



Michigan State University College of Engineering
ECE 480
Meng-Chiao, Lee
March 31th 2011

Custom logic Implementation in Camera Frame Buffer Demo Reference Design

Application note

Executive Summary

The Xilinx Spartan 3A-3400 DSP Video Starter Kit (VSK) provides users the functionality to develop a variety of video and image processing applications on Field Programmable Gate Array (FPGA). With the utilization of Xilinx's Embedded Development Kit (EDK) and System Generator, users can develop a custom application by modifying the reference design demo. The purpose of this application note is to guide the users to implement custom logic in the Camera Frame Buffer Demo reference design which is provided by Xilinx.



Table of Contents

Introduction	3
Purpose	3
Software Required	3
Hardware Required	3
Background Materials	3
System Generator	3
Xilinx Platform Studio (XPS)	4
Procedure.....	4
Conclusion.....	11
References	11



Introduction

There are variety of applications involve with FPGA boards such digital signal processing, medical imaging, computer vision, and radio astronomy. It's due to the reason that FPGAs contain array of programmable logic components and reconfigurable interconnects, the flexibility of FPGAs allows users to design a variety of applications. With the Xilinx's design tools, users can easily design and implement a new model into FPGAs without using hard description language (HDL).

Purpose

The objective of this application note is to demonstrate a process of implementing custom logic into the Camera Frame Buffer Demo reference design and simulating the design by using Xilinx Spartan 3A-3400 Xtreme DSP Video Starter Kit. This application note will emphasize the process of using System Generator and the Xilinx EDK to implement a custom logic instead of focusing on the logic itself. Therefore, it will utilize the vehicle detection logic which is a designed model base on the Camera Frame Buffer Demo reference design as a demo. The concept of this demonstration can further be used to implement specific custom logic by the users.

Software Required

- Xilinx ISE Design Suite 11.4
- System Generator 11.4
- MATLAB 2008a with Simulink

Hardware Required

- Xilinx Spartan 3A-3400 Xtreme DSP Video Starter Kit
- Monitor

Background Material

System Generator

System Generator is a digital signal processing design tool that enables the use of MATLAB/simulink to design a model for programming FPGA board. By using a Xilinx specific



blockset, all of the downstream FPGA implementation and necessary files would be generated to program the FPGA automatically.

Xilinx Platform Studio (XPS)

XPS is one of a software tool in The Embedded Development Kit (EDK) that enables designers to create customized embedded processor systems and integrate the designs into Xilinx FPGA boards. It provides a graphical user interface that allows users to perform their embedded system design tasks from design entry to design debug and verification.

Procedure

A) Download EDK demonstration file

1. Go to Xilinx web page by copying and pasting the following website on your browser, <http://www.xilinx.com/support/documentation/do-s3adsp-video-sk-uni-g.htm>.
2. Click the file **ug456_11_4_vsk_v3_0.zip** right under ***Spartan-3A DSP FPGA Video Starter Kit User Guide***.
3. Sign in to Xilinx with your user ID and password. If you don't have a Xilinx account yet, create one by clicking **Create Account** icon.
4. Click **Download Design File**, and save the file into any directory of your interest.
5. Extract the whole file into the same directory.

