Partial Reconfiguration of a Processor Peripheral Tutorial

PlanAhead Design Tool

UG744 (v 13.4) January 18, 2012





The information disclosed to you hereunder (the "Information") is provided "AS-IS" with no warranty of any kind, express or implied. Xilinx does not assume any liability arising from your use of the Information. You are responsible for obtaining any rights you may require for your use of this Information. Xilinx reserves the right to make changes, at any time, to the Information without notice and at its sole discretion. Xilinx assumes no obligation to correct any errors contained in the Information or to advise you of any corrections or updates. Xilinx expressly disclaims any liability in connection with technical support or assistance that may be provided to you in connection with the Information. XILINX MAKES NO OTHER WARRANTIES, WHETHER EXPRESS, IMPLIED, OR STATUTORY, REGARDING THE INFORMATION, INCLUDING ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT OF THIRD-PARTY RIGHTS.

© Copyright 2012 Xilinx, Inc. XILINX, the Xilinx logo, Virtex, Spartan, ISE, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. All other trademarks are the property of their respective owners.

Revision History

Date	Version	Revision
07/06/2011	13.2	Revalidated for the 13.2 release. Editorial updates only; no technical content updates.
10/19/2011	13.3	Updated the tutorial to use an AXI4-based design.
1/18/2012	13.4	Revalidated for the 13.4 release. Editorial updates only; no technical content updates.

The following table shows the revision history for this document.

Table of Contents

Revision History	2
Partial Reconfiguration of a Processor Peripheral Tutorial	4
Introduction	4
Tutorial Objectives	4
Getting Started	4
Tutorial Steps	8
Step 1: Creating a Processor Hardware System	9
Step 2: Creating a Software Project	13
Step 3: Creating a PlanAhead Project	16
Step 4: Defining a Reconfigurable Partition	21
Step 5: Adding Reconfigurable Modules	23
Step 6: Defining the Reconfigurable Partition Region	
Step 7: Running the Design Rule Checker	
Step 8: Creating the First Configuration, Implementing, and Promoting	
Step 9: Creating Other Configurations, and Implementing	
Step 10: Running Partial Reconfiguration to Verify Utility	35
Step 11: Generating Bit Files	
Step 12: Creating an Image, and Testing	
Conclusion	
Additional Resources	
Xilinx Resources	
Partial Reconfiguration Documentation	
PlanAhead Documentation	



Partial Reconfiguration of a Processor Peripheral Tutorial

Introduction

This tutorial shows you how to develop a partial reconfiguration design using the Xilinx® Platform Studio (XPS), Software Development Kit (SDK), and the PlanAhead[™] design tool. You will use XPS to create a processor hardware system which includes a lower-level module defining one Reconfigurable Partition (RP) and two Reconfigurable Modules (RMs). The two RM perform addition and multiplication functions. You will use SDK to create a software application which enables you to perform partial reconfiguration.

XPS and SDK are part of the Embedded Design Kit (EDK), which is included in the ISE® Design Suite Embedded and System Editions.

You will use PlanAhead to:

- Floorplan the design including defining a reconfigurable partition for the reconfigurable region
- Create multiple configurations and run the partial reconfiguration implementation flow to generate full and partial bitstreams.

You will use the ML-605 evaluation board to verify the design in hardware using a Compact Flash (CF) memory card to configure the FPGA device initially and then partially reconfigure the device using the AXI HWICAP peripheral by loading the partial bitstream files stored on the CF under the user software control.

This tutorial covers only a subset of the features contained in the PlanAhead tool bundled with ISE Design Suite Release. Other features are covered in other tutorials.

Tutorial Objectives

After completing this tutorial, you will be able to:

- Generate a processor system using XPS and SDK.
- Use the Partial Reconfiguration design flow capability in PlanAhead to generate full- and partialbitstreams to dynamically reconfigure an FPGA design using the AXI HWICAP peripheral.

Getting Started

Software Requirements

The PlanAhead tool is installed with the ISE Design Suite 13.4 software. For this tutorial, you must have the Embedded or System edition of the ISE Design Suite installed. Before starting the tutorial, ensure that the software is operational and the reference design is unzipped and installed.

E XILINX_{*}

For PlanAhead installation instructions and information, refer to *the ISE Design Suite 13: Installation, and Licensing Guide* on the Xilinx website:

http://www.xilinx.com/support/documentation/sw_manuals/xilinx13.4/iil.pdf

You must obtain a FlexLM license for Partial Reconfiguration to access the Partial Reconfiguration features. Contact your Xilinx Field Applications Engineer, or go to the Xilinx website at: http://www.xilinx.com/getproduct

Hardware Requirements

Xilinx recommends a minimum of 2 GB of RAM for use with this design for best performance.

Optionally, you can use an ML605 board and a USB download cable to test the hardware.

Locating the Tutorial Design Files

This tutorial uses a reference design, UG744_design_files.zip, which must be unzipped to a directory on your machine. Please note that the directory path you choose should not have a space in its name. You can download a copy of the reference design from the Xilinx website:

http://www.xilinx.com/support/documentation/dt_planahead_planahead13-4_tutorials.htm

Understanding the Processor System

This tutorial demonstrates how to implement a design that can be dynamically reconfigured using the AXI HWICAP peripheral.

The following figure shows a processor system. The design consists of a peripheral capable of performing a math function, having two unique capabilities: addition and multiplication.

You will verify the functionality with HyperTerminal under user application control. The dynamic modules are reconfigured using the AXI HWICAP peripheral.



