

# **KCU105 PCI Express Control Plane TRD User Guide**

***KUCon-TRD01***

**Vivado Design Suite**

**UG918 (v2015.4) November 24, 2015**

## Revision History

The following table shows the revision history for this document.

Date	Version	Revision
11/24/2015	2015.4	Released with Vivado Design Suite 2015.4 with no changes from previous version.
10/05/2015	2015.3	Released with Vivado Design Suite 2015.3 with minor textual edits.
06/30/2015	2015.2	Released with Vivado Design Suite 2015.2 with no changes from previous version.
05/05/2015	2015.1	Updated for Vivado Design Suite 2015.1. TRD ZIP file changed to <code>rdf0305-kcu105-trd01-2015-1.zip</code> . Updated Information about resource utilization for the base design and the user extension design in <a href="#">Table 1-1</a> and <a href="#">Table 1-2</a> . Added information about Windows 7 driver support of the reference design, updating these sections: <a href="#">Features</a> , <a href="#">Computers</a> , <a href="#">Software</a> , and <a href="#">Appendix A, Directory Structure</a> . Updated <a href="#">Figure 5-3</a> , <a href="#">Figure A-1</a> , and <a href="#">Table A-1</a> to include Windows information. The section <a href="#">Install TRD Drivers on the Host Computer (Windows 7)</a> was added to <a href="#">Chapter 2, Setup</a> . The section <a href="#">Using the QuestaSim/ModelSim Simulator</a> was removed from <a href="#">Chapter 4, Implementing and Simulating the Design</a> , because QuestaSim simulation is not supported in Vivado tool release 2015.1.
02/26/2015	2014.4.1	Initial Xilinx release.

# Table of Contents

Revision History .....	2
<b>Chapter 1: Introduction</b>	
Overview .....	5
Features .....	6
Resource Utilization.....	6
<b>Chapter 2: Setup</b>	
Requirements.....	8
Preliminary Setup.....	9
<b>Chapter 3: Bringing Up the Design</b>	
Set the Host System to Boot from the LiveDVD (Linux) .....	14
Configure the FPGA .....	15
Run the Design on the Host Computer.....	20
Test the Reference Design.....	25
Remove Drivers from the Host Computer (Windows Only) .....	26
<b>Chapter 4: Implementing and Simulating the Design</b>	
Implementing the Base Design .....	27
Implementing the User Extension Design .....	30
Simulating the Base Design Using Vivado Simulator.....	32
<b>Chapter 5: Targeted Reference Design Details and Modifications</b>	
Hardware .....	35
Data Flow .....	39
Software.....	40
Reference Design Modifications.....	41

**Appendix A: Directory Structure**

**Appendix B: Recommended Practices and Troubleshooting in Windows**

Recommended Practices ..... 45  
Troubleshooting ..... 45

**Appendix C: Additional Resources and Legal Notices**

Xilinx Resources ..... 46  
Solution Centers ..... 46  
References ..... 46  
Please Read: Important Legal Notices ..... 47

# Introduction

This document describes the features and functions of the PCI Express® Control Plane targeted reference design (TRD). The TRD comprises a base design and a user extension design. The user extension design adds custom logic on top of the base design. The pre-built user extension design in this TRD adds another AXI block RAM controller to the design and sets up ingress translations through BAR4 to access this memory space.

## Overview

The TRD targets the Kintex® UltraScale™ XCKU040-2FFVA1156E FPGA running on the KCU105 evaluation board. It demonstrates a control plane application using a PCI Express Endpoint block in a x1 Gen1 configuration. Simple base address register (BAR)-mapped read and write transactions are demonstrated using a kernel mode software driver controlled by the Control & Monitoring graphical user interface (GUI). The top-level block diagram of the TRD is shown in [Figure 1-1](#).

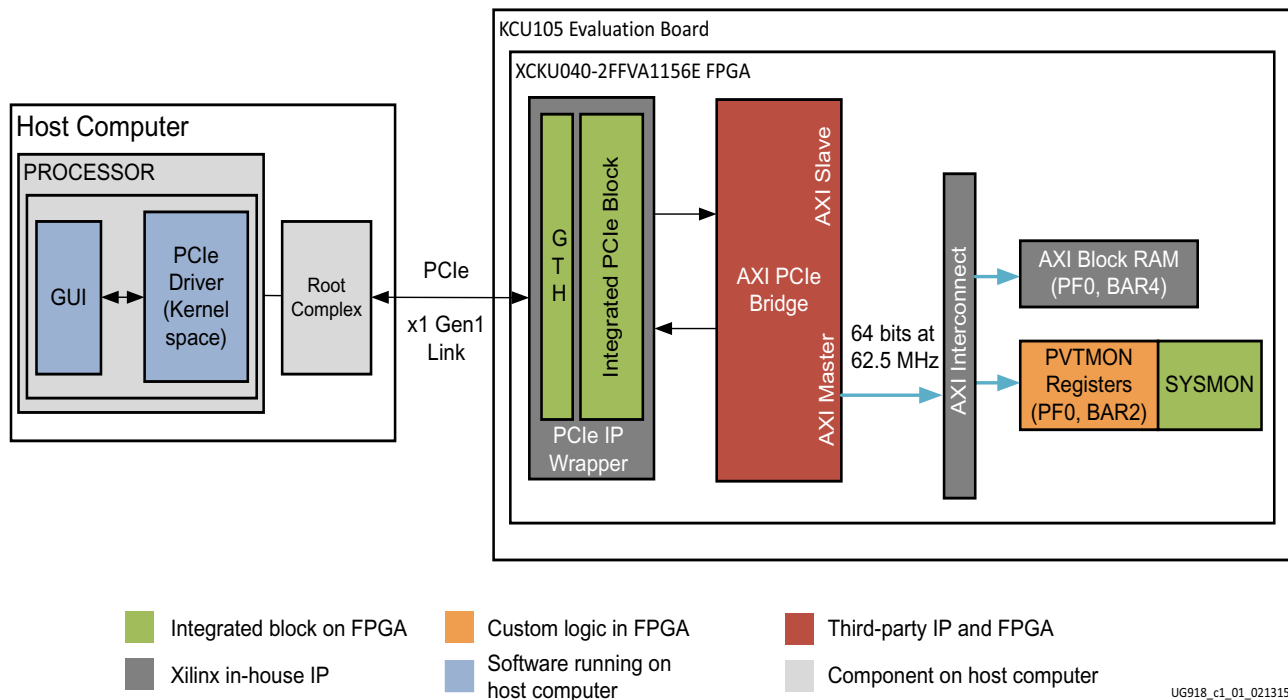


Figure 1-1: KCU105 PCI Express Control Plane Targeted Reference Design

An Expresso DMA Bridge Core from Northwest Logic (NWL) [Ref 1] is used to demonstrate PCIe-to-AXI conversion of transactions. The downstream slaves include a power, voltage, and temperature (PVT) module monitoring parameters from the FPGA system monitor and AXI block RAM IP targeted to BARs.

## Features

The TRD includes these features:

- PCIe x1 Gen1 Endpoint operating at 2.5 GigaTransfers per second (GT/s) per lane/direction
  - Single physical function with support for three 64-bit BARs
- DMA Bridge IP Core
  - Ingress address translation capability
  - AXI3 interface
- 64-bit kernel space drivers for Linux and Windows 7, which run on the host computer
- Control and monitoring graphical user interface (GUI)

## Resource Utilization

Table 1-1 and Table 1-2 list the resources used by the base and user extension designs after synthesis. Place and route can alter these numbers based on placements and routing paths. These numbers are to be used as a rough estimate of resource utilization. These numbers might vary based on the version of the TRD and the tools used to regenerate the design.

Table 1-1: Base Design Resource Utilization

Resource Type	Available	Used	Usage (%)
CLB registers	484,800	43,896	9.05
CLB LUT	242,400	27,431	11.32
Block RAM	600	22	3.66
MMCME3_ADV	10	1	10
Global Clock Buffers	240	3	1.25
BUFG_GT	120	5	4.17
SYSMONE1	1	1	100
IOB	520	16	3.08

**Table 1-1: Base Design Resource Utilization (Cont'd)**

Resource Type	Available	Used	Usage (%)
GTHE3_CHANNEL	20	1	5
GTHE3_COMMON	5	0	0

**Table 1-2: User Extension Design Resource Utilization**

Resource Type	Available	Used	Usage (%)
CLB Registers	484,800	44,395	9.16
CLB LUTs	242,400	27,817	11.48
Block RAM	600	24	4
MMCME3_ADV	10	1	10
Global Clock Buffers	240	3	1.25
BUFG_GT	120	5	4.17
SYSMONE1	1	1	100
IOB	520	16	3.08
GTHE3_CHANNEL	20	1	5
GTHE3_COMMON	5	0	0

# Setup

This chapter identifies the hardware and software requirements, and the preliminary setup procedures required prior to bringing up the targeted reference design.

---

## Requirements

### Hardware

#### *Board and Peripherals*

- KCU105 board with the Kintex® UltraScale™ XCKU040-2FFVA1156E FPGA
- USB cable, standard-A plug to micro-B plug (Digilent cable)
- Power supply: 100 VAC–240 VAC input, 12 VDC 5.0A output
- ATX power supply
- ATX power supply adapter

#### *Computers*

A *control* computer is required to run the Vivado® Design Suite and configure the on-board FPGA. It can be a laptop or desktop computer with any operating system supported by Vivado tools, such as Redhat Linux or Microsoft® Windows 7.

The reference design test configuration requires a host computer comprised of the chassis, containing a motherboard with a PCI Express slot, monitor, keyboard, and mouse. A DVD drive is also required if a Linux operating system is used. If a Windows 7 operating system is used, the 64-bit Windows 7 OS and the Java SE Development Kit 7 must be installed.

### Software

Vivado Design Suite 2015.4 is required. The Fedora 20 LiveDVD, on which the TRD software and GUI run, is only required if a Linux operating system is used.

---

## Preliminary Setup

Complete these tasks before bringing up the design.

### Install the Vivado Design Suite

Install Vivado Design Suite 2015.4 on the control computer. Follow the installation instructions provided in *Vivado Design Suite User Guide Release Notes, Installation, and Licensing* (UG973) [Ref 2].

### Download the Targeted Reference Design Files

1. Download `rdf0305-kcu105-trd01-2015-4.zip` from the *Xilinx Kintex UltraScale FPGA KCU105 Evaluation Kit - Documentation & Designs website*. This ZIP file contains the hardware design, software drivers, and application GUI executables.
2. Extract the contents of the file to a working directory.
3. The extracted contents are located at `<working_dir>/kcu105_control_plane`.

The TRD directory structure is described in [Appendix A, Directory Structure](#).

### Install TRD Drivers on the Host Computer (Windows 7)

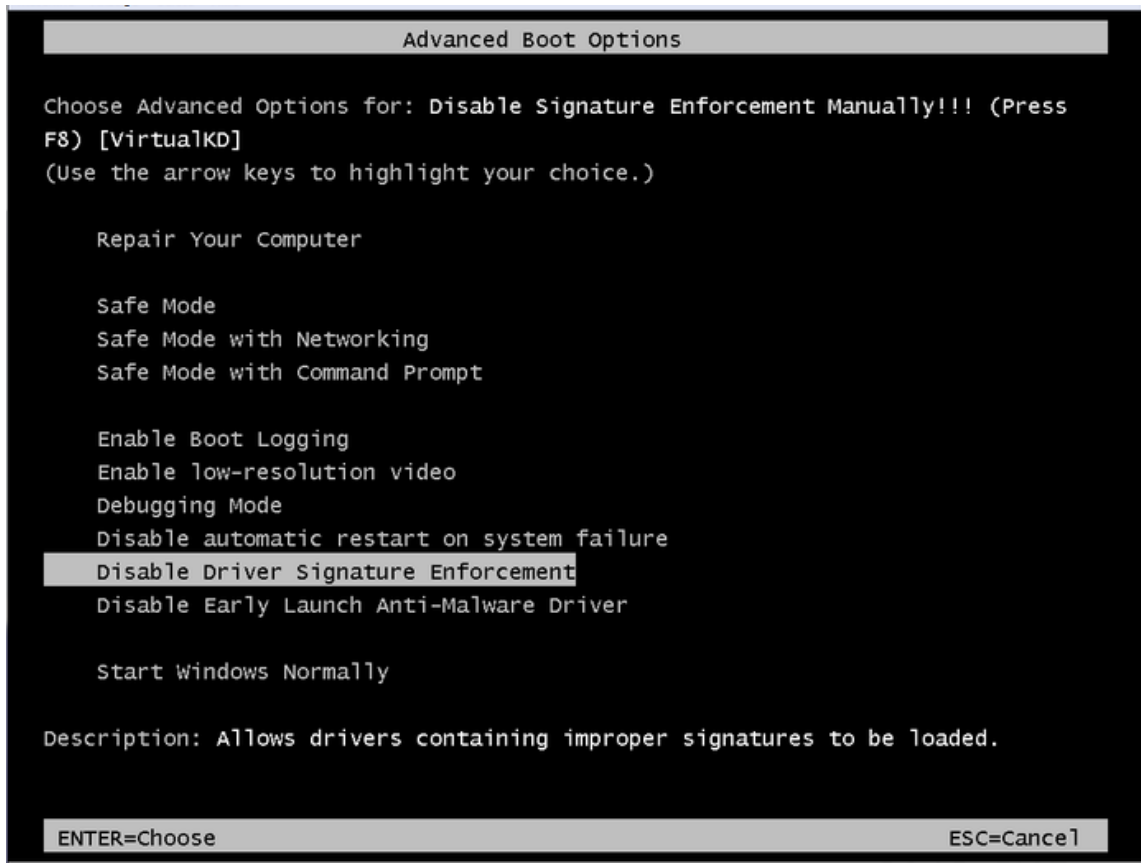
**Note:** This section provides steps to install KUCon-TRD drivers and is only applicable to a host computer running Windows 7 64-bit OS. If running Linux, proceed to [Set DIP Switches, page 11](#).