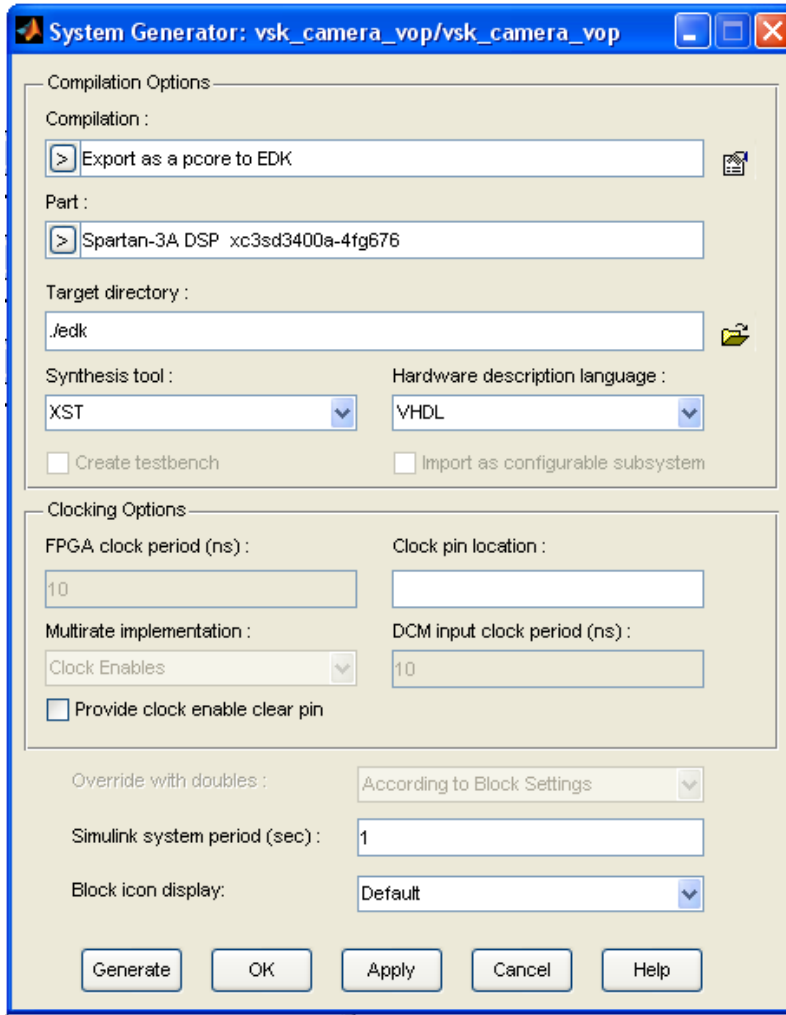
B) Download vehicle detection model

1. Go to Spring 2011 ECE480 Team 4 web page by copying and pasting the following website on your browser, <http://www.egr.msu.edu/classes/ece480/capstone/spring11/group04/documents.html>.
2. Download **vehicle_detection.mdl**, and save the file into any directory of your interest.

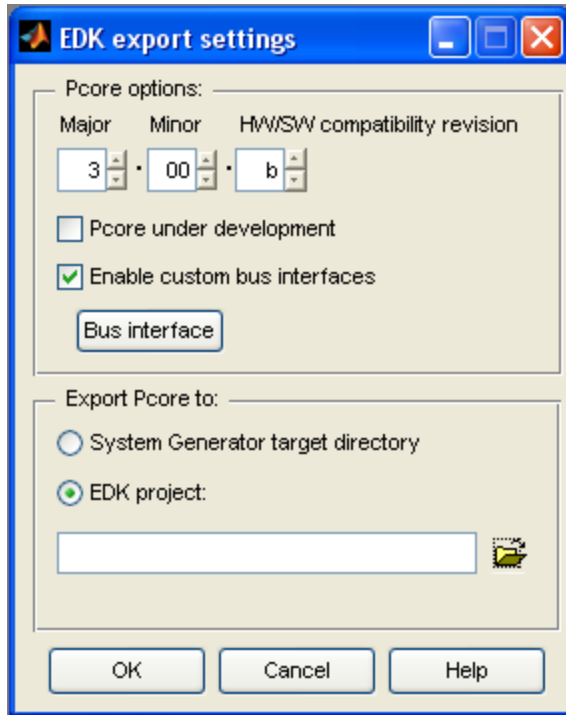
C) Pcore and XPS file generation

1. Double-click the file **vehicle_detection.mdl** to open the file and you will see the following window on your screen.

- Once the program is opened, we can start to generate the PCORE that will be integrated in the embedded system design in XPS and all necessary files by using system generator. Double-click the **System Generator** token in the model to bring up a **compilation options** window as following.