Figure 1: Top-Level Design

Project Directory Structure

The directory structure is:



Figure 2: The Project Directory

- The edk\ directory is used to create a processor system.
- The resources \ directory contains:
 - Source files used to generate the netlists of the addition and multiplication functions,
 - A pre-compiled netlist for the addition and multiplication functions in Math and associate sub-directories, and
 - A software application to demonstrate the functionality.
- The math processor core (pcore) that:
 - Provides necessary processor bus connections
 - Provides the required peripheral services (in this case, one slave register and a software reset)
 - Is a placeholder for the math functionality module
- The image\ directory is used to hold the generated full configuration bitstream file in the System ACETM format and partial bitstream files.
- The image_solution\ directory contains the final system.ace and partial bit files for a quick test.

Tutorial Steps

This tutorial is separated into steps, followed by general instructions and supplementary detailed steps allowing you to make choices based on your skill level as you progress through the lab.

- Step 1: Creating a Processor Hardware System
- Step 2: Creating a Software Project
- Step 3: Creating a PlanAhead Project
- Step 4: Defining a Reconfigurable Partition
- Step 5: Adding Reconfigurable Modules
- Step 6: Defining the Reconfigurable Partition Region
- Step 7: Running the Design Rule Checker
- Step 8: Creating the First Configuration, Implementing, and Promoting
- Step 9: Creating Other Configurations, and Implementing
- Step 10: Running Partial Reconfiguration to Verify Utility
- Step 11: Generating Bit Files
- Step 12: Creating an Image, and Testing

Step 1: Creating a Processor Hardware System

Creating a Processor System Using the Base System Builder (BSB) Wizard in XPS

- 1. Select **Start > Programs > Xilinx Design Suite 13.4 > EDK > Xilinx Platform Studio** to open XPS.
- 2. In the Getting Started page, click **Create New Project Using Base System Builder** to open a **Create New XPS Project using BSB Wizard** dialog box.
- 3. Browse to the reconfig_peripheral_lab\edk\ directory.
- 4. Click Save.
- 5. Keep the default options of using ISE tools and AXI System as the interconnect type, and click **OK**.

You will create a system for a Virtex®-6 ML605 evaluation platform.

- 6. In Board and System Selection form, select **Xilinx** as a Board Vendor.
- 7. In Board Name field, select Virtex-6 ML605 Evaluation Platform.
- 8. In Board Revision field, select **D**.
- 9. Click **Next** with other default options selected.
- 10. Select **50.00 MHz** from the Processor Frequency drop-down menu.
- 11. Select **64 KB** from the Local Memory Size drop-down menu.
- 12. In the selected peripherals list on the right, remove all devices *except*:
 - RS232_Uart_1
 - SysACE_CompactFlash
- 13. Click **RS232_Uart_1** and configure it with a baud rate of **115200**.
- 14. Click Finish.
- 15. If the Next Step dialog box opens, click **OK** to start using Platform Studio and open the **System Assembly View** window as shown in the following figure.

Step 1: Creating a Processor Hardware System

ALL	Bus Interfaces	Ports	Addresses			
XMM	Name		Bus	Name	IP Type	IP Version
	axi4lite_0 microblaze_0_0 microblaze_0_0 microblaze_0_0 microblaze_0_0 microblaze_0_0 microblaze_0_0 debug_module SysACE_Comp RS232_Uart_1 clock_generate proc_sys_reset	dimb ilmb bram_blo d_bram_c i_bram_ct e actFlash or_0 _0	ck trl rl		 ☆ axi_interco ☆ Imb_v10 ☆ Imb_v10 ☆ microblaze ☆ bram_block ☆ Imb_bram_i ☆ Imb_bram_i ☆ mdm ☆ axi_sysace ☆ axi_uartlite ☆ clock_gene ☆ proc_sys_re 	1.05.a 2.00.b 2.00.b 8.20.b 1.00.a 3.00.b 3.00.b 2.00.b 1.01.a 1.02.a 4.03.a 3.00.a

Figure 3: Displaying the System Assembly View

Adding the Required IPs to the Processor System

1. Copy the <code>reconfig_peripheral_lab/resources/math_v1_00_a/</code> folder to the <code>reconfig_peripheral_lab/edk/pcores/</code> folder.

Partial Reconfiguration Design Details

Examine the user_logic.vhd file located in

reconfig_peripheral_lab\resources\math_v1_00_a\hdl\vhdl\. It declares a component that will be used in reconfigurable partition at line 133. The same is instantiated at line 158. The data inputs to the component are clocked at lines starting at 191. The reset input to the component is a combination of the hardware bus reset and software reset. The software reset is generated by a soft_reset block located at line 310 in math.vhd file located in the same directory. The software reset is necessary to reset the reconfigured logic after reconfiguring the partition.

Note: If line numbering is hidden from view in XPS, turn line numbers on as follows:

1. Select Edit > Preferences > ISE Text Editor.

- 2. Click to select the **Show line numbers** check box.
- 3. Click **Apply** and then **OK**.
- 2. Rescan the User Repositories in XPS by selecting **Project > Rescan User Repositories**.

In the IP Catalog tab, MATH displays in the $\tt USER$ folder under the <code>Project Local pcores</code> folder.

- 3. Expand the USER folder.
- 4. Select MATH.
- 5. Double-click **MATH** to add an instance of the IP to the System Assembly.
- 6. A properties form will be displayed. Click **OK** twice to add the IP with the default settings and connect it to the microblaze_0 instance.



7. In the IP Catalog tab, select the FPGA Internal Configuration Access Port (v2.01.a) IP (axi hwicap) under the FPGA Reconfiguration folder, right-click and select Add IP.

This adds the instance of the IP to the System Assembly View.

