Fast Passive Parallel (FPP) Configuration Mode

Theory

FPP is another configuration scheme where the FPGA device is configured with configuration data stored in external flash. In this scheme, the configuration process is controlled by an external host/microprocessor hence it is known as a passive type of configuration scheme. This allows the microprocessor to change the target FPGA device’s functionality while the system is in operation by reconfiguring it. This scheme is considered the fastest among all the configuration schemes as the data transfer occurs over several data lines (8-, 16- and 32-bits data width) which in turns leads to shorter programming time. The figure below gives a general idea on the relationship of the microprocessor, external flash and the FPGA:

Note: Image taken from Cyclone V handbook. Other devices might require the resistors to be pulled up to different VCC. Please refer to the respective handbook/Pin connection guideline.

Altera provides Parallel Flash Loader (PFL) IP core that is able to program and control FPGA configuration with data from external flash. This walkthrough will focus on the PFL IP’s implementation in Altera’s CPLD which will act as the “brain” to control the configuration process.
Parallel Flash Loader (PFL) IP Core

PFL IP core when instantiated in Altera’s CPLD is capable of programming Common Flash Interface (CFI) flash, quad Serial Peripheral Interface (SPI) flash or NAND flash memory devices via the device JTAG interface. It then controls the FPGA configuration process by facilitating the data transfer process from either one of the flash formats stated above. Below are two figures that show the programming and configuration process respectively.

Note:
- You can use the PFL IP core to either program the flash memory, configure your FPGA or both at the expense of more logic elements (LE).
- For more in depth explanation of the PFL IP core, please refer to the user guide: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/ug/ug_pfl.pdf
Quartus MAX V Design & Settings

1. This is a simple PFL design: File: PFL_MV.qar. Copy and paste this .qar into your project directory. Restore this design by going to Project → Restore Archived Project.

2. Pin assignments are performed in Assignment → Pin Planner. Refer to development kit reference manual for information on pin location if you wish to change pin assignments.

3. Go to Assignment → Device → Device & pin options → Unused Pins. Ensure that the unused pins are reserved as input tri-stated.

4. You can check the settings in the PFL IP core by going to Tools → Qsys and open PFL.qsys.
- FPP x 16 is chosen as the development kit's default MSEL pins setting support FPP x 16.
- This option bit holds the addressing information of the configuration data stored in the CFI flash. 18000 (in hex) is chosen as the CFI flash has been preprogrammed with the option bit stored at 18000. CFI flash memory map obtained from the user guide of the development kit:
### Table A-1. Byte Address Flash Memory Map

<table>
<thead>
<tr>
<th>Block Description</th>
<th>KB Size</th>
<th>Address Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unused</td>
<td>128</td>
<td>0x03FE.0000 - 03FF.FFFF</td>
</tr>
<tr>
<td>User software</td>
<td>28,800</td>
<td>0x023C.0000 - 03FD.FFFF</td>
</tr>
<tr>
<td>Factory software</td>
<td>8192</td>
<td>0x01BC.0000 - 023B.FFFF</td>
</tr>
<tr>
<td>zipfs (html, web content)</td>
<td>4096</td>
<td>0x017C.0000 - 01BB.FFFF</td>
</tr>
<tr>
<td>User hardware 2</td>
<td>8064</td>
<td>0x00FE.0000 - 017B.FFFF</td>
</tr>
<tr>
<td>User hardware 1</td>
<td>8064</td>
<td>0x0080.0000 - 00FD.FFFF</td>
</tr>
<tr>
<td>Factory hardware</td>
<td>8064</td>
<td>0x0002.0000 - 007F.FFFF</td>
</tr>
<tr>
<td>PFL option bits</td>
<td>32</td>
<td>0x0001.8000 - 0001.FFFF</td>
</tr>
<tr>
<td>Board information</td>
<td>32</td>
<td>0x0001.0000 - 0001.7FFF</td>
</tr>
<tr>
<td>Ethernet option bits</td>
<td>32</td>
<td>0x0000.8000 - 000.7FFF</td>
</tr>
<tr>
<td>User design reset vector</td>
<td>32</td>
<td>0x0000.0000 - 000.7FFF</td>
</tr>
</tbody>
</table>

5. Compile the design by going to Processing → Start Compilation to obtain the .pof
Quartus FPGA Design & Settings

1. This is a simple LED design File:FPP.qar. Copy and paste this .qar into your project directory. Restore this design by going to Project → Restore Archived Project.

2. Pin assignments are performed in Assignment → Pin Planner. Refer to development kit reference manual for information on pin location if you wish to change pin assignments.

