

Using FPGA-Based Parallel Flash Loader with the Quartus II Software

December 2007, ver. 1.0

Application Note 478

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.



PFL Programming	To o per	configure the FPGA and program the flash device using the PFL, form the following first two steps. The third step is optional.
Flow	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.
	Fig	ure 2 shows the flow of programming a flash device using the PFL.

Figure 1. PFL Interface During Flash Programming





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



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.

Figure 4 shows the programming flow in the Quartus II software for a 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 Megafunction

The following procedure describes how to instantiate the PFL logic in user design with the PFL megafunction. 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 megafunction variation**.
- 3. Click Next. Page 2a appears (Figure 5).

Figure 5. MegaWizard Plug-In Manager [page 2a] Dialog Box

MegaWizard Plug-In Manager [page 2a]		
Which megafunction would you like to customize? Select a megafunction from the list below	Which device family will you be using? Which type of output file do you want AHDL VHDL Verilog HDL What name do you want for the output D:work/tpga_pfl.v Return to this page for another or Note: To compile a project successfu your design files must be in the project ibraries specified in the User Libraries p box (Assignments menu). Your current user library directories an	Cyclone III
	Cancel < Ba	ck Next > Finish

- 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

沟 🔹 Parallel Flash Lo	ader		
Parameter 2 EDA 3 Summary		_	About Documentatio
Settings			
	Cura	rently selected device family	Cyclope III
fpga_pfl			Match project/defaul
pfl_nreset pfl_flash_access_request →			
pfl_flash_access_granted	What operating mode(s) will be used?	Flash Programming	~
flash_addr[230]	Which flash device will you be using?	CFI 256 Mbit	
flash_data[150] → flash_nce →	What is the flash interface data width?	16 bits 🔽	
flash_nwe → flash_noe →	Set flash bus pins to tri-state when not in use	2	
	Flash Programming		
	Flash programming IP optimization target	Speed 🔽	
	Flash programming IP FIFO size	16 🗸	words
	FPGA Configuration		
	What is the external clock frequency?		MHz
	What is the flash access time?	100	ns
	What is the byte address of the option bits, in hex?	0x0	
	Which FPGA configuration scheme will be used?	PS (passive serial)	~
	What should occur on configuration failure?	Halt	~
	What is the byte address to retry from on failure?		
	Include input to force reconfiguration		
	Ratio between input clock and DCLK output?	1 🗸	
	Use burst mode reads to reduce configuration time (Intel P30 or P33 ONLY)		

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 1. PFL Megafunction Parameter Se	ttings			
Megafunction Parameter	Description			
Flash device	Density of the device to be programmed.			
Flash interface data width	Data width of the device to be programmed.			
Tri-state flash bus	Tri-state all pins interfacing with the flash device when the PFL does not need to access the flash device.			
Flash programming IP optimization target	The flash programming IP can be optimized for speed or area. An IP optimized for speed means that the time required for flash programming is shorter, but the megafunction uses more logic elements. An IP optimized for area means that the IP requires less logic elements, but the time required for flash programming is longer.			
FIFO size	If the flash programming IP is optimized for speed, the PFL uses additional logic elements to implement FIFO as a temporary storage for the programming data used during flash programming. There is a 16-word or 32-word option for the FIFO size. With a larger FIFO size, the programming time is shorter.			

- 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**. Page 5, the Summary page, appears (Figure 7). This page shows the files that will be created for the megafunction. Choose any additional file types that you want to create.



Figure 7. PFL Megafunction Summary

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.

Figure 8. PFL Megafunction Symbol



Table 2 describes the functions of the PFL input and output ports.

Table 2. Functions of the PFL Inpu	it and Output P	orts
Port	Description	Function
pfl_nreset	Input	Reset pin. Pull low to reset PFL.
pfl_flash_access_granted	Input	Used for system-level synchronization. This pin can be driven by an external host that controls access to the flash device. 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.
plf_flash_access_request	Output	Used for system-level synchronization. This pin can be connected to an external host if needed. The PFL drives this pin high whenever JTAG accesses the flash device.
flash_addr[230]	Output	Connects to PADD [230] bus of the FPGA.
flash_data[150]	Output	Connects to DATA [150] bus of the FPGA.
flash_nce	Output	Connects to the flash_nCE pin of the FPGA. A low signal enables the flash device.
flash_nwe	Output	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 [150] bus is valid.
flash_noe	Output	Connects to the nOE pin of the FPGA. Driving the nOE pin low during read operation enables the flash device outputs on the DATA [150] bus.

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.

