Programming Files**. The **Convert Programming Files** dialog box appears, as shown in Figure 3-2.

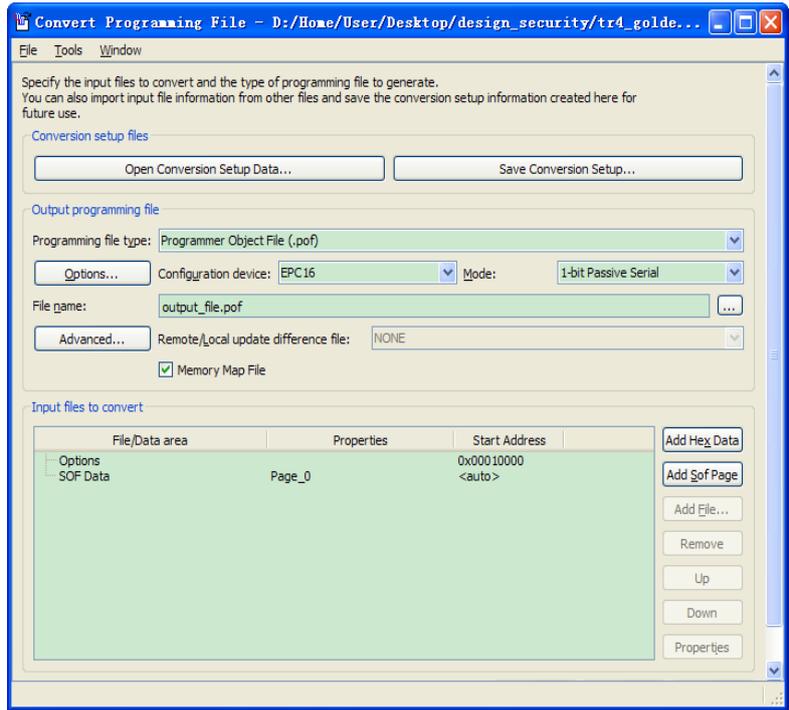


Figure 3-2 Convert Programming Files Dialog Box

- a. In the **Convert Programming Files** dialog box, select the programming file type from the **Programming** file type list. In this demonstration, select “Programmer Object File(.pof)”.
- b. In the **Configuration** device dialog box, select CFI_512Mb, match the flash type on TR4 board.
- c. In the **Mode** dialog box, select “Fast Passive Parallel”.
- d. Type the file name in the **File** name field, or browse to and select the file.
- e. Under the Input files to convert section, click SOF Data.
- f. Click **Add File** to open the **Select Input File** dialog box.
- g. Browse to the unencrypted SOF file and click Open. Still highlight SOF Data. Click **Properties**, SOF Data **Properties** dialog box appears, and set each item as shown in Figure 3-3.

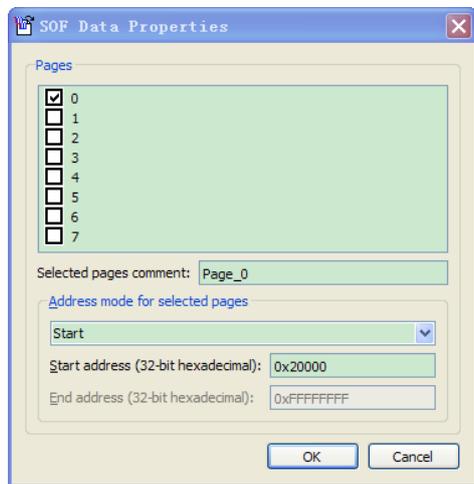


Figure 3-3 SOF Data Properties dialog box

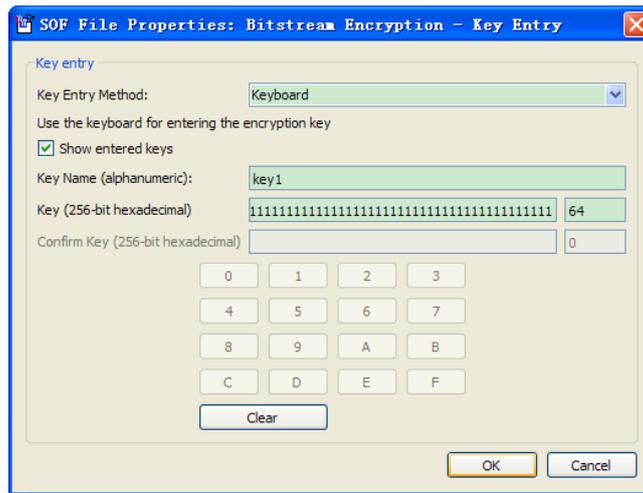


Figure 3-6 Key Entry dialog box

- p. Click OK. The **Convert Programming Files Dialog Box** shown as Figure 3-7.

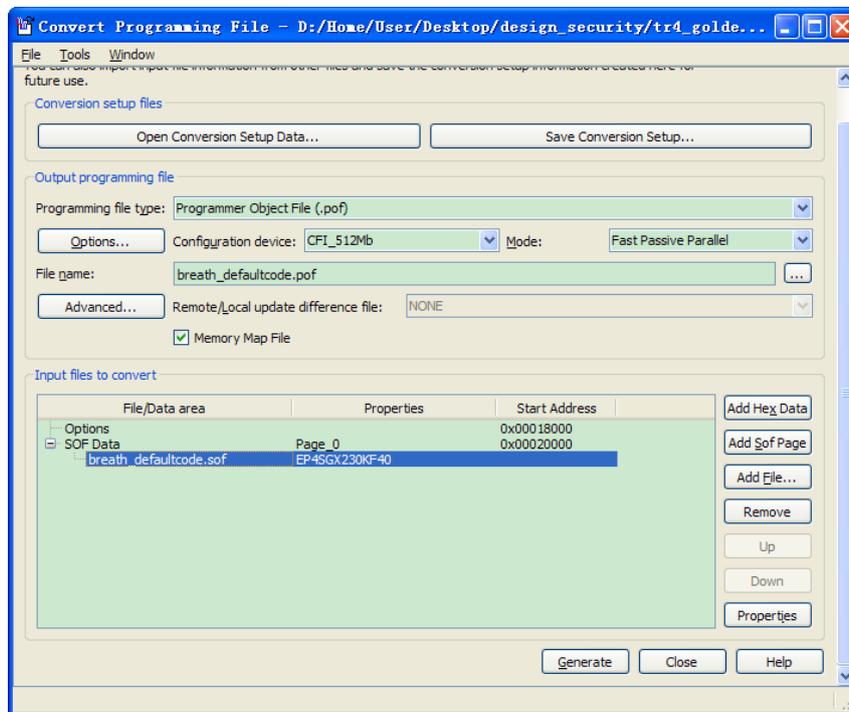


Figure 3-7 Convert Programming Files Dialog Box

- 8 In the **Convert Programming Files** dialog box, click OK. The <filename>.ekp and encrypted configuration file are generated in the same project directory.