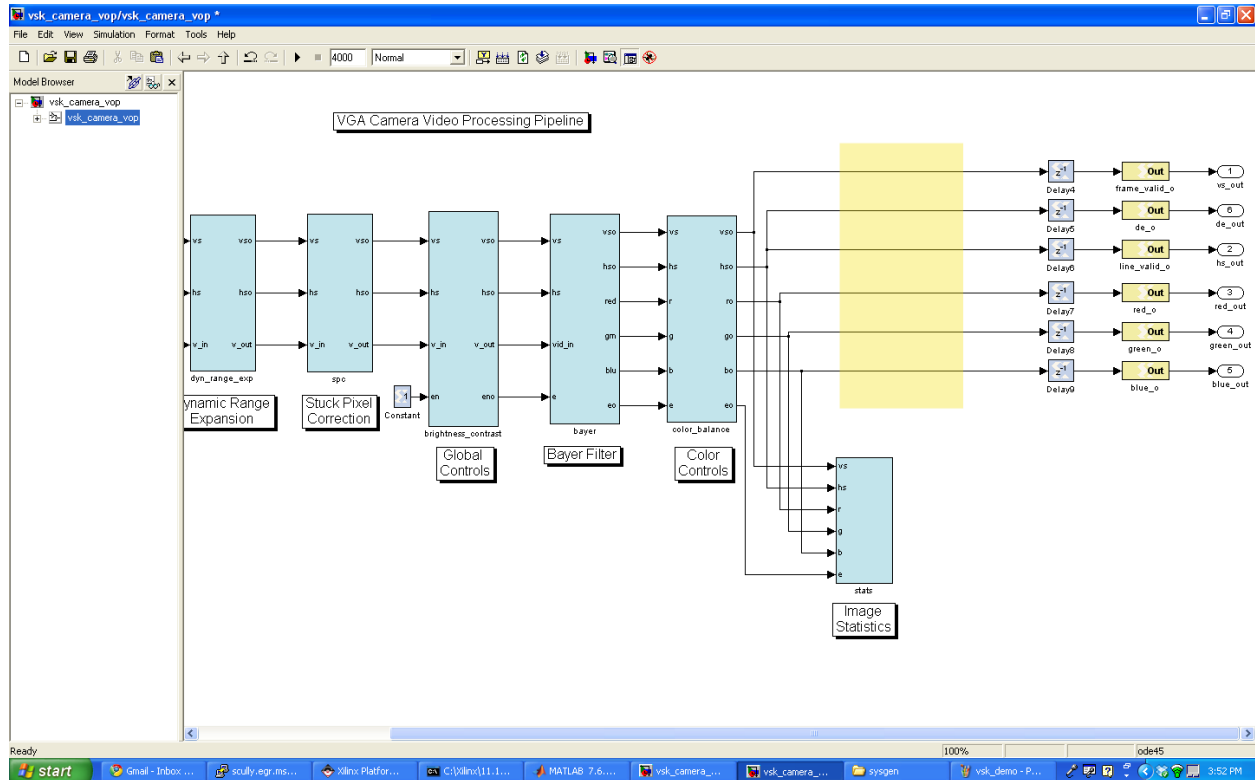
- Under compilation, make sure the **Export as a pcore to EDK** is selected.
- Now we are going to select the specific project where the Pcore will be exported. Click the icon at the end of the line to bring up a **EDK export settings** window as following figure.



5. Under **Export Pcore to:**, select **EDK project** and click the folder to browse for the project. You can find the file from the same directory where you store unzip the **ug456_11_4_vsk_v3_0.zip** and **ug456_11_4_vsk_v3_0 > Demonstrations > EDK > EDK_Demonstration > Camera_Frame_Buffer_Demo > system.xmp**. Select **OK** when the file is found.
6. The next step is to specify the target directory for the Pcore. We will select the same directory as the original Pcore used in the demo to save the need to connect the Pcore to the proper buses once the project is opened in XPS. Click the folder and browse the Pcore from you saved location and then **ug456_11_4_vsk_v3_0 > Demonstrations > EDK > EDK_Demonstration > Camera_Frame_Buffer_Demo > Pcores**.
7. Now, you are ready to generate the files. Click **Generate** button to start the process. A small window will pop up to show the processing. (It usually take couple minutes to generate the files) Once the generating process is done, you can close the MATLAB program.

For the future design, you can start from the model in the demo located at **ug456_11_4_vsk_v3_0 > Demonstrations > EDK > EDK_Demonstration > Camera_Frame_Buffer_Demo > Pcores > vsk_camera_vop_plbw_v3_00_b > sysgen > vsk_camera_vop.mdl**.

In this model, you can insert a custom logic after the Color Correction block in the highlighted area shown in the figure below.