3. Go to Assignment → Device → Device & pin options → Configuration. Ensure that the correct checkboxes are ticked:

   ![Configuration Options](image)

   Note: - Ensure that the configuration scheme matches with the one set in the PFL IP core.

4. Compile the design by going to Processing → Start Compilation to obtain the .sof

5. You can also open another project revision: led_2 that has a different LED pattern. Compile the design to obtain the other .sof.
Programming file generation

As we are going to program the design into the CFI flash, we need to generate a programming file that is compatible to the CFI flash: programmer object file (.pof)

1. Open the Convert Programming File GUI by going to File → Convert Programming Files.
2. Set as according to the following screenshot:

Note:
- The start address for SOF data is 0x80000 (in hex) as we just want to store this design in the user hardware 1 as shown in the CFI Memory Map table above. The start address can be modified by going to “Properties”.
- Options start address is left blank as only the SOF data is programmed into the CFI flash. If you are working with a blank CFI flash, then you have to choose a location to store the option bit so that the PFL IP knows the addresses of your configuration data. Ensure that the options bits are stored in unused portion of the flash device and the start address resides on a 8-KB boundary. The option bit address can be changed by clicking at “Options/Boot info”.

3. Generate the .pof file by clicking on the generate button.
**Setting up the development kit**

1. Set up the Cyclone V GX development kit according to the factory default settings. You can refer to “Setting up the board” in Chapter 4 of the development kit user guide.
2. The MSEL pins setting are controlled by MAX V design, however as the default pins setting point to FPP x 16 configuration scheme, no additional action is required.

**Demo-Bringing up the design in hardware**

The MAX V will now be programmed with the PFL IP core design created above. We will then use the PFL to program the CFI flash and then control the FPGA configuration with the data from CFI flash.

**Programming PFL design into MAX V**

1. Ensure the USB Blaster is connected correctly to the JTAG port of the Cyclone V GX development kit.
2. In Quartus, go to Tools → Programmer.
3. Configure the hardware setup to point to USB Blaster I/II.
4. Perform “auto-detect” to detect the physical devices in the JTAG chain.
5. Right click on the detected MAX V device and choose “change file”.
6. Browse to the generated .pof and click ok.
7. Tick on “Program/Configure”.
8. Click start to begin the programming process.
9. After completion of the programming process, perform “auto-detect” once again. You should be able to see the CFI flash attached to the MAX V device.

**Note:**
- You will be able to restore the original MAX V design by going to `<installation directory>kits\cycloneVGX_5cgxfc7df31es_fpga\examples\max5 and program the max5.pof.`

**Programming the LED design into CFI flash**

As the CFI flash contains important information, it is advisable to examine out the content of the CFI flash before any flash programming is done so that you will be able to reprogram it back to its original state if any mistake occur. To examine the CFI flash:

1. Tick on “examine” for the CFI flash detected at step 9 (ensure that no other boxes are ticked).
2. Click start and wait for the programmer to examine the CFI content.
3. Right click the examined .pof and choose save. You will be prompted to key in the option bit location (18000). Store this .pof elsewhere for emergency usage.
4. Right click again and choose “change file”.

5. Browse to the LED design .pof and click ok.
6. Tick on “Program/Configure”.

Note:
- Make sure you tick only on “Page_0”. Ticking on the upper box will erase the whole CFI flash and reprogram it with only page_0 and option bit. If the option_bits box is ticked, it will create a mismatch between the option bit address in the PFL IP and the CFI flash thus causing the FPP configuration to fail.

<table>
<thead>
<tr>
<th>File</th>
<th>Device</th>
<th>Checksum</th>
<th>Usercode</th>
<th>Program/ Configure</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;none&gt;</td>
<td>5CGXFC7D6F31ES</td>
<td>00000000</td>
<td>&lt;none&gt;</td>
<td></td>
</tr>
<tr>
<td>K:Kevin/Technical Share...</td>
<td>5M22102F256</td>
<td>009E09C7</td>
<td>009E09AF</td>
<td></td>
</tr>
<tr>
<td>CFM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UFM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>K:Kevi/Technical Share...</td>
<td>CF1_512Mb</td>
<td>0211E52</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Page_0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTION_BITS</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

7. Click start to begin the programming process.
8. Push the dipswitch at SW3.3 to OFF and hit on PGM_Load push button. You will then see the LED design on the board.

Note:
- Pushing the dipswitch to OFF indicate 001 at the PGM input of the PFL IP core. This allows the PFL IP to read the configuration data at User Hardware 1 and configuration is triggered by the PGM_Load push button. You can include your own design at the Factory hardware by changing the address in the section IV above. You can then trigger reconfiguration by pushing the dipswitch to ON (000) and hit the PGM_Load push button. You can then restore the CFI flash content with the .pof examined out earlier.