 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

Convert Programm	ing Files			
Specify the input files to co You can also import input f future use.	onvert and the type of pro ile information from other l	gramming file to genera files and save the conv	ate. version setup information o	created here for
Conversion setup files				
Open Con <u>v</u>	ersion Setup Data		Save Conversion Setup	
Output programming file				
Programming file type:	Programmer Object File	e (.pof)		•
Options	Configuration device:	CFI_256MB 💌	Mode: Active Parallel	-
File <u>n</u> ame:	/files/256MB_Flash.p	of		
	Remote/Local update of	lifference file: NONE		-
	🔽 Memory Map File			
Input files to convert				
File/Data area		Properties	Start Address	Add <u>H</u> ex Data
SOF Data	41	Page_0	<auto></auto>	
i i 3C25 De:	master sign.sof	EP3C25F324		Add Sof Data
High Byte (D[1	58])			Add <u>File</u>
····· Low Byte (D[7.	U]J			Remove
				Un
				Down
				<u>P</u> roperties
			<u>G</u> enerate	Close

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.



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.

Figure 10. Converting Programming Files fo	[•] Word-Wide Multi-Device	Configuration Chain
--	-------------------------------------	---------------------

🔓 Convert Programm	ing Files			
Specify the input files to c You can also import input future use.	onvert and the type of pro file information from other f	gramming file to generate. iles and save the conversion	setup information o	created here for
Conversion setup files—				
Open Cony	ersion Setup Data	<u>S</u> ave	e Conversion Setup	
Output programming file				
Programming file type:	Programmer Object File	; (.pof)		•
Options	<u>Configuration device</u> :	CFI_256MB	e: Active Parallel	•
File <u>n</u> ame:	D:/temp/files/256MB_	Flash_Multi.pof		
	Remote/Local update o	lifference file: NONE		Y
Input files to convert-				
File/Data area		Properties	Start Address	Add <u>H</u> ex Data
SOF Data	Master sign 1 sof	Page 0 EP3C25E324	0x00020000	Add <u>S</u> of Data
⊡High Byte (D[1 3C25_De	(58]) sign_3.sof	EP3C25F324		Add <u>F</u> ile
⊡… Low Byte (D[7	0]) .sian 2 sof	EP3025E324		<u>R</u> emove
		2. 00201021		Цр
				Down
				Properties
			<u>G</u> enerate	Close

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).

✓ 0 1 1 2 3 4 5 6 7 Selected pages comment: Page_0 Address mode for selected pages Start Start Start Start Start Start Start Start Start Ox20000 End address (32-bit hexadecimal): OxFFFFFFF	Pages	
2 3 4 5 6 7 Selected pages Address mode for selected pages Start Start Start address (32-bit hexadecimal): 0x20000 End address (32-bit hexadecimal): 0xFFFFFFF	₩ 0	
4 5 6 7 Selected pages comment: Page_0 Address mode for selected pages Start Start Start address (32-bit hexadecimal): 0x20000 End address (32-bit hexadecimal): 0xFFFFFFF		
☐ 6 ☐ 7 Selected pages comment: Page_0 Address mode for selected pages Start ▼ Start ▼ Start address (32-bit hexadecimal): 0x20000 End address (32-bit hexadecimal): 0xFFFFFFFF		
Selected pages comment: Page_0 Address mode for selected pages Start Start Start Cx20000 End address (32-bit hexadecimal): 0xFFFFFFF	□6 □7	
Selected pages comment: Page_0 Address mode for selected pages Start Start address (32-bit hexadecimal): 0x20000 End address (32-bit hexadecimal): 0xFFFFFFF		
Address mode for selected pages Start Start address (32-bit hexadecimal): 0x20000 End address (32-bit hexadecimal): 0xFFFFFFF 0xFFFFFFFF 0xFFFFFFFF	Selected pages comment: Page_0	
Start Start address (32-bit hexadecimal): 0x20000 End address (32-bit hexadecimal): 0xFFFFFFF	Address mode for selected pages	
Start address (32-bit hexadecimal): 0x20000 End address (32-bit hexadecimal): 0xFFFFFFFF	Start	•
End address (32-bit hexadecimal): 0xFFFFFFF	Start address (32-bit hexadecimal):	0x20000
	End address (32-bit hexadecimal):	0xFFFFFFFF

Figure 11. SOF Data Properties Dialog Box

- 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.

