



Out-Of-Box GPIO Demo Example Design for the Arty Evaluation Board

Version 1.0

Page 1 Copyright © 2017 Avnet, Inc. AVNET, "Reach Further," and the AV logo are registered trademarks of Avnet, Inc. All other brands are the property of their respective owners. LIT# 5184-7A35T_Arty_OOB_GPIO_demo_V1

Document Control

Document Version:	1.0
Document Date:	8/12/2015

Prior Version History

Version	Date	Comment
1.0	8/12/2015	Initial release for Vivado 2015.2

Contents

1	Ov	verview	5
2	Ob	bjectives	5
3	Re	eference Design Requirements	5
3	.1	Software	5
3	.2	Hardware	5
4	Su	upplied Files	6
5	Set	etting Up the Arty Evaluation Board	7
6	PC	C Setup	8
6	5.1	Installing the UART Driver and Virtual COM Port	8
6	.2	Installing a Serial Console on a Windows 7 Host	8
7	Ru	unning the Demo Files	9
7	.1	Applications Download	10
7	.2	GPIO Demo	12
8	Cre	reate the MicroBlaze System	14
8	.1	Hardware Design Block Diagram	14
8	.2	Description of Hardware Modifications	15
8	.3	Vivado IP Integrator	16
	8.3.	3.1 Install Board Definition Files	
	8.3.	3.2 Create the Vivado IPI Block Design Project	16
	8.3.	3.3 Import and Recreate the Block Design	
	8.3.	3.4 Implement the Design	
	8.3.	3.5 Review the Bitstream Settings	24
-	8.3.	3.6 Export Vivado Hardware Design to the SDK	
9	Cre	reate the Software Applications	
9	.1	Add Custom Repository	
9	.2	Create the GPIO Demo software application	
	9.2.	2.1 Examine the GPIO Demo Linker Script	
10	9.2. Dra	2.2 Running the GPIO Demo Software Application	
10	Pro	Ogram the GPIO Demo Application in QSPI Flash	
1	0.1	Program the QSPI Flash Using the Vivado Hardware Manager	
	10.1	1.1.1 Prepare the QSPI Programming File	
1	10.1 0.2	Program the OSPI Flash Using the SDK	
11	0.2 (Ani	nogram the Continuant only the ODK	
	' 'PI		

12	Арр	pendix II: Installation of Board XML Files	47
13	Арр	pendix III: Windows 260 Character Path Limit	48
14	Арр	pendix IV: Getting Help and Support	49
14	4.1	Avnet Website	.49
14	4.2	Xilinx Website	.49
14	4.3	Digilent Website	.49

1 Overview

This document describes a simple MicroBlaze[™] design implemented and tested on the Avnet/Digilent Arty Evaluation Board. This example design integrates pushbutton and DIP switch user input with a custom Pulse Width Modulator (PWM) peripheral to manipulate the brightness and display pattern of LEDs on the board This example design demonstrates continuously reading the DIP switches and using that value to manipulate the brightness of LEDs on the board via a custom PWM peripheral. The lower the value read from the DIP switches, the dimmer the LEDs become.

This tutorial shows how to build a MicroBlaze Hardware Platform and then create, build, and run a software application on the Avnet/Digilent Arty Evaluation Board.

2 **Objectives**

When you have completed this tutorial, you will know how to do the following:

- Build a MicroBlaze hardware platform integrating a custom IP peripheral.
- Set up an SDK workspace.
- Add an example software application.
- Run the example hardware and software design to manipulate the LED brightness.
- Program the QSPI Flash memory.

3 Reference Design Requirements

The following items are required for proper completion of this tutorial.

3.1 Software

The software requirements for this reference design are:

- Linux, Windows 7, Windows 8.1 (ug973-vivado-release-notes-install-license.pdf)
- Xilinx Vivado Design Edition 2015.2, with SDK

3.2 Hardware

The hardware setup used by this reference design includes:

- Computer with 1 GB RAM and 1 GB virtual memory (recommended) (www.xilinx.com/design-tools/vivado/memory.htm)
- Avnet/Digilent Arty Evaluation board
- USB-A to USB-micro B cable
- Power supply (optional)

4 Supplied Files

The following directory structure is included with this reference design:

demo: Contains the files to run receive test program: config_fpga.bat: Batch file to configure the FPGA with the "bootloop" bitstream of the MicroBlaze processor system. cp from sdk.bat: Batch file to copy hardware bitstream and software executables from the SDK workspace. demo_gpio_app.bat: Batch file to run the commands to load the FPGA bitstream of the hardware design and download the software application. **design 1 wrapper.bit:** The golden FPGA bitstream of the hardware design. design 1 wrapper.mmi: The golden BRAM map file to integrate the bootloop executable with the FPGA bitstream. **download.bit:** The golden FPGA bitstream integrated with the bootloop application. download_bit.tcl: TCL command file for downloading the bitstream to the board. gpio demo.elf: The golden MicroBlaze executable for the GPIO demo application. **load_bits.tcl:** TCL command file for downloading the bitstream to the board. sleep.exe: Utility to pause execution of the batch file. xmd.ini: Command file used by XMD to download start execution of the software application.

- doc: Contains the documentation for this reference design.
 7A35T_Arty_OOB_GPIO_demo_VIV2015_2.pdf: This document.
- **IPI:** Batch file and Vivado TCL scripts to create new Vivado project and create MicroBlaze system block design.

IPI_repo: Repository of files and IP needed to create the MicroBlaze hardware platform.

program_flash: Batch file and Vivado TCL scripts to program the QSPI Flash memory.

sdk_repo: Repository for the GPIO demo software application source files and BSP settings.

sdk_workspace: Empty folder to use to create new SDK workspace for this design.

IPI_solution.zip: Contains pre-built Vivado IPI project for this design.

sdk_solution.zip: Contains pre-built SDK project for this design.

5 Setting Up the Arty Evaluation Board

Refer to the following figure and perform the following steps to set up the board for running the example design.



1. Plug a USB cable into the PC and the combination JTAG & UART port (J10) (this will also power the board). If the USB port on the hub or host PC cannot supply enough power you may alternatively power the board via the 12V power barrel jack (J12).

6 PC Setup

6.1 Installing the UART Driver and Virtual COM Port

The USB UART driver is built into the device driver for the JTAG interface and is included with the Xilinx Vivado tools installation. As long as the Vivado tools are installed, the USB UART will be recognized when the board is plugged into the host PC.

See <u>Appendix I: Determining the Virtual COM Port</u> for information on identifying the COM port in use on the host PC.