This section includes steps to set up and install KUCon-TRD drivers in a host computer running the Windows 7 64-bit OS.

#### ***Disable Driver Signature Enforcement***

**Note:** Windows only allows drivers with valid signatures obtained from trusted certificate authorities to load in Windows 7 64-bit OS. Windows drivers provided for this reference design do not have a valid signature. Therefore, you have to disable driver signature enforcement on the host computer, as follows:

1. Power up the host system. Press **F8** to go to the Advanced Boot Options menu.
2. Select the **Disable Driver Signature Enforcement** option as shown in [Figure 2-1](#), and press **Enter**.



UG918\_c2\_01\_040315

Figure 2-1: Disable Driver Signature Enforcement

## Install Drivers

1. From the Windows explorer, navigate to the folder in which the reference design is downloaded (<dir>\kcu105\_control\_plane\software\windows\) and run the setup file with Administrator privileges, as shown in [Figure 2-2](#).

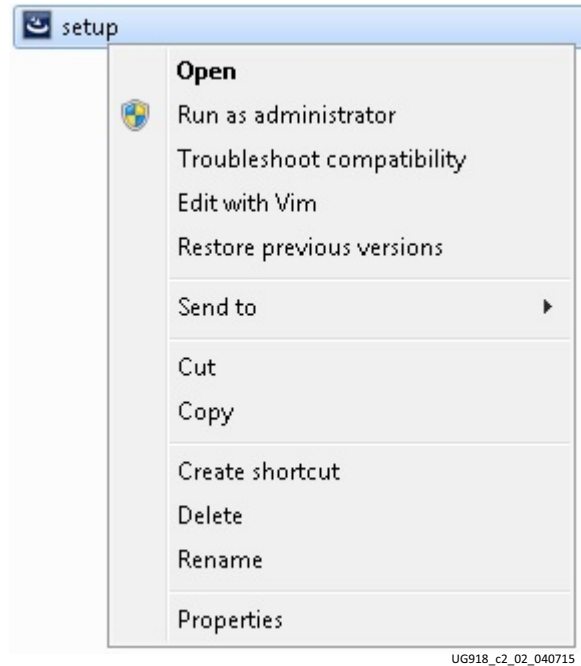


Figure 2-2: Run the Setup File with Administrator Privileges

2. Click **Next** after the InstallShield Wizard opens.
3. Click **Next** to install to the default folder; or click **Change** to install to a different folder.
4. Click **Install** to begin driver installation.
5. A warning screen displays as the drivers are installed, because the drivers are not signed by a trusted certificate authority yet. To install the drivers, ignore the warning message and click **Install this driver software anyway**. This warning message pops up two times. Repeat this step.
6. After installation is complete, click **Finish** to exit the InstallShield Wizard.

## Set DIP Switches

Ensure that the DIP switches and jumpers on the KCU105 board are set to the factory default settings as identified in the *Kintex UltraScale FPGA KCU105 Evaluation Board User Guide* (UG917) [\[Ref 3\]](#).

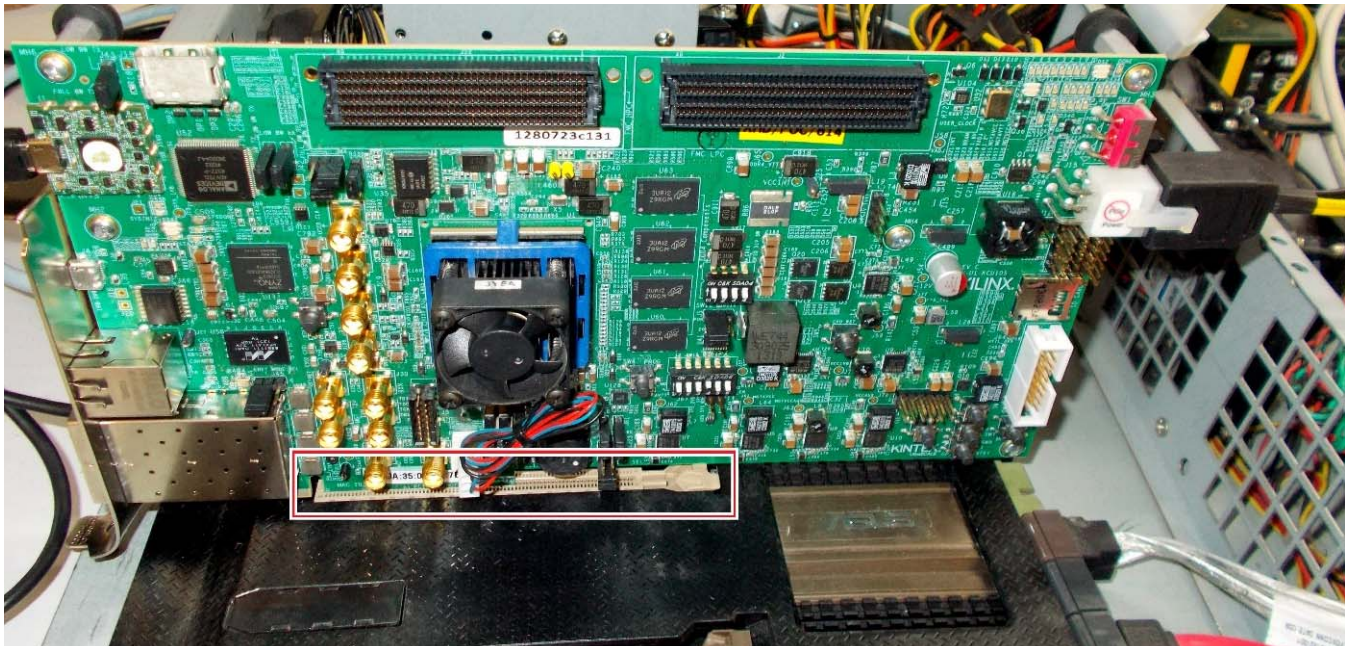
## Install the KCU105 Board

1. Remove all rubber feet and standoffs from the KCU105 board.
2. Power down the host chassis and disconnect the power cord.



**CAUTION!** Remove the power cord to prevent electrical shock or damage to the KCU105 board or other components.

3. Ensure that the host computer is powered off.
4. Open the chassis. Select a vacant PCIe Gen3-capable expansion slot and remove the expansion cover at the back of the chassis.
5. Plug the KCU105 board into the PCIe connector slot, as shown in [Figure 2-3](#).

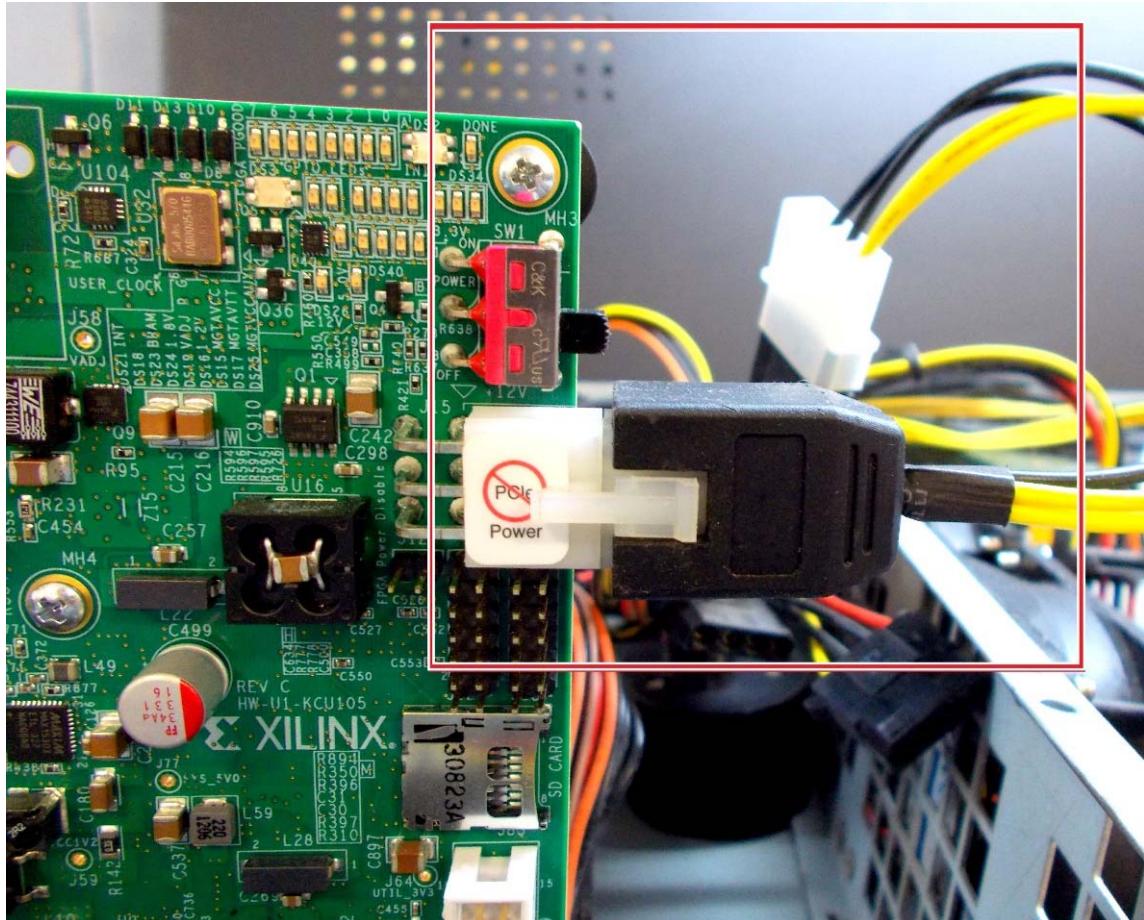


UG918\_c3\_01\_091314

Figure 2-3: PCIe Connector Slot

6. Connect the ATX power supply to the KCU105 board using the ATX power supply adapter cable as shown in [Figure 2-4](#).

**Note:** A 100 VAC–240 VAC input, 12 VDC 5.0A output external power supply can be substituted for the ATX power supply.



UG918\_c3\_02\_040715

**Figure 2-4: Power Supply Connection to the KCU105 Board**

7. Slide the KCU105 board power switch SW1 to the ON position (ON/OFF is marked on the board).

# Bringing Up the Design

This chapter describes how to bring up and test the targeted reference design.

---

## Set the Host System to Boot from the LiveDVD (Linux)

**Note:** This section is only applicable to host computers running Linux. If running Windows 7, proceed to [Configure the FPGA](#).

1. Power on the host system and stop it in BIOS to select options to boot from the DVD drive. BIOS options are entered by pressing DEL, F12, or F2 keys on most computers.

**Note:** If an external power supply is used instead of the ATX power, the FPGA can be configured first. Then power on the host system.

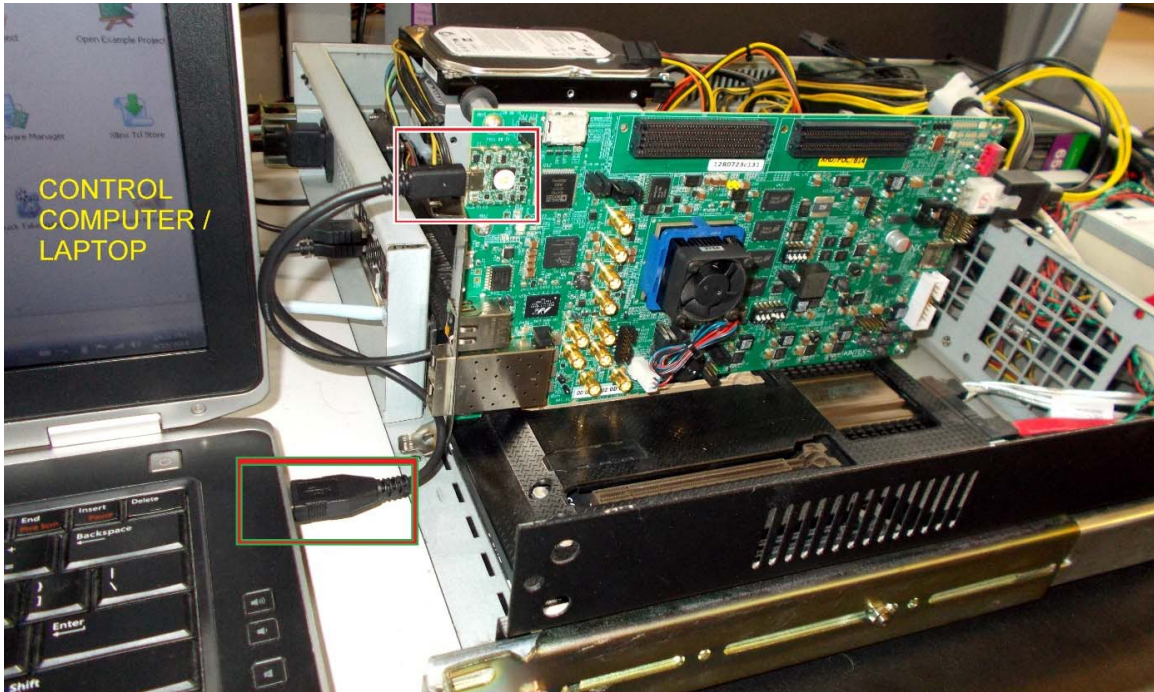
2. Place the Fedora 20 LiveDVD into the DVD drive.
3. Select the option to boot from DVD.