Figure 12. Start Address at 0x20000 Using Byte Addressing

🔓 Convert Programm	ing Files				
Specify the input files to co You can also import input f future use.	onvert and the type of pro- ile information from other f	gramming file to genera files and save the con-	ate. version se	etup information c	reated here for
Conversion setup files—					
Open Con <u>v</u>	ersion Setup Data		<u>S</u> ave C	Conversion Setup	
Output programming file Programming file	Programmer Object File	e (.pof)			
Options	<u>C</u> onfiguration device:	CFI_256MB V	Mode:	Active Parallel	
File <u>n</u> ame:	/files/256MB_Flash.p	of		,	
	Remote/Local update o	difference file: NONE			
	🔽 Memory Map File	,			
Input files to convert					
File/Data area		Properties		Start Address	Add <u>H</u> ex Data
SOF Data	faster			0x00020000	Add <u>S</u> of Data
High Byte (D[1	58]) 0])	EI 30231 324			Add <u>File</u>
					<u>R</u> emove
					Цр
					Down
					<u>Properties</u>
				<u>G</u> enerate	Close

- 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

Hardware Setup USB-Blaster [USI Enable real-time ISP to allow backgrour	3-0] nd programming (for MAX II devices)		M	ode: JTAG		2	 Progress 	s:	0%	
No Start	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase	ISP CLAMP
Mi Stop Auto Detect Charge File Charge Down		I Devices ince family ACEX1K ACEX1K ACEX1K APEX20K APEX20K APEX20K APEX20K Cyclone II Cyclone II Cyclone II Enhanced Configurati EPC1 EPC2 EPC1 EPC2 EPC1 EPC2 EPC2 EPC1 EPC2 EPC1 EPC2 EPC1 EPC3 ACE ACTORA ACTORAC	on Devices	Device PP PP PP PP PP PP PP PP PP P	name 3C10 3C10E144 3C10U256 3C120 3C120F38 3C25E144 3C25F324 3C25F484 3C55F484 3C55F484 3C55F486 3C55L484 3C55F280 3C55E144 3C55L48		OK	New Import. Export. Edit Remov Check Unchec	×	

- 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).

Chain1.cdf*								
🛓 Hardware Setu	IDSB-Blaster [USB-0]	Mode: JTAG		Progres	s:	0%	
Enable real-time	ISP to allow background p	programming (for MAX II devices)						
🌇 Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examin
Stop	<none></none>	EP3C25	00000000	<none></none>				
Auto Dataat	J 1		Delete	Del				
Auto Detect]		Add File					
Delete			Change File					
Add File	1		Add IPS File					
Change File			Change IPS File					
			Delete IPS File					
y Save File]		Attach Flash Devic	:e				
Add Device			Delete Flash Devic	ice •				
ա Աթ	1		Add Device					
Down			Change Device.					
			Up	Alt+Up /	Arrow			
			Down	Alt+Dow	n Arrow			
			ISP CLAMP State B	Editor				
			Properties					

Figure 14. Attaching Flash Device

Figure 15. Selecting Flash Device

ASC devices	CFI_128MB CFI_16MB MCFI_256MB CFI_32MB CFI_312MB CFI_512MB CFI_64MB CFI_64MB CFI_64MB	New Import Export Edit Remove Check Uncheck
-------------	---	---

- 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).

Figure 16. Adding POF for Flash Programming Device

Start File	Device	Checksum	Usercode	Program/ Configure	Verify Blank Check	Examine Ser
Stop <a>	EP3C25	00000000	FFFFFFF			
Auto Detect	LFI_236MB					
Delete	Select New Progra	amming File				X
Delete	Look in:) files		•	+ E 🕂 📼	1.
Add File						·
Change File		256MB_Flash.pot				
Save File	My Recent					
	Documents					
Add Device						
Up	Desktop					
Down						
	My Documents					
	-					
	My Computer					
	My Network Fi	e name: 256	MB_Flash.pof		-	Open N

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

17

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

🖺 Chain1.cdf*										
🔔 Hardware Setup	D USB-Blaster [USB-0]			Mode: JTAG		-	Progres	is:	0%	
🔲 Enable real-time I	SP to allow background programming (for	MAX II devices)								
🏓 Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase
🖬 Stop	Factory default PFL image D:/temp/files/256MB_Flash.pof	EP3C25 CFI_256MB	00000000	FFFFFFF						
Auto Detect	LPage_0									
X Delete										
🍅 Add File										
ピ Change File										
🗳 Save File										
😂 Add Device										
📫 Up										
Down	<									>



🖺 Chain1.cdf*									_	
🔔 Hardware Setup	USB-Blaster [USB-0]			Mode: JTAG		-	Progres	is:	0%	
🔲 Enable real-time IS	SP to allow background programming (for	MAX II devices)								
🟓 Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Security Bit	Erase
Stop.	<none></none>	EP3C25	0000000	FFFFFFF						
	D:/temp/files/256MB_Flash.pof	CFI_256MB	00000000							
Auto Detect	···Page_u				⊻					
X Delete										
🍰 Add File										
👺 Change File										
Save File										
Add Device										
🕐 Up										
Down	<									>

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.