8. Click **OK** twice to accept the default settings and connect the IP to the microblaze 0 instance.

Note that the IP cores are added, interface connections are made, and the addresses are automatically assigned.

Connecting the Ports

- 1. In the System Assembly View, select the **Ports** tab.
- 2. Expand the axi hwicap 0 instance.
- 3. Select Hardware > Launch Clock Wizard.
- 4. In the Clock Wizard form, select **clk_50_0000MHz** for the ICAP_Clk of the axi_hwicap_0 instance and select <AUTO> under the source column, and click **OK**.

A warning message will appear.

5. Click **OK** to close the form.

The connection appears as shown in Figure 4.

•	Bus Interfaces	Ports	Addresses		
N	lame			Connected Port	Direction
E	External Ports				
e	axi4lite_0				
E	microblaze_0_	dlmb			
Ð	microblaze_0_	ilmb			
E	microblaze_0				
	microblaze_0_	bram_blo	ck		
E	microblaze_0_	d_bram_c	trl		
E	microblaze_0_	i_bram_ct	rl		
E	debug_modul	е			
E	axi_hwicap_0				1.2
	ICAP_Clk			clock_generator_0::CLKOUT0	/ I
	IP2INTC_Ir	pt			/ 0
	(BUS_IF) S	IXA_		Connected to BUS axi4lite_0	-
Ð	SysACE_Comp	actFlash			
E	RS232_Uart_1				
Đ	math_0				
E	clock_generate	or_O			
0	proc_sys_reset	_0			

Figure 4: Connecting Clock Source to ICAP

Partial Reconfiguration Design Details

The axi_hwicap pcore allows separate clock domain for the hwicap so it can be run at 100 MHz when the system is run at a higher speed. In this tutorial, the system clock is 50.00 MHz and hence, we are running the entire design in a single clock domain.



Generating Netlists

1. To run the Platform Generator, select **Hardware > Generate Netlist**.

This generates the peripheral and system netlists, and the system.bmm files, all of which are used during implementation in the PlanAhead tool.

Step 2: Creating a Software Project

Once the hardware netlist is generated, use the Software Development Kit (SDK) available with EDK to:

- Create a software project
- Import the provided source files
- Compile the provided source file
- Generate an executable file

Exporting Hardware Design to SDK, and Creating a Board Support Package

Be sure to add xilfatfs library support.

- 1. In XPS, select **Project > Export Hardware Design to SDK** to launch SDK.
- 2. Uncheck Include bitstream and BMM file.
- 3. Click Export & Launch SDK.

A workspace location dialog box will appear.

- 4. Browse to the reconfig_peripheral_lab\edk\SDK\SDK_Workspace directory, and click OK to open SDK after importing hardware specification of the system.
- 5. In SDK, select **File > New > Xilinx Board Support Package**.

Notice that the default Project Name is Standalone_bsp_0 and the OS is Standalone.

6. Click **Finish** with default settings.

The Board Support Package Settings window opens.

7. Check the **xilfatfs** check box to select the FAT file system support for the Compact Flash card.

	Name	Version	Description
	lwip140	1.00.a	IwIP TCP/IP Stack library: IwIP v1.4.0, Xilinx adapter v1.00.a
1	xilfatfs	1.00.a	Provides read/write routines to access files stored on a FAT16/32 file system
	xilflash	3.00.a	Xilinx Flash library for Intel/AMD CFI compliant parallel flash
	xilisf	2.04.a	Xilinx In-system and Serial Flash Library
	xilmfs	1.00.a	Xilinx Memory File System

Figure 5: Selecting File System Support

8. Click **OK** to accept the settings and close the form.

Creating a Xilinx C Project

- 1. Select File > New > Xilinx C Project.
- 2. Type **TestApp** for the Project Name.
- 3. Select **Empty Application** in the Project Application Template pane.
- 4. Click **Next**.

- 5. Select Target an Existing Board Support Package.
- 6. Click Finish.

Generating a Test Application

- 1. In the Project Explorer view, select **TestApp**.
- 2. Right-click and select Import.
- 3. Double-click General.
- 4. Double-click File System.
- 5. Browse to the reconfig_peripheral_lab\resources\TestApp\src\ folder.
- 6. Click OK.
- 7. Select main.c and xhwicap_parse.h.
- 8. Click Finish.

This compiles the source files and generates <code>TestApp.elf</code> in the <code>reconfig_peripheral_lab\edk\ TestApp\Debug\</code> folder.

Partial Reconfiguration Design Details

Examine the reconfig peripheral lab\resources\TestApp\src\main.c file.

This code includes a function, beginning on line 164, which loads a partial bit file from the CompactFlash and writes to the ICAP.

The calls to this function, beginning on line 433, instruct the program to load a specific partial bit file and then assert software reset.

When the blank bitstream is loaded, the software reset is not required since there is no real logic residing in the reconfigurable region.

Generating a Linker Script

Be sure that the Heap and Stack sizes are set to 2048 (0x800).

- 1. In SDK Project Explorer view, select **TestApp**.
- 2. Right-click and select Generate linker script.
- 3. Change the Heap size and the Stack size to **2048**.

