2018 ASEE ANNUAL CONFERENCE & EXPOSITION
JUNE 24–27, 2018
SALT PALACE CONVENTION CENTER
SALT LAKE CITY, UT

Paper ID #21983

Bridging the Gap between Computer Engineers and Software Developers by Incorporating the PYNQ Platform into a Graduate Course on Embedded System Design Using FPGA

Dr. Chirag Parikh, Grand Valley State University

Chirag Parikh is an Associate Professor of Electrical and Computer Engineering at Grand Valley State University, Grand Rapids, Michigan. He received his B.S. degree from University of Mumbai, India in 2000. He received both his M.S. and Ph.D. degrees in Electrical and Computer Engineering from The University of Texas at San Antonio, Texas in 2003 and 2007 respectively. His research interest is in area of digital systems, hardware modeling, and cryptography and also published various papers in the same. He has taught C programming, Microcontroller Applications and Introductory/Advanced Digital courses at the undergraduate level. He has also taught a graduate level course in the area of Advanced FPGA Implementation.

Ryan T. Aldridge, Grand Valley State University

Bridging the gap between Computer Engineers and Software Developers by incorporating the PYNQ platform into a Graduate Course on Embedded System Design Using FPGA (Work-In-Progress)

Abstract

Today, Field Programmable Gate Arrays (FPGA) play a very important role in several embedded applications used in the area of defense systems, automotive, bioinformatics, cryptography, consumer electronics and many more. Despite their potential as high-performance computing platforms, they are rarely used within datacenters and other general-purpose computing infrastructure. Due to complicated process flow for developing and implementing applications on an FPGA, application developers without any hardware design background find it difficult to adapt and develop FPGA applications. Xilinx recently introduced the PYNQ platform that enables engineers and programmers to develop embedded systems at a higher abstraction level without having the necessary hardware design background.

At Grand Valley State University, we have partnered with Xilinx, a leading manufacturer of FPGAs and a leading provider of programmable platforms to develop a graduate level course for Computer Engineering curriculum to bridge the gap between computer engineers and software developers. This course would allow students from engineering and computer science majors to be able to develop and implement applications on FPGAs using Python programming language and overlays that are similar to software libraries. In this paper, we describe the structure of the course along with the associated topics and laboratory exercises.

I. Introduction

Today, Field Programmable Logic Devices (FPLDs) are considered as an alternative to Application Specific Integrated Circuits (ASICs) in designing complex digital systems. Because of the need to shorten the design cycle time and to be able to make changes to the design rapidly, these programmable devices have become very popular. Field Programmable Gate Arrays (FPGAs), a type of FPLD, plays a very important role in several embedded applications used in the area of defense systems, automotive, bioinformatics, cryptography, consumer electronics and many more. The immensely parallel architecture of FPGAs means that running computations on an FPGA will give better performance, lower power, lower latency, and higher throughput than software. However, despite their potential as high-performance computing platform, they are rarely used within datacenters and other general-purpose computing infrastructure.

Typically, the FPGA programming model has mostly been hardware-centric [1]. As FPGAs become a standard component of the computing environment, with users expecting the hardware to be software-defined, they must be accessible not just by hardware developers, but by software developers too. FPGAs have been around for many years to solve hardware design problems. Their programmability was done exclusively in terms familiar to hardware designers instead of via any programming language designed for software development. New FPGA designs aimed at supporting software development instead of just hardware replacement designs, coupled with new software development tools, make FPGA programming worth a serious look by software developers. However, due to complicated process flow for developing and implementing applications on an FPGA, application developers without any hardware design background find it difficult to adapt and develop FPGA applications.