6.2 Installing a Serial Console on a Windows 7 Host

Starting with Windows 7, Microsoft no longer includes the HyperTerminal terminal emulator software. However, this example design requires use of terminal emulation software for a serial console connection to the Arty board. A suitable free and open-source replacement for HyperTerminal is TeraTerm. Download and install instructions for TeraTerm can be found at http://en.sourceforge.jp/projects/ttssh2. As an alternative the Terminal applet in the Xilinx SDK may also be used.

7 Running the Demo Files

You can load the FPGA and run the software application without building the design by using the demo scripts and the pre-built bitstream and elf files. You must have the Xilinx tools (including the SDK) installed on your host, and have the hardware set up and connected as described in <u>Setting Up the Arty Evaluation Board</u>.

Refer to the figure below when running the GPIO demo software application. Note the location of the pushbutton switches, DIP switches, and LEDs.



7.1 Applications Download

- 1. Start a serial terminal session and set the serial port parameters to **115200** baud rate, **no** parity, **8** bits, **1** stop bit and no flow control.
- 2. Open a command window in the **<installation>\demo** folder and enter:

demo_gpio_app.bat

3. The FPGA bitstream will be downloaded, followed by the executable file for the software application. Do not close the command window.

```
x
Administrator: C:\Windows\system32\cmd.exe
  **** SW Build 1266856 on Fri Jun 26 16:35:25 MDT 2015
                                                                            ٠
  **** IP Build 1264090 on Wed Jun 24 14:22:01 MDT 2015
   ** Copyright 1986-2015 Xilinx, Inc. All Rights Reserved.
                                                                            Ξ
source C:/Xilinx/Vivado/2015.2/scripts/updatemem/main.tcl -notrace
Loading bitfile design_1_wrapper.bit
Loading data files...
Updating memory content...
Creating bitstream...
Writing bitstream download.bit...
update_mem: Time (s): cpu = 00:00:12 ; elapsed = 00:00:12 . Memory (MB): peak =
469.137 ; gain = 410.398
INFO: [Common 17-206] Exiting updatemem at Fri Aug 14 16:08:11 2015...
Configuring FPGA
****** Xilinx Microprocessor Debugger (XMD) Engine
****** XMD v2015.2 (64-bit)
 **** SW Build 1266856 on Fri Jun 26 16:35:25 MDT 2015
   ** Copyright 1986-2015 Xilinx, Inc. All Rights Reserved.
Executing user script : download_bit.tcl
Configuring Device 1 (xc7a35t) with Bitstream -- download.bit
...10..20...30..40..50...60..70..80...90..100Successfully downloaded bit file.
JTAG chain configuration
 Device ID Code IR Length Part Name
        0362d093
1
                     6 xc7a35t
C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_OOB_GPI0_demo_VIV2015_2\demo>
```

4. When the executable has finished loading and is ready to run you should see the following in your serial terminal window:



7.2 GPIO Demo

The GPIO demo software application allows the user to interact with the pushbutton and DIP switches on the board to change the display on the LEDs.

 Press the **BTN0** pushbutton switch to select the option to print the PWM value that controls the brightness of the LEDs. The PWM value is a calculation of the value read from the DIP switches (SW0-SW3). The DIP switches represent a hexadecimal value. The computed result is used as input to the PWM that controls the brightness of other LEDs on the board (LD4-LD7). The software prints the new PWM value each time the value read from the DIP switches changes. Notice that the LEDs on the board get dimmer and brighter as the DIP switches are toggled. The DIP switch value is multiplied by 20,000 and written to the PWM peripheral to modulate the brightness of the LEDs. The greater the computed value the brighter the LEDs will be. Do not close the command window. Press the **BTN3** switch to return to the main menu.

ECOM18:115200baud - Tera Term VT	
File Edit Setup Control Window Help	
	^

** Choose Task: BTN0: Print PWM value. BTN1: 'Cylon' LED display. BTN2: Scrolling LED display. BTN3: Return to this menu.	
Switch BTN0 pressed. Printing PWM value. Press BTN3 to exit. Toggle DIP switches to change LED brightness. LED PWM Value = 300000	-

2. Press **BTN1** to select the pattern to display a 'Cylon' (walking back and forth) pattern on the RGB LEDs (LD0-LD3). The color will also change as the LEDs cycle back and forth. Do not close the command window. Press the **BTN3** switch to stop the display and return to the main menu.

💆 COM18:115200baud - Tera Term VT	x
File Edit Setup Control Window Help	
	*

Choose Task: BTN0: Print PWM value. BTN1: 'Cylon' LED display. BTN2: Scrolling LED display. BTN3: Return to this menu.	
Switch BTN1 pressed. Display 'Cylon' pattern on RGB LEDs in R,G,B sequence. Press BTN3 to exit.	-

Press BTN2 to select the scrolling color LED display on the RGB LEDs (LD0-LD3). The RGB LEDs will scroll through the combination of red, green, and blue for each LED. Press the BTN3 switch to return to the main menu.

🚇 COM18:115200baud - Tera Term VT	x
File Edit Setup Control Window Help	
	^

Choose Task: BTN0: Print PWM value. BTN1: 'Cylon' LED display. BTN2: Scrolling LED display. BTN3: Return to this menu.	
Switch BTN2 pressed. Scroll through each combination of R,G,B on LEDs. Press BTN3 to exit.	•

8 Create the MicroBlaze System

8.1 Hardware Design Block Diagram

The following figure shows a high-level block diagram of the hardware design. The design requires:

- MicroBlaze processor
- 32KB of BRAM
- UART Port
- GPIO for LEDs and switches
- QSPI Flash
- Interrupt Controller
- Custom PWM peripheral



Figure 1 – Reference Design Block Diagram

8.2 Description of Hardware Modifications

This design was created using the Vivado IP Integrator Block Design flow with the following modifications:

- MicroBlaze processor settings:
 - Select the Typical Predefined Configuration
 - Specify 32KB for local memory size
- AXI_QSPI settings
 - Enable Quad mode
 - Set the slave device to Micron (Numonyx)
 - Disable the STARTUP2E primitive
- Clocking Wizard settings
 - Set clk_out1 to 100MHz (system clock)
 - Disable the reset input
- AXI_UARTLite
 - Set baud rate to 115200

8.3 Vivado IP Integrator

Vivado IP Integrator (IPI) provides a rich graphical environment in which to create and customize MicroBlaze processor systems. The integrated TCL command window allows for running simple commands. In fact, most functions and tasks in the Vivado GUI are run as TCL commands. The TCL command window can also be used to automate complex tasks like creating a MicroBlaze system from scratch that is capable of running the software application. Follow the steps below to import and implement a pre-built known-good MicroBlaze system block design.

