AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Hardware Tutorial (ISE Design Suite 14.5)

UG914 (v2.0) April 23, 2013





Notice of Disclaimer

The information disclosed to you hereunder (the "Materials") is provided solely for the selection and use of Xilinx products. To the maximum extent permitted by applicable law: (1) Materials are made available "AS IS" and with all faults, Xilinx hereby DISCLAIMS ALL WARRANTIES AND CONDITIONS, EXPRESS, IMPLIED, OR STATUTORY, INCLUDING BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE; and (2) Xilinx shall not be liable (whether in contract or tort, including negligence, or under any other theory of liability) for any loss or damage of any kind or nature related to, arising under, or in connection with, the Materials (including your use of the Materials), including for any direct, indirect, special, incidental, or consequential loss or damage (including loss of data, profits, goodwill, or any type of loss or damage suffered as a result of any action brought by a third party) even if such damage or loss was reasonably foreseeable or Xilinx had been advised of the possibility of the same. Xilinx assumes no obligation to correct any errors contained in the Materials or to notify you of updates to the Materials or to product specifications. You may not reproduce, modify, distribute, or publicly display the Materials without prior written consent. Certain products are subject to the terms and conditions of the Limited Warranties which can be viewed at http://www.xilinx.com/warranty.htm; IP cores may be subject to warranty and support terms contained in a license issued to you by Xilinx. Xilinx products are not designed or intended to be fail-safe or for use in any application requiring fail-safe performance; you assume sole risk and liability for use of Xilinx products in Critical Applications: http://www.xilinx.com/warranty.htm#critapps.

Automotive Applications Disclaimer

XILINX PRODUCTS ARE NOT DESIGNED OR INTENDED TO BE FAIL-SAFE, OR FOR USE IN ANY APPLICATION REQUIRING FAIL-SAFE PERFORMANCE, SUCH AS APPLICATIONS RELATED TO: (I) THE DEPLOYMENT OF AIRBAGS, (II) CONTROL OF A VEHICLE, UNLESS THERE IS A FAIL-SAFE OR REDUNDANCY FEATURE (WHICH DOES NOT INCLUDE USE OF SOFTWARE IN THE XILINX DEVICE TO IMPLEMENT THE REDUNDANCY) AND A WARNING SIGNAL UPON FAILURE TO THE OPERATOR, OR (III) USES THAT COULD LEAD TO DEATH OR PERSONAL INJURY. CUSTOMER ASSUMES THE SOLE RISK AND LIABILITY OF ANY USE OF XILINX PRODUCTS IN SUCH APPLICATIONS.

© Copyright 2012–2013 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Vivado, Virtex, Zynq, 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

The following table shows the revision history for this document.

Date	Version	Revision
09/04/2012	1.0	Initial Xilinx Release.
11/05/2012	1.1	Changed "system_top - STRUCTURE" to "Design Sources and system_top -STRUCTURE" under Create the Hardware Platform and Customize the Embedded Hardware Platform. Updated Figure 1-12, Figure 1-21, and Figure 1-22. Added step a under Export the BIST Hardware Platform to SDK. Changed "enter the new connection name" to "select from the drop down menu" in step 6 under Connect the Ports.
04/23/2013	2.0	Updated for ISE® Design Suite 14.5. Changed board_test_app_console.elf to board_test_app_Console.elf in step 9 under Executing the System, page 9. Under Connect the Ports, page 27, step 6 changed. Replaced Figure 1-5, Figure 1-6, Figure 1-7, Figure 1-10, Figure 1-12, Figure 1-14, Figure 1-15, Figure 1-17, Figure 1-18, Figure 1-19, Figure 1-20, Figure 1-21, and Figure 1-22. Enhanced Appendix A, Additional Resources.

Table of Contents

C	hapter 1: KC705 Embedded Kit MicroBlaze Processor Subsystem Hardware Tutorial
	Introduction
	Hardware and Software Requirements 5
	Prerequisites
	System Overview
	Included Files and Systems
	Executing the System
	Hardware Design Flow 12
	Summary
	Next Steps

Appendix A: Additional Resources

Xilinx Resources	33
Solution Centers	33
Further Resources	33
References	34



Chapter 1

KC705 Embedded Kit MicroBlaze Processor Subsystem Hardware Tutorial

Introduction

The Kintex®-7 FPGA embedded kit conveniently delivers the key components of the Xilinx® Embedded Targeted Design Platform (TDP) required for developing embedded software and hardware in a wide range of applications in the broadcast, industrial, medical, and aerospace and defense markets. Hardware designers now have immediate access to a pre-integrated MicroBlazeTM processor subsystem that includes the most commonly used peripheral IP cores, enabling the designers to begin at once developing their custom logic.

This tutorial guides the designer through the three steps required to examine, modify and test the MicroBlaze processor subsystem with the KC705 evaluation board:

- 1. Loading and executing a design from pre-built bitstream and ELF files
- 2. Examining and rebuilding a design
- 3. Adding IP from the Xilinx IP catalog to an embedded system

It is advisable to read <u>UG913</u>, *Getting Started with the Kintex-7 FPGA KC705 Embedded Kit* before going through this tutorial. Additional information related to EDK design flow can be found in <u>UG683</u>, *EDK Concepts, Tools, and Techniques*.

Hardware and Software Requirements

Included with the Kintex-7 FPGA embedded kit:

- Xilinx KC705 evaluation board
- One USB Type-A to Mini-B cable
- One USB Type-A to Micro-B cable
- Ethernet cable
- ISE® Design Suite: Embedded Edition which includes:
 - Integrated Software Environment (ISE)
 - Embedded Development Kit (EDK)
 - Software Development Kit (SDK)

Additional Requirements:

• Host PC with serial communications utility program (e.g., Tera Term)

Prerequisites

Prerequisites required to run the basic tutorial:

- KC705 embedded kit MicroBlaze processor subsystem
- Proper hardware setup and software installation, as described in <u>UG913</u>, *Getting Started with the Kintex-7 FPGA KC705 Embedded Kit*
- Familiarity with the KC705 embedded kit MicroBlaze processor subsystem, documented in <u>DS669</u>, AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Data Sheet
- Familiarity with <u>UG683</u>, *EDK Concepts, Tools, and Techniques*, specifically these sections:
 - Introduction
 - Using Xilinx Platform Studio
 - Working with the Embedded Platform
 - General knowledge of FPGAs, digital design concepts, and microprocessors
- Basic familiarity with the Xilinx Design Tools
- Basic VHDL/Verilog knowledge

System Overview

This tutorial is based on the KC705 embedded kit MicroBlaze processor subsystem. Readers are encouraged to refer to <u>DS669</u>, *AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Data Sheet* during the execution of this tutorial.