The technology and the Electronic Design Automation (EDA) tools [9] employed by the industry in the design of digital system hardware have evolved significantly in the past several years. Xilinx recently introduced the PYNQ [2] platform that enables engineers and programmers to develop embedded systems at a higher abstraction level without having the necessary hardware design background. PYNQ provides a Python interface to allow overlays (hardware libraries) in the *Programmable Logic* (PL) to be controlled from Python running in the *Processing System* (PS). FPGA design is a specialized task which requires hardware engineering knowledge and expertise. PYNQ overlays are created by hardware designers, and wrapped with this PYNQ Python API. Software developers can then use the Python interface to program and control specialized hardware overlays without needing to design an overlay themselves. This is analogous to software libraries created by expert developers which are then used by many other software developers working at the application level. Since the scope of potential applications is growing rapidly, it is imperative to educate hardware and software engineers in these new techniques.

At Grand Valley State University, we have collaborated with Xilinx [3], a leading manufacturer of FPGAs and a leading provider of programmable platforms to develop a graduate level course for Computer Engineering curriculum to bridge the gap between computer engineers and software developers. This course would allow students from engineering and computer science majors to be able to develop and implement applications on FPGAs using Python programming language and overlays that are similar to software libraries. This paper describes our experience in teaching the students to develop applications on the new PYNQ platform. The paper is organized as follows: the next section describes the main features of the PYNQ Platform course and introduces our integrated lecture / learning activity / laboratory approach. Then we talk about the teaching tools in the form of hardware and software that we use in this course. Next, we describe the development of our lecture material and laboratory exercise and how they build towards the final project that we assign to students. We then present a questionnaire that we would be asking students to fill out followed by our concluding remarks.

II. Course Organization

EGR680: PYNQ Platform is the graduate level course to be offered to Computer Engineering as well as Computer Science students in the School of Engineering at Grand Valley State

University who are interested in developing applications on FPGA. The semester long course will include approximately three hours of lecture per week integrated with laboratory/learning activity session. Topics covered in the course are divided into 4 areas: hardware design using Verilog [4], embedded system design concepts using ZYNQ platform, software programming using PYTHON and integration on PYNQ platform.

Hardware design using Verilog will briefly cover Verilog language concepts in a period of 3 weeks. Embedded System Design concepts [8] include topics like elements of embedded system design, programmable logic device architecture, tools for design and prototyping, design of advanced FPGA-based systems with hard ARM processors and interface to hard-core processors[5], creation and incorporation of IP cores and real-time debugging on FPGA hardware. These topics will also be covered in a period of 3 weeks. Software programming using PYTHON will cover the basics of PYTHON programming along with GUI application development in a period of 3 weeks. Integration on the PYNQ platform includes combining the knowledge students gained in the past 9 weeks and develop an application on the PYNQ hardware that will utilize a GUI-based application developed in Python to interact with the PYNQ hardware.

In our ECE program [5], the much needed knowledge of working with digital system fundamentals is covered in two lower-level courses; *EGR224: Introduction to Digital Systems* (taken by sophomores) and *EGR426: Integrated Circuit System Design* (taken by seniors). Whereas in our CIS program, this much needed knowledge is covered in their lower-level course; *CIS351: Computer Organization and Assembly Language* (taken by Juniors). These set of courses provide the necessary knowledge for our undergraduate students who plan to take the course discussed in this paper as an elective.

The main goals of the course are (1) to teach students the fundamental concepts in the 4 areas mentioned before and (2) to clearly illustrate the way in which advanced FPGA-based systems are designed on PYNQ platform, using computer aided design (CAD) tools. During the lecture session of the course, the first 90 minutes are used to present the theory materials in the form of power-point slides and journal articles to not only reflect the current trends in FPGA-based embedded system design but also enforce the basic concepts needed by the engineering and the computer science students. During the remaining 90 minutes of the lecture session, students are required to complete learning activities and laboratory exercises. These activities and exercises are “hands-on” meaning that the students are given the instructions in form of tutorial guides by the instructor. These tutorials provide guidelines for the laboratory and the necessary tools for its execution. Based on this, the students had to design a solution to the given task and implement it using the design methodology presented earlier in the class.