8.3.1 Install Board Definition Files

1. If you haven't already done so, please follow the instructions in <u>Appendix II</u> to install the board definition XML file set for the Arty evaluation board. The board definition file identifies the interfaces on the Arty board and allows us to target the board directly in the Vivado tools by selecting it from a menu.

5, Vivedo 2015.2	×
Elle Flow Iools Window 1940	Q Search commands
VIVADO. Productivity. Multiplied.	XILINX ALL PROGRAMMABLE.
Quick Start	
Create New Project Open Project	
Tasks	
Manage IP Open Hardware Manager Xilox Td Store	
Information Center	
Documentation and Tutorials Quick Take Videos Release Notes Guide	
Td Console	- 0 C ×
Type a Tol command here	

8.3.2 Create the Vivado IPI Block Design Project

1. Navigate to the Windows Desktop or Start menu and launch Vivado.

2. If you are using the Vivado Design Suite on a Windows[®]-7 host, there is a known issue that Vivado file path names often exceed the maximum allowed by Windows. Please see Appendix III for more details. A workaround for this is to change the temp folder that Vivado uses. Create your own temporary directory named C:\temp, and force Vivado to use the new folder with the following TCL command:

set_param "pr	roject.customTmpDirForArchive"	C:/temp
---------------	--------------------------------	---------

Td (Conso	ske	_ D & X
「「御御田御郎×		me_param "project.customTmpDirForArchive" C:/temp C:/temp	
		€	

2. Another workaround for the 260 character path length limitation is to use the Windows subst command to substitute a drive letter for a long file path. Substitute the drive letter V: (or any other unused drive letter on your Windows PC) for the path where the zip archive of this example design was extracted. This can be done in the Vivado TCL Console:

cd <installation>
exec subst V: .

Td Console	- 0 0 ×
<pre>\$ set_param "project.customTmpDirForArchive" C:/temp C:/temp od /Projects/DRC_Dev/7A3ST_Arty/7A3ST_Arty_008_GPI0_demo_VIV2015_2 exec subst v: . * * * * * * * * * * * * * * * * * *</pre>	*
Type a Tcl command here	

8.3.3 Import and Recreate the Block Design

1. At this point we can create the Vivado project and import the pre-built MicroBlaze processor system block design. Open the **TCL Console** by selecting the tab at the bottom of the Vivado window and enter the following command to import the block design:

cd ./IPI source ./design_1.tcl



2. You will see the following window as the block design is imported and recreated. This process may take a few minutes.

😓 Tcl Command Line	×
X Running create_bd_cell	<u>C</u> ancel
	Background

- 3. Open the **design_1.tcl** file in your favorite text editor and identify the important Vivado tasks that the script automates for us:
 - Create the new project in the **project_1** folder and select the **XC7A35T** FPGA device
 - Select the **Arty** as the target board
 - Set the custom IP repository folder location to ../IPI_repo/ip_repo
 - Build the MicroBlaze processor system block design
 - Generate the top-level HDL wrapper for the block design
 - Add the **design_1.xdc** constraints file to the design

3.6	# Create the new project in the project 1 folder and select the XC7135T FDGL device
82	reate moder project i form / moder i _mart wolationadd.it
100	constraints and a state of the second state of the
35	# Select the Artix-7 XC7A15T 'Arty' as the target board
40	set property board part digilenting.compartypart011.1 [current project]
41	
42	# Set the custom IP repository folder location to/IPI_repo/ip_repo
45	set param bd.enableIpSharedDirectory true
6.6	set repos local "./, ./IFI repo/ip repo"
45	set property ip repo paths "\$(repos local)" [current fileset]
56	update ip catalog -rebuild
47	
4.5	# Build the MicroBlaze processor system block design
49	source design 1_bd.tcl
50	generate_target (synthesis simulation implementation) [get_files ./project_l/project_l.sros/sources_l/bd/design_l/design_l.bd]
51	Control Control of the second s
52	.# Generate the top-level HDL wrapper for the block design
53	make_wrapper -files [get_files ./project_1/project_1.arcs/sources_1/bd/design_1/design_1.bd] -top
54	add files -norecurse ./project_l/project_l.srcs/sources l/bd/design_l/hdl/design_l_wrapper.v
5.5	update_compile_order -fileset sources_1
5.6	update compile order -fileset sim 1
53	
58	# Add the design 1.xdc constraints file to the design
5.9	add files -fileset constrs 1 -norecurse .//IPI repo/xdc/design 1.xdc
60	import_files -fileset constrs_1 .//IFI_repo/xdc/design_1.xdc



4. When the design is done building you should see a Vivado window similar to this:

5. The Block Design should look similar to this:



6. Right click in the block design **Diagram** window and select **Validate Design**. There should not be any errors or warnings. Click **OK** to continue.

15	Properties	Ctrl+E
×	Delete	Delete
h.	Сору	Ctrl+C
nh.	Paste	Ctrl+V
٩	Search	Ctrl+F
R	Select All	Ctrl+A
₽	Add IP	Ctrl+I
6	IP Settings	
	Validate Design	F6
V	Validate Design Expand/Collapse	F6
	Validate Design Expand/Collapse Create Hierarchy	F6
	Validate Design Expand/Collapse Create Hierarchy Create Comment	F6 ▶
•	Validate Design Expand/Collapse Create Hierarchy Create Comment Create Port	F6
	Validate Design Expand/Collapse Create Hierarchy Create Comment Create Port Create Interface Port	F6 Ctrl+K Ctrl+L
Ø	Validate Design Expand/Collapse Create Hierarchy Create Comment Create Port Create Interface Port Regenerate Layout	F6 Ctrl+K Ctrl+L

8.3.4 Implement the Design

1. We are now ready to build the design. Navigate to the **Project Manager** pane and click **Generate Bitstream**. This will run all previous steps to generate the IP netlists, synthesize the design, and run implementation. You will see a window reminding you that synthesis hasn't been completed yet. Click **OK** to continue.



2. Click **Save** if you are prompted to save the block design.

0	Save proje	t before a	enerating bit	stream?	
Data to	Save				
	7 <u>B</u> lock Desig	ın - design	_1		
			Save	Don't Save	Cancel

3. You will see a window reminding you that synthesis and implementation hasn't been completed yet. Click **Yes** to continue. This will take a few minutes.