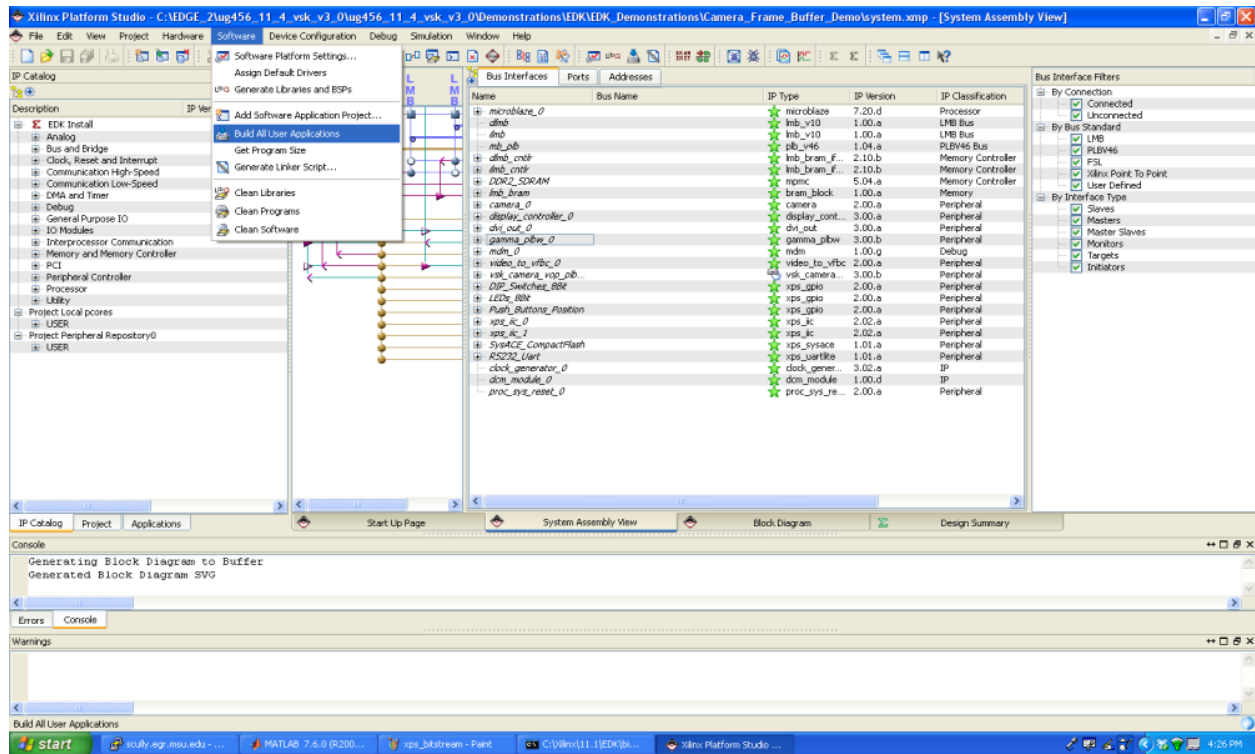


D) Hardware implementation and simulation

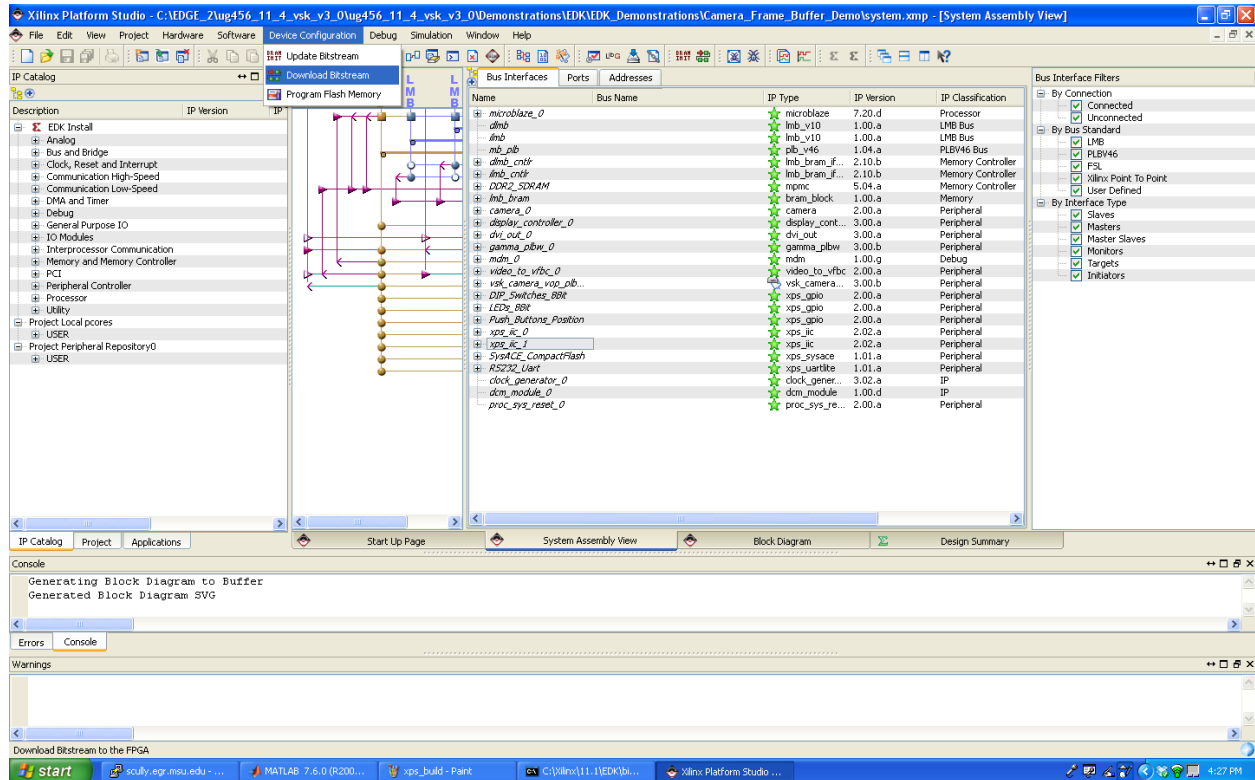
1. Open Xilinx Platform Studio 11. It is located at **All program > Xilinx ISE Design Suite 11 > EDK > Xilinx Platform Studio**.
2. Once the program is opened, it will pop up a prompt and ask for users to select a blank XPS project or open a recent project. Select **Open a recent project** and browse the same system.xmp file as step C-5 located at **ug456_11_4_vsk_v3_0 > Demonstrations > EDK > EDK_Demonstration > Camera_Frame_Buffer_Demo > system.xmp**. And then select **OK**. The project will be opened and you should see the following on your screen.

-
- The screenshot shows the Xilinx Platform Studio (Vivado) interface. The top menu bar includes File, Edit, View, Project, Hardware, Software, Device Configuration, Debug, Simulation, Window, and Help. The 'Hardware' menu is open, and the 'Generate Bitstream' option is highlighted. The main workspace displays a block diagram of a system with various IP blocks and interconnects. The 'IP Catalog' pane on the left shows a list of available IP blocks, and the 'Bus Interfaces' pane on the right shows a table of connected IP blocks and their properties.
- | Name | Bus Name | IP Type | IP Version | IP Classification |