3.2 Using Parallel Flash Loader

The FPGA-based PFL is a soft intellectual property (IP) core within the FPGA that bridges the JTAG and parallel flash interfaces. With the PFL, you can use the serial programming bitstream from the JTAG interface to control the flash data, address, and control pins for flash programming. The JTAG interface simplifies the flash programming process because it reduces the number of

pins required and shares the same interface as the flash device. For more information about PFL, please refer Altera application note "*AN478: Using FPGA-Based ParallelFlash Loader with the Quartus II Software*".

3.2.1 Instantiation the Parallel Flash Loader Megafunction

The project is designed base **Block Diagram/Schematic File**. Perform the following steps to generate the PFL instantiation:

1. On the Tools menu in the Quartus II software, click **MegaWizard Plug-In Manager**.
2. On page 1, click **Create a new custom megafunction variation**. Click **Next**. Page 2a appears As shown in Figure 3-8.

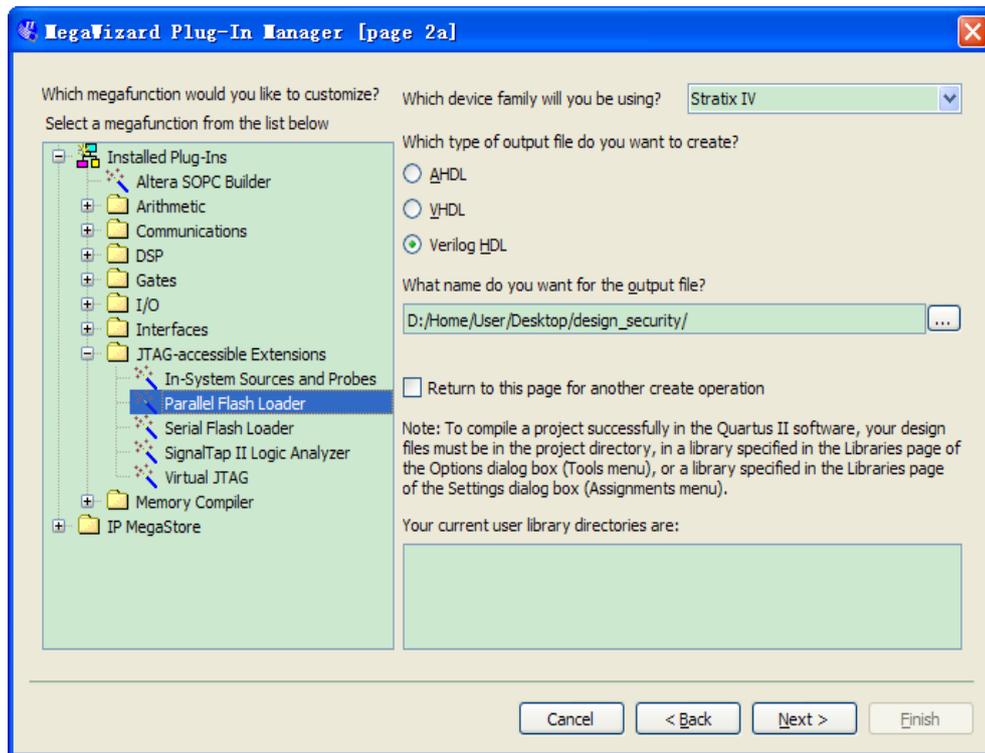


Figure 3-8 MegaWizard Plug-In Manager [page 2a] Dialog Box

3. In the megafunction list, expand the **JTAG-accessible Extensions** folder and select **Parallel Flash Loader**.
4. Make sure in the device family list, select **Stratix IV**.
5. In the output file list, select **Verilog HDL** output file type, As shown in Figure 3-8.
6. Click **Next**. **General** page appears and specify value as shown in Figure 3-9.

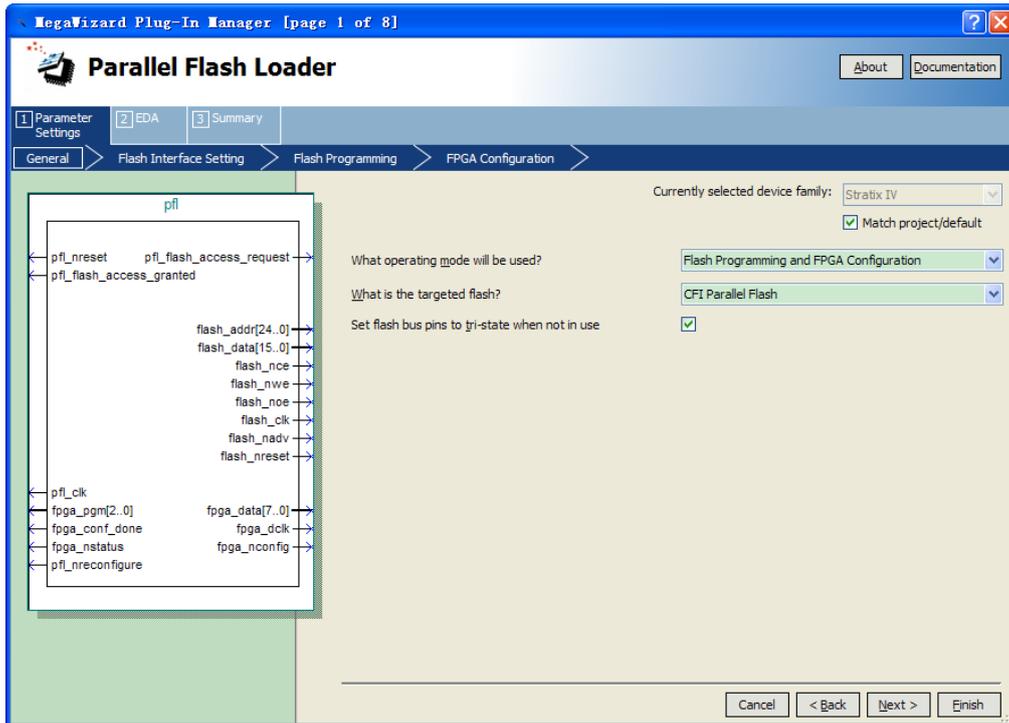


Figure 3-9 Setting the PFL Megafunction Parameters(Step 1)

- Click Next. **Flash Interface Setting** page appears and specify value as shown in Figure 3-10.