Included Files and Systems

The Tutorial_Sandbox directory provides the working area for this tutorial. It is part of the KC705 embedded kit. The hardware portion of the Tutorial Sandbox contains two subsystems: Built-In Self Test (BIST) and the partially built video demo system. The video demo system consists of video IP cores that are added to the system to develop a complete, functioning video demo. Figure 1-1 shows the structure of the Tutorial_Sandbox directory.





The completed tutorial resides in the Tutorial_Completed directory, the structure of which is shown in Figure 1-2.



Figure 1-2: Tutorial Completed Directory Structure

Executing the System

The ready_for_download directory contains the pre-built bitstream and ELF files for the KC705 embedded kit MicroBlaze processor BIST subsystem. The directory structure of the KC705_System is shown in <u>DS669</u>, *AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Data Sheet*.

The pre-built bitstream and ELF files are used to verify that the software is properly installed and that the board is set up correctly. To execute the system using these files:

- 1. Connect a USB Type-A to Micro-B cable between the KC705 evaluation board USB JTAG connector and the host computer.
- 2. Verify that a USB Type-A to Mini-B cable is properly connected to the KC705 evaluation board USB UART connection and the host computer.
- 3. If the board is not already powered on, power on the KC705 board.
- 4. Open and configure a serial communications terminal utility program with the settings shown in Figure 1-3.

Tera Term: Serial port setu	up X
Port:	СОМЗ • ОК
Baud rate:	9600 -
Data:	8 bit 🔹 Cancel
Parity:	none 🔻
Stop:	1 bit 🔹 Help
Flow control:	none 🔻
Transmit dela O msec	y c/char 0 msec/line
	LIG914 at 02 0718

Figure 1-3: Serial Communications Utility Configuration

Note: A procedure for setting up a UART connection is provided in Appendix B of <u>UG913</u>, *Getting Started with the Kintex-7 FPGA KC705 Embedded Kit.*

- 5. Open a command shell with the ISE Design tool and EDK environment settings. Refer to Appendix C of <u>UG913</u> for ISE tool chain installation and licensing help. If necessary, set the environment variables by running the settings32.bat script file located in the Xilinx directory of the ISE Design Suite installation area.
- 6. At the command prompt, enter the appropriate command:

On a Windows XP system:

C:\Xilinx\14.x\ISE_DS\settings32.bat

On Windows 7 system:

C:\Xilinx\14.x\ISE_DS\settings64.bat

7. Execute these commands to download the design and connect to the MicroBlaze processor:

\$ cd KC705_Embedded_Kit/KC705_System/ready_for_download

\$ xmd

XMD% fpga -f download.bit

Note: This command downloads the hardware bitstream into the FPGA but does not download the software application.

XMD% connect mb mdm

Note: This command connects to the MicroBlaze processor debug module.

8. XMD allows low-level visibility into the design. There are several useful XMD commands to allow for the peeking and poking of registers and memory locations within the system. For example, writes to and reads from the internal block RAM are accomplished as follows:

XMD% mwr 0xC000000 0xDEADBEEF

The value 0xDEADBEEF is written to location 0xC0000000.

XMD% mrd 0xC000000

The value 0xDEADBEEF should be returned:

C000000: DEADBEEF

As an example of register access, write to the GPIO register which outputs data to the 8-bit GPIO LEDs on the KC705 board:

XMD% mwr 0x40600000 0xAA

The value written to this register is now reflected in the GPIO LEDs. The next write changes the LED display:

XMD% mwr 0x40600000 0x55

A read from this register always returns 0x0000000, because this is an output-only register.

Note: Set all of the DIP switches labeled GPIO DIP SW (SW11) to the ON position. Then read from the GPIO register which inputs data from the DIP switches:

XMD% mrd 0x40700000

The value 0x000000F should be returned:

40700000: 0000000F

Change the DIP switch settings and verify that the correct value is read from the GPIO register.

Note: Switch 1 is the most significant bit.

For more information about XMD and the commands available within XMD, see <u>UG111</u>, *Embedded System Tools Reference Manual*, Xilinx Microprocessor Debugger (XMD).

9. To download and execute the test software application, enter these commands at the XMD command prompt:

XMD% dow board_test_app_Console.elf

XMD% con

Figure 1-4 shows the resulting output for the KC705 board test application on the serial communication terminal utility program.

```
* *
   Xilinx Kintex-7 FPGA KC705 Evaluation Kit
                               * *
------
** Xilinx Kintex-7 BIST MENU
                   * *
_____
Choose Feature to Test:
[1] uart [2] led
[3] iic [4] flash
[5] timer [6] rotary switch
[7] switch [8] sd card
[9] lcd [A] xadc
[B] button [C] ethernet loopback
[D] ddr3 external memory (16MB)
[E] ddr3 external memory (complete)
[F] bram internal memory
[0] Exit
                          UG914_c1_04_080612
```

Figure 1-4: Board Test Application Menu

- Execute the chosen tests and then select **0** to exit. Details about the test options can be found in the <u>UG915</u>, AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Software Tutorial.
- 11. When software execution is complete, enter these commands:

XMD% stop XMD% rst XMD% exit

Hardware Design Flow

The design flow for creating the MicroBlaze processor subsystem BIST and modifying the system to include the benchmarking cores requires these steps:

- 1. Create the Hardware Platform
- 2. Export the BIST Hardware Platform to SDK
- 3. Customize the Embedded Hardware Platform
- 4. Implement and Test the Design

The BIST system is used to demonstrate execution of steps one and two, and the partially built video demo system is used to demonstrate execution of steps three and four.