III. Teaching Tools

One of the objectives of this course is to provide students with an experience of software down development flow [6] rather than hardware-up. To fulfill this objective, students had to be exposed to FPGA-specific hardware and software tools.

A. Hardware

The PYNQ-Z1 development board shown in Figure 1 incorporates several features that we will utilize in this course for our purposes.

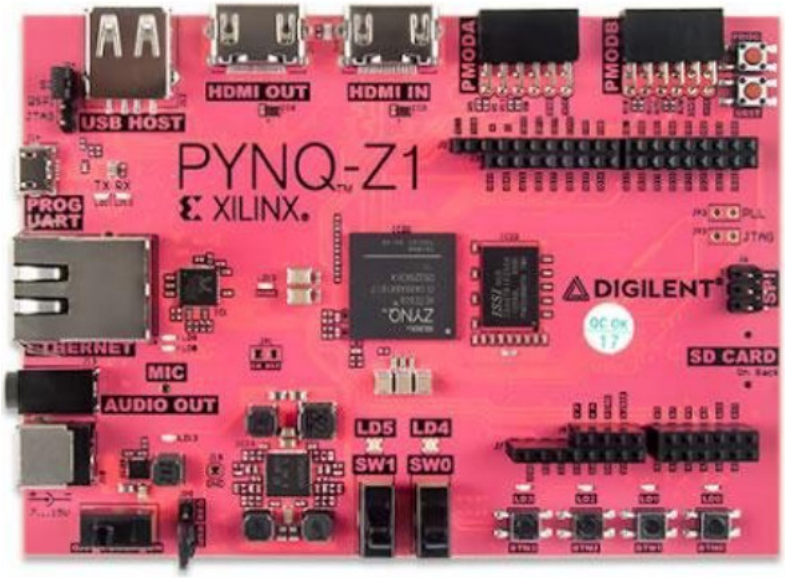

Figure 1: PYNQ-Z1 Development Board

The features include:

- ZYNQ XC7Z020-1CLG400C FPGA that houses Cortex-A9 processor with 13,300 Logic slices, 630KB of fast block RAM, and on-chip analog-to-digital converter (XADC)
- Low bandwidth (SPI, UART, CAN I2C) and High bandwidth (Ethernet, USB) peripheral controllers
- 512MB DDR3, 16MB Quad-SPI Flash and MicroSD slot to load PYNQ-Z1 boot image to use the PYNQ framework
- 4 LEDs, 4 push buttons, 2 slide switches and 2 RGB LEDs for I/O purposes
- Expansion connectors
    - Two standard PMOD ports (16 Total FPGA I/O)
    - Arduino/chipKIT shield connector (49 Total FPGA I/O)
- Bundled with Xilinx Vivado software tool used for synthesis and analysis of HDL designs
- Jupyter Notebook framework is deployed in an embedded context

We require students to purchase the PYNQ-Z1 board at an academic price for $65 from Digilent [7]. The PYNQ environment comes with a standard overlay that supports HDMI and Audio inputs and outputs, as well as two 12-pin PMOD connectors and an Arduino-compatible connector that can interact with Arduino shields.

B. Software

PYNQ-Z1 development board is compatible with the web edition of Xilinx tools, which are also freely-downloadable from the Xilinx website. This software tool will allow students to synthesize (compile) their designs, perform timing analysis, simulate their design for different stimuli and configure the target device with the programmer to develop fairly sophisticated digital systems of high complexity. However, our laboratory is also equipped with complete version of Xilinx tools that includes variety of sophisticated tools like ISE Foundation, Chipscope-Pro, Vivado and Software Development Kit (SDK).

The default overlay of the PYNQ framework instantiates several MicroBlaze processor cores to drive the various I/O interfaces. PYNQ also offers an API and extends common Python libraries and packages to include support for Bitstream programming, directly access the programmable fabric through Memory-Mapped I/O (MMIO) and Direct Memory Access (DMA) transactions without requiring the creation of device drivers and kernel modules. Our work builds upon these APIs and Overlay concepts to develop applications.