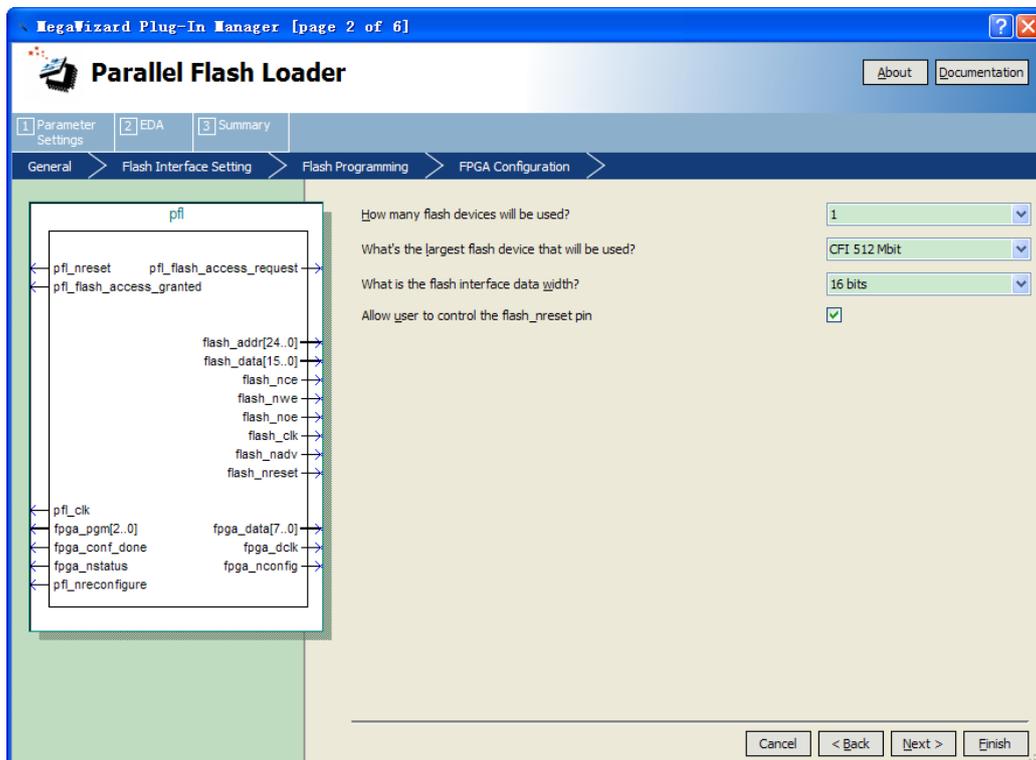


Figure 3-10 Setting the PFL Megafunction Parameters(Step 2)

- Click Next. **Flash Programming** page appears and specify value as shown in Figure 3-11.

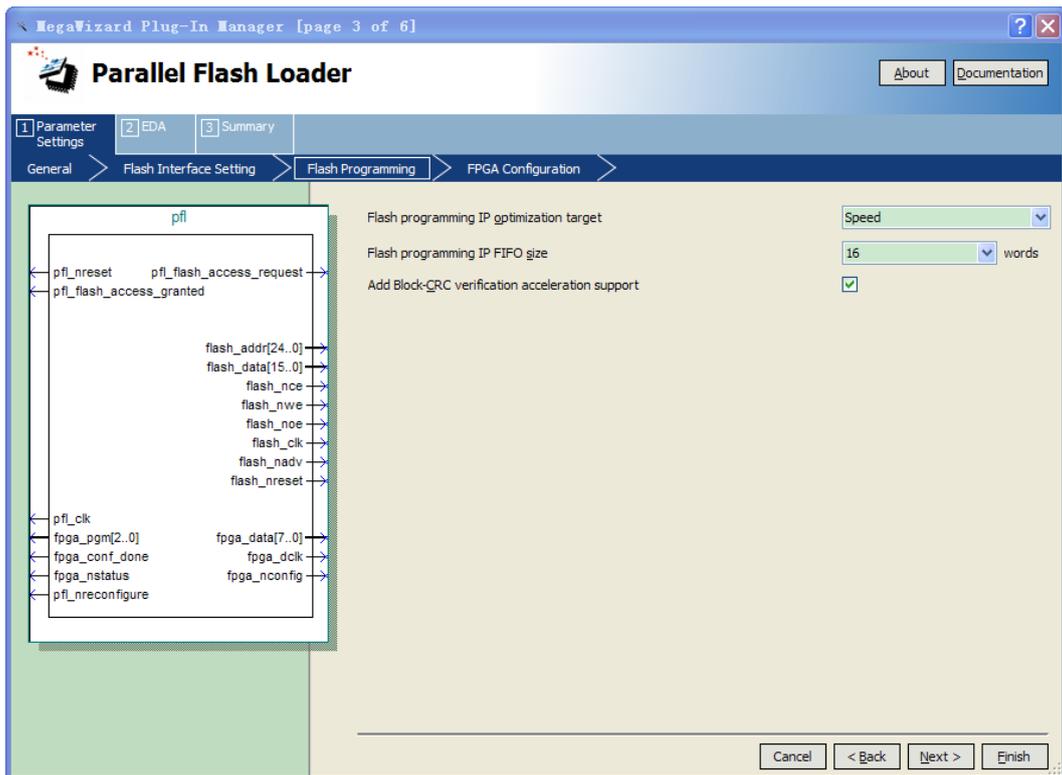


Figure 3-11 Setting the PFL Megafunction Parameters(Step 3)

