Embedded System Learning Platform for Developing Economies

Tinashe Chamunorwa¹(✉), D. Ursutiu¹,², C. Samoila¹,³, and Horia Alexandru Modran¹

¹ Transilvania University of Brasov, Brașov, Romania
{chamunorwa.tinashe,udoru,csam,horia.modran}@unitbv.ro
² Romanian Academy of Scientists, Bucharest, Romania
³ Romanian Academy of Technical Sciences, Brașov, Romania

Abstract. The continual research of electronic embedded system platforms for teaching and learning is of paramount importance. This is to increase the impact on a universities’ ability to lead in technological advancement with goals to enhance innovation and accelerate the concept-to-deployment process. Technological progress in the fields of electronics, wireless communication, cognitive computing, and robotics has caused almost everything which connects to electricity to have a small processor and sensor embedded with itself [1]. Cognitive or Intelligent embedded systems are the ‘core’ of trends such as: reduced energy consumption, deep learning applications, improved security for embedded devices, cloud connectivity and mesh networking, and visualization tools with real time data.

This paper is aimed at stimulating design and innovation in electronics education through the rapid prototyping of configurable embedded systems. It also covers remote access functionality is also shown using cloud services. Divided into two parts, this paper gives design examples for both an elevator controller and a Data Acquisition (DAQ) system design. Focus was on the Programmable System on Chip (PSoC) 6 based kit, PSoC Creator software, and Laboratory Virtual Instrumentation Engineering Workbench (LabVIEW) as rapid prototyping and learning platforms for digital and analog system designs. The Universal Digital Block (UDB) editor in PSoC Creator software was used to configure PSoC chip digital blocks and design a Finite State Machine (FSM) based elevator controller, which acquired digital signals and gave corresponding output. LabVIEW was used as a signal analysis tool and was also used to send results for online access and display. The PSoC and LabVIEW ecosystems are utilized here to bring an innovative paradigm into the embedded system design. These two platforms, especially combined with the virtual instrumentation concept, offer configurability and monetary value. This paper shows that the use of these ecosystems for the purpose of electronics education can accelerate learning and bridge the gaps within the online environments that students and universities find increasingly necessary.

Keywords: AFE · FPGA · FSM · LabVIEW · PSoC · UDB · Verilog

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2021
M. E. Auer and T. Rüttmann (Eds.): ICL 2020, AISC 1329, pp. 617–627, 2021.
https://doi.org/10.1007/978-3-030-68201-9_60
1 Introduction

Networked embedded systems are an essential part of the electronics field and can be divided into analog and digital. This paper explains the implementation of a configurable mixed-signal system to stimulate project-based learning and remote laboratories.

The PSoC 6 platform is a mixed-signal configurable embedded platform; hence, it fits well for this purpose as it includes a variety of components like AD/DA converters, DMA controllers, logic elements, flip-flops, and lookup tables, which all diversify learning. The structure of this paper introduces the digital design and then goes on to discuss the data acquisition system. The growing importance of digital electronics systems design makes this an essential part of electronics. Digital systems have become less hardware-based and now are comprised mostly of microprocessors, firmware, and software [2].

This paper seeks to close the gap between discrete logic gates implementation and the advanced Field Programmable Gate Arrays (FPGA) implementations. The need to pursue this subject was confirmed in a survey conducted with Electronic Engineering graduates.

This shortcoming has resulted in numerous digital electronics-based laboratory designs ending at gate level in the lab. The PSoC comes as an intermediate method between these two extremes. The design method discussed in this paper works as a jumpstart to the new student as it does not require deep knowledge about digital systems. It encourages the implement-as-you-learn concept. An FSM-based elevator controller design was implemented, with results displayed on PSoC 6 kit. A data acquisition system was also implemented, with the results displayed both on LabVIEW and online.