4. When prompted select **Open Implemented Design** and click **OK**:

Bitstream Generation Completed					
Bitstream Generation successfully completed.					
Open Implemented Design					
View Reports					
Open <u>H</u> ardware Manager					
Don't show this dialog again					
OK Cancel					

8.3.5 Review the Bitstream Settings

In addition to specifying the system I/O and timing constraints, the system constraints file, design_1.xdc, also specifies the parameters for creating the FPGA configuration bitstream. Because the Arty Evaluation board is configured via a Quad SPI flash memory we must be sure to create a properly formatted bitstream for this interface. Also, the default behavior of the bitstream generation tool is to create an internal pulldown on all unused pins. This can sometimes cause unintended consequences for board interfaces like the DDR3 memory. Follow the steps below to review the bitstream generation options.

1. Navigate to the **Sources** tab and expand the **Constraints** branch until the **design_1.xdc** constraints file is visible. Double-click to open the file.

Sources _ 🗆 🗠 🗵				
🔍 🔀 🖨 📑 📳 🛃				
🖃 🔂 Design Sources (1)				
.v) (1)				
🛱 🖓 Constraints (1)				
🖻 🔂 constrs_1 (1)				
Hierarchy IP Sources Libraries Compile Order				
👃 🖧 Sources 🔄 Design 🖉 🔤 Signals 🖉 📓 Board				

2. Scroll down to near the end of the file and locate the **#bitgen settings** section. These are the constraints and parameters that instruct Vivado to create a properly formatted bitstream for the Arty Evaluation board.

dc ^
^
=
-
F

3. These constraints are actually created by selecting options in the Bitstream Settings GUI. With the implemented design open in memory, go to the **Flow Navigator** pane and scroll to the bottom to find and select **Bitstream Settings** under **Program and Debug.**

Program and Debug						
	🏀 Bitstream Settings					
	🔚 Generate Bitstream					
\triangleright	💕 Open Hardware Manage					

4. In the Project Settings window click on **Configure additional bitstream settings**.



5. In the **Edit Device Properties** window select the **Configuration** tab and compare the settings set in the GUI to those specified in the system constraints file. Click **OK** when done:

e vis davy w exit ine pro	ng ann mg ann um ngur ann i progen aes na ' pror current seogrif, seraun raises are set ann martait.	
t+	Configuration	
eneral ordiguration	Configuration Setup	
nfiguration Modes artup cryption	Configuration Rate (Mriz) 66 • C	
adback	Enable external configuration dock and set divide value DISABLE * C	
	Configuration Voltage 3.3 - C	
	Configuration Bank Voltage Selection	
	BPI Configuration	
	Ist Read cycle	
	Page Size (bytes) 1 * C	
	Synchronous Mode DISABLE + C	
	SPI Configuration	
	Enable SPI 32-bit address style	
	Bus width	
	Enable the PPGA to use a failing edge dock for SPI data capture $\fbox{125}$	
	MultBoot Settings	
		11

8.3.6 Export Vivado Hardware Design to the SDK

 All software aspects of the design are performed inside a SDK Workspace. To generate an empty workspace based on the hardware platform built in Vivado navigate to File → Export → Export Hardware from the Vivado GUI. Accept the default
 <Local to Project> export directory location and check the box to Include bitstream then click OK to continue.

🚴 Export Hardware
Export hardware platform for software development tools.
Include bitstream
Export to: 🛜 <local project="" to=""> 👻</local>
OK Cancel

2. Navigate to **File** → **Launch SDK** from the Vivado GUI. In the Launch SDK window click on the Workspace drop list and select **Choose Location**.

🚴 Launch SDK					
Launch software development tool.					
Exported location: 🛜 <local project="" to=""> 👻</local>					
Workspace: Si <local project="" to=""> Si <local project="" to=""></local></local>					
Choose Location					

3. Navigate to the <installation>\sdk_workspace folder to create a new SDK workspace. Make sure there are NO SPACES in this path. The Xilinx SDK does not tolerate spaces in this file path. Click Select and then click OK to continue.

Choose Loca		
		▶ & ■ ■ X Ø ☆ 3
	Production Product	
	IPI_repo	
	ig- ig- ig- grag-flash ig- ig- ig- sdk_repo	
	sdk_workspace	Select Cancel

4. After a few seconds the SDK will show a GUI similar to the one shown below:

		Ouick Acce		
Design of the second	(Internet in the second	QUICK ALCE		
Project topioner (A) Project topioner (A) B % V V design_1_wrapper_hw_platform_0 B & drivers	design_1_wrapper_hw_platform_0 Hardware Platform Specif	fication	An outline is not available.	
 design_l_bd.tcl design_l_wrapper.bit design_l_wrapper.mmi system.hdf 	Target FPGA Device: 7a35ti Created With: Vivado 2015.2 Created On: Fri Aug 14 15:06:27 2015 Address Map for processor microblaze_0	1		
	PWM_w_int_0 0x44a00000 0x44a0ffff xni_gpio_0 0x40000000 0x4000ffff axi_gpio_1 0x40010000 0x4000ffff axi_guid_spi_0 0x44a10000 0x400ffff axi_guid_spi_0 0x44a10000 0x40a0ffff microblaze_0_xii_intc 0x41200000 0x40a0ffff microblaze_0_local_memory_dlmb_bram_if_cntbr 0x00000000 0x00007fff microblaze_0_local_memory_ilmb_bram_if_cntbr 0x00000000 0x00007fff IP blocks present in the design			
	PWM_w_int_0 PWM_w_int 1.0 axi_gpio_0 axi_gpio 2.0 axi_gpio_1 axi_gpio 2.0			
🖨 Terget Connections 🐹 🍰 💭 😁 🗖	🛃 Pro 🕅 🧑 Tesks 🔲 Con 🛄 Pro 🧬 Ter 😁 🗔 📗 SDK Log 🕅		la la 🖷 🗆	
S G Hardware Server S G Linux TCF Agent S G QEMU TcfGdbClient	0 items 0 escription 0 items 0	IFO 1 Launchi IFO 1 Process IFO 1 XSDB se	ing XSOB server: xsdb.bat + sing command line option erver has started success	

9 Create the Software Applications

The next steps are to create the GPIO demo and SPI SREC bootloader software applications.

9.1 Add Custom Repository

Adding a custom repository to the SDK workspace is a convenient way to integrate software libraries and applications into your software development. We need to add a repository to the SDK workspace for the GPIO demo software application. Including the GPIO demo application in a repository greatly simplifies the task of adding new software sources or application projects to the SDK workspace.