9. Click Next. **FPGA Configuration** page appears and specify value as shown in Figure 3-12.

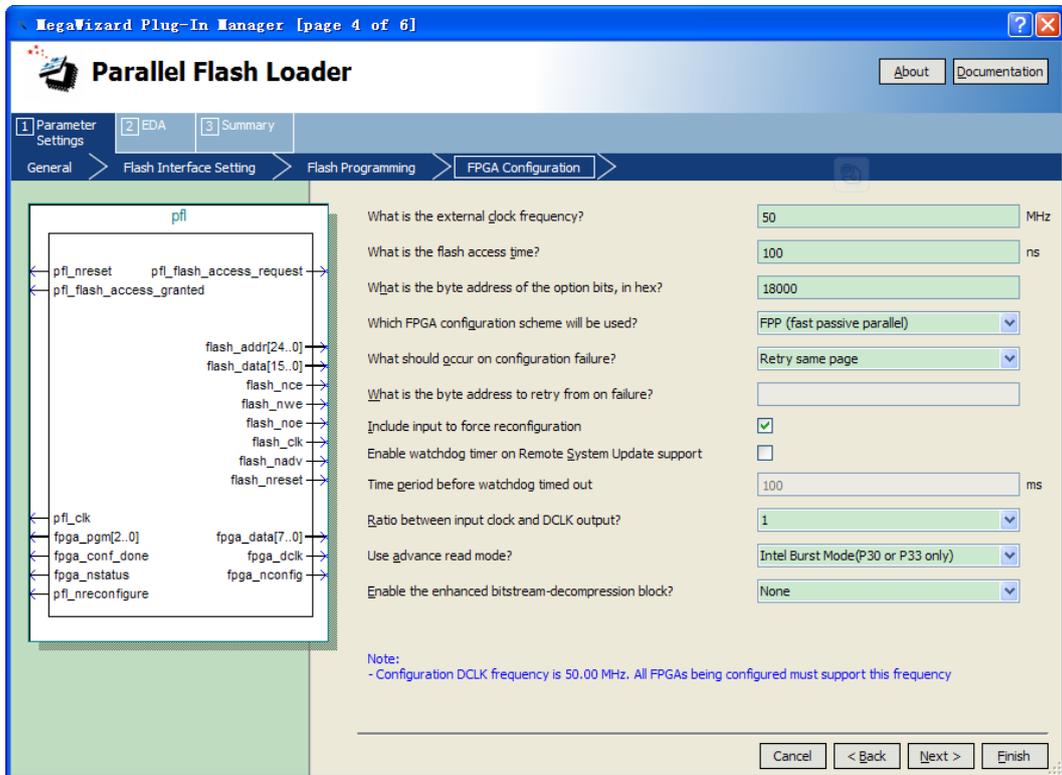


Figure 3-12 Setting the PFL Megafunction Parameters(Step 4)

10. Click Next. Page 4 appears, listing the simulation files needed for the PFL megafunction. No

simulation file will be listed for the megafunction because the PFL does not have any simulation files and it cannot be simulated

11. Click **Next**. **Summary** page appears, s shown in Figure 3-13 This page shows the files that will be created for the megafunction. Choose any additional file types that you want to create.

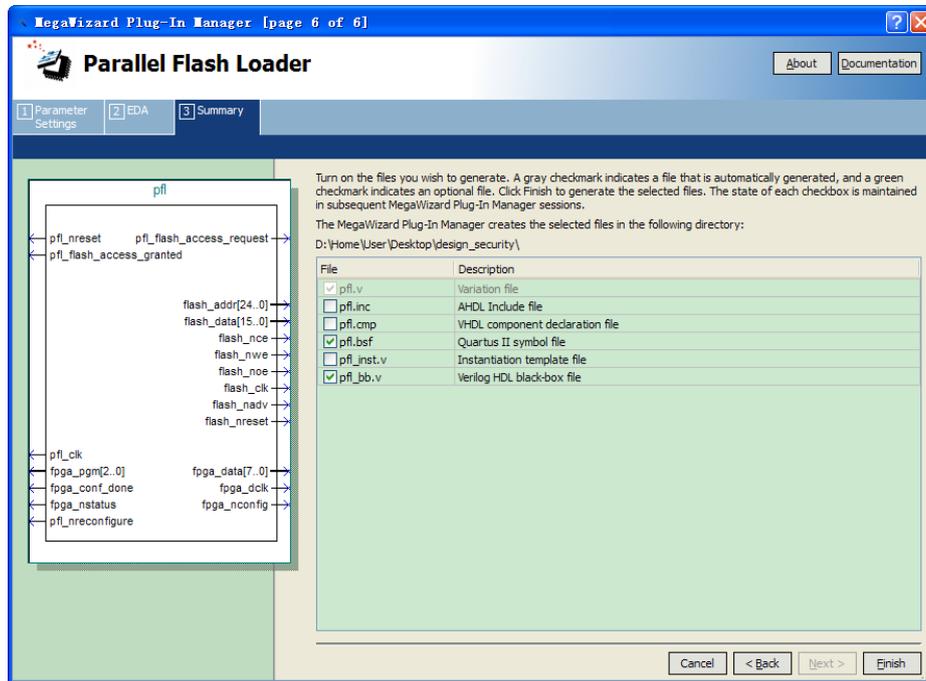


Figure 3-13 Setting the PFL Megafunction Parameters(Step 5)

12. Click **Finish**. The Quartus II software generates the PFL megafunction in the form of the HDL file selected on Figure 3-8 and any additional files selected on Figure 3-13.

3.2.2 Input and Output Ports of the Parallel Flash Loader Megafunction

Figure 3-14 shows the symbol for the PFL megafunction. The functions of the PFL input and output ports, please refer Altera application note "[AN478: Using FPGA-Based ParallelFlash Loader with theQuartus II Software](#)".

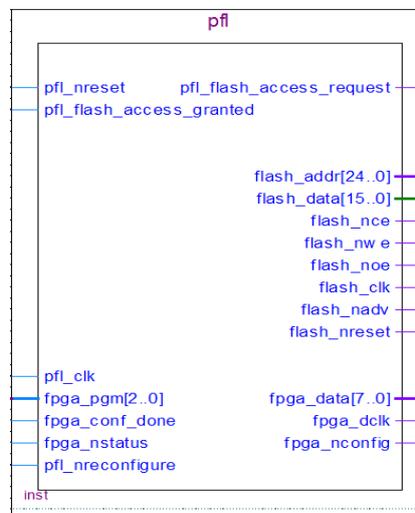


Figure 3-14 PFL Megafunction Symbol

This section only describe how to connect the input and output ports of the PFL megafunction as shown in Table 3-1. Please reference *"TR4_UserManual"* for more details.

