Introduction

This application note explains the use of the FPGA-based parallel flash loader (PFL) in programming a parallel flash device before configuring an FPGA through the active parallel (AP) configuration scheme.

The AP configuration scheme configures Altera® FPGAs using industry-standard parallel flash devices. For high-density FPGAs, such as the Altera Cyclone® III devices, the parallel flash device reduces the configuration time through the parallel interface and provides a higher memory capacity to store configuration data. However, the parallel flash device does not support the Joint Test Action Group (JTAG) interface, and therefore does not support direct device programming through JTAG.

With the FPGA-based PFL, you can use the FPGA’s JTAG interface to perform in-system programming for the parallel flash device. The PFL enables you to program the flash device indirectly before configuring an Altera FPGA with the AP configuration scheme.

For more information about Cyclone III devices and the supported parallel flash devices, refer to the Configuring Cyclone III Devices chapter in volume 1 of the Cyclone III Device Handbook.

FPGA-Based 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.

Figure 1 shows the PFL interface in relation to the JTAG interface and parallel flash device during flash programming.
To configure the FPGA and program the flash device using the PFL, perform the following first two steps. The third step is optional.

1. **PFL Configuration**

   Configure the FPGA with the PFL to establish a bridge between the JTAG and parallel flash device interfaces. You may bypass this step if the FPGA is already configured with the PFL.

2. **Flash Programming**

   Use the serial programming bitstream from the Quartus® II software, through the JTAG interface, to program the parallel flash device.

3. **Reconfiguration (Optional)**

   After programming the flash, you may choose to reset the FPGA and configure the updated design using the AP configuration scheme.

*Figure 2* shows the flow of programming a flash device using the PFL.
Quartus II Software Support

The Quartus II software provides the Quartus II Programmer tool for you to configure the FPGA or program the parallel flash device or perform both functions together. In the case of a single-step operation, the Programmer will first configure the FPGA before programming the flash device.

To program the flash device, use either one of the following methods:

- “Factory Default PFL”
- “PFL in User Design”

Factory Default PFL
The Quartus II software provides a factory default PFL for flash programming.
Using a factory default PFL eliminates the need to create a PFL design, but you must first configure the FPGA with the PFL before programming the flash device.

Figure 3 shows the programming flow in the Quartus II software for a factory default PFL.

**Figure 3. Programming Flow for Factory Default PFL**

- Add SOF(s) for conversion to POF
- Convert POF for flash device
- Attach POF to the Quartus II Programmer for flash programming
- The Quartus II software automatically enables factory default PFL in Programmer
- Configure PFL
- Program parallel flash device

**PFL in User Design**

Alternatively, you can instantiate the PFL logic in the user design. The PFL programming logic can be generated in the Quartus II software by using the PFL megafunction. If the FPGA already contains the PFL logic in the user design, you do not have to reconfigure the FPGA. This enables the PFL to program the flash without interrupting the operation of the FPGA.
Using the PFL in the Quartus II Software

Figure 4 shows the programming flow in the Quartus II software for a PFL in user design.

Figure 4. Programming Flow for PFL in User Design

Using the PFL in the Quartus II Software

The following three major steps describe how to use the PFL with the Quartus II software. Each step is further explained in the subsections that follow:

1. Instantiation of PFL logic in user design using the PFL megafunction. This procedure also explains the functions of PFL input and output ports.

2. Conversion of the SRAM object file (SOF) that contains FPGA configuration data to a Programmer object file (POF) for parallel flash programming.

3. Programming the parallel flash device using a POF in the Quartus II Programmer.
This application note applies to the Quartus II software version 7.1 and later, but the screenshots shown are captured using the Quartus II software version 7.2.

Instantiation of PFL Logic in User Design Using the Parallel Flash Loader Megafuction

The following procedure describes how to instantiate the PFL logic in user design with the PFL megafuction. You may skip this procedure if you are using a factory default PFL to program the flash device.

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 megafuction variation.
3. Click Next. Page 2a appears (Figure 5).

Figure 5. MegaWizard Plug-In Manager [page 2a] Dialog Box
4. In the megafunction list, expand the JTAG-accessible Extensions folder and select Parallel Flash Loader.

5. In the device family list, select Cyclone III.

6. In the output file list, select the HDL output file type (Verilog HDL was chosen for the example in Figure 5).

7. Specify the output file name as `<project directory>\<file name>`.

8. Click Next. Page 3 appears (Figure 6).

**Figure 6. Setting the PFL Megafunction Parameters**

![Parallel Flash Loader settings](image)
9. Specify values for the PFL megafunction parameters listed in Table 1. For FPGA-based PFL, the Operating Mode parameter settings are grayed out.