## Configure the FPGA

While in BIOS, program the FPGA with the BIT file:

1. Connect the standard-A plug to micro-B plug USB cable to the JTAG port on the KCU105 board and to the control computer laptop as shown in [Figure 3-1](#).

**Note:** The host system can remain powered on.



UG918\_c3\_03\_042115

**Figure 3-1: Connect the USB Cable to the KCU105 Board and Control Computer**

**Note:** [Figure 3-1](#) shows a Rev C board. The USB JTAG connector is on the PCIe panel for production boards.

2. Launch the Vivado® Integrated Design Environment (IDE) on the control computer:
  - a. Select **Start > All Programs > Xilinx Design Tools > Vivado 2015.4 > Vivado 2015.4**.
  - b. On the getting started page, click **Open Hardware Manager** (Figure 3-2).



UG918\_c3\_04\_091314

Figure 3-2: Vivado IDE Getting Started Page, Open Hardware Manager

3. Open the connection wizard to initiate a connection to the KCU105 board:
  - a. Click **Open New Target** (Figure 3-3).

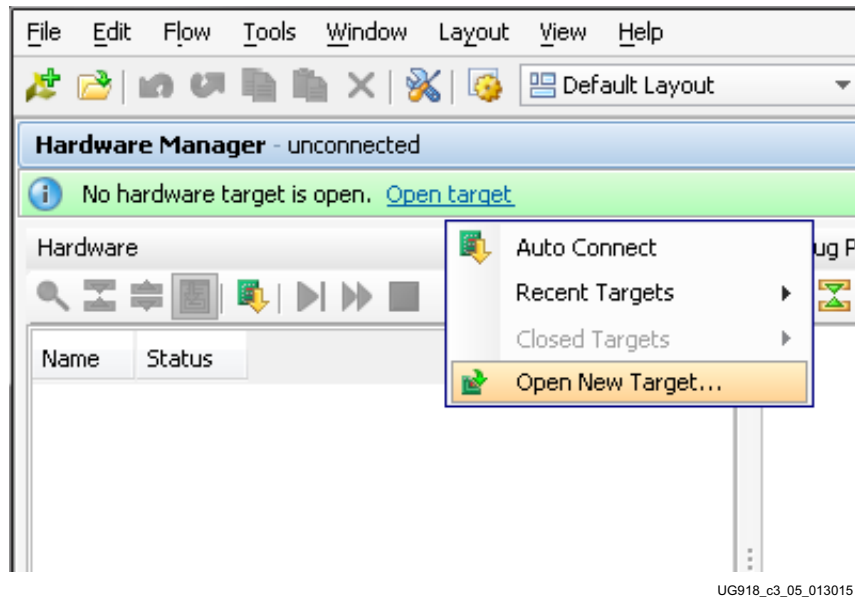
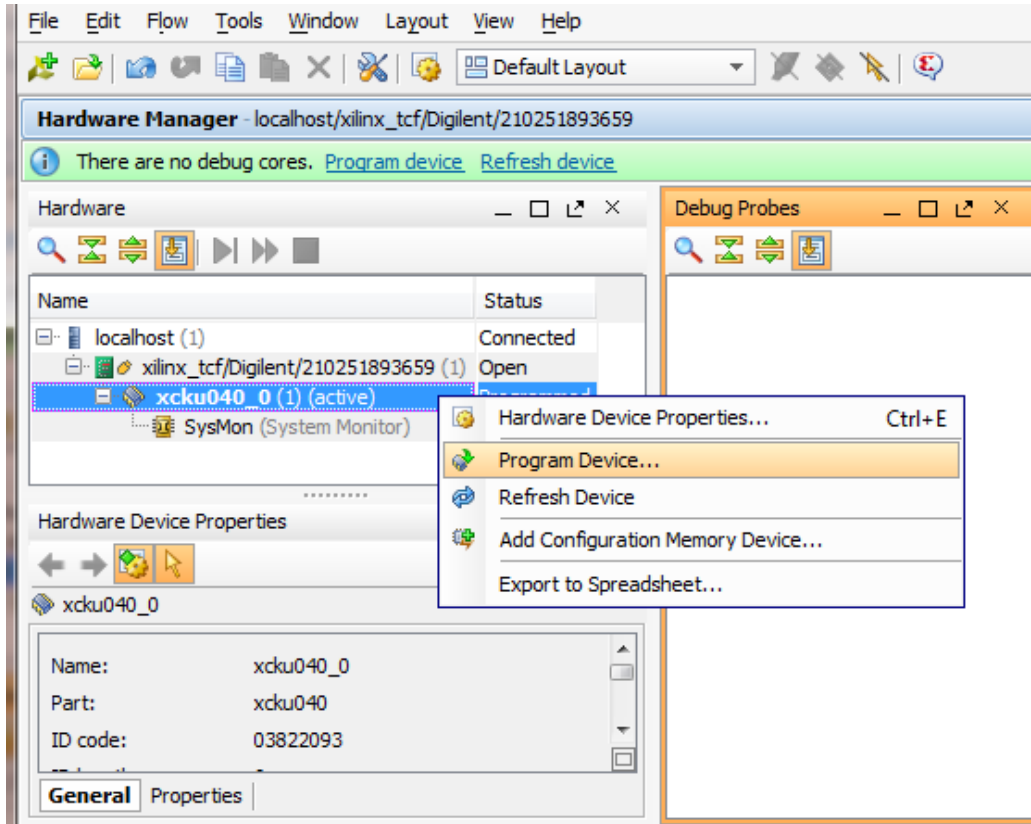


Figure 3-3: Using the User Assistance Bar to Open a Hardware Target

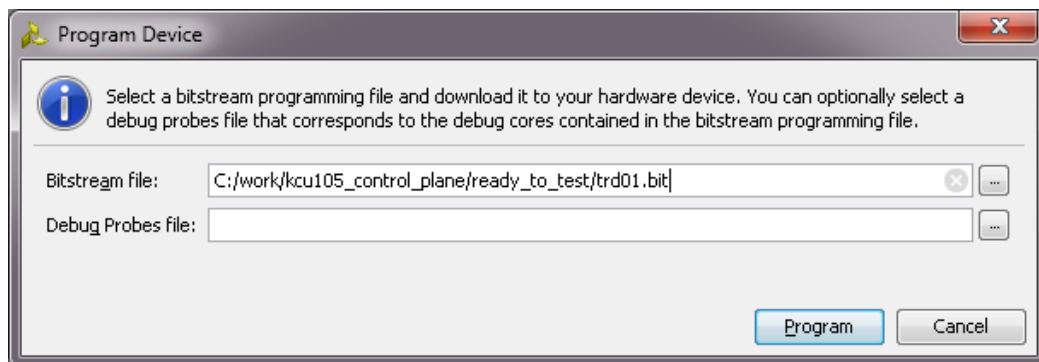
4. Configure the wizard to establish connection with the KCU105 board by selecting the default value on each wizard page. Click **Next** > **Next** > **Next** > **Finish**.
  - a. In the hardware view, right-click **xcku040** and click **Program Device** (Figure 3-4).



UG918\_c3\_06\_091314

Figure 3-4: Select Device to Program

- b. In the **Bitstream file** field, browse to the location of the BIT file `<working_dir>/kcu105_control_plane/ready_to_test/trd01.bit` and click **Program** (see Figure 3-5).



UG918\_c3\_07\_011415

Figure 3-5: Program Device Window

5. Check the status of the design by observing the GPIO LEDs positioned at the top right corner of the KCU105 board (Figure 3-6). After FPGA configuration, the LED status from left to right indicate
  - LED position **1**: Heartbeat LED, flashes if the PCIe user clock is present
  - LED position **0**: ON if the PCIe link is UP

**Note:** The LED position numbering used here matches with the LED positions on the board.

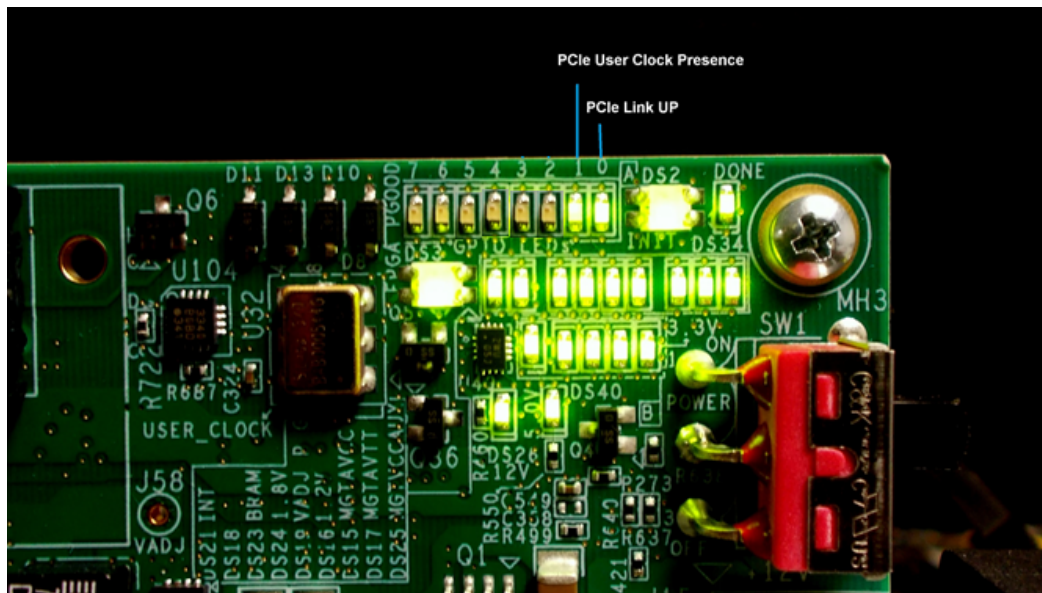


Figure 3-6: GPIO LED Indicators

6. Exit the BIOS and let the system boot.
7. On most systems, this gives a second reset on the PCIe connector, which should discover the device during enumeration.
  - To know that the PCIe Endpoint is discovered, see [Check for PCIe Devices, page 20](#).
  - If the PCIe Endpoint is not discovered, reboot the system. Do not power off.

## Run the Design on the Host Computer

This section provides instructions to run the reference design on either a host computer with Linux, or a host computer with Windows 7.

### Run the Design on a Linux Host Computer

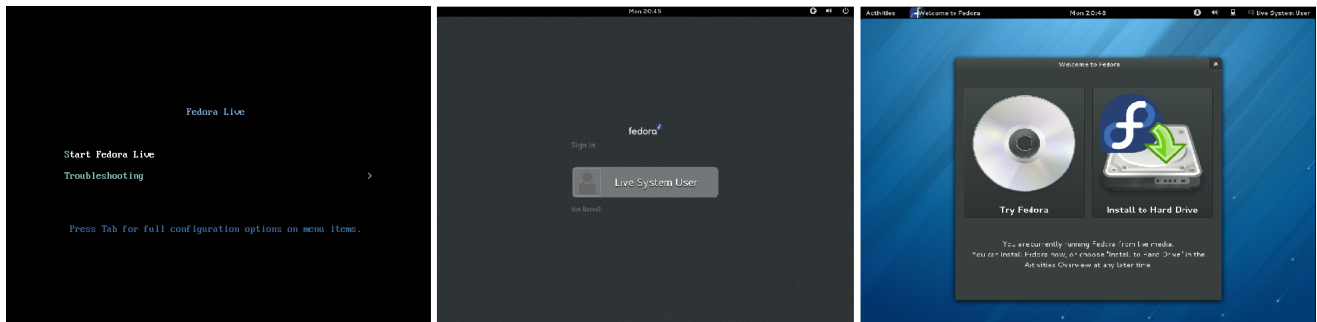
#### Setup

This section describes how to set up the reference design using the Linux drivers and the Fedora 20 LiveDVD.

Figure 3-7 shows different boot stages of Fedora 20. After you reach the third screen, shown in Figure 3-7, click the **Try Fedora** option, then click **Close**. It is recommended that you run the Fedora operating system from the DVD.



**CAUTION!** If you want to install Fedora 20 on the hard drive connected to the host system, click the **Install to Hard Drive** option. **BE CAREFUL!** This option erases any files on the hard disk!