Figure 19. Adding SOF for PFL

chain.cdf*										
Hardware Setup	USB-Blaster [USB-0]			Mode	JTAG		Progress	s:	0%	
Enable real-time I	SP to allow background program	nming (for MAX II devi	ices)		,					
A Start	File	Device	Che	cksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine	Secur Bit
th Stop										
Auto Detect		Select Program	ming File							X
✓ Delete		Look in:	🗀 files			•	+ 🔁	d 🕅	•	
Add File 📐		À	user_des	ign.sof						
Change File		My Recent								
Save File										
Add Device		Desktop								
lu Up										
Down		Mu Documents								
		-								
		My Computer								
				11				22		
		My Network Places	File name:	use	er_design.sof			-	Ope	n
			Files of type:	Pro	ogramming Files	(*.sof;*.pof;*.jam;	*.jbc;*.ekp	* 💶	Can	cel
	<		100							

- 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).

Figure 20. Attaching Flash Device

🖺 chain.cdf*					. 🗆 🗙				
🔔 Hardware Setup	USB-Blaster [USB-0]								
Mode:	JTAG				•				
Progress:		0%							
Enable real-time ISP to allow background programming (for MAX II devices)									
🏓 Start	File	Device	Checksum	Usercode Program/ Configure	Verify				
🖬 Stop	/files/user_design.sof	EP3C25F324	0015045F	FFFFFFF 🔽					
Auto Detect			Delete	Del					
× Delete	Add File Change File								
F Add File			Save File						
Change File			Change IPS File.						
			Delete IPS File		_				
Save File			Attach Flash Dev Change Flash De	vice					
Add Device			Delete Flash Dev	/ice					
1 th Up			Add Device						
🜵 Down					_				
			Down	Alt+Down Arrow					
			ISP CLAMP State	e Editor					
	<		Properties		>				

Figure 21. Selecting Flash Device

Uevice raming I∏ASC devices Ø Flash Memory	Device name CF_128MB CF_16MB CF_16MB CF_255MB CF_32MB CF_512MB CF_512MB CF_512MB CF_64MB CF_8MB	New Import Export Edit Remove Check Uncheck
--	---	---

- 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).

Figure 22. Adding POF for Flash Programming Device

🖺 chain.cdf*						
🔔 Hardware Setup	D USB-Blaster [USB-0]		Mode: JTAG		✓ Progress:	0%
Enable real-time I	SP to allow background programm	ing (for MAX II devices)				
🏓 Start	File	Device	Checksum	Usercode	Program/ Configure	/erify Blank- Examine
Stop	/files/user_design.sof	EP3C25F324 CEL 256MB	0015045F	FFFFFFF		
Auto Detect	Select	t New Programming	File		-	
X Delete		Look in: 🗀 files			★ ← €	
Add File Change File Save File Add Device Down Down	My Do Du My D My U My T	Recent cuments resktop rocuments rocuments rocuments rocuments rocuments resktop rocuments rocum	Flash.pof	*.pof)		Open Cancel
	<	101				

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 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

🖺 chain.cdf*							[
🔔 Hardware Setup.	USB-Blaster [USB-0]		Mode: JTAG		Progress:		0%	
Enable real-time IS	6P to allow background programming (for	MAX II devices)						
🏴 Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine
📲 Stop	/files/user_design.sof /files/256MB_Flash.pof	EP3C25F324 CFI_256MB	0015045F I 0015045F	FFFFFFF	>			
Auto Detect	LPage_0							
🗙 Delete								
🍰 Add File								
🞬 Change File								
Save File								
😂 Add Device								
The Up								
Down	<							>

Figure 24. Disabling PFL Image in User Design

🖺 chain.cdf*								
🔔 Hardware Setup	USB-Blaster [USB-0]		Mode: JTAG		▼ Progress	:	0%	
🔲 Enable real-time IS	SP to allow background programming (fo	r MAX II devices)						
🏴 Start	File	Device	Checksum	Usercode	Program/ Configure	Verify	Blank- Check	Examine
-We Shop	/files/user_design.sof	EP3C25F324	0015045F	FFFFFFF				
The stop	/files/256MB_Flash.pof	CFI_256MB	0015045F					
Auto Detect	L Page_0				✓			
X Delete								
🚵 Add File								
👺 Change File								
🕒 Save File								
😂 Add Device								
🕐 Up								
🐌 Down								
		1111		J		_		>

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.

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 3. Document Revision History						
Date and Document Version	Changes Made	Summary of Changes				
December 2007 v1.0	Initial release.	—				



101 Innovation Drive San Jose, CA 95134 www.altera.com Technical Support: www.altera.com/support/ Literature Services: literature@altera.com Copyright © 2007 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make change es to any products and services at any time without notice. Altera assumes no responsibility or liability

arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