1. In the SDK GUI navigate to **Xilinx Tools** \rightarrow **Repositories.**



2. Under **Local Repositories** click on **New** to create a repository that will only be visible and usable by this particular SDK workspace.

sok Preferences		
type filter text	Add, remove or change the order of SDK's software repositories.	$\phi \bullet \phi \bullet \bullet$
▷ General ▷ Ant	Local Repositories (available to the current workspace)	
⊳ C/C++		New
 Help Install/Update 		Remove

3. Navigate to <installation>\sdk_repo and click OK.

Browse For Folder	×
Choose a repository directory. A repository directory typ the 'drivers', 'bsp' or 'sw_services' sub-directories.	vically contains
A 35T_Arty_OOB_GPIO_demo_VIV	2015_2
🍑 demo	
🌙 doc	
Þ 🍌 IPI	
⊳ 🍌 IPI_repo	
program_flash	
> 🔒 sdk_repo	
> 🔒 sdk_workspace	*
Folder: sdk_repo	
Make New Folder OK	Cancel

4. Click on **Rescan Repositories**, then click **Apply**, then click **OK**. The new repository is now ready to use.

son Preferences					
type filter text	Add, remove or change the order of SDK's software repositories.	⇔ ▼ ⇒ ▼			
b General	Local Repositories (available to the current workspace)				
▷ Ant ▷ C/C++	C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_OOB_GPIO_demo_VIV2015_2\sdk_repo	New			
⊳ Help					
Install/Update		Remove			
 Java Remote Systems 		Up			
Run/Debug		Down			
⊳ Team		Palativa			
Terminal		Relative			
Boot Image	Global Repositories (available across workspaces)				
BSP Preferences		New			
Flash Programming Hardware Specification		Remove			
Log Information Level		Up			
XMD Startup		Down			
	SDK Installation Repositories				
	C:/Xilinx/SDK/2015.2/data\embeddedsw				
	Descent Descention in a				
	Rescan Repositories				
	Nate Local reportery attings take proceedings over alphal reporting attings				
	Note, Local repository settings take precedence over global repository settings.				
	Restore Defaults Apply				
(?)	ОК	Cancel			

9.2 Create the GPIO Demo software application

1. Go to File → New → Application Project

ok C/	′C++ -	design_1	_wrapper_h	nw_platform	n_0/syster	m.hdf - Xil	inx SDK		
File	Edit	Source	Refactor	Navigate	Search	Project	Xilinx T	ools	Run Window Help
	New					Alt+Sł	nift+N ▶	9	Application Project
	Open	File							SPM Project
	Close	A11				Ctrl+Sh	trl+W		Board Support Package Project

2. Name the project **gpio_demo** and accept the **default location** and **Target Hardware** settings. Accept the default **OS Platform** and **Language**. Accept the default to let the SDK Create New board support package named **gpio_demo_bsp**. Click **Next** to continue.

New Project	
Application Project Create a managed make application project.	F
Project name: gpio_demo	
Use generic rotation Location: C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_OOB_GPIO_de Choose file system:	B <u>r</u> owse
OS Platform: standalone	•
Target Hardware Hardware Platform: design 1 wrapper hw platform 0	▼ New
Processor: microblaze_0	
Target Software Language: ○ C ○ C++ Board Support Package: ○ Create New ○ gpio_demo_bsp ○ Use existing 	Ţ
(?) < <u>B</u> ack <u>Next</u> > <u>Finish</u>	Cancel

3. Select the **GPIO Demo Application** template and click **Finish** to continue. This application appears in this list because it is in the repository we added earlier. The source code for the applications and BSP, along with the BSP settings, will be added to the workspace and compiled.

sox New Project	
Templates Create one of the available templates to gener application project.	rate a fully-functioning
Available Templates:	
Peripheral Tests GPIO Demo Application Dhrystone Empty Application Hello World IwIP Echo Server Memory Tests RSA Authentication App SREC Bootloader SREC SPI Bootloader Xilkernel POSIX Threads Demo Zynq DRAM tests Zynq FSBL	This application demonstrates interacting with the push buttons to control the board LEDs and manipulate the brightness of LEDs via DIP switches and a PWM.
Reck Net	t > Finish Cancel

4. When compilation is complete the SDK should display a Console tab similar to this:

🖹 Problems 🕢 Tasks 📮 Console 🛛 🗐 Properties 🖉 Terminal 📗 SDK Log	
CDT Build Console [gpio_demo]	
'Invoking: MicroBlaze Print Size'	*
mb-size gpio_demo.elf tee "gpio_demo.elf.size"	
text data bss dec hex filename	
15004 436 2272 17712 4530 gpio_demo.elf	
'Finished building: gpio_demo.elf.size'	*

9.2.1 Examine the GPIO Demo Linker Script

1. Navigate the **gpio_demo** software application project source tree in the **Project Explorer** pane and double-click on the **Iscript.Id** linker script:



2. Notice that the **Stack Size** and Heap Size are both set to **0x400 (1KB)** and that all of the code sections are mapped to be located in **BRAM** (you will need to scroll the window to see all of the code sections). Click the a on the file tab to close the file.

👔 system.hdf 🛛 👔 system.mss	🛐 lscript.ld 🖾		
Linker Script: lscript.ld			
A linker script is used to control where d In this page, you can define new memor	lifferent sections of an execu ry regions, and change the a	table are placed in me ssignment of sections	emory. to memory
Available Memory Regions			
Name		Base Address	Size
microblaze_0_local_memory_ilmb_bra	am_if_cntlr_microblaze_0	0x0000050	0x00
Stack and Heap Sizes			
Stack Size 0x400			
Heap Size 0x400			
Section to Memory Region Mapping			
Section Name	Memory Region		
.text	microblaze_0_local_me	emory_ilmb_bram_if_c	ntlr_microb
.init	microblaze_0_local_me	emory_ilmb_bram_if_c	ntlr_microb
.fini	microblaze_0_local_me	emory_ilmb_bram_if_c	ntlr_microb
.ctors	microblaze_0_local_me	emory_ilmb_bram_if_c	ntlr_microb
		10.00	

9.2.2 Running the GPIO Demo Software Application

Because the MicroBlaze processor system is configured with 32KB of BRAM, this is enough to hold the gpio_demo software application. The FPGA can be configured with the hardware bitstream merged with this software application, and immediately begin software execution. No need for an intermediate bootloader. We will configure the FPGA now with our hardware bitstream that includes the gpio_demo software application.