This project augurs well with the United Nations’ 2030 Agenda for Sustainable Development. It also addresses primary innovation challenges for developing countries in an effort to develop capacity to learn, adopt and diffuse current and new knowledge and technologies to promote sustainable and inclusive development [3]. The global market for the embedded systems industry was valued at $68.9 billion in 2017 and is expected to rise to $105.7 billion by the end of 2025 [4]. Knowing this, it is imperative that tertiary institutions tap into this market.

There exists a considerable body of work on the use of embedded platforms for online laboratories. One such work proposes an intelligent online interface supported by a dedicated hardware architecture that maps the circuits while they are being designed by the students. The hardware is based on low cost microcontrollers and port expanders. The system is capable of matching the current circuit with the target circuit in a web-based environment as well as finding the possible faults in the current circuit. The system has been shown to use a prototype board for digital electronics which can be scaled to implement bigger interfaces and more complex functions [5].
2 Digital System Design Experimental Steps

2.1 Programmable System on Chip (PSoC)

The PSoC 6 Bluetooth Low Energy (BLE) Prototyping Kit was used. This is a low-cost hardware platform that enables the design and debug of PSoC 6 MCUs. The kit features the EZ-BLE Creator module which is scalable and reconfigurable platform architecture. It combines reconfigurable and programmable digital and analog blocks with flexible automatic routing. The CYBLE-416045-02 also includes digital programmable logic, high performance Analog-to-Digital Conversion (ADC), low-power comparators, and standard communication and timing peripherals. Of particular interest to this project is the use of Universal Digital Blocks (UDB) for the rapid prototyping of the digital elevator controller.

2.2 Setting up a Universal Digital Block (UDB) Library and UDB Components

A UDB is a flexible, programmable digital block, shown in Fig. 1 inside a PSoC device, that is designed to realize synchronous state machines. The UDB Editor is a tool which allows the creation of UDB-based designs with just a very basic knowledge of digital logic or Verilog code. Using this graphical tool, one can drag, drop, and then configure hardware without having to write Verilog code [6]. The writing of Verilog code for digital designs is usually a turn off for many beginners. The UDB Editor takes care of many internal configuration details simply by specifying the parameters of the UDB blocks on the design canvas. The tool translates a design to Verilog in real time and shows how the UDB blocks translate to Verilog hardware description language (HDL).

![Fig. 1. Universal Digital Block](image-url)
A library project was created in PSoC Creator software and was named a UDB library. The UDB document was then used to create a new UDB component, in this case an FSM. The component was targeted for PSoC 6 kit.

2.3 Finite State Machine Design

In the UDB library environment, state machine components were connected in three states and six transitions. In this case, the Moore type state machine design was used and gives outputs depending on the current state. The resultant state machine and state table are shown in Fig. 2 and Table 1, respectively.

![Fig. 2. The State Machine depicting the Moore Elevator](image_url)

| Input          | Next State | Output                      |
|----------------|------------|-----------------------------|
| Floor_1 UP_button | Floor_2    | Floor1_light = 1
|                 |            | Floor2_light = 0
|                 |            | Floor3_light = 0 |
| Floor_1 DOWN_button | Floor_1    | Floor1_light = 1
|                 |            | Floor2_light = 0
|                 |            | Floor3_light = 0 |
| Floor_2 UP_button | Floor_3    | Floor1_light = 0
|                 |            | Floor2_light = 1
|                 |            | Floor3_light = 0 |
| Floor_2 DOWN_button | Floor_1    | Floor1_light = 0
|                 |            | Floor2_light = 1
|                 |            | Floor3_light = 0 |
| Floor_3 UP_button | Floor_3    | Floor1_light = 0
|                 |            | Floor2_light = 0
|                 |            | Floor3_light = 1 |
| Floor_3 DOWN_button | Floor_2    | Floor1_light = 0
|                 |            | Floor2_light = 0
|                 |            | Floor3_light = 1 |
Figure 2 and Table 1 show that the initial state is Floor_1, which is also the reset state. When an UP_button is pressed, the elevator moves from Floor_1 to Floor_2 and the arrival at Floor_2 is indicated by the corresponding LED. The routine is also followed for the transition from Floor_2 to Floor_3. Pressing the DOWN_button follows the same routine in the opposite direction.