Table 3-1 the input and output ports of the PFL megafunction connection

Name	Connection
Pfl_nreset	VCC
Pfl_flash_access_granted	Pfl_flash_access_request
Pfl_clk	OSC_50_BANK2
Fpga_pgm[2..0]	GND
Pfl_nreconfigure	VCC
Flash_add[24..0]	FSM_A[25..1]
Flash_data[15..0]	FSM_D[15..0]
Flash_nce	FLASH_CE_n
Flash_nwe	FSM_WE_n
Flash_noe	FSM_OE_n
Flash_clk	FLASH_CLK
Flash_nadv	FLASH_ADV_n
Flash_nreset	FLASH_RESET_n

After the connection finished as Table 3-1, the Block Diagram shown as Figure 3-15.

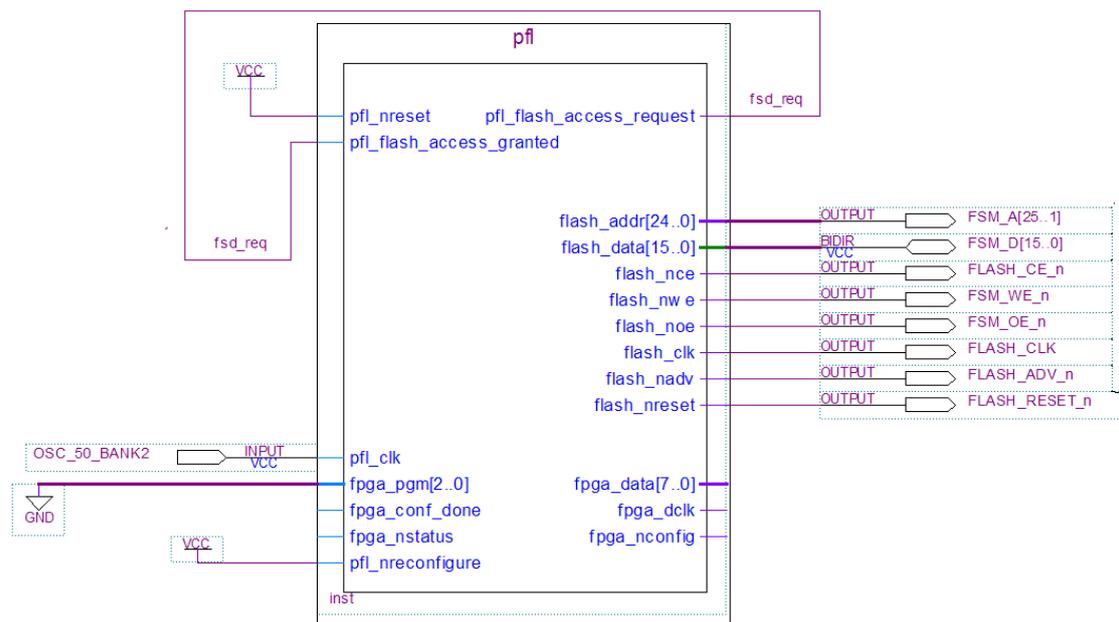


Figure 3-15 PFL symbol connection

After save it,click **Start Compilation** to compile the project, and <filename>.sof file are generated in the same project directory. Next, copy the <filename>.ekp and encrypted configuration file(.pof) generated in **"3.1 Generate the .ekp File and Encrypt the Configuration File"** to the same directory.

3.3 Programming the Parallel Flash Device, Encrypt Configuration

File, .ekp File

Perform the following steps to program the flash device in the Quartus II Programmer:

1. On the Tools menu in the Quartus II software, click **Programmer**.
2. In the Programmer window, click **Add File**. The **Select Programming File** dialog box appears, as shown in Figure 3-16.

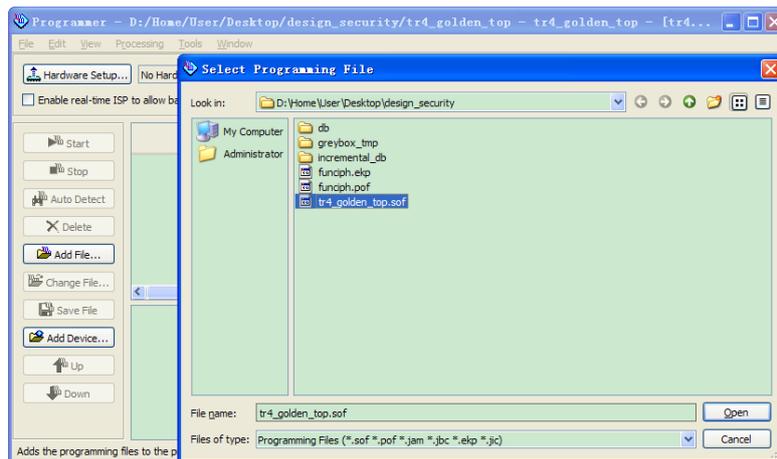


Figure 3-16 Adding SOF for PFL

3. Select the SOF of the user design that contains the PFL logic.
4. Click **Open**. The SOF name appears in the Programmer window.
5. Select and right-click the SOF you just added. Click **Attach Flash Device**, as shown in Figure 3-17. The **Select Flash Device dialog** box appears as shown in Figure 3-18.