Create the Hardware Platform

Open the Project

1. Start the PlanAheadTM tool.

On a Windows system:

Select Start > All Programs > Xilinx Design Tools > ISE Design Suite > PlanAhead

On a Linux system:

Enter **planAhead** at the command prompt.

- 2. If necessary, close out the previous project by selecting File > Close Project.
- 3. In Project Commands, select Open Project...
- 4. Browse to KC705_Embedded_Kit/Tutorial_Sandbox/HW/BIST and select kc705_system.ppr. Click Open.
- 5. In the Project Manager/Sources section, expand Design Sources and system_top -STRUCTURE and double click **system_i - system (system.xmp)**. This invokes XPS for the XPS subsystem.

Figure 1-5 shows the block diagram of the KC705 embedded kit BIST system.



Figure 1-5: BIST System

Examine the System

The System Assembly View (SAV) of the design provides visibility into the MicroBlaze processor subsystem bus interfaces, port connections, and address map.

Note: Refer to <u>UG683</u>, *EDK Concepts, Tools, and Techniques* for additional details or instructions pertaining to any of the steps outlined in this section of the tutorial.

Examine Bus Interfaces

1. To examine the bus structure of the MicroBlaze processor subsystem, select the System Assembly View tab below the workspace and then select the Bus Interfaces tab at the top edge of the workspace as shown in Figure 1-6.

For MicroBlaze processor masters (Instruction Cache (IC), Data Cache (DC), and Data Port (DP)), both AXI_MM and AXI_Lite interconnects are used. Two Local Memory Buses (LMB) are used by the MicroBlaze processor to interface to the local BRAM memory for the processor. These interfaces can be seen by clicking the + to the left of the microblaze_0 component. (See: Figure 1-6).



UG914_c1_06_032313

Figure 1-6: Bus Interfaces View of the MicroBlaze Processor Subsystem

Examine Ports

1. Select the Ports tab along the top edge of the workspace. By clicking the + to the left of any of the components, the connections to that component are visible. (See: Figure 1-7)

Bus interfaces i forts Au	nesses				Port Filters	
Name	Net Dir	rection Range Class	Freq Reset Pol S IP Type	Connected Por	By Interface	3
- Locannemory_cnur_r			X		✓ BUS	1
Linear_Flash			🚖 axi_emc		- V IO	
DDR3_SDRAM			🚖 axi_7series_ddrx		By Connection	
- clk_ref	clk_ref 🗘 I	CLK		clock_generat	🗕 🖌 🖌 Defaults	
- mem_refclk	mem_ref 🗘 I	CLK		clock_generat	 Connected 	
- freq_refclk	freq_refclk 🖨 I	CLK		clock_generate	Unconnected	
pll_lock	clock_ge 🗘 I			clock_generate	🖻 By Class	
- sync_pulse	sync_pulse 🖨 I	CLK		clock_generate	Clocks Only	
- iodelay_ctrl_rdy_i	No Conn 🖨 I				Clocks	
- iodelay_ctrl_rdy_o	No Conn 🗘 O			=	Resets Only	
xadc_device_temp_i	xadc_te 🗘 I	[11:0]		axi_xadc_0::[s	 Resets 	
xadc_device_temp_o	No Conn 🖨 O	[11:0]			Interrupts Only	
- init_calib_complete	phy_done 🗘 O				✓ Interrupts	
(BUS_IF) S_AXI	Connect 🗘			Connected to [✓ Others	
IO_IF) memory_0	Connect 🗘			Connected to E	By Direction	
+ logisdhc_0			🔫 logisdhc		✓ Inputs	
∓ axi2axi_mb			🙀 axi2axi_connector		✓ Outputs	
+ debug_module			🚖 mdm		✓ InOuts	
Interrupt Cntlr			🚖 axi intc			
+ AXI DMA Ethernet			🛨 axi dma			

UG914_c1_07_032313

Figure 1-7: Ports View of the MicroBlaze Processor Subsystem

Note: The visibility of the port connections for the system is controlled by the Port Filters as shown in Figure 1-8. Many of the peripherals have only default port connections to the AXI interconnect and these connections are not shown unless Defaults filter item is checked.

Port Filters		<<
🖃 By Interface		ш
- V BUS		lter
✓ IO		S
By Connection		
Defaults		
✓ Connected		
 Unconnected 		
🖻 By Class		
Clocks Only		
 Clocks 		
Resets Only		
✓ Resets		
Interrupts Only		
 Interrupts 		
✓ Others		
By Direction		
🖌 Inputs		
 Outputs 		
🖌 InOuts		
	UG914 c1 08 07	1812

Figure 1-8: Port Filters

Examine Addresses

1. Select the Addresses tab along the top edge of the workspace. Click the + to the left of the microblaze_0 address map to expand the map as shown in Figure 1-9. This workspace is used to modify the addresses of peripherals in the system.