<table>
<thead>
<tr>
<th>Table 1. PFL Megafunction Parameter Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Megafuction Parameter</td>
</tr>
<tr>
<td>------------------------</td>
</tr>
<tr>
<td>Flash device</td>
</tr>
<tr>
<td>Flash interface data width</td>
</tr>
<tr>
<td>Tri-state flash bus</td>
</tr>
<tr>
<td>Flash programming IP optimization target</td>
</tr>
<tr>
<td>FIFO size</td>
</tr>
</tbody>
</table>

10. Click Next. Page 4 appears, listing the simulation files needed for the PFL megafuction. No simulation file will be listed for the megafuction because the PFL does not have any simulation files and it cannot be simulated.

11. Click Next. Page 5, the Summary page, appears (Figure 7). This page shows the files that will be created for the megafuction. Choose any additional file types that you want to create.
12. Click Finish. The Quartus II software generates the PFL megafunction in the form of the HDL file selected on Page 2a and any additional files selected on Page 5.

**Input and Output Ports of the Parallel Flash Loader Megafunction**

This section explains the functions of the input and output ports of the PFL megafunction. Figure 8 shows the symbol for the PFL megafunction.
Table 2 describes the functions of the PFL input and output ports.

<table>
<thead>
<tr>
<th>Port</th>
<th>Description</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>pfl_nreset</td>
<td>Input</td>
<td>Reset pin. Pull low to reset PFL.</td>
</tr>
<tr>
<td>pfl_flash_access Granted</td>
<td>Input</td>
<td>Used for system-level synchronization. Pull this active-high pin permanently high if you want to use the PFL as the flash master. Pulling it low prevents JTAG from accessing the flash device.</td>
</tr>
<tr>
<td>plf_flash_access_request</td>
<td>Output</td>
<td>Used for system-level synchronization. Pull this pin high whenever JTAG accesses the flash device.</td>
</tr>
<tr>
<td>flash_addr[23..0]</td>
<td>Output</td>
<td>Connects to PADD[23..0] bus of the FPGA.</td>
</tr>
<tr>
<td>flash_data[15..0]</td>
<td>Output</td>
<td>Connects to DATA[15..0] bus of the FPGA.</td>
</tr>
<tr>
<td>flash_nce</td>
<td>Output</td>
<td>Connects to the flash_nCE pin of the FPGA. A low signal enables the flash device.</td>
</tr>
<tr>
<td>flash_nwe</td>
<td>Output</td>
<td>Connects to the nWE pin of the FPGA. Driving the nWE pin low during write operation indicates to the flash device that data on the DATA[15..0] bus is valid.</td>
</tr>
<tr>
<td>flash_noe</td>
<td>Output</td>
<td>Connects to the nOE pin of the FPGA. Driving the nOE pin low during read operation enables the flash device outputs on the DATA[15..0] bus.</td>
</tr>
</tbody>
</table>
Conversion of SRAM Object File to Programmer Object File for Parallel Flash Devices

To create a POF for the flash device, use the SOF(s) generated from the FPGA device. You can also add other non-configuration data into the POF by selecting the hexadecimal (HEX) file that contains the user data when you create the POF of the flash device.

Perform the following steps to combine multiple SOFs into one POF.

1. On the File menu in the Quartus II software, click **Convert Programming Files**. The **Convert Programming Files** dialog box appears (Figure 9).

   ![Figure 9. Converting Programming Files for Single-Device Configuration Chain](image)

2. In the **Programming file type** list, select **Programmer Object File (.pof)**.
3. In the **Configuration device** list, select the common flash interface (CFI) device with the correct density. For example, CFI_256 denotes a parallel flash memory with 256-Mbit capacity.

4. In the **Mode** list, choose **Active Parallel** for the configuration scheme.

5. In the **File name** box, specify the name of the output file.

Under **Input files to convert**, you can see the **SOF Data** hierarchy expanded to Configuration Master, Low Byte [D[7...0]] and High Byte [D[15..8]].

6. To add the SOF for a single-device configuration chain, select **Configuration Master** and click **Add File**.

7. Select the SOF to add and click **Open**.

For a multi-device configuration chain, you can add more than one SOF into the same page. The order of the SOFs should follow the order of the devices in the chain.

For devices in a byte-wide, multi-device configuration chain, add the SOFs to **Low Byte** [D[7...0]], according to the order of the devices in the chain.

For devices in a word-wide, multi-device configuration chain, add the SOFs to **Low Byte** [D[7...0]] and **High Byte** [D[15..8]], as shown in **Figure 10**, according to the order of the devices connected to the **DATA[7..0]** and **DATA[15..8]** buses respectively.

![Figure 10](image)

Word-wide device configuration is available only in the Quartus II software version 7.2 and later.

If you want to store the data from a SOF in another page, click **Add SOF Data**. A new line for **SOF Data** appears under the **Input files to convert** list. Add the SOF for the new page.
8. To set the page number and page name for the SOF_Data, select the SOF_Data and click Properties. The SOF Data Properties dialog box appears (Figure 11).
9. In the **Address mode for selected pages** list, select **Start**.