UG918\_c3\_09\_011415

Figure 3-7: Fedora 20 Boot Stages

#### Check for PCIe Devices

1. After the Fedora 20 OS boots, open a terminal and use **lspci** to see a list of PCIe devices detected by the host:

```
$ lspci | grep -i xilinx
```

The following is displayed:

```
03:00.0 Memory controller: Xilinx Corporation Device 8011
```

**Note:** If the host computer does not detect the Xilinx PCIe Endpoint, `lspci` does not show a Xilinx device.

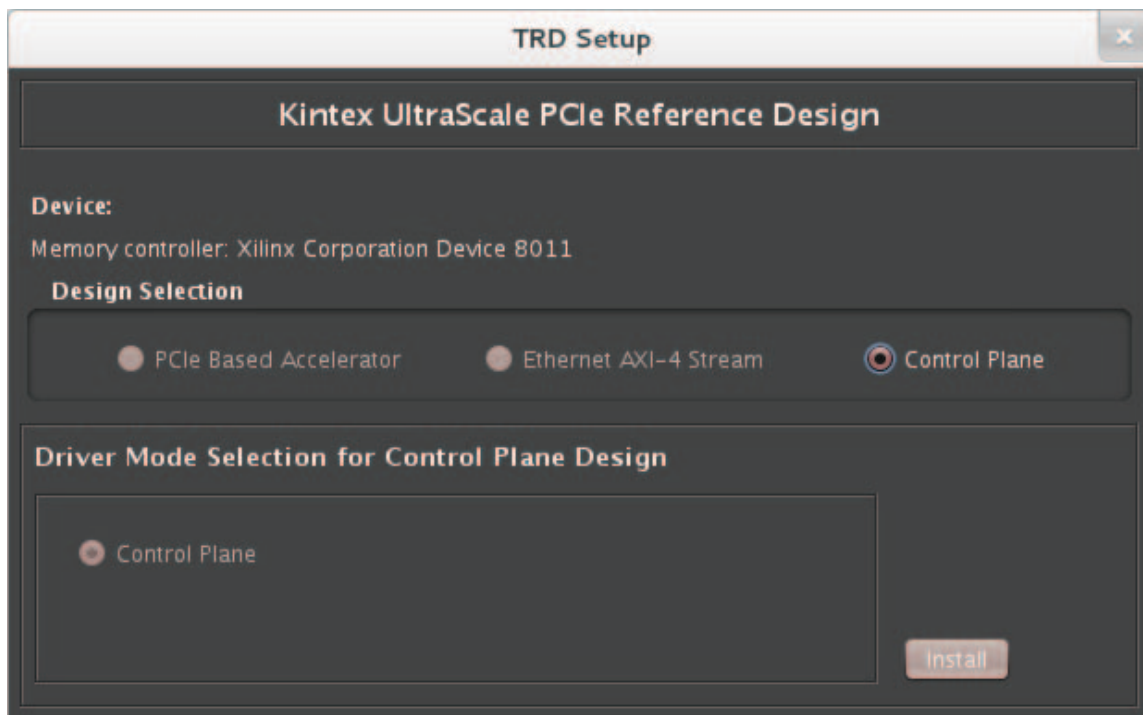
## Run the Design

1. Navigate to the `<working_dir>/kcu105_control_plane/software` folder and open a terminal. (The TRD files were extracted to your `<working_dir>` in [Download the Targeted Reference Design Files](#), page 9).

2. Enter:

```
$ cd /home/<working_dir>/kcu105_aximm_dataplane
$ su      --> command to login as super user
$ chmod +x quickstart.sh
$ ./quickstart.sh
```

3. The TRD setup screen is displayed ([Figure 3-8](#)) and indicates detection of a PCIe device with an ID of 8011, a control plane design selection, and a control plane driver mode. Click **Install** and the drivers are installed. This takes you to the Control and Monitoring GUI as shown in [Figure 3-12](#), page 25.



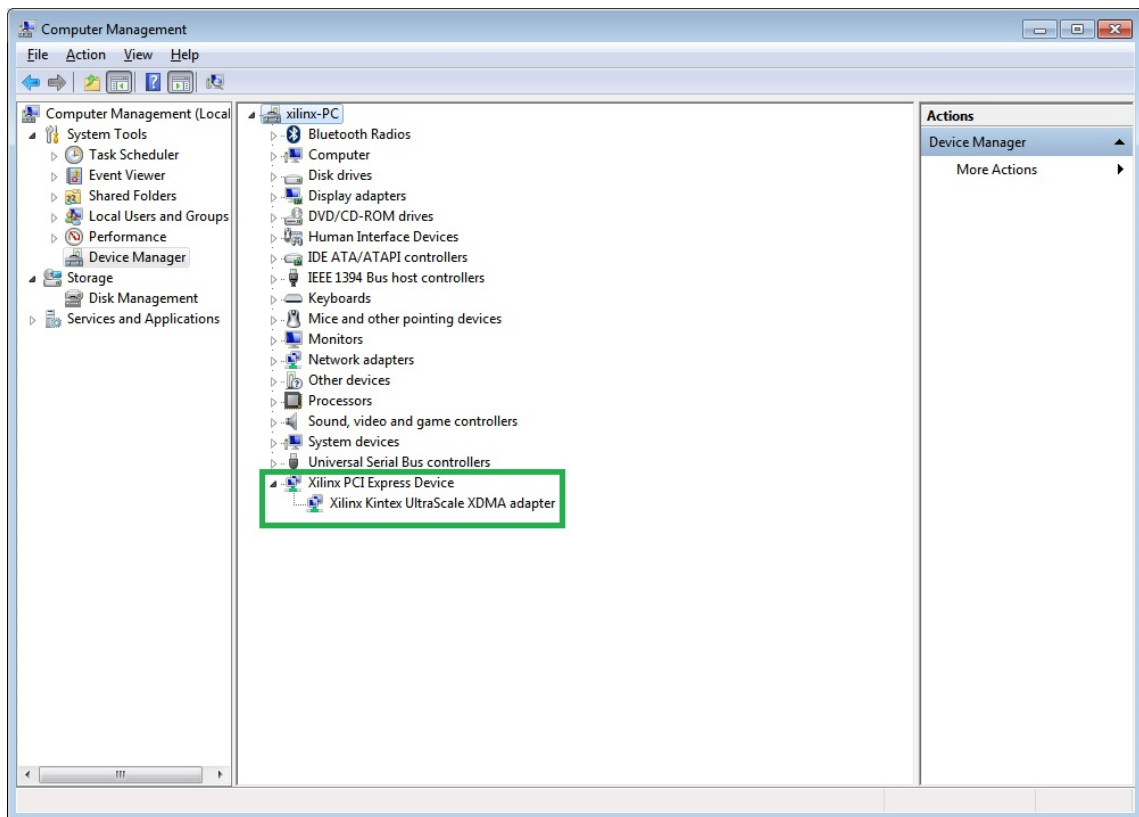
UG918\_c3\_10\_021815

Figure 3-8: TRD Setup Screen with a PCIe Device Detected

## Run the Design on a Windows 7 Host Computer

After booting the Windows OS, follow these steps:

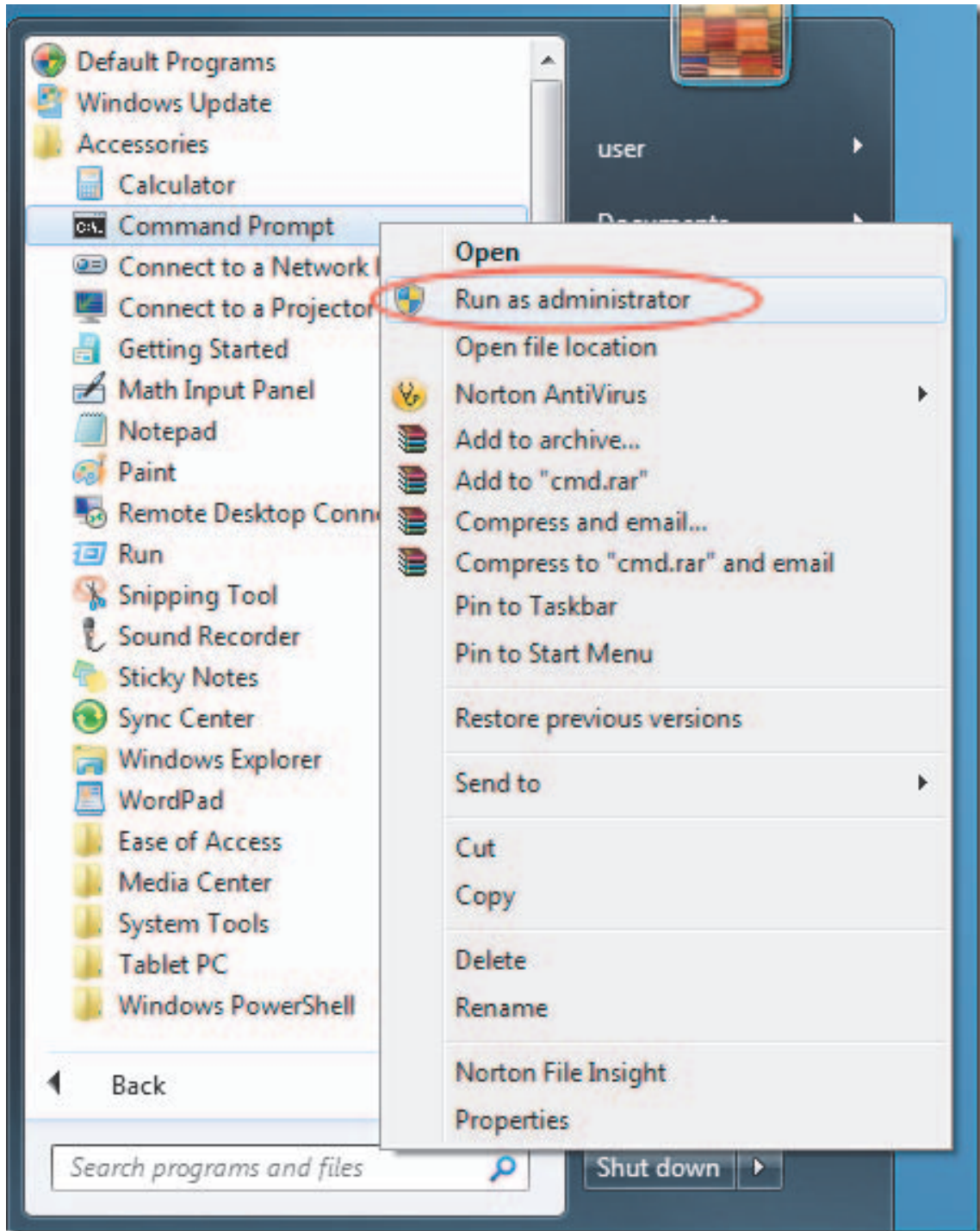
1. Repeat the steps in section [Disable Driver Signature Enforcement, page 9](#).
2. Open Device Manager (click **Start** > **devmgmt.msc** then press **Enter**) and look for the Xilinx PCI Express Device as shown in [Figure 3-9](#).



UG918\_c3\_11\_040715

Figure 3-9: Xilinx PCI Express Device in Device Manager

3. Open a command prompt with administrator privileges, as shown in [Figure 3-10](#).



UG918\_c3\_12\_040715

Figure 3-10: Command Prompt with Administrator Privileges

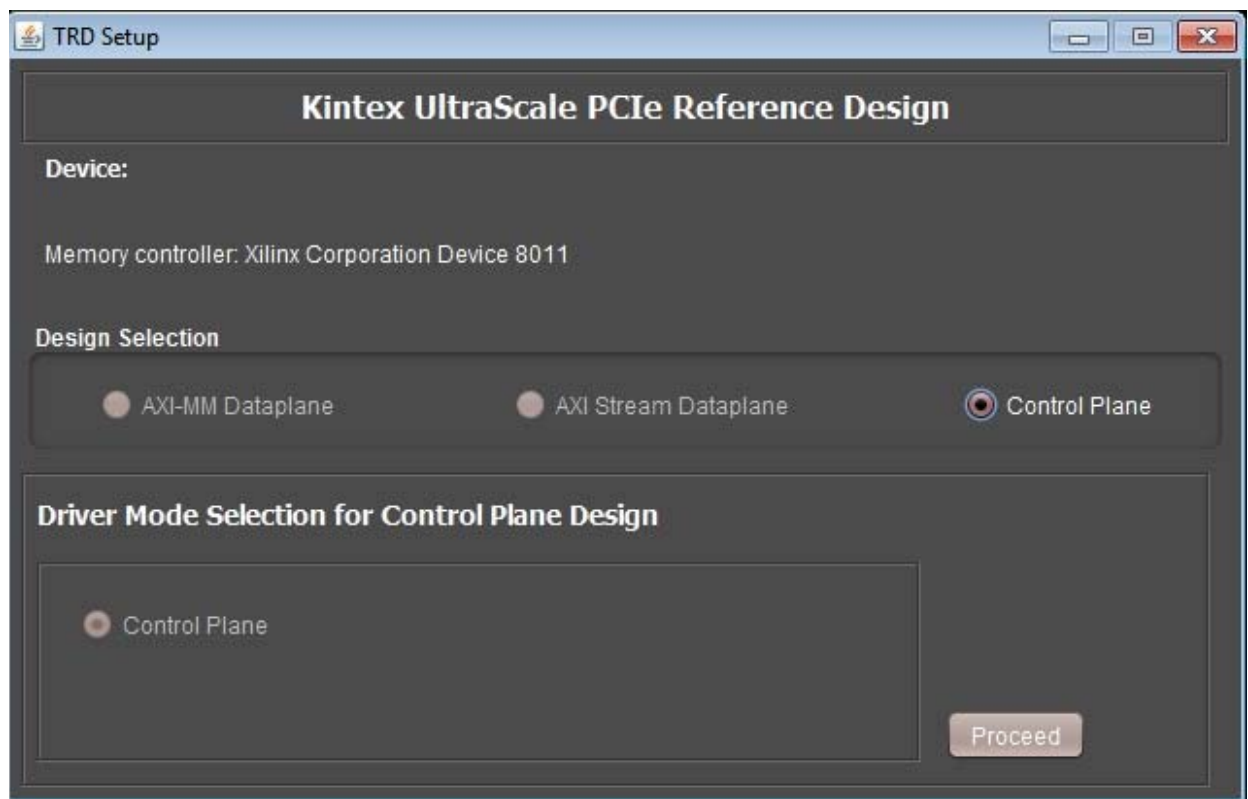
4. Navigate to the folder where the reference design is copied:

```
cd <dir>\kcu105_control_plane
```

5. Run the batch script `quickstart_win7.bat`:

```
quickstart_win7.bat
```

6. The screen in [Figure 3-11](#) shows the TRD Setup screen of the GUI. Click **Proceed** to test the reference design. This step takes you to the Control and Monitoring GUI as shown in [Figure 3-12, page 25](#).