IV. Lecture Materials, Sample Laboratory Assignments and Projects

The lecture session will involve four independent tracks namely; hardware design using Verilog, embedded system design concepts using ZYNQ platform, software programming using PYTHON and integration on PYNQ platform. The necessary information will be provided in the form of technical papers, slide presentations, and relevant handouts. The lecture will consist of the following parts:

Part A: Hardware Design Using Verilog

- ***Introduction to FPGA architecture:*** Students will learn about various programmable logic devices and their architectures. Emphasis will be placed on exploring the PYNQ hardware.

- ***Introduction to Verilog:*** Verilog is the HDL used in this course. Students will learn to write behavioral descriptions from the software perspective of their design using Verilog and perform simulations.

Part B: Embedded System Design Concepts Using ZYNQ platform

- ***Xilinx Tools:*** Students will learn to do logic design on FPGA using Xilinx Vivado, embedded system design, configuring and adding custom IP using Xilinx SDK, real-time debugging using Chipscope and software profiling to evaluate software performance.

Part C: Software Programming Using PYTHON

- ***Introduction to PYTHON:*** Students will learn about the basics of the PYTHON language including topics such as I/O, conditional statements, loops, files, functions, classes and GUI. Students will use Jupyter Notebook (an open source web application) to develop and test their PYTHON code. More emphasis would be placed on GUI-based applications since students for their final project would need that knowledge.

Part D: Integration on PYNQ Platform

- ***Integration:*** Students will learn how to integrate default overlay of PYNQ framework with the GUI-based PYTHON code for rapid application development on FPGA.

In addition to the lecture session, students will also be required to complete several laboratory exercises assigned every week. The complexity of the laboratory exercises will gradually increase every week. The initial laboratory exercises will require students to familiarize themselves with the working knowledge of Verilog, a hardware description language used for design entry. The next set of laboratory exercises will require a higher level of understanding and abstraction due to the nature of the tools and the design process of the embedded systems. Then we would move on to PYTHON-based laboratory exercises.

The objectives of these laboratories and projects are:

- To help students understand and assimilate lecture material
- To give students practical experience with the process of design and implementation of FPGA-based embedded systems using PYTHON
- To give hands-on experience with CAD tools used in the design process

The course laboratories are structured as 9 laboratory assignments and one project. These laboratories expose the students to a range of design activities, tool functionality and implementation technology. Each laboratory has a set of deliverables; typically including the circuit schematics or top-level block diagram, simulation waveform plots, demonstration of hardware implementation and a brief report.

The following is an outline of the 9 laboratory assignments and a project that we offer in the course:

- **Laboratory #1**: The purpose of the laboratory is to introduce the students to the design tools to be used in the course. Students will be introduced to Xilinx Vivado to develop, simulate, synthesize and implement their design of a N-bit counter on the PYNQ board.

- **Laboratory #2**: The purpose of the laboratory is to familiarize students with the behavioral modeling style using Verilog. Students will be provided with two seven-segment PMODs (see Figure 2) that they have to interface with the PYNQ board. Students will then be required to design a time-multiplexed seven-segment decoder that displays different 4-character messages based on the button presses.

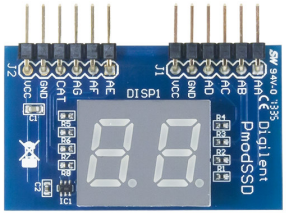

Figure 2: Digilent Pmod seven-segment display

- **Laboratory #3**: The purpose of the laboratory is to familiarize students with the state machine design to implement sequential designs. Students will be required to design a vending machine that accepts nickels and dimes, and dispenses gum, apple and yogurt. The top-level diagram of the vending machine is shown in Figure 3.