Step 2: Creating a Software Project

Generate a linker script				-	Stational Springho	subjective Property and States of States	X
Generate linker script							2
Control your application's me	mory map.						1
Project: TestApp Output Script: peripheral_lab\edk\SDK\SDK Modify project build settings Set generated script on all pr	_Export\TestAp as follows: oject build conf	p\src\ls	cript.ld	Browse	Basic Advanced Place Code Sections in: Place Data Sections in: Place Heap and Stack in: Heap Size:	microblaze_0_i_bram_ctrl_microblaze_0_d_bram_ctrl microblaze_0_i_bram_ctrl_microblaze_0_d_bram_ctrl microblaze_0_i_bram_ctrl_microblaze_0_d_bram_ctrl 2 KB	•
Memory	Base Address	Size		1	Stack Size.	Z ND	
microblaze_0_i_bram_ctrl	0x00000000	64					
Fixed Section Assignments							
3						Generate Canc	el

Figure 6: Generating a Linker Script

- 4. Click **Generate**.
- 5. Click **Yes** to overwrite the existing copy and recompile the application again.
- 6. Select **File > Exit** to close SDK.

Step 3: Creating a PlanAhead Project

Now that you have generated the required netlist files for the design, you will use the PlanAhead tool to:

- Floorplan the design
- Define reconfigurable partitions
- Add reconfigurable modules
- Run the implementation tools
- Generate full and partial bitstreams

In this step, you will create a new project.

Creating a PlanAhead Project, and Importing the Generated Netlist Files

- 1. To open PlanAhead, select **Start > Programs > Xilinx ISE Design Suite 13.4 > PlanAhead > PlanAhead**.
- 2. Click Create New Project.
- 3. Click Next.
- 4. Browse to and select the reconfig_peripheral_lab\ directory for the Project location.
- 5. Click Select.
- 6. Type **PlanAhead** for the Project name in the New Project wizard.

Project Name		-
Enter a nam will be store	e for your project and specify a directory where the project data files	R
Project name:	PlanAhead	0
Project location:	C:\PartialReconfiguration\reconfig_peripheral_lab	0
🔽 Create Proje	ct Subdirectory	
Project will be cre	ated at: C:\PartialReconfiguration\reconfig_peripheral_lab\PlanAhead	
in space with the city		

Figure 7: Project Name Page of the New Project Wizard

- 7. Click Next.
- 8. In the New Project Design Sources page, select **Specify synthesized (EDIF or NGC) netlist**.

9. Check the Enable Partial Reconfiguration option.

Note: If you forget to check the option, you can still enable it from the project (netlist based only) by selecting **Tools > Project Settings > General** and clicking the "Partial Reconfiguration Project" check box. This must be done before a partition can be defined as reconfigurable.

10. Click Next.

Note: The Enable Partial Reconfiguration option is available only if you have a license for Partial Reconfiguration.

	Vew Project
De	sign Source
	Specify the type of sources for your design. You can start with RTL or a synthesized EDIF.
0	Specify <u>R</u> TL Sources You will be able to run RTL analysis, synthesis, post-synthesis design analysis, planning and implementation.
	Import settings and sources from XST or Synplify project
0	Specify synthesized (EDIF or NGC) netlist You will be able to run post-synthesis design analysis, planning, and implementation.
	Enable Partial Reconfiguration
0	Create an I/ <u>O</u> Planning Project Do not specify design sources. You will be able to do port assignment and verification.
0	Import I <u>S</u> E Place & Route results You will be able to do post-implementation analysis of your design.
0	Import ISE Project Create a PlanAhead project from an ISE project file.
	< Back Next > Finish Cancel

Figure 8: Importing Synthesized Netlists

- 11. Click the **Add Files** button.
- 12. Browse to reconfig_peripheral_lab\edk\implementation\
- 13. Select all NGC files including the system.ngc file, and click OK.
- 14. In the Top column, click the radio button next to system.ngc to identify it as the top-level design file.

Step 3: Creating a PlanAhead Project

1	Id	Name	Тор	Location				
1	1	axi4lite_0_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
1 2	2	axi_hwicap_0_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
1 3	3	axi_hwicap_0_wrapper_fifo_generator_v8_3_1.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
4	4	axi_hwicap_0_wrapper_fifo_generator_v8_3_2.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
1 5	5	clock_generator_0_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
6	6	debug_module_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
1 7	7	math_0_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
8	8	microblaze_0_bram_block_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
9	9	microblaze_0_d_bram_ctrl_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
1	10	microblaze_0_dlmb_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
1	11	microblaze_0_i_bram_ctrl_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
1	12	microblaze_0_ilmb_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
1	13	microblaze_0_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
1	14	proc_sys_reset_0_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
a i 1	15	rs232_uart_1_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
a l 1	16	sysace_compactflash_wrapper.ngc	0	C:\PartialReconfiguration\reconfig_peripheral_la				
al 1	17	system.ngc	۲	C:\PartialReconfiguration\reconfig_peripheral_la				
(111		•				
		Add Elec	Ad	d Directories				

Figure 9: Selecting the Top-level Netlist File

15. Click Next.

The Add Constraint files (optional) page opens.

- 16. Click Add Files.
- 17. Browse to reconfig_peripheral_lab\edk\data
- 18. Select system.ucf.
- 19. Click **OK**.
- 20. Click **Next** to open the Product Family and Default Part page.
- 21. Make sure that the **xc6vlx240tff1156-1** part is selected. Otherwise, select the filters, and select the **xc6vlx240tff1156-1** part as shown in the following figure.

Step 3: Creating a PlanAhead Project

Choose a defau	ult Xilinx	cpart for your pro	ject. This can be	changed later.						
Filter										
Product Category	Gener	al Purpose	*	Package F Speed Grade		FF1156				
Family	Virtex	6	*			-1				
Sub-Family	Virtex6 LXT			Temp Grade			C *			
Search: Q- Device		I/O Pin Count	Available IOBs	LUT Elements	Flip	Flops	Block RAMs	DSPs	Gb Tra	
xc6vlx130tff11	56-1	1156	600	80000	160	000	264	480	20	
xc6vlx195tff11	56-1	1156	600	124800	249	600	344	640	20	
xc6vlx240tff11	56-1	1156	600	150720	301	440	416	768	20	
xc6vix365tff11	56-1	1156	600	227520	455040		416	5/6	20	
•		m							• 🗆	

Figure 10: Selecting the Target Device

- 22. Click Next.
- 23. Click Finish.