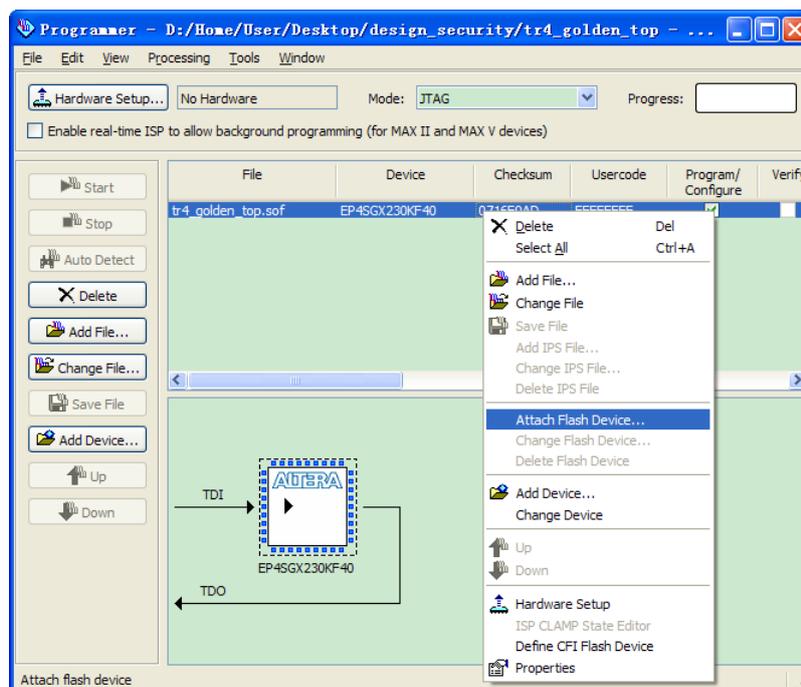


Figure 3-17 Attaching Flash Device

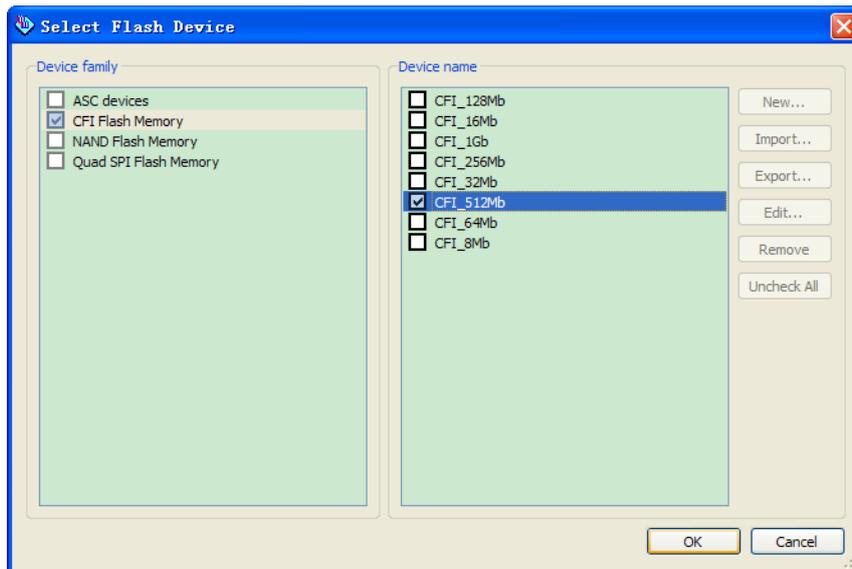


Figure 3-18 Selecting Flash Device

6. Under Device family, turn on **CFI Flash Memory**.
7. Under Device name, select the density of the flash device.
8. Click **OK** to go back to the Programmer window.
9. Select and right-click the device name. Click **Change File**. The **Select New Programming File** dialog box appears as shown in Figure 3-19.

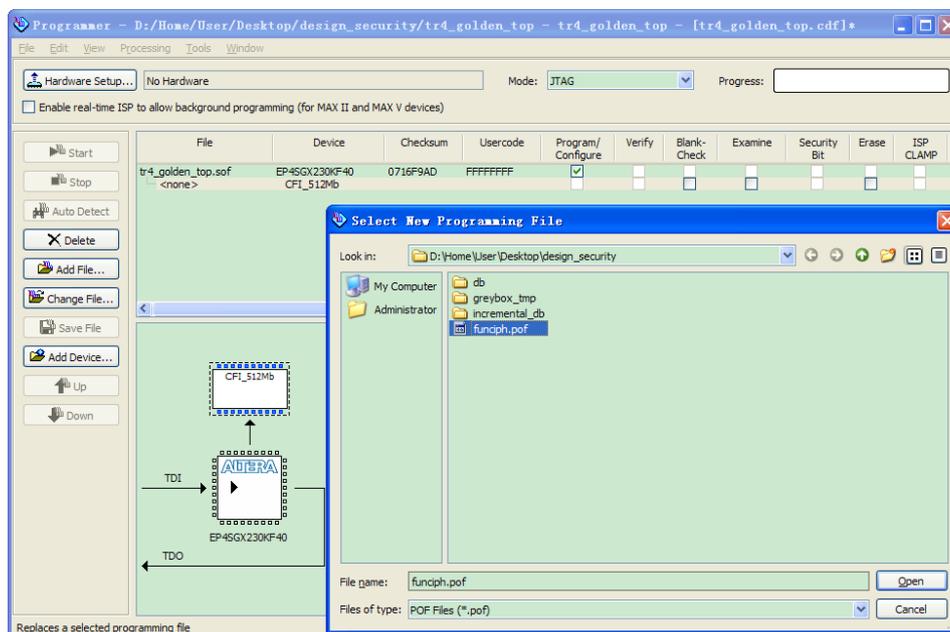


Figure 3-19 Adding POF for Flash Programming Device

10. Select the POF of the flash device and click **Open**.
11. Under **Program/Configure** column, turn on all check box, as shown in Figure 3-20