UG918\_c3-13\_040715

Figure 3-11: GUI - TRD Setup Screen

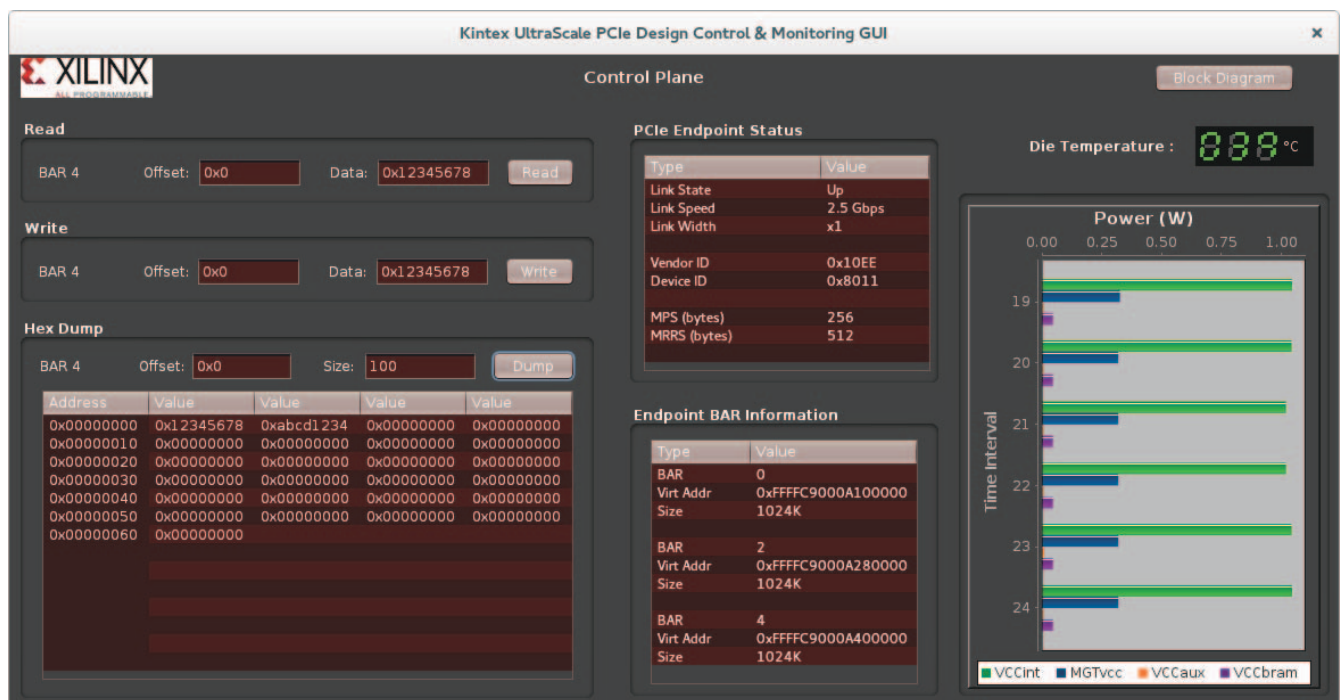
## Test the Reference Design

The control and monitoring GUI, shown in Figure 3-12, provides information on power and FPGA die temperature, (PVTMON is mapped to BAR2), PCIe Express Endpoint link status, virtual address assigned to BARs, and user options to access BAR-mapped regions.

The following can be done through the main control and monitoring GUI:

- Read an address offset from BAR4.
- Write to an address offset from BAR4 by providing the offset value and data value to be written.
- Obtain a dump of the data from a specific address offset from BAR4.

You can view the block diagram by clicking Block Diagram in top right corner of the screen (Figure 3-12).



UG918\_c3\_14\_040715

Figure 3-12: Control & Monitoring GUI

Click the **X** mark on the top right corner to close the GUI. On a Linux host computer, this step uninstalls the drivers and returns the GUI to the TRD Setup screen. Close the TRD Setup screen and power off the host machine and then the KCU105 board. On a Windows host computer, this step returns to the TRD Setup screen.

To uninstall the drivers on a Windows host computer, use the following steps.

---

## Remove Drivers from the Host Computer (Windows Only)

---



**IMPORTANT:** *Shutdown the host computer and power off the KCU105 board. Then use the following steps to remove the Windows drivers.*

---

1. Power on the host computer and from Windows Explorer, navigate to the folder in which the reference design is downloaded (`<dir>\kcu105_control_plane\software\windows\`). Run the setup file with administrator privileges.
2. Click **Next** after the InstallShield Wizard opens.
3. Select **Remove** and click **Next**.
4. Click **Remove** to remove drivers from the host system.
5. Click **Finish** to exit the wizard.

# Implementing and Simulating the Design

This chapter describes how to implement and simulate the targeted reference design. The time required to do so can vary from system to system depending on the control computer configuration.

**Note:** All the steps mentioned in this chapter to run simulation and implementation should be run on the control PC that has Vivado® tools installed.

**Note:** In Windows, if the path length is more than 260 characters, then design implementation or simulation using the Vivado Design Suite might fail. This is due to a Windows OS limitation. Refer to Refer to the following AR for more details: [KCU105 Evaluation Kit Master Answer Record \(AR 63175\)](#).

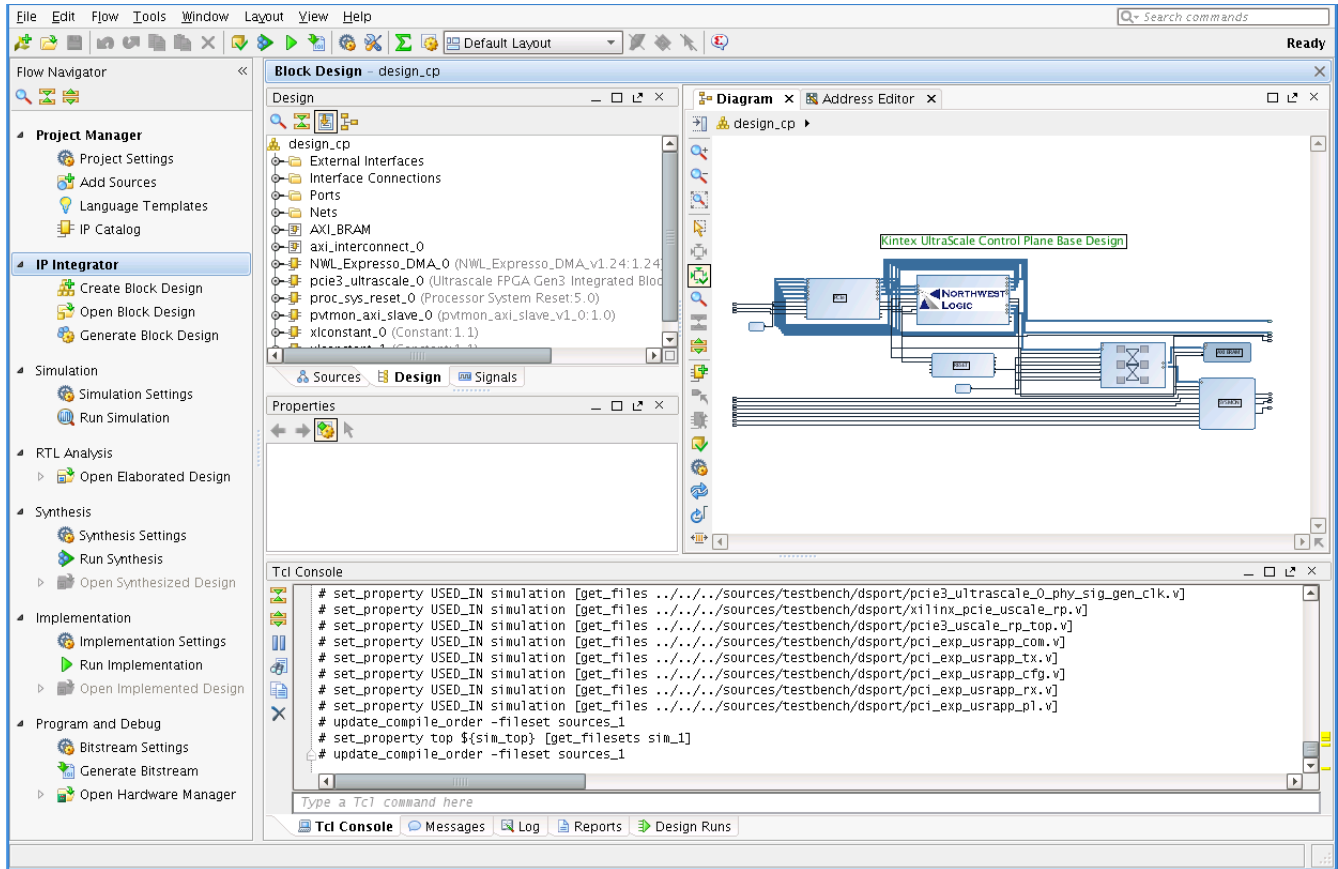
---

## Implementing the Base Design

1. If not already done so, copy the reference design ZIP file to the desired directory on the control PC and unzip the ZIP file. (The TRD files were extracted to your <working\_dir> in [Download the Targeted Reference Design Files, page 9](#).)
2. Open a terminal window on a Linux system with the Vivado environment set up, or open a Vivado tools Tcl shell on a Windows system.
3. Navigate to the `kcu105_control_plane/hardware/vivado/scripts/base` folder.
4. To run the implementation flow, enter:

```
$ vivado -source trd01_base.tcl
```

This opens the Vivado Integrated Design Environment (IDE), loads the block diagram, and adds the required top file and Xilinx design constraints (XDC) file to the project (see [Figure 4-1](#)).

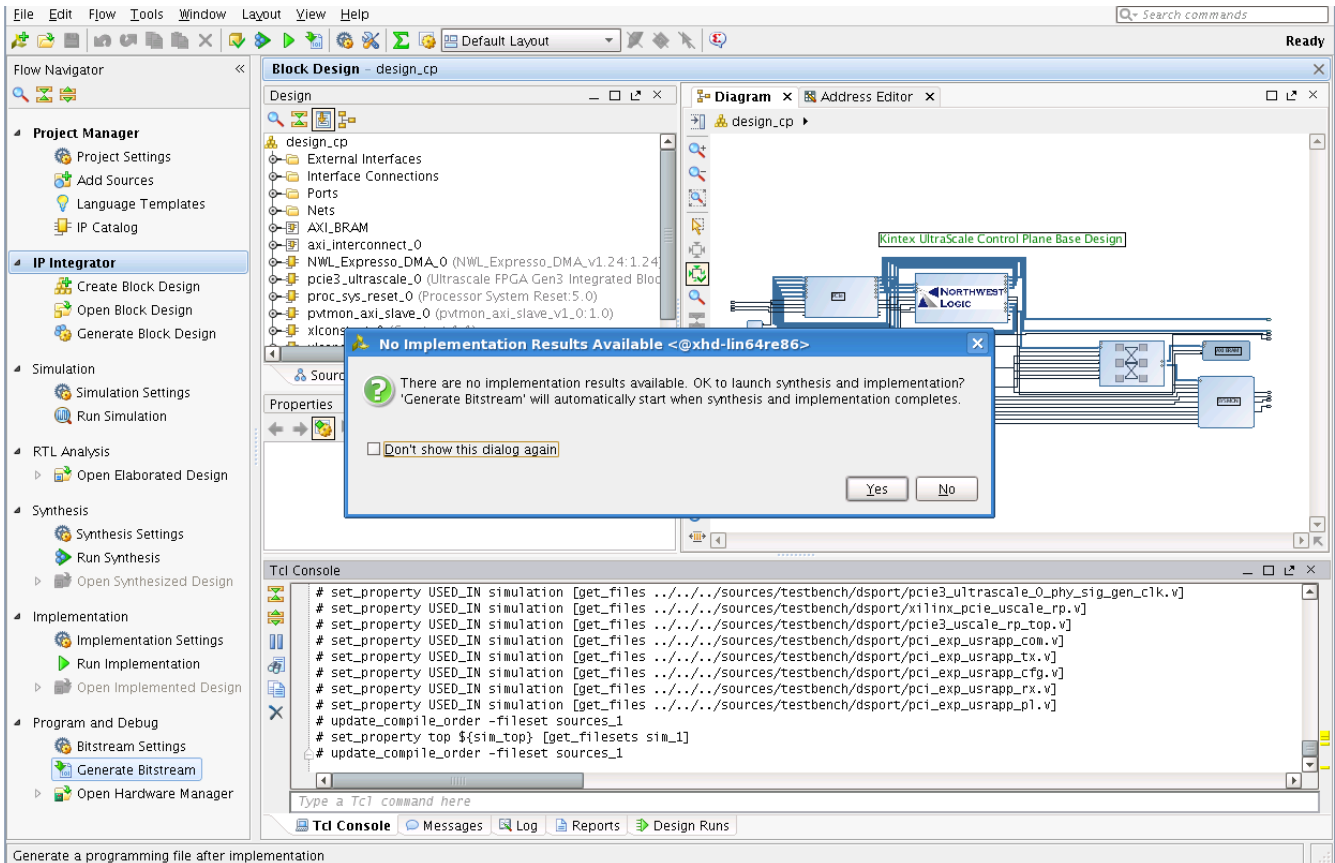


UG918\_c4\_01\_011415

Figure 4-1: Base Design—Project View

- In the Flow Navigator panel, click the **Generate Bitstream** option which runs synthesis, implementation, and generates the BIT file (see Figure 4-2). Click **Yes** if a window indicating No Implementation Results are available is displayed. The generated bitstream can be found under the following directory:

kcu105\_control\_plane/hardware/vivado/runs\_base/trd01.runs/impl\_1.