2.4 UDB Component Generation

In the Universal Digital Block editor, the FSM was then generated into a symbol and added to the UDB library. The generated FSM symbol was then added to a new project in PSoC Creator software and the inputs and outputs were added to it, as shown in Fig. 3.

In Fig. 3, the UP_Button and DOWN_Button digital inputs were given pin assignments in the PSoC Creator software which corresponded with the switches in the PSoC 6 kit. The elevator FSM block processes and acts on these signals according to the embedded FSM. The outputs gave the expected outcome as switches were being pressed. The elevator_clock was set at 10 Hz for prototyping purposes and so that the transitions and outputs would be observable. The main idea behind this is to utilize the PSoC Creator software and PSoC kits as tools for rapid digital system design and as a learning platform. The elevator serves as a good example because it has common functionality and it represents a full system level design. The FSM based elevator controller works as a digital input data acquisition system.

The FSM technique has been chosen because it serves as a foundation for complete digital system controller designs and implementations. A useful formalism for designing more complex digital circuits is that of the FSM, whereby the circuit's function is broken down into a collection of states and rules which determine when the system moves from one state to another state [7]. The FSM based design, as stated, is the entry point for a student to have a feel for real-life designs. At this stage, students can design controllers for systems like a vending machine, a subway entrance turnstile, a heating system, an automated subway system, a self-driving car system, and an elevator. While the FSM can implement any digital design algorithm, it becomes overcomplicated when dealing with data. It is therefore necessary to include the
controller and data path in the digital system design because they are more efficient to process and memorize data under the machine’s control. They optimize the sharing of duties between the FSM and an external architecture. For the sake of this paper, this technique has been avoided to circumvent a steep learning curve for the student at the foundational level.