- 1. If not already done, connect the Arty Evaluation board to the host PC as described earlier in <u>Setting Up the Arty Evaluation Board</u>.
- 2. Start a serial terminal session using your terminal software of choice and set the serial port parameters to **115200** baud rate, **no** parity, **8** bits, **1** stop bit and no flow control.
- 3. In the SDK main menu, select Xilinx Tools → Program FPGA or click on the and the SDK toolbar:

File Edit Source	Refactor Navigate S	Search Project	Xilinx Tools Run Window Help
	§ ▼ % ▼ 🗟 🥃 ŧ	🛱 🐹 🖬 🔽 ·	 Repositories Board Support Package Settings
Project Explorer	3 - 8	system.hdl	S Generate linker script

4. Accept the defaults for the **Hardware Platform** and **Connection** and verify that the **gpio_demo** application is selected as the **ELF/MEM File to Initialize in Block RAM**. Click **Program** to continue:

SOK Program FPGA			×			
Program FPGA Image: Specify the bitstream and the ELF files that reside in BRAM memory						
Hardware Configura	tion		_			
Hardware Platform:	design_1_wrapp	er_hw_platform_0	•			
Connection:	Local		New			
Device:	Auto Detect		Select			
Bitstream:	design_1_wrapp	Search Browse				
Partial Bitstream						
BMM/MMI File:	design_1_wrapp	per.mmi	Search Browse			
Software Configurati	ion					
Processor		ELF/MEM File to Initialize in	Block RAM			
design_1_i/microbla	aze_0	.sdk_workspace\gpio_demo	\Debug\gpio_demo.elf 👻			
		III	+			
?		Prog	ram Cancel			

5. You will see the following window while the FPGA bitstream is downloaded:

Sor Progress Information	
Initializing Bitstream with ELF data	·
	Cancel Details >>

6. You should see the following on the serial console:



7. You are now ready to run the GPIO demo software application. The steps to run the application are the same as <u>running the demo</u> you probably used earlier, except the steps of downloading the bitstream and application executable are already completed.

10 Program the GPIO Demo Application in QSPI Flash

Starting in version 2015.1, the Xilinx SDK now has the capability to program the QSPI flash from the SDK GUI. Previously the Vivado Hardware Manager had to be used for Flash memory programming tasks. Both methods are described here for reference.

10.1 Program the QSPI Flash Using the Vivado Hardware Manager

10.1.1 Prepare the QSPI Programming File

All the components have been assembled at this point, and the last step is to create a FPGA boot image. We will use the Vivado tools to create a boot image for the Artix-7 FPGA combining the FPGA bitstream (with integrated SPI SREC bootloader executable in BRAM) and GPIO demo executable into a single **boot.mcs** Flash programming file.

1. Go back to the command window you opened earlier for <u>running the demo</u>. If you closed that command window you can open another from the SDK. In the SDK GUI navigate to



Xilinx Tools \rightarrow **Launch Shell** or click the \boxed{D} icon on the toolbar.

2. Change directories to the **program_flash** folder:

cd ..\program_flash



3. Run the **cp_from_sdk.bat** batch file to copy the needed files from their locations in the folders of the SDK workspace:

cp_from_sdk.bat

X Administrator: C:\Windows\system32\cmd.exe Microsoft Windows [Version 6.1.7601] ٠ Copyright (c) 2009 Microsoft Corporation. All rights reserved. Ξ C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_00B_GPI0_demo_VIV2015_2\sdk_workspace> cd ..\program_flash C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_OOB_GPIO_demo_VIV2015_2\program_flash> cp_from_sdk.bat C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_00B_GPI0_demo_VIV2015_2\program_flash> set SDK_PATH=..\SDK_Workspace C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_00B_GPI0_demo<u>_VIV2015_2\program_fla</u>sh> copy /Y ...SDK_Workspace\design_1_wrapper_hw_platform_0\design_1_wrapper.bit . 1 file(s) copied. C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_00B_GPI0_demo_VIV2015_2\program_flash> copy /Y ..\SDK_Workspace\design_1_wrapper_hw_platform_0\design_1_wrapper.mmi . 1 file(s) copied. C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_OOB_GPI0_demo_VIV2015_2\program_flash> copy /Y ...\SDK_Workspace\gpio_demo\Debug\gpio_demo.elf . 1 file(s) copied. C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_OOB_GPI0_demo_VIV2015_2\program_flash>

- 4. Run the **make_mcs.bat** batch file to create the boot.mcs file needed to program into QSPI Flash. This batch file assumes the Vivado tools have been installed in their default locations and checks to see if the required system environment variables have been set. You will need to edit the batch file if your installation of the Vivado tools is in a different folder. The batch file will run the following commands:
 - Merge the gpio_demo.elf with the FPGA bitstream of the MicroBlaze system:

```
cmd /C updatemem -force -meminfo design_1_wrapper.mmi -bit
design_1_wrapper.bit -data gpio_demo.elf -proc
design_1_i/microblaze_0 -out bootload.bit
```

Start Vivado in batch mode to run the TCL script to create the boot.mcs QSPI programming file:

```
call C:\\Xilinx\\Vivado\\%VIVADO_VER%\\.\\bin\\vivado.bat -
mode batch - source make_mcs.tcl
```

- The **make_mcs.tcl** file:

```
write_cfgmem -force -format MCS -size 16 -interface SPIx4 -
loadbit " up 0 ./bootload.bit" boot.mcs
```

23 Administrator: C:\Windows\system32\cmd.exe INFO: [Common 17-206] Exiting updatemem at Fri Aug 14 15:50:38 2015... ۰ ****** Vivado v2015.2 (64-bit) **** SW Build 1266856 on Fri Jun 26 16:35:25 MDT 2015 **** IP Build 1264090 on Wed Jun 24 14:22:01 MDT 2015 ** Copyright 1986-2015 Xilinx, Inc. All Rights Reserved. source make_mcs.tcl # write_cfgmem -force -format MCS -size 16 -interface SPIx4 -loadbit " up 0 ./bo otload.bit" boot.mcs Creating config memory files... Creating bitstream load up from address 0x00000000 Loading bitfile ./bootload.bit Writing file ./boot.mcs Writing log file ./boot.prm Configuration Memory information _____ File Format MCS Interface SPIX4 Size 16M 0x00000000 Start Address End Address 0x00FFFFFF Addr1 Addr2 Date File(s) 0x00000000 0x0021728B Aug 14 15:50:38 2015 ./bootload.bit INFO: [Common 17-206] Exiting Vivado at Fri Aug 14 15:50:43 2015... C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_00B_GPI0_demo_VIV2015_2\program_flash>