UG918\_c4\_02\_011415

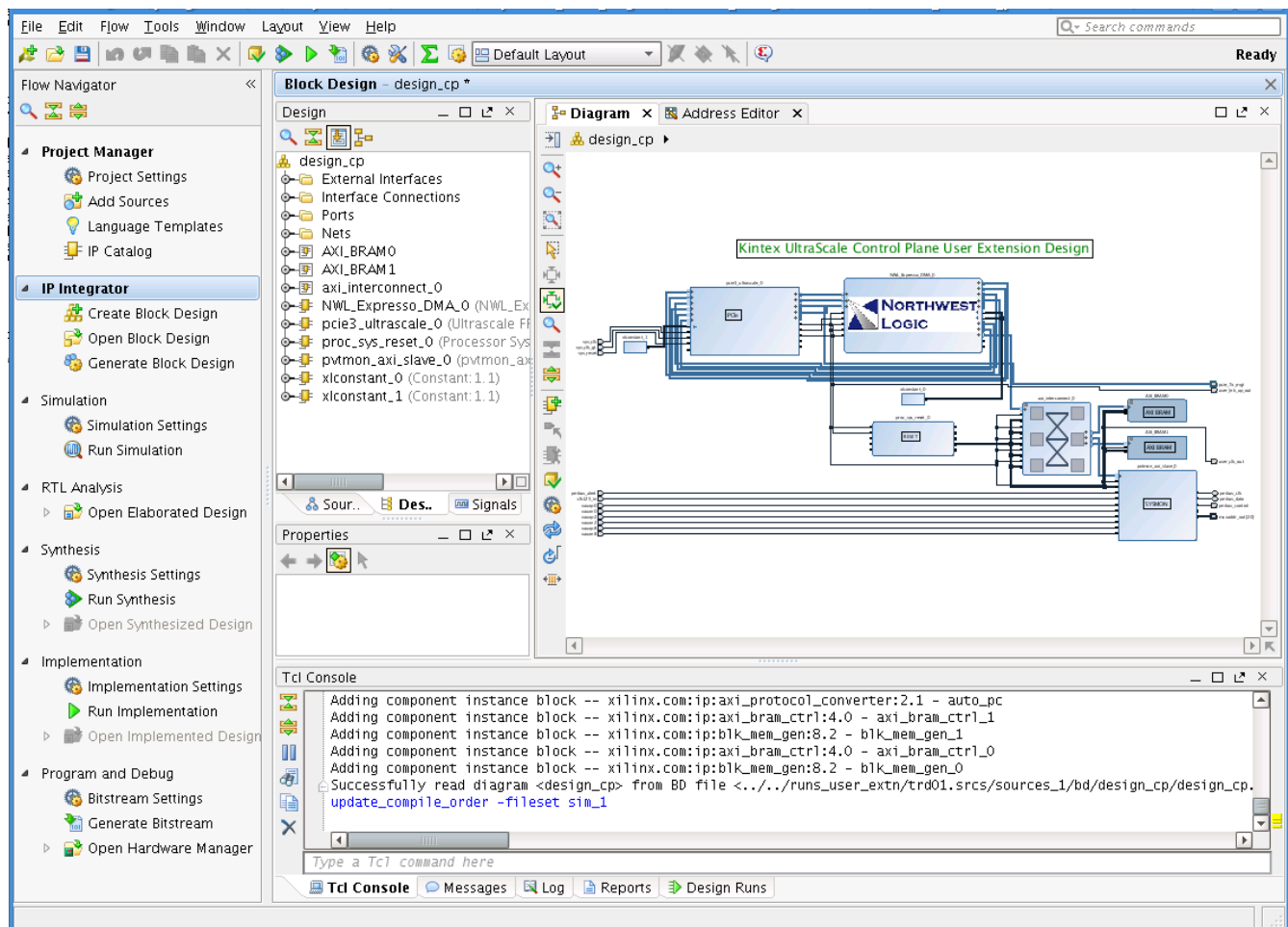
Figure 4-2: Base Design—Generate Bitstream

## Implementing the User Extension Design

1. Open a terminal window on a Linux system with the Vivado environment set up, or open a Vivado tools Tcl shell on a Windows system.
2. Navigate to the `kcu105_control_plane/hardware/vivado/scripts/user_extn` folder.
3. To run the implementation flow, enter:

```
$ vivado -source trd01_user_extn.tcl
```

This opens the Vivado IDE, loads the block diagram, and adds the required top file and XDC file to the project (see [Figure 4-3](#)).

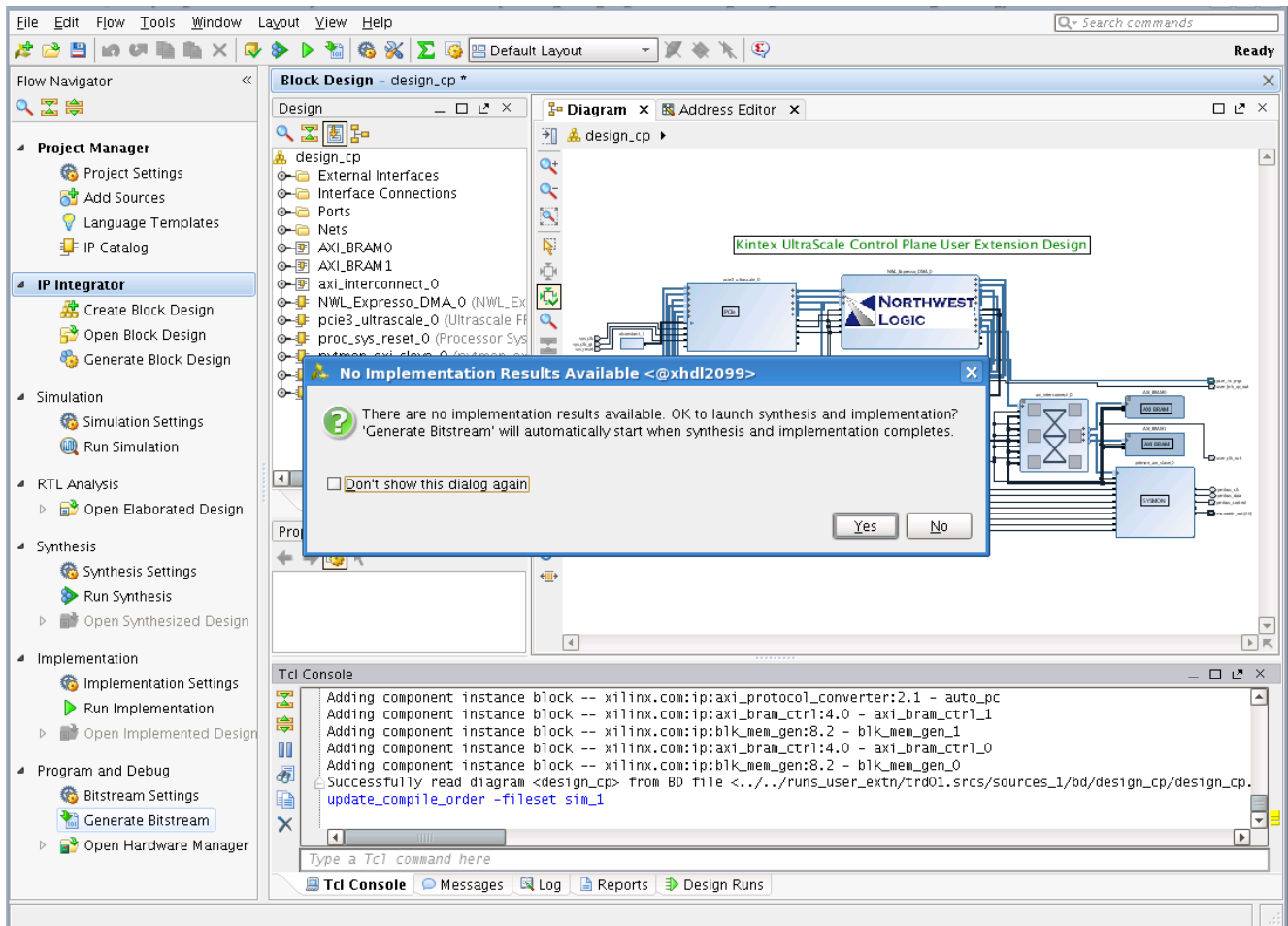


UG918\_c4\_03\_011415

Figure 4-3: User Extension Design—Project View

- In the Flow Navigator panel, click the **Generate Bitstream** option which runs synthesis, implementation, and generates the bit file (see Figure 4-4). The generated bitstream can be found under the following directory:

kcu105\_control\_plane/hardware/vivado/runs\_user\_extn/trd01.runs/  
impl\_1/



UG918\_c4\_04\_011415

Figure 4-4: User Extension Design—Generate Bitstream

---

## Simulating the Base Design Using Vivado Simulator

The targeted reference design can be simulated using the Vivado simulator. The testbench and the Endpoint PCIe IP block are configured to use the PHY Interface for PCI Express (PIPE) mode simulation.

The test bench initializes the bridge, does one double word (DW) write to BAR-mapped address space, reads back from the same address, and compares the data with expected pattern.

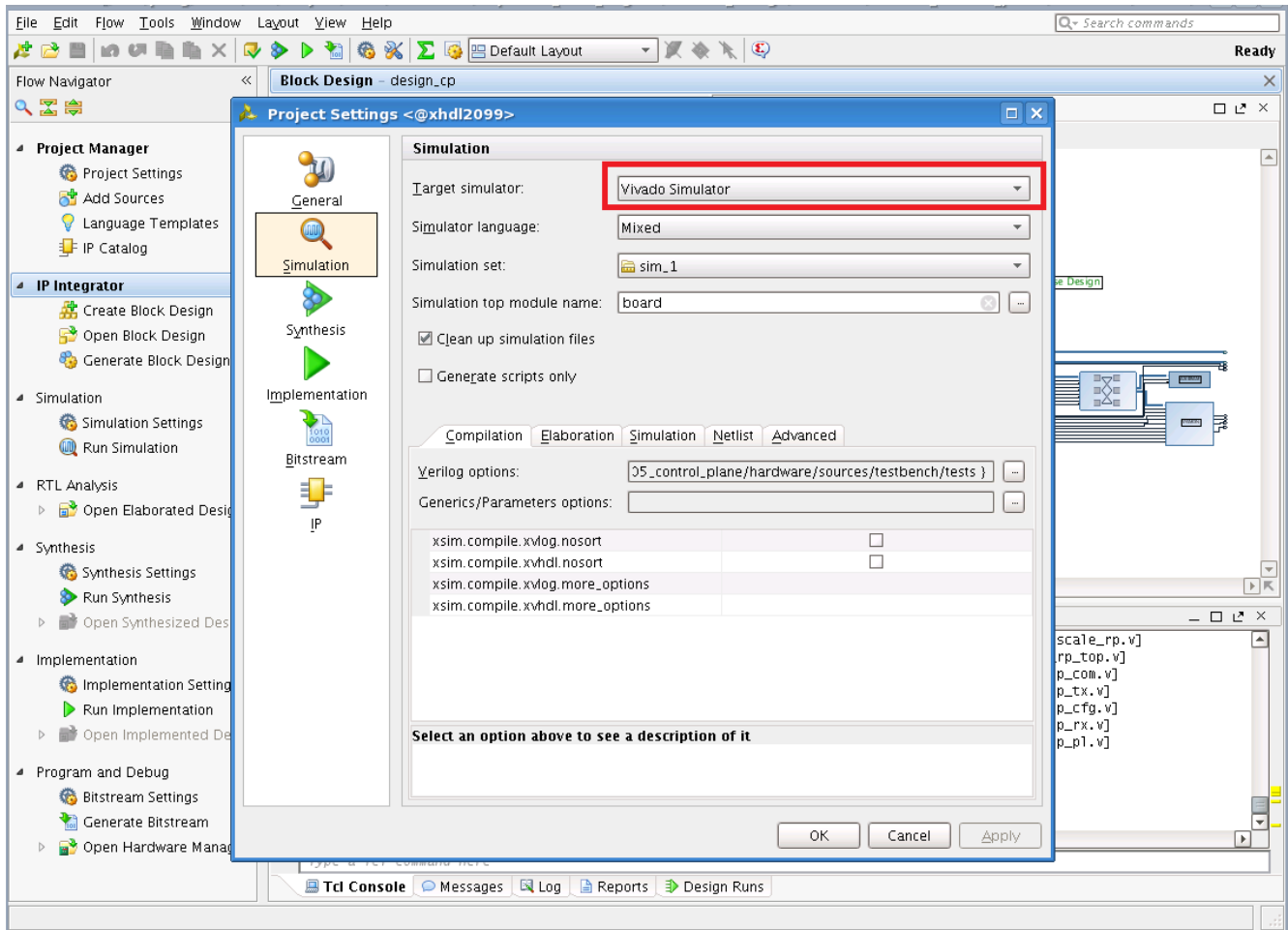
Simulation setup is provided only for the base design and not for the pre-built user extension design.