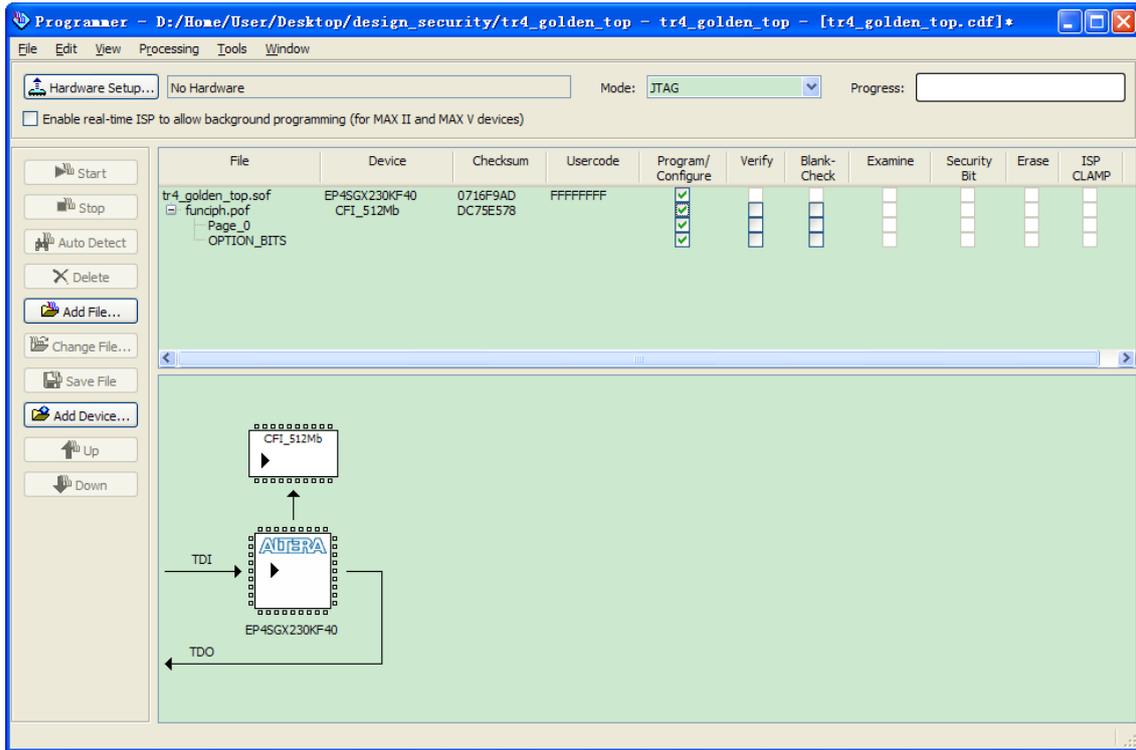


Figure 3-20 Quartus II Programmer Showing PFL and POF of Flash Device

12. Click **Start** to configure the PFL and program the flash device. Then the Encrypted <filename>.pof file is stored in flash device on TR4 board.
13. Highlight <filename>.sof file, click **delete**. All files in the programmer window are excluded. Then click **Add file**, shown as Figure 3-16, select <filename>.ekp, and click **Open**, The programmer shown as Figure 3-21.

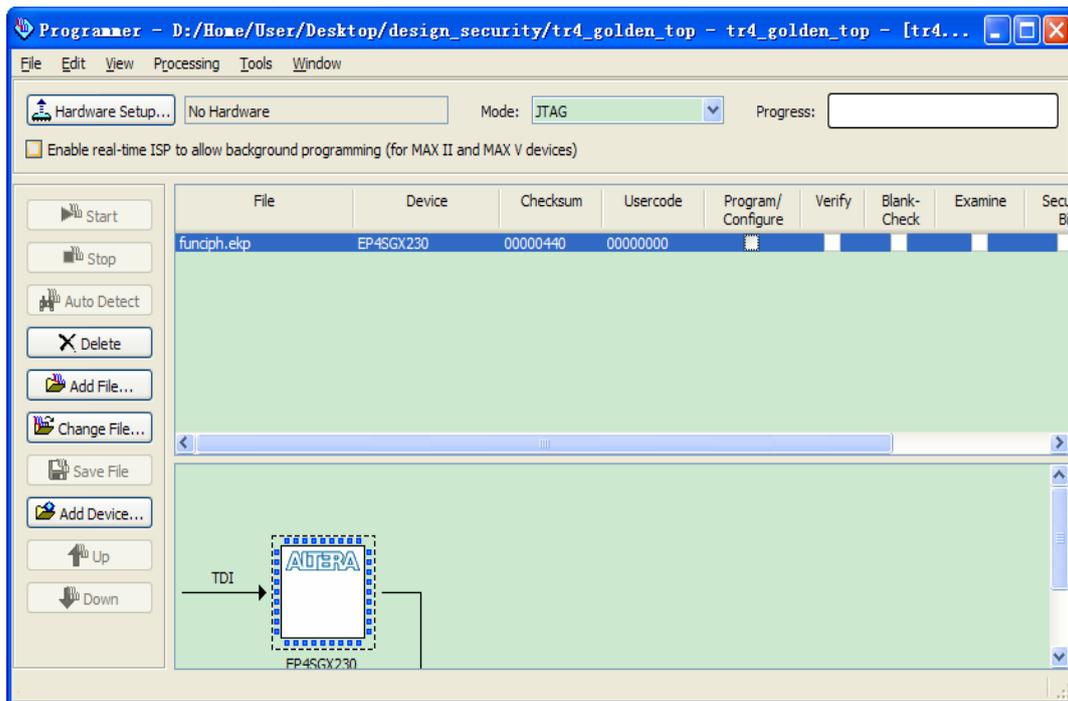


Figure 3-21 programmer .ekp file

14. Under **Program/Configure** column, turn on all check box.
15. Click **Tools** and select **Option**, the **Option** window appears ,as is show in Figure 3-22.

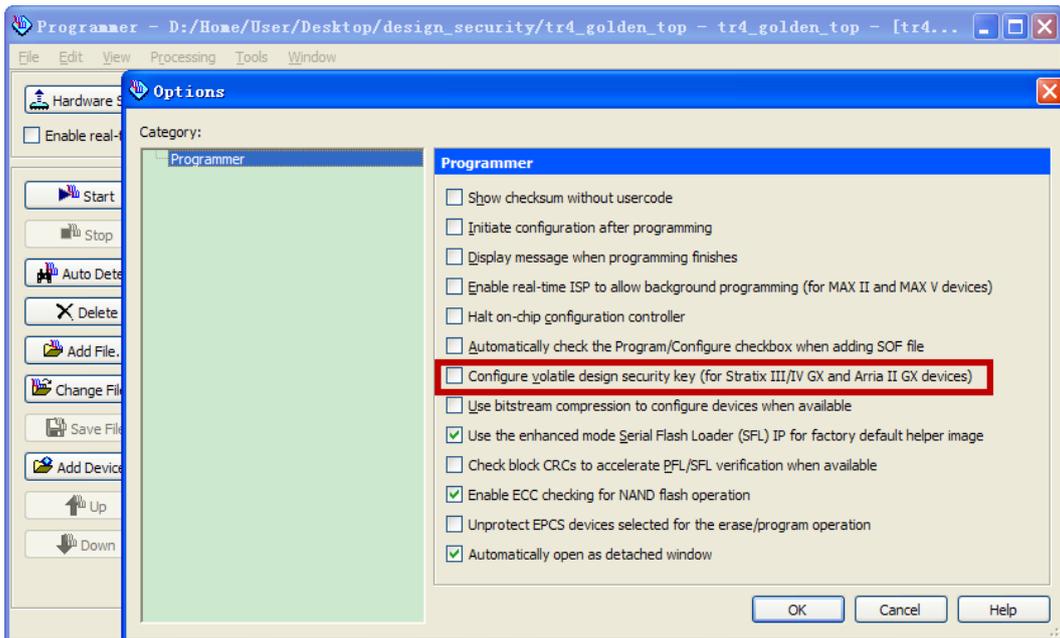


Figure 3-22 Option window

16. Make sure the item “*Configure volatile design security key(for Stratix III/IV GX and Arria II GX device)*” is not selected.
17. Click **Start**. The progress successful in moment. As is shown in Figure 3-22