Ð	Bus Interfaces Ports	Addresses						(🖆
h	nstance	Base Name	Base Address	High Address	Size	Bus Interface(s	Bus Name	
E	microblaze_0's Address							
	 LocalMemory_Cntlr_D 	C_BASEADDR	0×00000000	0X00001FFF	8К 🕃	SLMB	dlmb	
	LocalMemory_Cntlr_I	C_BASEADDR	0×00000000	0X00001FFF	8К 🕻	SLMB	ilmb	
	 Interrupt_Cntlr 	C_BASEADDR	0×40100000	0X4010FFFF	64K	S_AXI	axi4lite_0	
	debug_module	C_BASEADDR	0x40200000	0X4020FFFF	64K	S_AXI	axi4lite_0	
	Dual_Timer_Counter	C_BASEADDR	0x40300000	0X4030FFFF	64K	S_AXI	axi4lite_0	
	RS232_Uart_1	C_BASEADDR	0x40400000	0X4040FFFF	64K	S_AXI	axi4lite_0	
	 Push_Buttons_5Bits 	C_BASEADDR	0x40500000	0X4050FFFF	64K	S_AXI	axi4lite_0	
	LEDs_8Bits	C_BASEADDR	0x40600000	0X4060FFFF	64K	S_AXI	axi4lite_0	
	DIP_Switches_4Bits	C_BASEADDR	0×40700000	0X4070FFFF	64K	S_AXI	axi4lite_0	
	- LCD_GPIO	C_BASEADDR	0x40800000	0X4080FFFF	64K	S_AXI	axi4lite_0	
	- ROTARY_GPIO	C_BASEADDR	0x40900000	0X4090FFFF	64K	S_AXI	axi4lite_0	
	IIC_EEPROM	C_BASEADDR	0x40A00000	0X40A0FFFF	64K	S_AXI	axi4lite_0	
	logisdhc_0	C_REGS_BASEADDR	0x40B00000	0X40B0FFFF	64K	S_AXI	axi4lite_0	
	axi_xadc_0	C_BASEADDR	0x40D00000	0X40D0FFFF	64K	S_AXI	axi4lite_0	
	Linear_Flash	C_S_AXI_MEM0_BASEADDR	0×48000000	0x4FFFFFFF	128M	S_AXI_MEM	axi4lite_0	
	AXI_DMA_Ethernet	C_BASEADDR	0×50000000	0X5000FFFF	64K	S_AXI_LITE	axi4lite_0	
	 Soft_Ethernet_MAC 	C_BASEADDR	0×50100000	0X5013FFFF	256K	S_AXI	axi4lite_0	
	DDR3_SDRAM	C_S_AXI_BASEADDR	0×80000000	0xBFFFFFFF	1G 🕻	S_AXI	axi4_0	
	axi2axi_mb	C_S_AXI_RNG00_BASEADDR	0×80000000	0xBFFFFFFF	1G 🕻	S_AXI	axi_mm_mb	
	Internal_BRAM	C_S_AXI_BASEADDR	0xC0000000	0xC000FFFF	64K	S_AXI	axi_mm_mb	
	1							
Ľ								
							UG914 c1 09 (080612

Figure 1-9: Addresses View of the MicroBlaze Processor Subsystem

Configure IP Cores

The IP cores within the system are configured to the required functionality for this embedded system. The configurations of the IP cores are described in the System Configuration section of <u>DS669</u>, *AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Data Sheet*.

1. With the Bus Interfaces tab selected in the System Assembly view, right-click the IP core of interest, as shown in Figure 1-10.

E LCD_GPIO	🚖 axi_gpio
LEDs_8Bits	🚖 axi_gpio
Push_Buttons_5Bits	🚖 axi_gpio
ROTARY_GPIO	🚖 axi_gpio
IIC_EEPROM	🚖 axi_iic
Dual_Timer_Counter	Configure ID
RS232_Uart_1	Configure IP
i axi_xadc_0	Show Ports for selected IPs
clock_generator_0	Show Forts for Selected in S
proc_sys_reset_0	View MPD
clock_generator_1	View ID Medifications (Change Les)
DIP_Switches_4Bits	view iP Modifications (Change Log)
•	View Helper IP Modifications (Change Log) 🕨 📃
	View PDF Datasheet
aster/Slave 🚬 Target <initiator td="" 🔌="" 🕻<=""><td>Drawsa UDI Cauraa</td></initiator>	Drawsa UDI Cauraa
se (paid) 🐻 License (eval) 🤜	Browse HDL Sources
scontinued	Delete Instance
gn Summary 🗙 🗙	8
	Make This IP Local
	UG914_c1_10_032313

Figure 1-10: Menu Option to Configure IP

2. Select **Configure IP...** and the configuration wizard for that core is displayed. Select the tab to show the particular feature or parameter of interest. Figure 1-11 shows the configuration wizard for the Dual_Timer_Counter core.

🍪 XPS Core	e Config - Dual_Timer_Counter - axi_timer_v1_03_a	×
Component Instance Name Dual_Ti	mer_Counter	
CaptureTrigO GenerateOutO CaptureTrigI GenerateOutO CaptureTrigI GenerateOutI Freeze PWMO S_AXI Interrupt	User System Interconnect Settings for BUSIF Image: All The Width of Counter in Timer Only One Timer is present TRIG0 Active Level TRIG1 Active Level GEN0 Active Level GEN1 Active Level	
_ SHOW AILFORTS		
	ОКС	ancel Help

UG914_c1_11_072012

Figure 1-11: **Dual Timer Counter - AXI_Timer Configuration Wizard**

3. Click **Cancel** to return to the System Assembly View since no changes are needed to the cores in the system at this time.

Generate the Hardware Platform

- 1. Close the XPS project by selecting **File > Exit**.
- 2. In the Project Manager/Sources pane, click system_top STRUCTURE.
- 3. In the Flow Navigator pane, double click **Generate Bitstream** under the Program and Debug section, as shown in Figure 1-12.

Note: Because the design is not yet implemented, a message window will open to indicate that no implementation results are available. Click **Yes** to close the message window.



UG914_c1_12_032313

Figure 1-12: Generating the Hardware Bitstream

Export the BIST Hardware Platform to SDK

Because the BIST hardware platform is used with the Software Development Kit (SDK) as described in <u>UG915</u>, AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Software Tutorial, information about the hardware platform must be provided to the SDK to allow development of software platforms and applications. Use the following steps to export this information to the SDK:

- 1. When bitstream generation is complete:
 - a. In the Flow Navigator pane, double click **Open Implemented Design** under the Implementation section.
 - b. Under Sources, click system_i system (system.xmp).
 - c. Under the File tab, select **Export** and click **Export Hardware** to open the dialog shown in Figure 1-13.

Export Har	dware		23
Expor	t hardware platform for SDK.		
Options			
Source:	🗾 system.xmp		-
Export to:	🛜 <local project="" to=""></local>		-
🔽 Include	Bitstream		
Export	Hardware		
📃 Launch	SDK		
		ОК	Cancel
		11001	4 01 12 071912

Figure 1-13: Exporting BIST Hardware Platform to SDK

d. The exported hardware description files are stored in this location:

KC705_Embedded_Kit/Tutorial_Sandbox/HW/BIST/kc705_system.sdk/SDK/SDK_Export

2. Copy the SDK_Export directory into the SW area of the Tutorial sandbox for use with the procedure described in <u>UG915</u>, *AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Software Tutorial*. The existing directory can be overwritten.