|-----------------------|----------|----------------|------------|-------------------|
| microblaze_0 | | microblaze | 7.20.d | Processor |
| lmb_v10 | | lmb_v10 | 1.00.a | LMB Bus |
| lmb_v10 | | lmb_v10 | 1.00.a | LMB Bus |
| plb_v46 | | plb_v46 | 1.04.a | PLB46 Bus |
| lmb_bram_0 | | lmb_bram_0 | 2.10.b | Memory Controller |
| lmb_bram_1 | | lmb_bram_1 | 2.10.b | Memory Controller |
| mpmc | | mpmc | 5.04.a | Memory Controller |
| bram_block | | bram_block | 1.00.a | Memory |
| camera_0 | | camera | 2.00.a | Peripheral |
| display_cont | | display_cont | 3.00.a | Peripheral |
| drv_out_0 | | drv_out_0 | 3.00.a | Peripheral |
| gamma_pwm_0 | | gamma_pwm | 3.00.b | Peripheral |
| mdm_0 | | mdm | 1.00.0 | Debug |
| video_to_vfbc_0 | | video_to_vfbc | 2.00.a | Peripheral |
| vsk_camera_vop_pwm... | | vsk_camera... | 3.00.b | Peripheral |
| DSP_Switches_000 | | xps_gpio | 2.00.a | Peripheral |
| LED_000 | | xps_gpio | 2.00.a | Peripheral |
| Push_Buttons_Position | | xps_gpio | 2.00.a | Peripheral |
| xps_ic_0 | | xps_ic | 2.02.a | Peripheral |
| xps_ic_1 | | xps_ic | 2.02.a | Peripheral |
| SysADC_CompactFlash | | xps_sysadc | 1.01.a | Peripheral |
| AS232_uart | | xps_uartlite | 1.01.a | Peripheral |
| clock_generator_0 | | clock_gener... | 3.02.a | IP |
| dcm_module_0 | | dcm_module | 1.00.d | IP |
| proc_sys_reset_0 | | proc_sys_re... | 2.00.a | Peripheral |