The project is created. The Project Manager pane displays the modules present in the design.



Figure 11: Design Hierarchy in PlanAhead

Step 4: Defining a Reconfigurable Partition

This design has one reconfigurable partition that must explicitly be defined.

Defining a Reconfigurable Partition (RP) With a Black Box Reconfigurable Module (RM).

1. Click **Netlist Design** to invoke the netlist files parser.

This is necessary as we want to access a lower-level module to define a reconfigurable partition.

A warning message indicating that one instance will be converted to a black box because the netlist file for it is missing. This is expected because no netlist has been associated with this module yet.

A Netlist tab displays the hierarchical view of the system.

- 2. Click **OK**.
- 3. Expand the math_0 instance.
- 4. Select math_0/USER_LOGIC_I/rp_instance in the Netlist view.

Netlist	_ D & >
X 🕅 🖪	
💥 system	
H- Rets (1626)	
Primitives (26)	
B- RS232 Uart 1 (rs232 uart	1 wrapper)
SysACE_CompactFlash (sysa	ce_compactflash_wrapper)
axi4lite 0 (axi4lite 0 wrappe	er)
- axi hwicap 0 (axi hwicap 0	wrapper)
dock generator 0 (dock ge	nerator 0 wrapper)
- a debug_module (debug_modu	le_wrapper)
- math_0 (math_0_wrapper)	
🕀 - 🫅 Nets (284)	
Primitives (209)	
math_0/USER_LOGIC_I/	rp_instance (rp)
microblaze_0 (microblaze_0_	wrapper)
- microblaze_0_bram_block (m	icroblaze_0_bram_block_wrapper)
microblaze_0_d_bram_ctrl (n	<pre>icroblaze_0_d_bram_ctrl_wrapper)</pre>
microblaze_0_dlmb (microblaze)	ze_0_dlmb_wrapper)
microblaze_0_i_bram_ctrl (mi	croblaze_0_i_bram_ctrl_wrapper)
microblaze_0_ilmb (microblaze)	e_0_limb_wrapper)
H- a proc sys reset 0 (proc sys	reset 0 wrapper)

Figure 12: Netlists Hierarchy View

- 5. Right-click and select **Set Partition**.
- 6. Click Next twice.

The Set Partition dialog box will appear.

7. Select Add this Reconfigurable module as a black box without a netlist.



8. Type **math_BB** in the RM name field since the partition does not yet have a defined netlist.

Reconfigurable Module	Name
Enter a name for the new 'math_0/math_0/USER_LC	Reconfigurable Module for instance DGIC_I/rp_instance'.
Reconfigurable Module Name:	math_BB
Netlist already available	for this Reconfigurable Module
Add this Reconfigurable	Module as a black box without a netlist

Figure 13: Setting a Partition

- 9. Click Next.
- 10. Click Finish.

Note: The black box icon has changed to a diamond shape.

Step 5: Adding Reconfigurable Modules

This design has two Reconfigurable Modules (RMs) for the Reconfigurable Partition (RP). In this step, you will add the two modules.

Adding Two Reconfigurable Modules: Adder and Multiplier

- 1. In the Netlist window, select the math_0/USER_LOGIC_I/rp_instance.
- 2. Right-click and select Add Reconfigurable Module.
- 3. Click Next.

The Add Reconfigurable Module dialog box displays.

- 4. In the Reconfigurable Module Name field, type **adder**.
- 5. Verify that **Netlist already available for this Reconfigurable Module** is selected.

Reconfigurable Module Name	
Enter a name for the new Reconfigurable Module for instan 'math_0/math_0/USER_LOGIC_I/rp_instance'.	nce 📢
Reconfigurable Module Name: adder	0
Netlist already available for this Reconfigurable Module	
Add this Reconfigurable Module as a black box without a	netlist

Figure 14: Adding a Reconfigurable Module

- 7. Click Next.
- 8. Browse to reconfig_peripheral_lab\resources\Math\adder\ and select the rp.ngc file.
- 9. Click Open.



Step 5: Adding Reconfigurable Modules

Specify the EL	IF or NGC netlist that contains the Reconfigurable Module
Top Netlist File:	tialReconfiguration\reconfig_peripheral_lab\resources\Math\adder\rp.ngc 🕥 [Add Directories
Copy Sour	es into Project

Figure 15: Locating the adder Version of math.ngc

- 10. Click Next twice.
- 11. Click Finish.
- 12. In the Netlist pane, expand Reconfigurable Modules hierarchy under math_0/USER_LOGIC_I/rp_instance to view the adder RM entry.
- 13. Follow the steps in Step 5 to add a **multiplier** RM from the reconfig_peripheral_lab\resource\Math\multiplier\rp.ngc directory. Name the RM **mult**.

The Netlist window displays three Reconfigurable Modules (including the black box) for the math Reconfigurable Partition.

The multiplier module is active (with a check mark) as it was the most recent netlist to be added to the project.