Copy:

```
KC705_Embedded_Kit/Tutorial_Sandbox/HW/BIST/kc705_system.sdk/
SDK/SDK_Export
```

to:

KC705_Embedded_Kit/Tutorial_Sandbox/SW/board_test_app

The SDK is run using the procedure described in <u>UG915</u>, AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Software Tutorial.

Download and Verify BIST Design

- The steps for testing the newly generated bit file are the same as those used to generate the System section where pre built bit and ELF files are tested. The only difference is that the new bit file should be loaded from KC705_Embedded_Kit/ Tutorial_Sandbox/HW/BIST/kc705_system.runs/impl_1/download.bit
- 2. The UART output will be displayed and should look similar to Figure 1-4, page 11.
- 3. Execute the chosen tests and then select **0** to exit. Refer to <u>UG915</u>, *AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Software Tutorial* for details about these test options.

At this point in the tutorial, the KC705 embedded kit MicroBlaze processor subsystem is successfully built, verified, and exported for use with the SDK. The next section in this tutorial explains how to customize an embedded hardware platform. Alternately, if no modifications to the hardware subsystem are required, proceed with the tutorial presented in UG915, KC705 Embedded Kit MicroBlaze Processor Subsystem Software Tutorial.

Customize the Embedded Hardware Platform

In this section of the tutorial, the partially built video demo system from the Tutorial Sandbox directory is modified to include an additional AXI_VDMA for developing a complete video system. After building the hardware, the video demonstration application uses a Web server to display the video throughput data.

Every data point consists of a data transfer of a gigabyte by means of multiple AXI Video DMAs. Each Video DMA transfers the video data to and from the DDR memory through the AXI interconnect and the DDR controller. The Perf_AXI core monitors the transactions on the AXI interface to provide data throughput calculations.

Note: For More details on the video demo, refer to <u>DS669</u>, AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Data Sheet.

The steps to customize the MicroBlaze processor subsystem include:

1. Add IP from the Xilinx IP Catalog.

Note: The VDMA which will be added to the partially built video demo system is highlighted in Figure 1-14.

- 2. Connect the Bus Interfaces.
- 3. Connect the Ports.



UG914_c1_14_041213



To Add the AXI VDMA Core:

- 1. The PlanAhead tool should already be running (from the previous section of this tutorial).
- 2. Close the BIST project by selecting **File > Close Project**.
- 3. In Project Commands, select **Open Project...**
- 4. Browse to KC705_Embedded_Kit/Tutorial_Sandbox/HW/Video_Demo and select video_demo.ppr. Click OK.
- 5. Under Sources, expand Design Sources and system_top -STRUCTURE and double click **system_i system (system.xmp)**. This invokes XPS for the XPS subsystem.
- 6. In the EDK_Install section, Expand the DMA and Timer list by clicking the respective+. The AXI Video DMA IP core is now displayed.

Note: The description field can be made wider by placing the cursor near the divider between the fields until it changes into two vertical bars with two opposing horizontal arrows. Click and drag to adjust the width.

7. Right-click the AXI Video DMA and select Add IP, as shown in Figure 1-15.

IP Catalog	⊕ D Ø X	•	Bus Interface	s Ports	Addresses
12 🖲 📳 🕃 🕷 🗐		R 2	Namo		Bus Name
Description		III			bus Name
Communication Low-Speed	d		Push Button	s 5Bits	
DMA and Timer			+ ROTARY GP	0	
🚽 📩 AXI Central DMA					
🛉 🛧 AXI Datamover			SCALER_0		
🚽 🛧 AXI DMA Engine			SCALER_2		
👷 AXI FIFO Memory Ma	apped To Stre.		Dual_Timer_	Counter	
🛉 🛧 AXI Watchdog Timei	r =		axi_tpg_0		
🛉 🙀 AXI Timer/Counter			i axi_tpg_2		
🚽 🛉 AXI Video DMA	Add IP				
The Debug					
	View MPD			40	
General Purpose IO	View IP Modifi	cations (Chan	ge Log)	MA MA	
+ Interprocessor Communic	View Helper IF	P Modification	(Change Log)	MA	
+ Memory and Memory Cor	View PDF Det		(enange zog, ·	MA	
+ PCI	VIEW PDF Data	asneet			
	Make This IP L	.ocal		► Target <ir< td=""><td>nitiator 🌢 Connec</td></ir<>	nitiator 🌢 Connec
Search IP Catalog:	Clear	+ Production	BLicense (paid)	BLicense (e	val) 🗟 Local 🋓
🍪 Project 🎯 IP Catalog		8	System As	sembly Vie	w

UG914_c1_15_032313

Figure 1-15: **Adding the AXI_VDMA Core**

Note: Warning messages about the axi_vdma or other cores added into the system not being accessible from any processor in the system can be safely ignored. The procedure to connect the bus interfaces is discussed later in this tutorial.

- The XPS Core Configuration dialog is automatically invoked (See: Figure 1-16, page 24). This dialog sets AXI interconnect parameters for the core such as register slicing.
- 9. In the Configuration dialog, make the following settings:
 - Component instance name: **DVI_IN_1_VDMA**
 - In the USER tab:
 - Select Enable Asynchronous Clocks within the VDMA options
 - Clear Enable Channel within MM2S Channel options
 - Do following changes within the S2MM Channel options:
 - Set Memory Map Data Width as 64 from pull down menu
 - Select Allow Unaligned Transfers (DRE)
 - Set Maximum Burst Size to **256** from pull down menu
 - Set Line Buffer Depth to 4096
 - In the SYSTEM tab:
 - Base Address: **0x50900000**
 - High Address: 0X5090FFFF
 - In the Interconnect Settings for BUSIF tab, M_AXI_S2MM section
 - User Register Slice on all channels (AW,AR,W,R and B): AUTOMATIC
 - Write Data FIFO Depth: 512 (BRAM)
 - In Interconnect Settings for BUSIF tab, S_AXI_LITE section
 - User Register Slice on all channels (AW,AR,W,R and B): AUTOMATIC
 - Click Ok