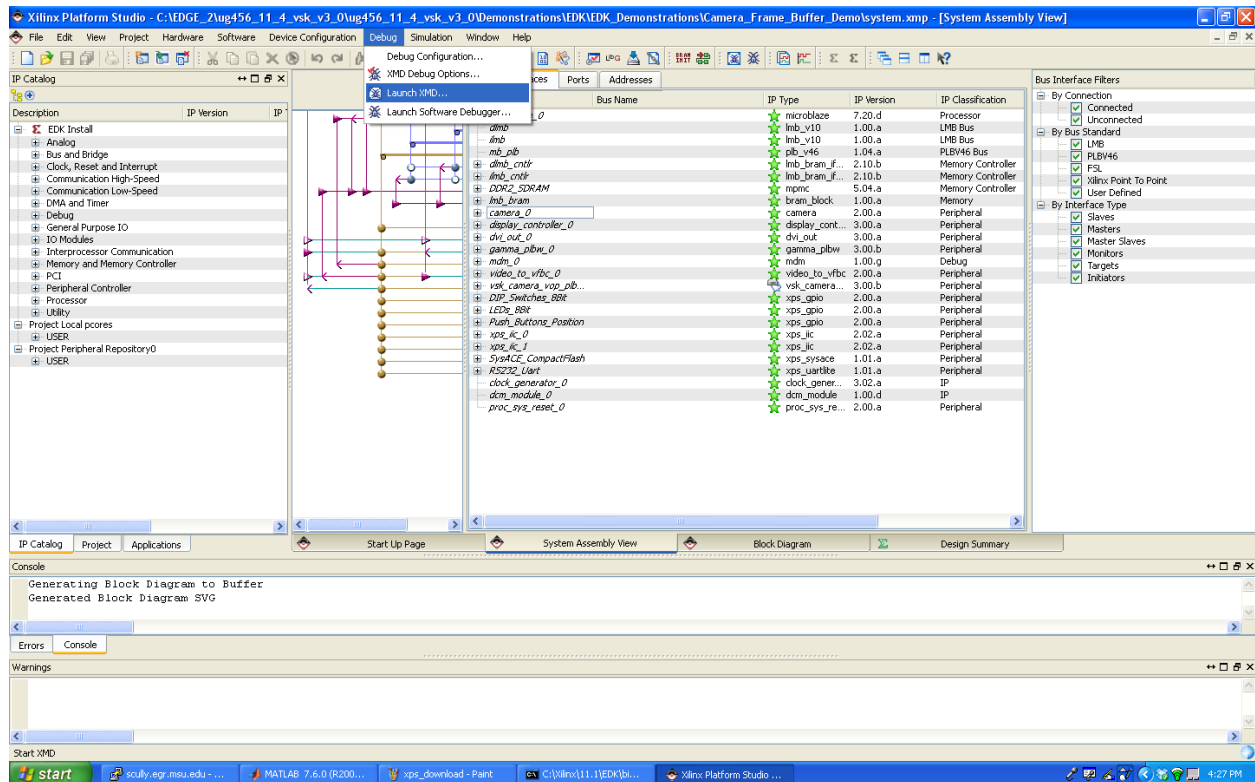
- Build the software application to create an executable file that will run the Microblaze processor. Under the main menu, select **Software > Build All user Applications**.



- Once the bitstream and software application is created, we can download the bitstream into the board. Under the main menu, select **Device Configuration > Download Bitstream**.



6. Now, we are ready to simulate the model. Under the main menu, select **Debug > Launch XMD**. A command will pop up on the screen.





7. Once the command window is ready to accept input, type **dow Camera_Frame_Buffer_Sw/executable.elf** and then hit **enter** to access into the executable file.
8. Finally, type **run** and hit **enter** to execute the simulation. Now, you can see the output on the monitor you connected.

Conclusion

This application note demonstrates a method to implement a custom logic into the FPGA board by using System Generator and Xilinx Platform Studio. It shows all the necessary process in order to achieve the goal. Once the concept is understood, the users can further design a more advance model into the FPGA boards.

References

- “System Generator for DSP: Getting Started Guide”
- “Platform Studio User Guide: Embedded Development Kit EDK 6.2i”