10. In the **Start address** box, type `0x20000` to specify the start address using byte addressing.

    The Cyclone III AP configuration scheme configures from the default boot address at `0x20000` using byte addressing or `0x10000` using word addressing. To configure the AP configuration scheme from a different boot address, execute the JTAG instruction `APFC_BOOT_ADDR` to change the boot address of the Cyclone III device.

    For more information on how to use the JTAG instruction, refer to the *Configuring Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*.

11. Click **OK**. The Start Address for the SOF data indicates `0x20000`, as shown in **Figure 12**.
Using the PFL in the Quartus II Software

Figure 12. Start Address at 0x20000 Using Byte Addressing

12. Alternatively, you can also store user data in a HEX file. Perform the following steps to store user data in a HEX file.

   a. Under Input files to convert, click Add Hex Data. The Add Hex Data dialog box appears.

   b. Under Addressing mode, select the addressing mode you require. Turn on Set start address and specify the start address.

   c. In Hex file box, specify the name of the HEX file.

   d. Click OK.

You cannot create the POF of the flash device by using only the HEX file. You must also add in a SOF for the FPGA when creating the POF.
13. Click Generate to create the POF.

**Programming the Parallel Flash Device Using Factory Default PFL**

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 Device. The Select Devices dialog box appears, as shown in Figure 13.

**Figure 13. Adding SOF for PFL**

3. Under Device Name, select the device name to add.

4. Click OK. The device name appears in the Programmer window.

5. Select and right-click the device name you just added and click Attach Flash Device, as shown in Figure 14. The Select Flash Device dialog box appears (Figure 15).
Figure 14. Attaching Flash Device

![Attaching Flash Device Image]

Figure 15. Selecting Flash Device

![Selecting Flash Device Image]
6. Under **Device family**, turn on **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 (Figure 16).

10. Select the POF of the flash device and click **Open**.

    You can only program one flash device in the chain at one time as the Quartus II Programmer only allows you to attach the POF of the flash device to one FPGA in the chain at a time. To program the flash device of another FPGA in the chain, you must delete the flash device POF for the first FPGA and add in the flash device POF for the next FPGA in the chain.
11. Under **Program/Configure** column, turn on the check box for **Page 0** of the POF you added.

The Quartus II Programmer automatically enables a factory default PFL image, as shown in **Figure 17**. To bypass PFL configuration, disable the factory default PFL image by turning off its associated check box under the **Program/Configure** column, as shown in **Figure 18**.

To erase or program the entire flash device, turn on the check box associated with the POF. To erase or program a particular page of the flash device, turn on the check box associated with the page.

---

**Figure 17. Quartus II Programmer Showing Factory Default PFL Image and POF of Flash Device**
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 19.

12. Click **Start** to configure the PFL and program the flash device.

**Programming the Parallel Flash Device Using PFL in User Design**

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 19.
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 20. The Select Flash Device dialog box appears (Figure 21).
Using FPGA-Based Parallel Flash Loader with the Quartus II Software

Figure 20. Attaching Flash Device

Figure 21. Selecting Flash Device
6. Under Device family, turn on 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 (Figure 22).

10. Select the POF of the flash device and click Open.

Figure 22. Adding POF for Flash Programming Device
You can only program one flash device in the chain at one time as the Quartus II Programmer only allows you to attach the POF of the flash device to one FPGA in the chain at a time. To program the flash device of another FPGA in the chain, you must delete the flash device POF for the first FPGA and add in the flash device POF for the next FPGA in the chain.

11. Under Program/Configure column, turn on the check box for the SOF and Page 0 of the POF you just added, as shown in Figure 23. With this set-up, the Quartus II Programmer configures the SOF of the user design with PFL logic first before programming the flash device. To bypass PFL configuration, disable the SOF by turning off its associated check box under the Program/Configure column, as shown in Figure 24.

To erase or program the entire flash device, turn on the check box associated with the POF. To erase or program a particular page of the flash device, turn on the check box associated with the page.

Figure 23. Quartus II Programmer Showing PFL Image of User Design and POF of Flash Device
12. Click **Start** to configure the PFL and program the flash device.

With the Quartus II Programmer, you can program, verify, erase, or blank-check the configuration data pages and user-data page separately, provided the FPGA contains the PFL. You can bypass the PFL configuration step if the FPGA already contains the PFL configuration.

**Conclusion**

The PFL feature available in Altera FPGAs that support the AP configuration scheme 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.
Using FPGA-Based Parallel Flash Loader with the Quartus II Software

Referenced Documents

This application note references the following document:

- *Configuring Cyclone III Devices* chapter in volume 1 of the *Cyclone III Device Handbook*

Document Revision History

Table 3 shows the revision history for this application note.

<table>
<thead>
<tr>
<th>Date and Document Version</th>
<th>Changes Made</th>
<th>Summary of Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>December 2007 v1.0</td>
<td>Initial release.</td>
<td>—</td>
</tr>
</tbody>
</table>