10.1.2 Program the QSPI Flash Using a TCL Script

- 1. If not already done, connect the Arty Evaluation board to the host PC as described earlier in <u>Setting Up the Arty Evaluation Board</u>.
- 2. Start a serial terminal session using your terminal software of choice and set the serial port parameters to **115200** baud rate, **no** parity, **8** bits, **1** stop bit and no flow control
- 3. Run the **program_qspi.bat** batch file to program the QSPI Flash. This batch file assumes the Vivado tools have been installed in their default locations and checks to see if the required system environment variables have been set. You will need to edit the batch file if your installation of the Vivado tools is in a different folder. The batch file will run the following command:
 - Start Vivado in batch mode to run the TCL script to program the boot.mcs file to QSPI Flash:

```
call C:\\Xilinx\\Vivado\\%VIVADO_VER%\\.\\bin\\vivado.bat -
mode batch - source program_qspi.tcl
```

X Administrator: C:\Windows\system32\cmd.exe Erase Operation successful. A Performing Program and Verify Operations... Program/Verify Operation successful. INFO: [Labtoolstcl 44-377] Flash programming completed successfully program_hw_cfgmem: Time (s): cpu = 00:00:01 ; elapsed = 00:00:55 . Memory (MB): peak = 273.738 ; gain = 3.625 # endgroup # boot_hw_device [lindex [get_hw_devices] 0] INFO: [Labtools 27-32] Done pin status: HIGH # disconnect_hw_server ****** Webtalk v2015.2 (64-bit) **** SW Build 1266856 on Fri Jun 26 16:35:25 MDT 2015 **** IP Build 1264090 on Wed Jun 24 14:22:01 MDT 2015 ** Copyright 1986-2015 Xilinx, Inc. All Rights Reserved. source C:/Projects/DRC_Dev/7A35T_Arty/7A35T_Arty_00B_GPI0_demo_VIV2015_2/program flash/.Xil/Vivado-14040-avnet-wks2/webtalk/labtool webtalk.tcl -notrace INFO: [Common 17-186] 'C:/Projects/DRC_Dev/7A35T_Arty/7A35T_Arty_OOB_GPIO_demo_V IV2015_2/program_flash/.Xil/Vivado-14040-avnet-wks2/webtalk/usage_statistics_ext _labtool.xml' has been successfully sent to Xilinx on Fri Aug 14 15:52:36 2015. For additional details about this file, please refer to the WebTalk help file at C:/Xilinx/Vivado/2015.2/doc/webtalk_introduction.html. INFO: [Common 17-206] Exiting Webtalk at Fri Aug 14 15:52:36 2015... disconnect_hw_server: Time (s): cpu = 00:00:00 ; elapsed = 00:00:06 . Memory (MB): peak = 273.738 ; gain = 0.000 # close hw INFO: [Common 17-206] Exiting Vivado at Fri Aug 14 15:52:36 2015... C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_00B_GPI0_demo_VIV2015_2\program_flash>

4. When the QSPI Flash programming has completed you should see the Arty board reconfigure and launch the GPIO Demo application. This can also be tested by pressing the **PROG** switch on the Arty board (near LD8 and the USB JTAG/UART port). You are now ready to run the GPIO Demo software application. The steps to run the application are the same as <u>running the demo</u> you probably used earlier, except the steps of downloading the bitstream and application executable are already completed. Open a command window in the **<installation>\demo** folder and run the **cp_from_sdk.bat** batch file script to copy the new bitstream and software ELF files to the demo folder. Go back and rerun the <u>GPIO Demo</u>. This concludes this design tutorial.

COM18:115200baud - Tera Term VT	
File Edit Setup Control Window Help	
<pre>File Edit Setup Control Window Help ************************************</pre>	

10.2 Program the QSPI Flash Using the SDK

- 1. If not already done, connect the Arty Evaluation board to the host PC as described earlier in Setting Up the Arty Evaluation Board.
- 2. Start a serial terminal session using your terminal software of choice and set the serial port parameters to **115200** baud rate, **no** parity, **8** bits, **1** stop bit and no flow control.
- We must first create a bitstream with the QSPI SREC bootloader application merged with the FPGA bitstream. In the SDK main menu, select Xilinx Tools → Program FPGA or click on the and the SDK toolbar:

File E	dit Source	Refactor	Navigate	Search	Project	Xilin	x Tools Run Window Help		
C •		🖲 🗕 炎	- 🗟 🕃	器 🐹	2	0	 Repositories Board Support Package Settings 		
						Ŋ	Board Support Package Settings		
Pro	ject Explorer	3	- E] [] s	ystem.hdf		Board Support Package Settings Generate linker script		

4. Accept the defaults for the **Hardware Platform** and **Connection** and verify that the **gpio_demo** application is selected as the **ELF/MEM** File to Initialize in **Block RAM**. Click Program to continue:

SOK Program FPGA					×
Program FPGA					->-
Specify the bitstrean	M memory		<u> </u>		
Hardware Configura	tion				
Hardware Platform:	design_1_wrapp	er_hw_platform_0			
Connection:	Local	Local			
Device:	Auto Detect	Auto Detect			
Bitstream:	design_1_wrap	design_1_wrapper.bit			Browse
Partial Bitstream					
BMM/MMI File:	design_1_wrapper.mmi			Search	Browse
Software Configurat	ion				
Processor		ELF/MEM File to	Initialize in Blo	ck RAM	
design_1_i/microbla	aze_0	.sdk_workspace\g	ipio_demo\Del	bug\gpio_d	emo.elf 👻
•					•
?			Program		Cancel

5. You will see the following window while the FPGA bitstream is downloaded:

w Progress Information		
Initializing Bitstream with ELF data		
	Cancel	Details >>

6. In the SDK GUI navigate to **Xilinx Tools** \rightarrow **Program Flash**.

File Edit Source Refactor	Navigate Se	arch Project	Xilin	x Tools Run Window Help
Project Explorer S?				Repositories Board Support Package Settings Generate linker script
			*	Program FPGA
▲ ¹ / _↓ design_1_wrapper_hw_pla	tform_0	Linker Sc		Program Flash
🗁 cache		A linker script		Dump/Restore Data File