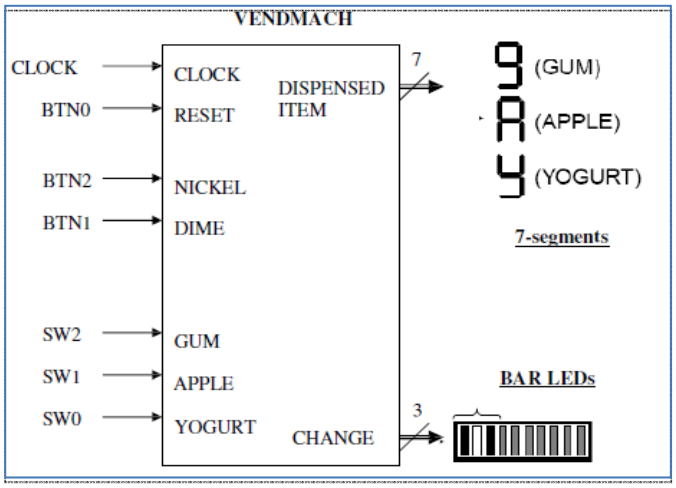

Figure 3: Block diagram of a vending machine

- **Laboratory #4**: The purpose of the laboratory is to introduce the students to create a simple ARM Cortex-A9 processor system targeting PYNQ board using the Vivado and SDK tools. Students will then required to design a complete hardware and software processor system by adding additional peripheral cores to the existing design.

- **Laboratory #5**: The purpose of the laboratory is to familiarize the students with the process of creating custom IP and then adding them to the existing design. Students will use the design developed in Lab #4, create a custom IP of seven-segment display and then add that to the existing design along with push-buttons, switches and UART to develop an application

that shows the configuration of the switches on the terminal window and seven-segment PMOD when a button is pressed.

- **Laboratory #6**: The purpose of the laboratory is to introduce the students to the PYTHON language and Jupyter Notebook for developing PYTHON applications. Students will then be asked to design an ATM machine and demonstrate several ATM transactions.
- **Laboratory #7**: The purpose of the laboratory is to introduce the students to some of the advanced topics of PYTHON programming such as file pointers and functions. Students will still be asked to design an ATM machine using functions and print a transaction summary to a file.

- **Laboratory #8**: The purpose of the laboratory is to introduce students to GUI-based PYTHON programming. Students will still be asked to design a GUI-based ATM machine and demonstrate several ATM transactions. A sample view of GUI ATM interface is shown in Figure 4 below.