### Running Simulation Using the Vivado Simulator

1. Open a terminal window on a Linux system with the Vivado environment set up, or open a Vivado tools Tcl shell on a Windows system.
2. Navigate to the `kcu105_control_plane/hardware/vivado/scripts/base` folder.
3. To run simulation, enter:

```
$ vivado -source trd01_base.tcl
```

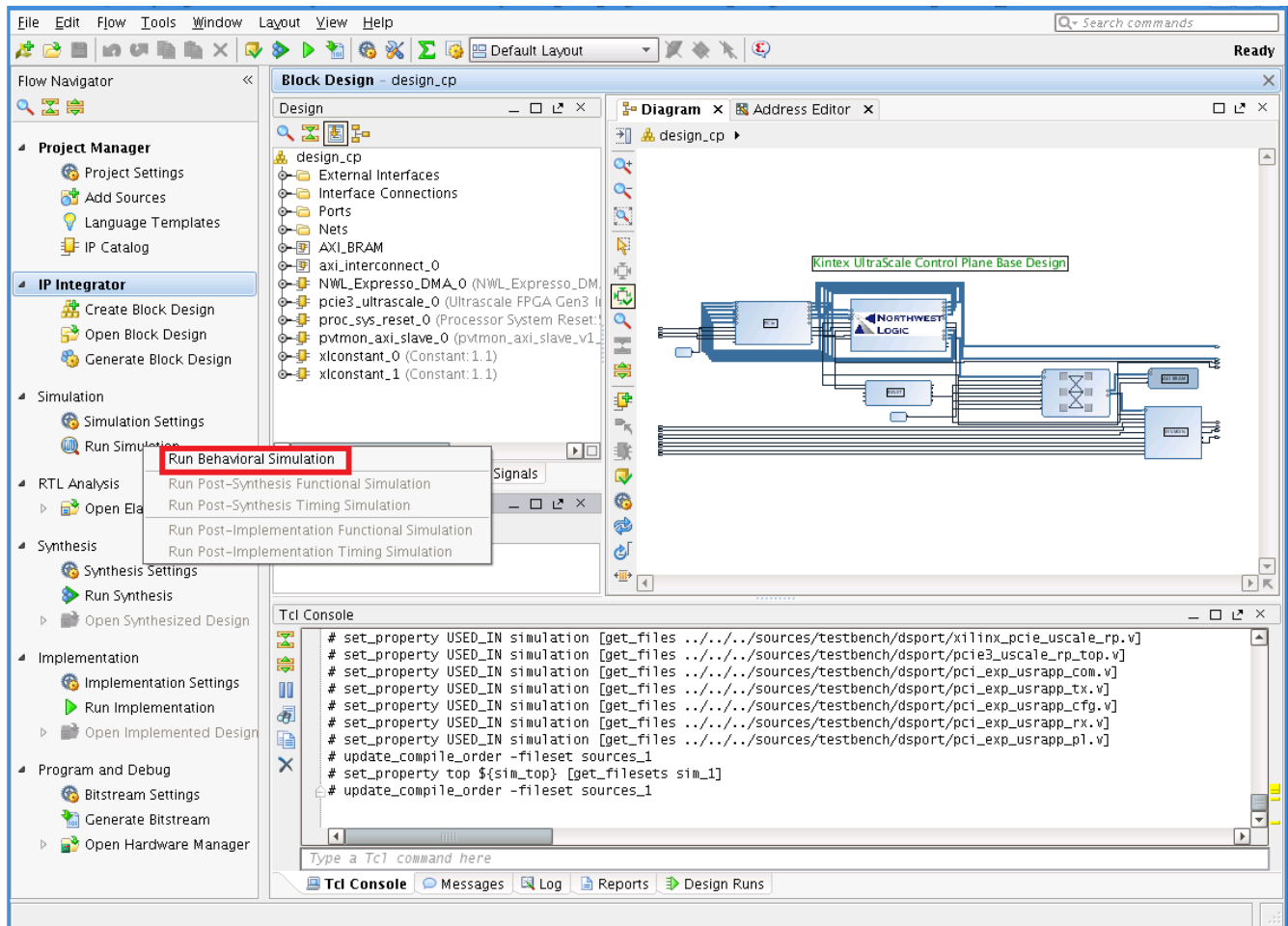
This opens the Vivado IDE with the target simulator set to the Vivado Simulator (Figure 4-5).



UG918\_c4\_05\_011415

Figure 4-5: Base Design Project Settings for Simulation

- In the Flow Navigator panel, under Simulation, click **Run Simulation** and select **Run Behavioral Simulation**. This generates all the simulation files, loads the Vivado simulator, and runs the simulation. The result is shown in Figure 4-6.



UG918\_c4\_06\_011415

Figure 4-6: Base Design Behavioral Simulation using the Vivado Simulator

# Targeted Reference Design Details and Modifications

This chapter describes the TRD hardware design and software components in detail, and provides modifications to add an additional AXI block RAM controller to the design.

## Hardware

The functional block diagram in [Figure 5-1](#) identifies the different TRD hardware design components. Subsequent sections discuss each of the components in detail.

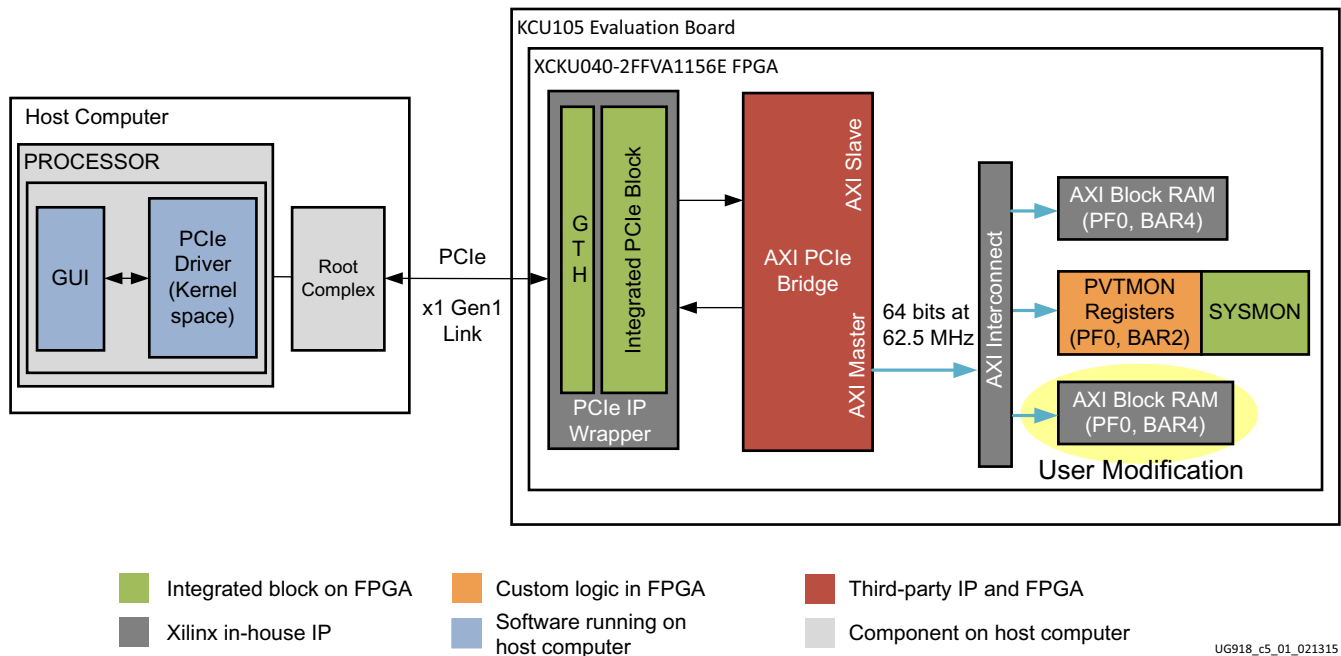


Figure 5-1: TRD Functional Block Diagram

## Endpoint Block for PCI Express

The PCI Express IP for is used in the following configuration:

- x1 Gen1 line rate (2.5 GT/s/lane/direction)
- Three 64-bit BARs

See *LogiCORE IP UltraScale FPGAs Gen3 Integrated Block for PCI Express Product Guide* (PG156) [Ref 4] for more information.

## DMA Bridge Core

The DMA bridge core includes an AXI-PCIe bridge and Espresso DMA in one netlist bundle. See the Northwest Logic Espresso DMA Bridge Core website to obtain a user guide [Ref 5].

**Note:** The IP netlist used in the reference design supports a fixed configuration where the number of DMA channels and translation regions is fixed; for higher configurations of the IP, contact Northwest Logic.

**Note:** The Northwest Logic Espresso IP provided with the design is an evaluation version of the IP. It times out in hardware after 12 hours. To obtain a full license of the IP, contact Northwest Logic.

## AXI-PCIe Bridge

The AXI-PCIe bridge translates protocol to support transactions between the PCIe and AXI3 domains. It provides support for two ingress translation regions to convert PCIe BAR-mapped transactions to AXI3 domain transactions.

### Bridge Initialization

The AXI-PCIe bridge consumes transactions hitting BAR0 in the Endpoint.

- The bridge registers are accessible from BAR0 + 0x8000.
- During ingress translation initialization:
  - Two ingress translations are enabled (0x800 and 0x820).
  - Address translation is set up as shown in Table 5-1.

For example, assume that the PCIe BAR2 physical address is 0x2E000000. Any memory read request targeted to address 0x2E000000 is translated to 0x44A00000.

Table 5-1: Address Translation Maps

Ingress Source Base	Ingress Destination Base	Comments
BAR2	0x44A00000	PCIe BAR2 mapped to power monitor slave
BAR4	0xC0000000	PCIe (BAR4) mapped to AXI block RAM

- During bridge register initialization:
  - Bridge base low (0x210) is programmed to (BAR0 + 0x8000).
  - Bridge Control register (0x208) is programmed to set the bridge size and enable translation.
- After bridge translation has been enabled, the ingress registers can be accessed with Bridge Base + 0x800.

### **Expresso DMA**

Key features of Expresso DMA are:

- High-Performance Scatter Gather DMA, designed to achieve full bandwidth of AXI and PCIe
- Separate source and destination scatter-gather queues with separate source and destination DMA completion Status queues
- DMA channels merge the source and destination scatter gather information

### **AXI Block RAM Controller**

The AXI block RAM controller provides block RAM with an AXI4 memory-mapped interface. This behaves as a register file in this design to which BAR-mapped transactions are targeted.

See *LogiCORE IP AXI Block RAM (BRAM) Controller Product Guide* (PG078) [Ref 6] for more details.

### **AXI Interconnect**

The AXI Interconnect is used to connect the various IPs together in a memory-mapped system. The interconnect is responsible for:

- Converting AXI3 transactions from AXI-PCIe bridge into AXI4 transactions for various slaves
- Decoding address to target appropriate slave

See *LogiCORE IP AXI Interconnect Product Guide* (PG059) [Ref 7] for more details.

## Power and Temperature Monitoring

The design uses a SYSMON block to provide system power and die temperature monitoring capabilities. The design uses a SYSMON block (17 channel, 200 ksps) to provide system power and die temperature monitoring capabilities. The block provides analog-to-digital conversion and monitoring capabilities. It enables reading of voltage and current on different power supply rails (supported on the KCU105 board) which are then used to calculate power.

A lightweight PicoBlaze™ controller is used to set up the SYSMON registers in continuous sequence mode and read various rail data periodically. The output from PicoBlaze is made available in block RAM and an FSM reads various rails from the block RAM (as shown in Figure 5-2), and updates the user space registers. These registers can be accessed over PCIe through the BAR-mapped region.

The AXI4 Lite IPIF core is used in the design and the interface logic between the block RAM and the AXI4 Lite IPIF reads the power and temperature monitor registers from block RAM. Providing an AXI4 Lite slave interface adds the flexibility of using the module in other designs.

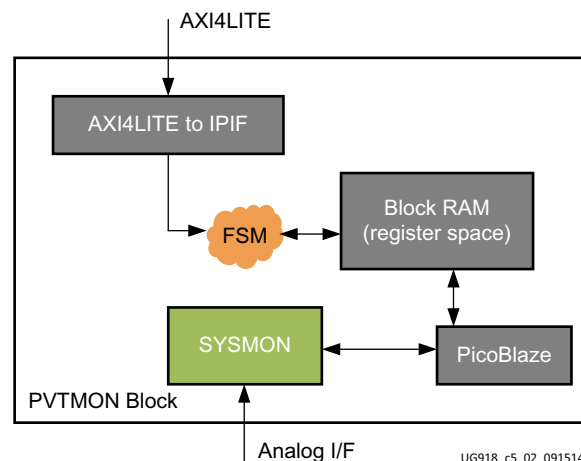


Figure 5-2: Power and Temperature Monitoring Block

See the *UltraScale Architecture System Monitor User Guide* (UG580) [Ref 8] for more details.

## Data Flow

This section summarizes the TRD data flow.

### *Transmit Path*

1. The GUI opens the driver interface for read/write functionality.
2. The GUI issues WRITE system calls for writing into BAR-mapped registers based on your input.
3. The driver writes appropriate values into BAR-mapped registers.