7. Programming the QSPI using the SDK requires two steps. The first step is to program the bitstream with the integrated QSPI SREC bootloader application to the QSPI Flash. Click Browse and navigate to the <installation>\sdk_workspace\design_1_wrapper_hw_platform_0 folder and select the download.bit bitstream file. Click Open to continue:

ize Vew folder				1	∃== ▼
J. Xil	^	Name	Size	Date modified	Туре
2014		\mu drivers		8/14/2015 3:17 PM	File folder
2015		design_1_wrapper.bit	2,141 KB	8/14/2015 3:17 PM	BIT File
bat_files	III	download.bit	2,141 KB	8/14/2015 3:37 PM	BIT File
3 7415T Att					
7A35T Arty Ethernetlite LwIP VIV2015	1				
7A35T Arty Ethernetlite LwIP VIV2015	2				
7A35T_Arty_OOB_GPIO_demo_VIV201!	5_2				
🎍 demo					
🔒 doc	-				

- 8. Select the following settings and click **Program** to continue:
 - Set the Offset to 0x0000000
 - Set the Flash Type to n25q128-3.3v-spi-x1_x2_x4
 - Enable Verify after flash

SOK	X				
Program Flash Memory					
Program Flash Mer	mory via In-system Programmer.				
Hardware Platform:	design_1_wrapper_hw_platform_0				
Connection:	Local New				
Device:	Auto Detect Select				
Image File:	C:\Projects\DRC_Dev\7A35T_Arty\7A35T_Arty_OOB_GPIO_demo_VIV2015_2\sdk_workspace\design_1_wrapper_hw_platform_0\download.bit Browse				
Offset:	0x0				
Flash Type	n25q128-3.3v-spi-x1_x2_x4				
FSBL File:	Browse				
Convert ELF to be	ootloadable SREC format and program				
🔲 Blank check after	erase				
🔽 Verify after flash					
?	Program Cancel				

9. You will see a status window while the bitstream is programmed to Flash:

SOK Progress Information	
Performing programming flash	
Performing Erase Operation	
	Cancel <u>D</u> etails >>

10. Test that the QSPI Flash programming was successful by pressing the PROG switch on the Arty board (near LD8 and the USB JTAG/UART port). You should see the Arty board reconfigure, launch the SREC bootloader, and boot the GPIO Demo application. You are now ready to run the GPIO Demo software application. The steps to run the application are the same as <u>running the demo</u> you probably used earlier, except the steps of downloading the bitstream and application executable are already completed. Open a command window in the <installation>\demo folder and run the cp_from_sdk.bat batch file script to copy the new bitstream and software ELF files to the demo folder. Go back and rerun the GPIO Demo. This concludes this design tutorial.

ECOM18:115200baud - Tera Term VT	
File Edit Setup Control Window Help	
<pre>/// Contor window rep /// // // // // // // // // // // // //</pre>	•

11 Appendix I: Determining the Virtual COM Port

Now you can connect the evaluation board's USB-to-UART port to one of the USB ports on your PC. The new hardware detection will pop up and enumeration of the driver will be started. Once finished a virtual COMx port is created and you are ready to setup a connection using Windows HyperTerminal or comparable serial terminal emulation utility. Follow these instructions to determine the COMx port assigned to the USB-to-UART bridge:



1. Open the Device Manager by right-clicking on **Computer**, select **Properties**, then click on the **Device Manager**.



2. In the Device Manager, scroll down to Ports and expand the list. You will see the USB Serial Port and its assigned COM port. In the example below, it is COM14. Make note of this COM port number for use with the serial terminal you will use elsewhere in this design tutorial. This concludes these USB UART driver and virtual COM port installation instructions.



12 Appendix II: Installation of Board XML Files

Many Avnet evaluation boards can now be targeted directly in the Xilinx Vivado tools. This greatly simplifies the task of building the MicroBlaze processor system and integrating system peripherals and board interfaces into your own custom designs. Follow the instructions below to install the board XML files for the Arty board into your Vivado installation folders:

- 1. Using your favorite web browser, download the zip archive of board XML files from the Digilent wiki site for the Arty Evaluation Board: <u>https://reference.digilentinc.com/_media/arty_boardfiles.zip</u>
- Extract the zip file in the download folder and in Windows Explorer, copy the arty folder from the <installation>\Arty_boardfiles\Vivado 2015 or later\board_files folder to <Vivado_install>\Xilinx\Vivado\2015.2\data\boards\board_files:



13 Appendix III: Windows 260 Character Path Limit

If you are using the Vivado Design Suite on a Windows-7 host, you may run into issues resulting from Vivado pathnames exceeding the maximum allowed. Vivado projects create a very deep file hierarchy, and it becomes very easy to violate the Windows limit if the project is not extracted near the root of the drive. This can even happen when the archive is decompressed, depending on where you choose to place the project in your existing file hierarchy.

It is not always convenient to place every Vivado project at the root of a drive. To work around this limitation, the recommended procedure is to place the Vivado archive in a shared folder on your host machine, then use Windows Explorer to map a network drive to the directory where the archive will be decompressed. This allows the Vivado project to be mapped to the root of the virtual (mapped) directory, eliminating any path issues.

Vivado also makes use of the Windows temp folder, which may be located several folders deep from the root drive, and this can also cause problems. You can create your own temporary directory in C:\temp, and force Vivado to use the new folder with the following TCL:

set_param "project.customTmpDirForArchive" C:/temp

For further information, see the Xilinx answer record at: <u>http://www.xilinx.com/support/answers/52787.html</u>.

14 Appendix IV: Getting Help and Support

14.1 Avnet Website

- Evaluation Kit home page with Documentation and Reference Designs <u>http://em.avnet.com/arty</u>
- Avnet support forums <u>http://community.em.avnet.com/</u>

14.2 Xilinx Website

- Details on the Artix-7 FPGA family are included in the following Xilinx documents:
 - Artix-7 Family Overview (<u>DS180</u>)
 - Artix-7 FPGA Data Sheet (DS181)
 - Artix-7 FPGA Configuration User Guide (UG470)
- For more detailed information about xilisf, please refer to the Xilinx OS and Libraries Document Collection located here: <u>www.xilinx.com/support/documentation/sw_manuals/xilinx2015_2/oslib_rm.pdf</u>
- For more detailed information about SDK, please refer to the SDK Help HTML: www.xilinx.com/support/documentation/sw_manuals/xilinx2015_2/SDK_Doc/index.html
- Xilinx support forums <u>http://www.xilinx.com/support.html</u>

14.3 Digilent Website

- Arty home page <u>http://www.artyboard.com</u>
- Arty Wiki https://reference.digilentinc.com/arty