Netist Image: Compact Flash System Image: Compact Flash Image: Compact Flash (sysace_compact flash_wrapper) Image: Compact Flash (sysace_compact flash_wrapper)
system System Primitives (1626) RS232_Uart_1 (rs232_uart_1_wrapper) SysACE_CompactFlash (sysace_compactflash_wrapper) SysACE_CompactFlash (sysace_compactflash_wrapper) Call axi4lite_0 (axi4lite_0_wrapper) Call axi_hwicap_0 (axi_hwicap_0_wrapper)
system Primitives (1626) RS232_Uart_1 (rs232_uart_1_wrapper) SysACE_CompactFlash (sysace_compactflash_wrapper) axi4lite_0 (axi4lite_0_wrapper) axi_hwicap_0 (axi_hwicap_0_wrapper)
Nets (1626) Primitives (26) RS232_Uart_1 (rs232_uart_1_wrapper) SysACE_CompactFlash (sysace_compactflash_wrapper) axi4lite_0 (axi4lite_0_wrapper) axi_hwicap_0 (axi_hwicap_0_wrapper)
Primitives (26) RS232_Uart_1 (rs232_uart_1_wrapper) SysACE_CompactFlash (sysace_compactflash_wrapper) area axi4lite_0 (axi4lite_0_wrapper) area axi_hwicap_0 (axi_hwicap_0_wrapper)
 RS232_Uart_1 (rs232_uart_1_wrapper) SysACE_CompactFlash (sysace_compactflash_wrapper) axi4lite_0 (axi4lite_0_wrapper) axi_hwicap_0 (axi_hwicap_0_wrapper)
Ð
Ð∽ @ axi4lite_0 (axi4lite_0_wrapper) ∄∽ @ axi_hwicap_0 (axi_hwicap_0_wrapper)
axi_nwicap_0 (axi_nwicap_0_wrapper)
The dealer and the O (dealer and the O and the O
I dock_generator_0 (dock_generator_0_wrapper)
math 0 (math 0 wranner)
(The Nets (284)
1 Primitives (209)
math 0/USER LOGIC I/rp_instance (rp)
E-C Reconfigurable Modules (3)
🕹 math_BB
🥎 adder
hen 🧇 mult
microblaze_0 (microblaze_0_wrapper)
microblaze_0_bram_block (microblaze_0_bram_block_wrapper)
microblaze_0_d_bram_ctrl (microblaze_0_d_bram_ctrl_wrapper)
H [] microblaze_0_dlmb (microblaze_0_dlmb_wrapper)
H- H microplaze_U_i_pram_ctri (microplaze_U_i_pram_ctri_wrapper)
The processes reset 0 (processes 0 wrapper)
a la hor ala leger a (hor ala leger a lumphe)

Figure 16: PlanAhead Project with adder and mult RMs Added



Step 6: Defining the Reconfigurable Partition Region

Next, floorplan the RP region. Depending on the type and amount of resources used by each RM, the RP region must be appropriately defined so it can accommodate any RM variant.

Setting the Reconfigurable Region

- 1. Select **Window > Physical Constraints**.
- 2. In the Physical Constraints tab, select **pblock_math_0/USER_LOGIC_I/rp_instance**.
- 3. Right-click and select **Set Pblock Size**.

Physical Constraints	-	Ð	×
netlist_1			
E- D ROOT			
	ce		

Figure 17: Setting Physical Constraints

- 4. Zoom to the top left quarter of the FPGA.
- 5. Move the cursor in the Device window.
- 6. Click and drag the cursor to draw a box that bounds SLICE_X8Y230:SLICE_X17Y239, as shown below.

Drawing a box around this region is required because the multiplier (mult) RM requires one DSP48E and the adder RM requires 32-bit tall carry chain.

The current grid coordinates are reported in the status bar at the bottom of the PlanAhead window.

At the completion of this step, the Set Pblock dialog box displays.

Step 6: Defining the Reconfigurable Partition Region



Figure 18: Closer View of the Pblock Area

- 7. In the Set Pblock dialog box, verify that **SLICE** and **DSP48** are checked as the resources to be reconfigured, shown in the following figure.
- 8. Click OK.

🖸 Set Pblock	×
Which resources do you wish pblock_math_0_USER_LOGIC_I_rp_instance to constrain? Grids	
✓ <u>SLICE</u> ✓ DSP48	
Select All Clear All	
OK Cancel	

Figure 19: Setting Pblock with SLICE and DSP48

Step 7: Running the Design Rule Checker

Xilinx recommends that you run a Design Rule Check (DRC) in order to detect errors as soon as possible.

Selecting and Running PR-specific DRCs

- 1. Select **Tools > Run DRC**.
- 2. Deselect All Rules.
- 3. Select Partial Reconfig.
- 4. Click **OK** to run the PR-specific design rules.

Results Name:	results_1		8
Output File:			
Rules to Chec	k: 14 of 95		0
9, 🕸 🛣			
All Rule	es (95)		
🕀 🔲 Ne	tlist (6)		
🕀 🛄 Clo	ock (4)		
🕀 🛄 Ba	nk (19)		
IO IO	B (13)		
Pla	cer (10)		
Flo	orplan (5)		
	BUT (1) D49 (6)		
	MB (3)		
🕀 🛄 Pa	rtition (10)		
🕀 🗹 Pa	rtial Reconfig (14)		
🕀 🛄 Te	am Design (3)		
🕀 🔝 FI	O (1)		
	Select All	Clear All	

Figure 20: Running Design Rule Checks

You will see warnings stating that Reconfigurable Modules (RMs) have not been implemented.

Step 8: Creating the First Configuration, Implementing, and Promoting

Now you can create and implement the first configuration.

Creating a New Strategy

Use the -bm option pointing to the system.bmm file for the new strategy.