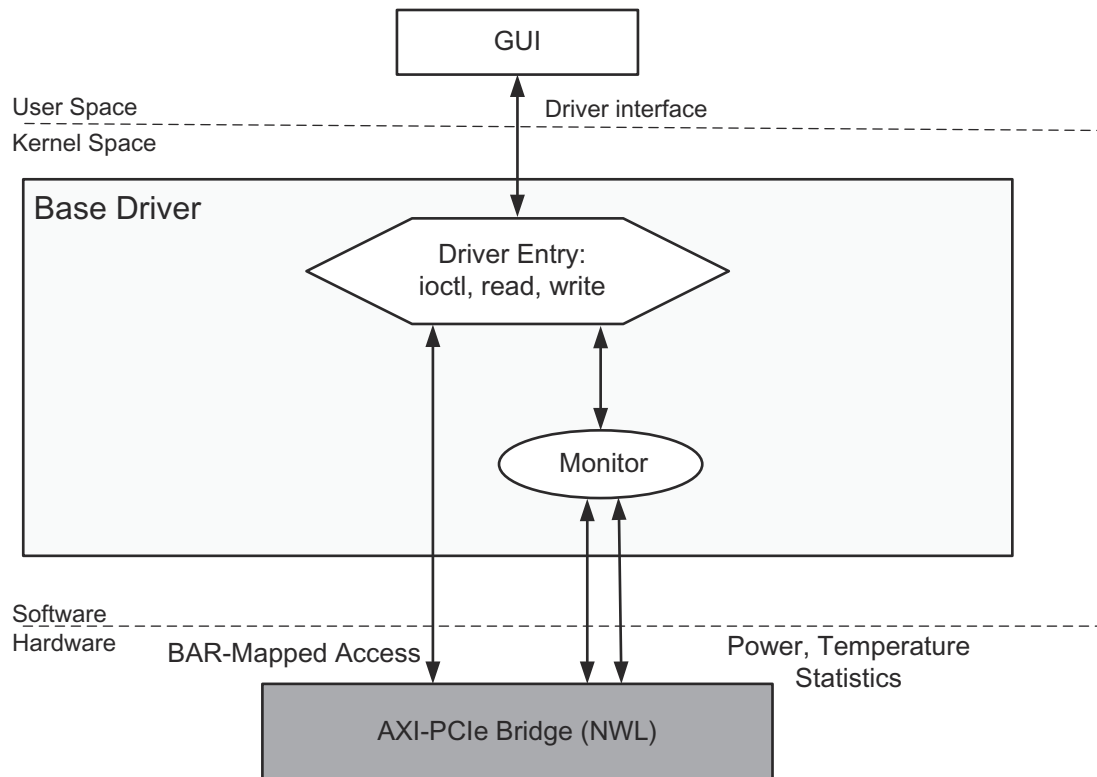
### *Receive Path*

1. The GUI issues a READ system call to read BAR-mapped registers based on your input.
2. The Character driver reads appropriate BAR-mapped registers and conveys the readings to the GUI.
3. The GUI displays the read BAR-mapped register information.

# Software

## Software Architecture

The software component of the TRD framework comprises a kernel space base driver module (see [Figure 5-3](#)) and a user space GUI that controls design operation. The software's building blocks are designed with scalability in mind. Additional user-space applications can be designed with the existing blocks.



UG918\_c5\_03\_040615

Figure 5-3: TRD Architecture

## Graphical User Interface

The user-space GUI is a Java-based GUI that provides these features:

- GUI management of the driver and device
  - In Linux, the GUI installs the selected design mode drivers and can configure and control device and test parameters.
  - In Windows, the GUI can configure and control device and test parameters.

- GUI front end graphical display of statistics collected from the underlying driver through the driver interface.

The TRD demonstrates the use of PCIe in control plane applications. A simple kernel driver on a host computer demonstrates BAR-mapped single double word (DW) register transfers.

Apart from generic GUI functionality described previously, the GUI allows you to read from or write to BAR-mapped registers in hardware and display them in a GUI window.

---

## Reference Design Modifications

Adding a pre-built additional AXI block RAM controller is included in the TRD as an extension of the base design. This section describes how to add an additional AXI block RAM controller to the design and set up ingress translations through BAR4 to access this memory space.

### ***Rebuilding Hardware***

A pre-built design script is provided that can be run to generate a bitstream with an additional AXI block RAM controller added. The additional block RAM is mapped to AXI address 0xD000\_0000. The steps needed to build the user modification design are described in [Chapter 4, Implementing and Simulating the Design](#).

### ***Software Modification***




---

**IMPORTANT:** *The pre-built user extension design can be tested only on Linux and not on Windows. There is no support for the user extension design in the Windows platform.*

---

In the software driver, access to newly added block RAM can be added as follows.

In the file `software/linux_driver_app/driver/ctrlplane/xpcie.c`, under the `InitBridge` function, the line can be changed as shown:

```
// - Program DST address to be AXI domain address for BRAM Controller
XIo_Out32((bar0_addr + REG_BRDG_BASE + REG_INGR_AXI_BASE + SECOND_TRANS
+ OFFSET_INGR_AXI_DST_LO ), 0xC0000000);
// - Program DST address to be AXI domain address for BRAM Controller
XIo_Out32((bar0_addr + REG_BRDG_BASE + REG_INGR_AXI_BASE + SECOND_TRANS
+ OFFSET_INGR_AXI_DST_LO ), 0xD0000000);
```

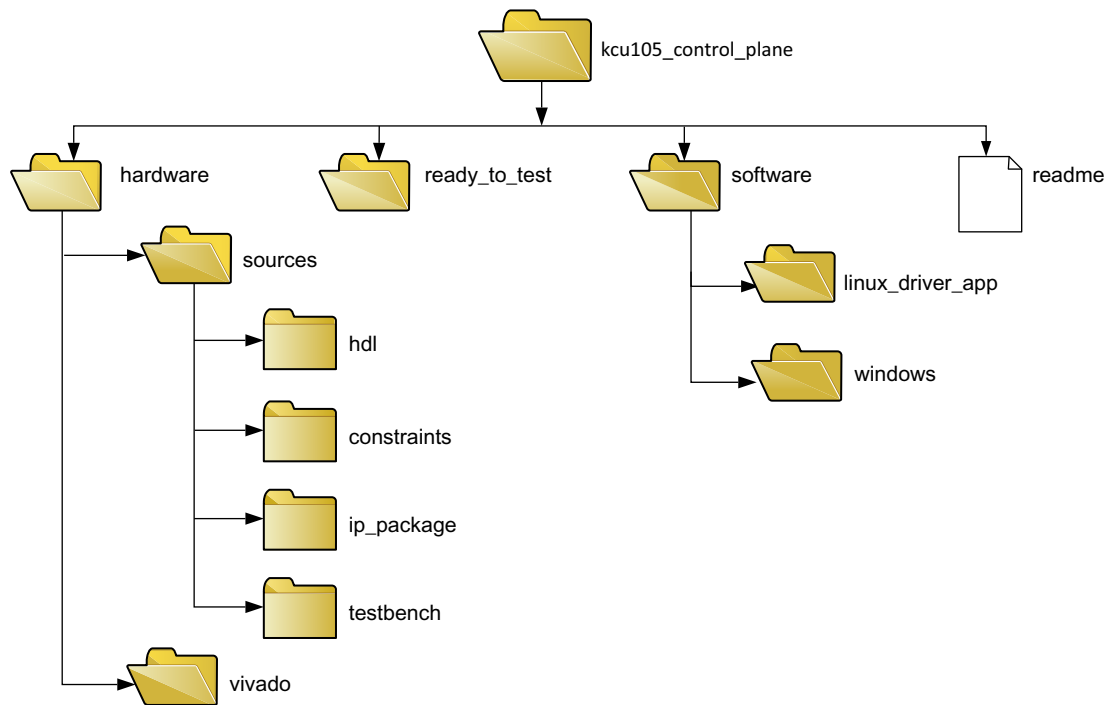
This maps the newly added block RAM controller to BAR4. With this minor change, the same GUI can be used for read/write access.

To enable read access to both block RAM controllers, an additional ingress translation aperture can be mapped and the ReadUserReg function in the driver can be used to access those registers. The display from the software driver can be seen in the system dmesg log.

The ability to read multiple user registers or block RAM controllers is not supported by the GUI, and the aperture size is currently limited to 4K.

# Directory Structure

The directory structure for the TRD is shown in [Figure A-1](#) and described in [Table A-1](#). For a detailed description of each folder, see the Readme file.



UG918\_aA\_01\_040315

Figure A-1: TRD Directory Structure

Table A-1: Directory Description

Folder	Description
readme	A TXT file that includes revision history information, steps to implement and simulate the design, required Vivado® tool software version, and known limitations of the design (if any).
hardware	Contains hardware design deliverables
sources	
hdl	Contains HDL files
constraints	Contains constraint files
ip_package	Contains custom IP packages
testbench	Contains test bench files
vivado	Contains scripts to create a Vivado Design Suite project and outputs of Vivado runs
ready to test	Contains the BIT file to program the KCU105 PCI Express® Control Plane application
software	Contains software design deliverables for Linux and Windows
linux_driver_app	
windows	

# Recommended Practices and Troubleshooting in Windows

---

## Recommended Practices

---



**RECOMMENDED:** *Make a backup of the system image and files using the Backup and Restore utility of the Windows 7 operating system before installing reference design drivers. (As a precautionary measure, a fresh installation of the Windows 7 OS is recommended for testing the reference design.)*

---

## Troubleshooting

**Problem:** The TRD Setup screen of the GUI does not detect the board.

**Corrective Actions:**

1. If the GUI does not detect the board, open **Device Manager** and see if the drivers are loaded under **Xilinx PCI Express Device**.
2. If the drivers are not loaded, check the PCIe Link Up LED on the board (see [Figure 3-6](#)).
3. If the drivers are loaded but the GUI is not detecting the board, remove non-present devices from Device Manager using the following steps.
  - a. Open a command prompt with Administrator privileges.
  - b. At the command prompt, enter the following bold text:  
**set devmgr\_show\_nonpresent\_devices=1**  
**start devmgmt.msc**
  - c. Click the **View** menu and select **Show hidden devices** on the Device Manager window.
  - d. Non-present devices are indicated by a lighter shade of text.
  - e. Look for all the Non Present/Hidden devices. Right-click each one, and select **Uninstall**. Remove the driver if prompted for it.
4. Invoke the GUI of the reference design and check if it detects the board.

# Additional Resources and Legal Notices

---

## Xilinx Resources

For support resources such as Answers, Documentation, Downloads, and Forums, see [Xilinx Support](#).

For continual updates, add the Answer Record to your [myAlerts](#).

---

## Solution Centers

See the [Xilinx Solution Centers](#) for support on devices, software tools, and intellectual property at all stages of the design cycle. Topics include design assistance, advisories, and troubleshooting tips.

---

## References

The most up-to-date information for this design is available on these websites:

[KCU105 Evaluation Kit website](#)

[KCU105 Evaluation Kit documentation](#)

[KCU105 Evaluation Kit Master Answer Record \(AR 63175\)](#)

These documents and sites provide supplemental material:

1. [Northwest Logic Espresso DMA Bridge Core](#)
2. *Vivado Design Suite User Guide Release Notes, Installation, and Licensing* ([UG973](#))
3. *Kintex UltraScale FPGA KCU105 Evaluation Board User Guide* ([UG917](#))
4. *LogiCORE IP UltraScale FPGAs Gen3 Integrated Block for PCI Express Product Guide* ([PG156](#))
5. [Northwest Logic PCI Express Solution](#)

6. *LogiCORE IP AXI Block RAM (BRAM) Controller Product Guide* ([PG078](#))
7. *LogiCORE IP AXI Interconnect Product Guide* ([PG059](#))
8. *UltraScale Architecture System Monitor User Guide* ([UG580](#))

---

## Please Read: Important Legal Notices

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 Xilinx's limited warranty, please refer to Xilinx's Terms of Sale which can be viewed at <http://www.xilinx.com/legal.htm#tos>; 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 such critical applications, please refer to Xilinx's Terms of Sale which can be viewed at <http://www.xilinx.com/legal.htm#tos>.

### 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.

### Fedora Information

Xilinx obtained the Fedora Linux software from Fedora (<http://fedoraproject.org/>), and you may too. Xilinx made no changes to the software obtained from Fedora. If you desire to use Fedora Linux software in your product, Xilinx encourages you to obtain Fedora Linux software directly from Fedora (<http://fedoraproject.org/>), even though we are providing to you a copy of the corresponding source code as provided to us by Fedora. Portions of the Fedora software may be covered by the GNU General Public license as well as many other applicable open source licenses. Please review the source code in detail for further information. To the maximum extent permitted by applicable law and if not prohibited by any such third-party licenses, (1) XILINX DISCLAIMS ANY AND ALL EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE; AND (2) IN NO EVENT SHALL XILINX BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Fedora software and technical information is subject to the U.S. Export Administration Regulations and other U.S. and foreign law, and may not be exported or re-exported to certain countries (currently Cuba, Iran, Iraq, North Korea, Sudan, and Syria) or to persons or entities prohibited from receiving U.S. exports (including those (a) on the Bureau of Industry and Security Denied Parties List or Entity List, (b) on the Office of Foreign Assets Control list of Specially Designated Nationals and Blocked Persons, and (c) involved with missile technology or nuclear, chemical or biological weapons). You may not download Fedora software or technical information if you are located in one of these countries, or otherwise affected by these restrictions. You may not provide Fedora software or technical information to individuals or entities located in one of these countries or otherwise affected by these restrictions. You are also responsible for compliance with foreign law requirements applicable to the import and use of Fedora software and technical information.

© Copyright 2014–2015 Xilinx, Inc. Xilinx, the Xilinx logo, Artix, ISE, Kintex, Spartan, Virtex, Vivado, Zynq, and other designated brands included herein are trademarks of Xilinx in the United States and other countries. PCI, PCIe and PCI Express are trademarks of PCI-SIG and used under license. All other trademarks are the property of their respective owners.