OVI_IN_1_VDMA		
DVI_IN_1_VDMA	User System Interconnect Settings for BUSIF ACLK Frequency Ratio Arbitration Priority Use register slice on AW channel Use register slice on AW channel Use register slice on AR channel Use register slice on R channel Use register slice on B channel Use register slice on B channel Use register slice on B channel Write Data FIFO Depth Read Data FIFO Depth Write Transaction Issuance Limit Read Transaction Issuance Limit Normal	HU F C
		OK Cancel Help
	A25_prmry_reset_out_n h25_prmry_reset_out_n mm25_princ_put15:01 mm25_princ_put2 mm25_putfer_empty s2mm_bro_cout s2mm_bro	12:5 prmry_roset_out_n 12:5 prmry_roset_out_n 13:5 prmry_roset_out_n 14:5 prmry_roset_out_n 15:5 prmry_roset_out

Figure 1-16: Setting AXI_VDMA Parameters

10. In the Instantiate and Connect IP dialog, which is automatically invoked, select **User will make necessary connections and settings** then click **OK**. The procedure to connect the bus interfaces will be covered next.

Connect the Bus Interfaces

1. Expand the DVI_IN_1_VDMA core in the System Assembly View (SAV) by clicking the respective +. Connect M_AXI_S2MM to AXI_MM_VIDEO34 using the pull-down menu as shown in (Figure 1-17).

Note: Ignore error messages related to bus interface and port connections.



Figure 1-17: Connecting DVI_IN_1VDMA Master Interface to AXI MM interconnect

2. For S_AXI_LITE, click the **No Connection** box. The Connection Dialog box appears. In the left column, select **AXI4Lite_0** as shown in Figure 1-18 and click **OK**.

Bus Interfaces Ports	Addresses	
Name	Bus Name	IP Туре
 clock_generator_1 DVI_IN_0_SWRST_FF 		★ clock_generator ★ util_flipflop
DVI_IN_1_SWRST_FF	😵 DVI_IN_1_V	/DMA.S_AXI_LITE Connection Dialog
TPG_0_SWRST_FF	Select AXI Interconnect	Select Master(s)
DVI_IN_0_SWRST_FF DVI_IN_1_SWRST TPG_0_SWRST TPG_2_SWRST video_mux_0 video_mux_1 proc_sys_reset_0 DVI_IN_1_VDMA S_AXI_LITE M_AXI_S2MM	No Connection axi4_0 axi_mm_mb axi_mm_video12 axi_mm_video34 axi4lite_0 axi4lite_1	No master selection available for shared bus
S_AXIS_S2MM		🔀 <u>C</u> ancel 🔗 <u>O</u> K

UG914_c1_18_032313

Figure 1-18: Connecting DVI_IN_1VDMA Lite Slave Interface to AXI Lite interconnect

3. Connect S_AXIS_S2MM to DVI_IN_1_S_AXIS_S2MM using the pull-down menu as shown in Figure 1-19.

e E	Bus Interfaces	Ports	Ac	ldresses					
Nar	me			Bus Name	2	ІР Туре			
	DVI_IN_0_SWRS	ST_FF				🔺 util_	flipflop		
	DVI_IN_1_SWRS	T_FF				🔺 util_	flipflop		
····· ·	TPG_0_SWRST_	FF				🔺 util_	flipflop		
	TPG_2_SWRST_	FF				🔺 util_	flipflop		
	DVI_IN_0_SWRS	T				🔺 util_	reduced_	logic	
	DVI_IN_1_SWRS	T				🔺 util_	reduced_	logic	
	TPG_0_SWRST					🔺 util_	reduced_	logic	
-	TPG_2_SWRST			No Conne	ection		reduced_	logic	
	video_mux_0			nerf mor	itor 0 AXIS		:_sel		
-	video_mux_1			pen_mor			:_sel		
-	proc_sys_reset_	0		DVI_IN_0	_S_AXIS_S2MM		:_sys_res	et	
	axi_interconnec	:t_0		SCALER_	0_VDMA_M_AXIS_M	1M2S	interconn	ect	
÷.	DVI_IN_1_VDMA	l		SCALER	O S AXIS S2MM		vdma		
	S_AXI_LITE			JUALLIN_	0_0_4/10_021111				
	M_AXI_S2MM	1		tpg_0_s2	mm				
	S_AXIS_S2MI	М		DVI_IN_1	_S_AXIS_S2MM				
				SCALER_	2_VDMA_M_AXIS_M	1M2S			
laster	r/Slave Þ Target	<initiator td="" 🤘<=""><td>Cor</td><td>SCALER_</td><td>2_S_AXIS_S2MM</td><td></td><td></td><td></td><td></td></initiator>	Cor	SCALER_	2_S_AXIS_S2MM				
nse (Discor	paid) 🔞License ntinued	(eval) 🤤	Lo	tpg_2_s2	mm		hent		
sign	Summary	×	8		System Assembly	View	×	8	
			_						

UG914_c1_19_032313

Figure 1-19: Connecting DVI_IN_1VDMA Stream Slave Interface to Video Streaming Master

- 4. Expand the axi2axi_video34 core in the Bus Interface tab of System Assembly View of the XPS GUI by clicking the respective +.
- 5. Click the **Bus Name** column for the S_AXI row. This brings up the master list dialog for the axi2axi_video34 connector.
- 6. Select DVI_IN_1_VDMA.M_AXIS_S2MM and click OK (see Figure 1-20).

Bus Interfaces Ports	Addresses	
Name	Bus Name	IP Туре
		🛓 axi2axi_connector
in axi2axi_video12 In axi2axi video34	🍪 axi2axi	_video34.S_AXI Connection Dialog
S_AXI	Select AXI Interconnect	Select Master(s)
M_AXI debug_module Interrupt_Cntlr AXI_DMA_Ethernet Soft_Ethernet_MAC DIP_Switches_4Bits GPIO_RST LCD_GPIO LEDs_8Bits Bush Buttons_5Bits	No Connection axi4_0 axi_mm_mb axi_mm_video12 axi_mm_video34 axi4lite_0 axi4lite_1	 ✓ SCALER_2_VDMA.M_AXI_MM2S ✓ SCALER_2_VDMA.M_AXI_S2MM ✓ TPG_2_VDMA.M_AXI_S2MM ✓ DVI_IN_1_VDMA.M_AXI_S2MM
ROTARY_GPIO		¥ <u>C</u> ancel ⊘K

UG914_c1_20_032313

Figure 1-20: Adding DVI_IN_1VDMA as AXI MM Master to AXI2AXI Connector

Connect the Ports

The section covers connecting clock pins, interrupts, and signals for the DVI_IN_1_VDMA. In the Port Filters section of the XPS window, ensure that **Defaults** is selected (refer to Figure 1-8, page 15).