- 1. Select **Tools > Options**.
- 2. Select **Strategies** in the left pane.
- 3. Select **ISE 13** in the **Flow** drop-down box.
- 4. Under PlanAhead Strategies, select ISE Defaults.
- 5. Click the + button to create a new strategy.



Figure 21: Creating a New Strategy

- 6. Name the new strategy **ISE13_BM**.
- 7. Click **OK**.
- 8. Under **Translate (ngdbuild)**, click in the **More Options** field.
- 9. Type -bmedk.implementation.system.bmm, and click Apply.

Running the Implementation Using Mult as a Variant

- 1. At the bottom of the PlanAhead tool user interface, select the **Design Runs** tab.
- 2. Select the **config_1** run.
- 3. In the Implementation Run Properties window, select the **General** tab.
- 4. In the Name field, type **mult** as the run name.
- 5. Click **Apply** to change the run name from config_1 to mult.

Step 8: Creating the First Configuration, Implementing, and Promoting

Implementation	Run Properties	_ C 7 ×
🔶 🔶 🍋	5	
mul		
Name:	mult	0
Part:	 xc6vlx240tff1156-1 (a	active)
Description:	ISE Defaults, including pa	icking rei
Status:	Not started	
Constraints:	constrs_1 (active) 💌	•
	1	
General 0	otions Monitor Reports	Messages Partitions
	Apply 🔯 C	ancel

Figure 22: Implementation Run Properties View

- 6. In the Options tab, change the Strategy to **ISE13_BM**.
- 7. Click Apply.
- 8. In the Partitions tab, click the Module Variant column drop-down button and select **mult** as the variant.
- 9. Click Apply.

Step 8: Creating the First Configuration, Implementing, and Promoting

mpleme	ntation Run	Propertie	25		_ 🗆	₽ ×
> mui	t					
ne				Mo	dule Variant	
Static	Logic					
math_	0/math_0/U	SER_LOG	iIC_1/rp_in	stance ma	th_BB	
				ma ado	th_BB der	
				mu	i i	
•		m				•

Figure 23: Implementation Run

- 10. In the Design Run window, select **mult**, and right-click and select **Launch Runs** to run the implementation.
- 11. Select Launch Runs on Local Host.
- 12. Click OK.
- 13. Click **Save** to save the project and run the implementation.

The implementation runs.

When implementation is finished running, a dialog box opens in which you can load the implemented results, or promote the implemented partitions, among other options.

- 14. Select the Promote Partitions radio button, and click OK.
- 15. In the Promote Partitions dialog box, click **OK** to promote the current configuration so the implemented results are available for the subsequent configurations.

Step 8: Creating the First Configuration, Implementing, and Promoting

Select Runs to p	promote
Run	Directory
- 🗹 mult C Static math	onfig_peripheral_lab\PlanAhead\PlanAhead.promote\Xmult Logic _0/math_0/USER_LOGIC_I/rp_instance - mult
	Select Implemented Clear All

Figure 24: Promoting Partitions



Step 9: Creating Other Configurations, and Implementing

After you have created the first configuration, the static logic implementation is reused for the rest of the configurations. Next, you will create the desired number of additional configurations and implement them.

Creating Multiple Runs

1. Select **Flow > Create New Runs**.

The Create Multiple Runs window opens.

- 2. Click Next twice.
- 3. In the Choose Implementation Strategies and Reconfigurable Modules page, change the name of the configuration from config_1 to **adder**.
- 4. Click **More**.
- 5. Change the name of config 1 to **black_box**.

reate Implem	Strateov		Make Active (ontional)	Partition Action
adder	ISE13_BM (ISE 13)	•		math_0/ma
lack_box	A ISE 13_BM (ISE 13)	•	0	math_0/ma
		8		

Figure 25: Creating Multiple Runs

- 6. In the adder configuration row, click the **Partition Action** field.
- 7. For the rp_instance row, click the Module Variant column drop-down arrow, and select **adder** as the variant to be implemented, as shown in Figure 26.

Step 9: Creating Other Configurations, and Implementing

Specify Partition		_	
Choose Module Variants for Reconfigur	able Modules and actio	ns	
Name	Module Variant	Action	Import from
Static Logic		Import	✓ C:\PartialReconfiguration\reconfig_peripheral_lab\PlanAhe
math_0/math_0/USER_LOGIC_I/rp_instance	adder	 Implement 	w N/A
	math_BB		
	adder		
	marc		
			OK Cancel

Figure 26: Selecting adder Module Variant

- 8. Click OK.
- 9. Similarly, select **math_BB** variant for the black_box run (row).
- 10. Click Next.
- 11. Select Launch Runs on Local Host.
- 12. Click Next.
- 13. Click **Finish** to run the implementations for both configurations.
- 14. Click **Cancel** when the runs are finished.



Step 10: Running Partial Reconfiguration to Verify Utility

Next, you will check to be sure that the static implementation, including interfaces to reconfigurable regions, is consistent across all configurations. To verify this, you can run the PR_Verify utility.

Running the PR_Verify Utility

Run the PR_Verify utility to make sure that there are no errors.

- 1. In the Configurations window, select any of the configurations.
- 2. Right-click, and select Verify Configuration.

Configuration			Module Variant
🚍 📣 🛧 mult (2)			
- ∰ Static 	Logic 😙 0/ma	Promote Partitions Unpromote Configurati	on
🖸 ✔ Static	Logic 🚺	Verify Configuration	
⊘v math_	0/mai	Load Configuration	
- Static	Logic	Export to Spreadsheet	
math_	0/math_0/	USER_LOGIC_I/rp_instance	math_BB

Figure 27: Verifying All Configurations

- 3. Press **Shift** and select all configurations.
- 4. Click **OK**.
- 5. The PR_Verify utility runs and reports that there were no errors.