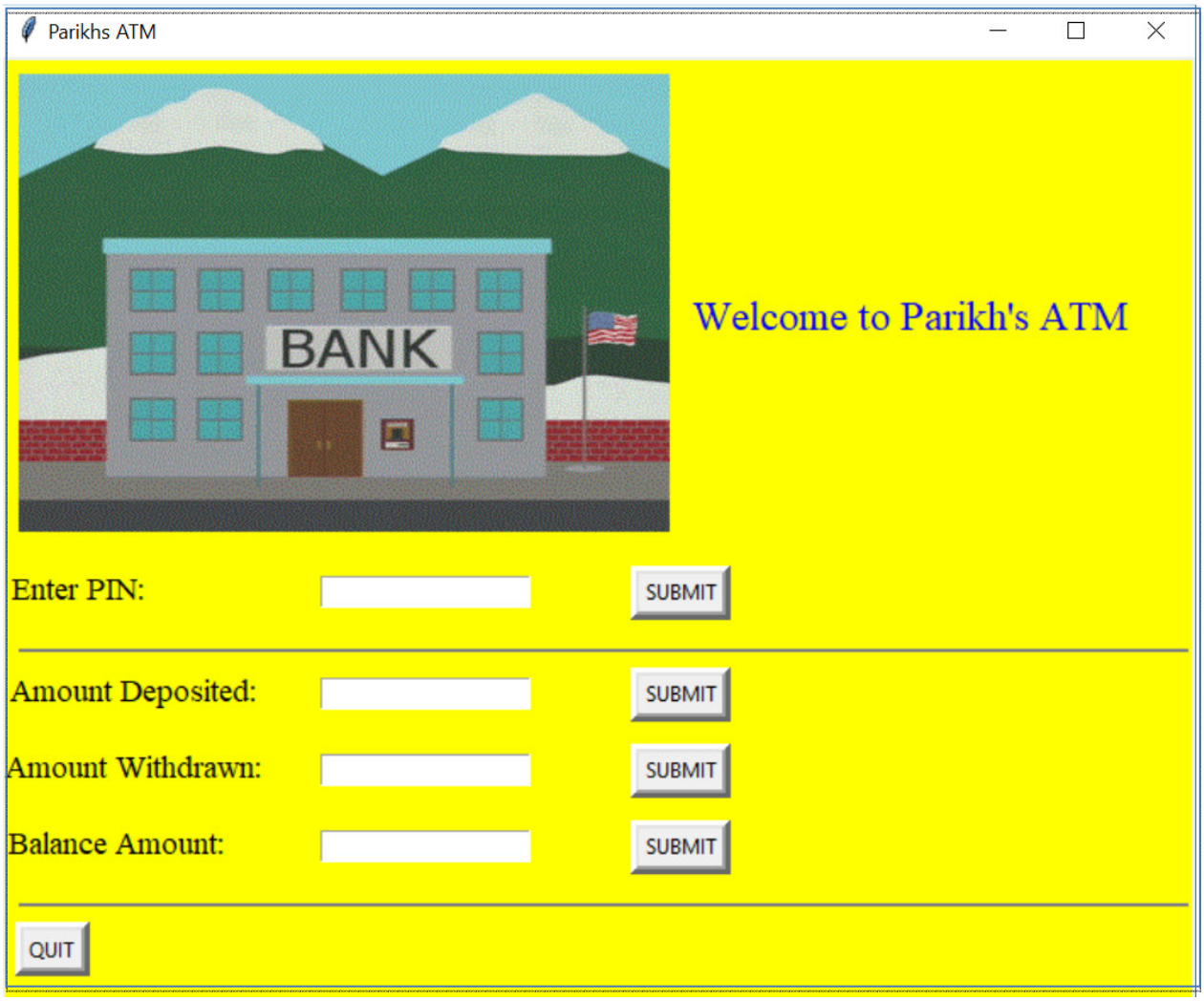

Figure 4: ATM GUI interface

- **Laboratory #9**: The purpose of the laboratory is to have students write a GUI-based PYTHON application in Jupyter Notebook, which is the only kernel (programming language) installed for Jupyter notebook in PYNQ distribution. We will incorporate the open-source Jupyter notebook infrastructure to run an Interactive Python (IPython) kernel and a web server directly on the ARM Cortex-A9 of the Zynq device.
- **Project**: The project is intended to allow the student to apply the knowledge that they have learned in subsequent laboratory exercises to develop a fully functional system that is written in PYTHON and uses existing hardware overlays. Students will be provided with the Grove adapter for PYNQ board to which they would connect the Grove LED bar add-on shield. Students will then be asked to develop a graphical user interface in PYTHON that would allow them to turn ON different LEDs on the LED bar by moving the slider from left to right on their interface. In addition they would also design an interface that enables the user to type in the desired beats per minute of an RGB LED, and select from several color options.

V. Assessment

The course "*PYNQ Platform*" will be offered for the first time during Fall 2018. It will be open to students from Computer Engineering, Computer Science and Electrical Engineering emphasis. In order to obtain finer-grained feedback from the students, at the end of the semester we will encourage the students to complete a customized anonymous questionnaire to learn the benefits of offering such cross-platform course to students from two disciplines mainly; engineering and computer science.

A. Student Learning Objectives

Listed below are the outcome-related learning objectives for this course that we plan to assess when the course is offered.