- 1. Click the **Ports** tab at the top of the System Assembly View.
- 2. Expand the DVI_IN_1_VDMA core in the System Assembly View by clicking the respective +.
- 3. In the name column of the port list for **s_axis_s2mm_aclk**, select the net name **sys_clk_s** using the pull-down menu.
- Expand (BUS_IF) S_AXI_Lite by clicking the respective +. In the name column for s_axi_lite_aclk, select the net name sys_clk_axilite_s using the pull-down menu in the Net column.
- 5. Expand (BUS_IF) M_AXI_S2MM by clicking the respective +. In the name column for **m_axi_s2mm_aclk**, select the net name **sys_clk_s** using the pull-down menu.
- 6. Expand (BUS_IF) S_AXIS_S2MM by clicking the respective +. In the name column for **s_axis_s2mm_tdata**, enter the new connection name, **0x00 & dvi2axi_tdata_1**.
- 7. In the name column for **s2mm_fsync_out**, enter the new connection name, **fsync_from_dvi_in_1_vdma**.
- 8. In the name column for **s2mm_introut**, enter the new connection name, **DVI_IN_1_VDMA_s2mm_introut**.
- 9. In the name column for **s2mm_prmry_reset_out_n**, enter the new connection name, **DVI_IN_1_S_AXIS_S2MM_RESET_OUT_N**.
- 10. Expand the Interrupt_Cntlr core in the System Assembly view by clicking the respective **+**.

- 11. Click the **Net** column for the INTR row. This brings up the Interrupt Connection Dialog.
- 12. In the left column select **DVI_IN_1_VDMA:s2mm_introut** and click the right arrow (see Figure 1-21).

<u> </u>			Interr	upt Cor	mec	tion Dialog		×	ass	Fre
nter	rupt Controller Inter	rupt_Cntlr 😫				Show Net Name				
Jnc	onnected Interrupt(s)			Con	nected Interrupt(s)		Priority		
	Instance Name	Port Name				Instance Name	Port Name	Low		
5	Push_Buttons_5Bits	IP2INTC_Irpt			9	DVI_IN_0_VDMA	s2mm_introut			
6	LEDs_8Bits	IP2INTC_Irpt			10	logisdhc_0	sd_int			
7	DIP_Switches_4Bits	IP2INTC_Irpt			11	RS232_Uart_1	IP2INTC_Irpt	T		
8	DVI_IN_0_VDMA	mm2s_introut			12	IIC_EEPROM	IIC2INTC_Irpt			
9	TPG_0_VDMA	mm2s_introut			13	Dual_Timer_Counter	Interrupt	,		
10	TPG_2_VDMA	mm2s_introut	=		14	Soft_Ethernet_MAC	INTERRUPT			
11	DVI_IN_1_VDMA	mm2s_introut			15	AXI_DMA_Ethernet	s2mm_introut			
12	DVI_IN_1_VDMA	s2mm_introut	-		16	AXI_DMA_Ethernet	mm2s_introut	▼ High	ER	
				1		Help	🕻 <u>C</u> ancel 🔗 <u>O</u> k		_	

Figure 1-21: Adding DVI_IN_1_VDMA_S2MM Interrupt to Interrupt controller

13. In the right column select **DVI_IN_1_VDMA:s2mm_introut** and click the up arrow under Priority Low to set this interrupt priority to **5** and click **OK** (see Figure 1-22).

Interrupt Connection Dialog Interrupt Controller Interrupt_Cntlr \$ Show Net Name Unconnected Interrupt(s) Instance Name Priority Instance Name Port Name Instance Name Priority 1 perf_monitor_0 PERF_Irpt Instance Name Port Name 2 debug_module Interrupt Interrupt Low 3 Internal_BRAM ECC_Interrupt Interrupt Interrupt Interrupt 4 Internal_BRAM ECC_UE Internal_BRAM SCALER_2_VDMA S2mm_introut Image: Control of the priority	
Interrupt Controller Interrupt_Cntlr Unconnected Interrupt(s) Instance Name Port Name I perf_monitor_0 PERF_Irpt 2 debug_module Interrupt 3 Internal_BRAM ECC_Interrupt 4 Internal_BRAM ECC_UE Show Net Name Connected Interrupt(s) Priority Instance Name Port Name 0 axi_xadc_0 IP2INTC_Irpt 1 CVC_DISPLAY Interrupt 2 TPG_2_VDMA s2mm_introut C	Class
Unconnected Interrupt(s) Unconnected Interrupt(s) Connected Interrupt(s) Priority Instance Name Port Name I perf_monitor_0 PERF_Irpt debug_module Interrupt I CVC_DISPLAY	
Instance Name Port Name 1 perf_monitor_0 PERF_Irpt 2 debug_module Interrupt 3 Internal_BRAM ECC_Interrupt 4 Internal_BRAM ECC_UE	
1 perf_monitor_0 PERF_Irpt 2 debug_module Interrupt 3 Internal_BRAM ECC_Interrupt 4 Internal_BRAM ECC_UE	
2 debug_module Interrupt 3 Internal_BRAM ECC_Interrupt 4 Internal_BRAM ECC_UE 5 SCALER_2_VDMA	
3 Internal_BRAM ECC_Interrupt 4 Internal_BRAM ECC_UE 3 SCALER_2_VDMA s2mm_introut	
4 Internal_BRAM ECC_UE 3 SCALER_2_VDMA s2mm_introut	
5 Push_Buttons_5Bits IP2INTC_Irpt 4 SCALER_2_VDMA mm2s_introut	
6 LEDs_8Bits IP2INTC_Irpt 5 DVI_IN_1_VDMA s2mm_introut	
7 DIP_Switches_4Bits IP2INTC_Irpt 6 TPG_0_VDMA s2mm_introut	INTER
8 DVI_IN_0_VDMA mm2s_introut 🗸 7 timebase_0 irq 🗸 High	CLK
Help 🔀 Cancel 🔗 OK	
MSuperseded ODiscontinued	

Figure 1-22: Setting DVI_IN_1_VDMA_S2MM Interrupt priority

- 14. As shown in Figure 1-23 Double click **axi_mm_video34** to open the configuration wizard
- 15. Select Master Read/Write Settings and click the Master/Slave Specific Settings tab.
- 16. Set Write FIFO Delay for DVI_IN_1_VDMA.M_AXI_S2MM to 1 and press **OK**.