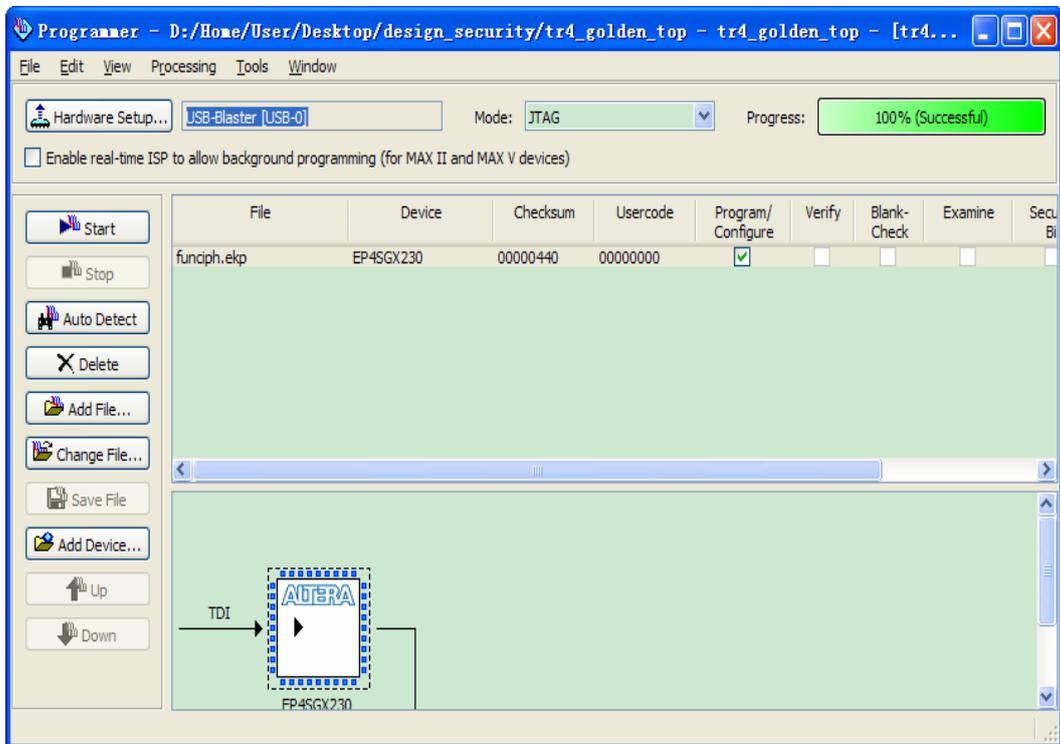


Figure 3-22 successfully progress

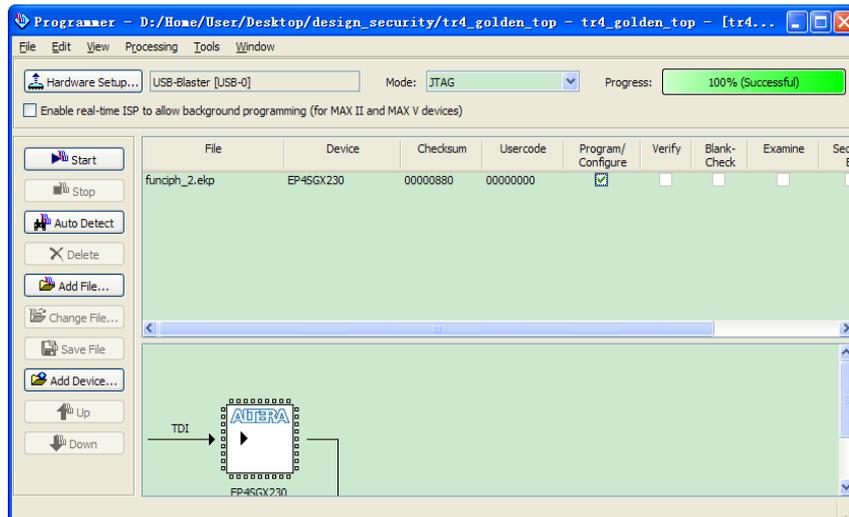


Figure 3-24 programmer the <filename>.ekp file to FPGA

Restart TR4, the TR4 will not work, because the key stored in flash is all "1", different the all "2" key retained in FPGA.

Again, programmer all "1" key to FPGA, restart TR4 board, the TR4 will function as user's design.

4.2 Take off battery

Now, take off battery installed on the bottom of TR4 board, and power on TR4 board, it will not work, because no key is maintained in the FPGA to correspond the key stored in flash. Even though battery installed into battery holder, TR4 still do not work, because the key which is previously stored in FPGA has already "lost" when the battery token off.

If programmer all "1" key to FPGA which is provide an external battery to retain the volatile key. And restart TR4 board, the TR4 board work well again.

5 Conclusion

in TR4 FPGA development kit with design security features work well to protect user's designs against unauthorized copying, reverse engineering, and tampering of configuration files. The PFL feature available in Altera FPGAs enables you to use in-system programming to program parallel flash devices. The Quartus II software provides the tools necessary for you to program the parallel flash device through the FPGA's JTAG interface.

6 Referenced Documents

This application note references the following document:

TERASIC TR4 CD

TR4_UserManual

TR4_Schematic

Altera application note and handbook

"AN556: Using the Design Security Features in Altera FPGAs".

"AN478: Using FPGA-Based ParallelFlash Loader with the Quartus II Software".

"Chapter 10: Configuration, Design Security, and Remote System Upgrades in Stratix IV Devices"

Document Revision History

Date	Version	Changes
2012.01.09	First publication	