At the end of this course, students will be able to:

1. *Apply advanced concepts in analyzing, designing and building digital systems (Outcomes: 1, 6)*
2. *Employ modern day tools in designing, testing and debugging complex digital systems (Outcomes: 1,7)*
3. *Rapidly prototype applications on programmable devices using high-level language (Outcomes: 1,6,7)*

B. Questions

The students will be asked six questions that will be rated on a five point scale, where '1' indicates "not at all" and '5' indicates "very much". Along with this, students will be given the opportunity to include written comments for additional three questions. This will help us to get valuable feedback for improvement of the course. The questions will be:

1. *The focus of this course was abstraction through a higher level language in order to utilize FPGA resources. Given that, do you feel that you gained enough knowledge on the hardware (Verilog) aspects of the course to be able to understand how the Python code communicates with the FPGA?*

   1 *2* *3* *4* *5*

2. *Do you feel that you learned enough of the Python language to be able to design your own projects?*

   1 2 3 4 5

3. *Do you feel that having to write code in 3 different languages (Verilog, C, Python) for a single course is too much to handle?*

   1 2 3 4 5

4. *How challenging was the course?*

   1 2 3 4 5

5. *From a computer science perspective, do you feel that this course would offer any benefit to computer science majors?*

   1 2 3 4 5

6. *From a computer engineering perspective, do you feel that this course would offer any benefit to computer engineering majors?*

   1 2 3 4 5

   Additional Questions

7. *Do you feel that there is a benefit of being able to program an FPGA using a high level language? If so, what are those benefits? If not, why?*

8. *Do you feel that the labs and projects were appropriate to facilitate the learning process? If not, why?*

9. *If you could change one thing from this course, what would it be?*

VI. Conclusion

With the new trend of software community shifting focus towards application development on FPGAs, it becomes imperative that the tools and languages needed for application development on FPGA need to go beyond conventional embedded system languages. Towards this trend, Xilinx recently released PYNQ as a productivity environment and platform for developers, combining the use of Python, its tools and libraries with the capabilities of programmable logic and ARM processors. This paper describes a graduate course on PYNQ Platform that allows Computer Engineering and Computer Science students to concurrently take this course to design applications on FPGAs using a software-down approach.

This paper highlights the significance of PYTHON programming language to reduce application development time on FPGAs by exploiting a tremendously rich and diverse set of packages, libraries and tools. This is accomplished by providing lecture materials, learning activities,

laboratory exercises and project that go hand-in-hand in order to educate the students with the working knowledge of PYNQ platform. In this course, the information was provided to the students in the form of slide presentations, tutorials and journal articles, which proved valuable in assisting and enabling the students to learn about the design techniques and the tools that are constantly changing. We also plan to obtain feedback from the students in terms of how valuable this course offering was to them and discuss any potential improvements that could be made to this course.

References

[1]. http://blog.liveedu.tv/enabling-fpgas-software-developers/

[2]. www.pynq.io

[3]. http://www.xilinx.com/univ

[4]. G. Schelle, D. Fay, D. Grunwald, D. Connors, J. Bennett. "*An Evolving Curriculum to Match the Evolution of Reconfigurable Computing Platforms*", 1st International Workshop on Reconfigurable Computing Education, 2006.

[5]. www.gvsu.edu/engineering

[6]. A. Schmidt, G. Weisz and M. French, "*Evaluating Rapid Application Development with Python for Heterogeneous Processor-based FPGAs*", 2017 IEEE 25th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM'17)

[7]. www.digilentinc.com

[8]. V. Sklyarov, I. Skliarova, "*Reconfigurable Systems in Education*", Field Programmable Logic and Application, 13th International Conference, FPL 2003, Lisbon, Portugal, September 1-3, 2003, Proceedings.

[9]. I. Vélez and J. Sevillano, "*A course to train digital hardware designers for industry*." IEEE Transactions on Education, 50(3): 236–243, August 2007.