Step 11: Generating Bit Files

After all the configurations have been validated by PR_Verify, you can generate full and partial bit files for the entire project.

Generating Full and Partial Bitstreams

- 1. In the Design Runs window, press **Shift** and select the following three designs runs:
 - mult
 - adder
 - black_box
- 2. Right-click and select Generate Bitstream.

This runs the bitstream generation process and generates full and partial bitstreams.

The bit files are placed in the mult, adder and black box directories under the reconfig_peripheral_lab\PlanAhead\PlanAhead.runs\ directory.

- 3. Click OK.
- 4. Save the project.
- 5. Close PlanAhead.

Step 12: Creating an Image, and Testing

For this step you need to open an EDK shell, and create both a download.bit and a system.ace file in the image\ directory. Copy the generated partial bit files, place them in the image\ directory, and name them adder.bit, mult.bit, and blank.bit.

Renaming Partial Bitstream Files, and Generating the system.ace File

- 1. Launch the EDK bash shell or ISE Design Suite command prompt as follows:
 - From XPS, select **Project > Launch Xilinx Shell**, or
 - From your Windows environment, select Start > Programs > Xilinx ISE Design Suite 13.4 > Accessories > ISE Design Suite Command Prompt.
- 2. In the Xilinx shell or command window, go to the reconfig_peripheral_lab\image\ directory.
- 3. Execute the following command to generate the download.bit file (with the software component included) from adder.bit (with the hardware component) only.

```
data2mem -bm ..\edk\implementation\system bd
```

```
-bt ..\PlanAhead\PlanAhead.runs\adder\adder.bit
```

```
-bd ..\edk\SDK\SDK_Export\TestApp\Debug\TestApp.elf tag microblaze_0 -o b download.bit
```

Hint: Copy the command text from this document and paste it in the shell or command window by right-clicking and selecting **Paste**.

This generates the download.bit in the image \ directory.

4. In the Bash shell, execute the following command to generate the system.ace file in the image\ directory.

```
xmd -tcl genace.tcl -jprog -target mdm -hw download.bit -board ml605 -
ace system.ace
```

5. Using Windows Explorer, copy and rename the following files, as shown in Table 1.

Table 1: Renaming partial bit files

File Name	Copy to Directory	Rename File To
<pre>reconfig_peripheral_lab\PlanAhead\PlanAhead.runs\adder\ adder_math_0_math_0_user_logic_i_rp_instance_adder_partia l.bit</pre>	\image	adder.bit
<pre>reconfig_peripheral_lab\PlanAhead\PlanAhead.runs\mult\mul t_math_0_math_0_user_logic_i_rp_instance_mult_partial.bit</pre>	∖image	mult.bit
<pre>reconfig_peripheral_lab\PlanAhead\PlanAhead.runs\black_bo x\black_box_math_0_user_logic_i_rp_instance_math_b b_partial.bit</pre>	\image	blank.bit

Copying the system.ace and Three Partial Bit Files on a Compact Flash Memory Card

- 1. Place a blank Compact Flash memory card in a Compact Flash writer.
- 2. Using Windows Explorer, copy the three partial bit files and the system.ace file from reconfig_peripheral_lab\image\ folder to the Compact Flash card.
- 3. Place the Compact Flash card in the ML605 board.
- 4. Set the SACE Mode pins (S1) to **0111 (dn-up-up-up)** to configure the FPGA device from the Compact Flash.
- 5. Connect your PC to the ML605 with the provided USB cable.
- 6. Install the driver, if necessary. For instructions, see the *ML605 Hardware User Guide*: http://www.xilinx.com/support/documentation/boards_and_kits/ug534.pdf
- 7. Start a HyperTerminal window, connecting using **COMx at 115200 baud** and power **ON** the ML605 board.
- 8. Press CPU Reset.
- 9. Follow the menu and test various reconfigurations.

Conclusion

In this tutorial, you created a processor system using XPS, added a user peripheral which included a place holder for the reconfigurable partition, and generated netlist files. Also, you created an application using SDK. Full bitstream as well as partial reconfiguration bitstreams were generated using the PlanAhead tool. Also, you generated an ACE file for Compact Flash memory card. You verified the functionality using the ML605 evaluation board.

XILINX_®

Appendix A

Additional Resources

Xilinx Resources

- *ISE Design Suite: Installation and Licensing Guide* (UG798): <u>http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_4/iil.pdf</u>
- *ISE Design Suite 13: Release Notes Guide* (UG631): <u>http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_4/irn.pdf</u>
- Product Support and Documentation:
 <u>http://www.xilinx.com/support/index.htm</u>
- Xilinx Glossary: http://www.xilinx.com/company/terms.htm
- Video Demonstrations:
 <u>http://www.xilinx.com/products/design_resources/design_tool/resources/index.htm</u>

Partial Reconfiguration Documentation

- *Partial Reconfiguration User Guide* (UG702): http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_4/ug702.pdf
- Partial Reconfiguration Tutorial (UG743): http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_4/PlanAhead_Tut orial_Partial_Reconfiguration.pdf

PlanAhead Documentation

- PlanAhead User Guide (UG632): http://www.xilinx.com/support/documentation/sw_manuals/xilinx13_4/PlanAhead_Use rGuide.pdf
- PlanAhead Methodology Guides: <u>http://www.xilinx.com/support/documentation/dt_planahead_planahead13-</u> <u>4_userguides.htm</u>
- PlanAhead Tutorials: http://www.xilinx.com/support/documentation/dt_planahead_planahead13-4_tutorials.htm