8		XPS C	ore Config - axi_mn	_video34 - axi_inter	connect_v1_06_a			×
Component Instance Name	xi_mm_video34							
General Control Interface	Master/Slave Specific Settings	Interconnect S	ettings for BUSIF					
Master Register Slices	Master	Read FIFO Depth	Write FIFO Depth	Read FIFO Delay	Write FIFO Delay	Read Issuing	Write Issuing	
Master Read/Write Settings	SCALER_2_VDMA:M_AXI_MM2S	θ (None) 💠	θ (None) 💠	1 \$	0 \$	4		Αυτο 📐
Slave ACLK & Other Settings	SCALER_2_VDMA:M_AXI_S2MM	θ (None) 💠	512 (BRAM) 💲	θ \$	1 🗘	AUTO 🔰 1	*	
Slave Register Slices	TPG_2_VDMA:M_AXI_S2MM	θ (None) 💠	512 (BRAM) 🗘	0 \$	1 🗘	AUTO 🔰 1	÷	_
Slave Read/Write Settings Slave Information	DVI_IN_1_VDMA:M_AXI_S2MM	θ (None) 💠	512 (BRAM) 🗘	θ \$	1 🗘	AUTO 📐		Αυτο 📐
							OK Car	icel Help

UG914_c1_23_071812

Figure 1-23: DVI_IN_1_VDMA Interconnect settings

At this point in the tutorial, additional IP from the Xilinx IP Catalog has successfully been added to the KC705 embedded kit MicroBlaze processor subsystem.

Implement and Test the Design

The design is ready to be synthesized, mapped, placed and routed in the chosen FPGA architecture.

- 1. Close the XPS project by selecting **File > Exit**.
- 2. Under Project Manager/Sources, click system_top STRUCTURE.
- 3. In the Flow Navigator, double click **Generate Bitstream** under the Program and Debug section, as shown in Figure 1-12, page 19.
- 4. Once the bit file is generated, Export hardware to SDK as explained in Export the BIST Hardware Platform to SDK, page 19.
- 5. Connect the FMC cards, HDMI cables, Ethernet cable and USB cables as shown in <u>UG913</u>, *Getting Started with the Kintex-7 FPGA KC705 Embedded Kit*, Kintex-7 FPGA Video Demonstration Hardware Setup.
- 6. Configure the FPGA with the generated bit file, KC705_Embedded_Kit/ Tutorial_Sandbox/HW/Video_Demo/MicroBlaze_Processor_Subsystem/ download.bit

- 7. Load and run the software ELF file, KC705_Embedded_Kit/Video_Demo/ ready_for_download/Video_Demo.elf
- 8. Open the webpage at IP 192.168.1.10 which shows the video demo web server running on the MicroBlaze processor subsystem.
- 9. Select different menu options on the web page and check the video output and data throughput for different possible options.

Note: For more details on Video Demo testing, different possible menu options and data throughput values refer to UG913, *Getting Started with the Kintex-7 FPGA KC705 Embedded Kit.*

Summary

The tasks accomplished in this tutorial are listed here:

- Loaded and executed a design from a pre-built bitstream and ELF
- Rebuilt a design
- Added IP from the Xilinx IP catalog to an embedded system

Next Steps

Next Step	Refer To
Develop the Software Platform	<u>UG915</u> , AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Software Tutorial
Advanced Debugging	UG029, ChipScope Pro Software and Cores User Guide and UG111, Embedded System Tools Reference Manual, GNU compiler tools
Simulate the Embedded System	UG683, EDK Concepts, Tools, and Techniques



Appendix A

Additional Resources

Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see the Xilinx Support website at:

www.xilinx.com/support.

For continual updates, add the Answer Record to your myAlerts:

www.xilinx.com/support/myalerts.

For a glossary of technical terms used in Xilinx documentation, see:

www.xilinx.com/company/terms.htm.

Solution Centers

See the <u>Xilinx Solution Centers</u> for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

Further Resources

The most up to date information related to the KC705 board and its documentation is available on the following websites.

The Xilinx Kintex-7 FPGA Embedded Kit product page:

www.xilinx.com/products/boards-and-kits/DK-K7-EMBD-G.htm

The Kintex-7 FPGA Embedded Kit - Known Issues and Release Notes Master Answer Record:

www.xilinx.com/support/answers/52970.htm

These documents provide supplemental material useful with this user guide:

- <u>UG913</u>, Getting Started with the Kintex-7 FPGA KC705 Embedded Kit
- <u>UG683</u>, EDK Concepts, Tools, and Techniques
- <u>DS669</u>, AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Data Sheet
- <u>UG111</u>, Embedded System Tools Reference Manual
- <u>UG915</u>, AXI Interface Based KC705 Embedded Kit MicroBlaze Processor Subsystem Software Tutorial
- <u>UG029</u>, ChipScope Pro Software and Cores User Guide

- <u>UG631</u>, ISE Design Suite 14: Release Notes, Installation, and Licensing
- <u>UG810</u>, KC705 Evaluation Board for the Kintex-7 FPGA User Guide
- <u>UG081</u>, MicroBlaze Processor Reference Guide Embedded Development Kit

References

AMBA AXI4-Stream Protocol Specification:

infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.ihi0051a/index.html