2.5 Verilog Code Generation

After the configuration of the FSM in the UDB editor and the generation of a UDB library component, a Verilog code was also generated. This Verilog generation is one of the main advantages for students to use PSoC Creator software. The student can learn how Verilog represents their design and can then edit the code if they want to add some functionality that may not be addressed by the UDB editor. The initial goal of having a functional full digital system will have been achieved. Figure 4 displays a snippet of the generated elevator controller Verilog code which shows the declaration of floors as states and their associated transitions. This code can then be ported into any Field Programmable Gate Arrays (FPGA), if the student so wishes.

```verilog
module Elevator_FSM (  
  input wire clock, 
  input wire UP_button, 
  input wire DOWN_button, 
  output wire Floor1_door, 
  output wire Floor2_door, 
  output wire Floor3_door;
/* Assignment of Combinatorial Variables =  
assign Floor1_door = (Floor1_light);  
assign Floor2_door = (Floor2_light);  
assign Floor3_door = (Floor3_light);
/*State Machine: StateMachine_1  
always @(posedge clock)  
begi
  case(StateMachine_1)  
    Floor_1 :  
        begin  
          Floor1_light <= (1);  
          Floor2_light <= (0);  
          Floor3_light <= (0);  
          if ((UP_button ) == 1'b1)  
            begin  
              StateMachine_1 <= Floor_2 ;  
            end  
          else if ((DOWN_button ) == 1'b1)  
            begin  
              StateMachine_1 <= Floor_3 ;  
          
Fig. 4. Verilog partial code depiction

Students can import the Verilog code, generated in the PSoC Creator environment, into LabVIEW FPGA Virtual Instruments (VIs) using the Hardware Description Language (HDL) Interface Node in LabVIEW software. One can test the functionality of the HDL code using an HDL simulator. A shell of the testbench should be generated
which includes clock and enable inputs. One must add code to the testbench shell to test the functionality [8].

3 Data Acquisition Experimental Steps

3.1 ADC and UART Configurations

In this experiment an external hand-touch signal (Vin) was inputted into the processor through an ordinary connector wire. The input pin was configured as a high-impedance analog drive mode. The signal was fed into an Analog-to-Digital Converter (ADC) which then passed on the digital signal to a Universal Asynchronous Receiver-Transmitter (UART) for onward transmission to the LabVIEW environment and displayed on the virtual graphs. The signal was then displayed in real-time on an online platform. The PSoC Creator software was used to configure an ADC component. During this configuration, parameters such as channel type, input mode and range, interrupt limits, data format, and timing were adjusted. In the UART, data format, transmission speed, bit order, data width, parity, and baud rate were also adjusted. The ADC and UART schematic connection is shown in Fig. 5.

![Fig. 5. Universal digital block](image)

The drag-and-drop function is an advantage offered by the PSoC Creator software environment. To configure UART components in some FPGA environments, a long code has to be written. In PSoC Creator software, this is done by just double-clicking the component after dragging it from the library. Thus, PSoC is well suited for concept-to-deployment acceleration.

For the design, which uses multiple sensors, the PSoC Analog Coprocessor kit can be easily utilized. This simplifies the design of sensor-based systems by delivering a reconfigurable and scalable architecture that integrates programmable Analog Front Ends (AFE). The PSoC Analog Coprocessor kit consists of a signal processing engine (32-bit Arm® Cortex®-M0+) that can calibrate and tune the AFE [9]. The AFE interfaces to humidity, temperature, contact, motion, and ambient light detection sensors.
3.2 Interfacing PSoC 6 with LabVIEW

The PSoC 6 was interfaced with LabVIEW using the National Instruments (NI) implementation of the Virtual Instrument Software Architecture (VISA) Application Program Interface (API). The NI-VISA API provides a programming interface to control Ethernet/LXI (LAN-based eXtensions for Instrumentation), GPIB (General Purpose Interface Bus), USB (Universal Serial Bus), PXI (Peripheral Component Interconnect), VXI (Versa Module Eurocard eXtensions for Instrumentation), and serial instruments in NI application development environments like LabVIEW. NI-VISA is recommended because it is interface independent, has platform portability and is easy to use. An NI-VISA communication to the PSoC 6 was performed by opening a VISA session.

![Block diagram of LabVIEW acquisition, process, and display](image)

**Fig. 6.** The LabVIEW block diagram used for acquisition, process and display

Figure 6 shows how LabVIEW was used to process and display signals which have been acquired from an external finger-touch signal detection. The use of software-based processing tools allows for the designer to experiment with different functionalities until the desired result is achieved. LabVIEW offers numerous examples which students may customize according to their requirements. This, in turn, accelerates the design process (Fig. 7).

![Graphical signal display](image)

**Fig. 7.** LabVIEW graphical signal display
3.3 Online Access

Once the results are analyzed and displayed in LabVIEW they can be accessed online through the internet. One of the platforms for online access is ThingSpeak™, an Internet of Things (IoT) analytics platform service that allows one to aggregate, visualize, and analyze live data streams in the cloud [10]. ThingSpeak provides instant visualizations of data posted by devices to the ThingSpeak webpage. It is often used for prototyping and proof of concept for IoT systems that require analytics. Figure 8 shows a LabVIEW block function which was used to publish data on a ThingSpeak webpage and could be accessed publicly. The Uniform Resource Locator (URL) is split and concatenated. An Application Programming Interface key (API key) is then included in the URL. Figure 9 is the sample result displayed on the ThingSpeak webpage. Data can also be sent in the reverse from the cloud to the computer.

![LabVIEW block diagram for website publishing](image1)

**Fig. 8.** LabVIEW block diagram for website publishing

![Waveform displayed on the ThingSpeak webpage](image2)

**Fig. 9.** Waveform displayed on the ThingSpeak webpage
4 Conclusion

This paper has shown that PSoC and LabVIEW ecosystems can be used for rapid prototyping of embedded system designs. PSoC kits can also be used to bridge the gap between discrete logic gates laboratory and the advanced FPGA implementations. Although the UDB editor lacks versatility compared to coding with Verilog or using FSMs with datapaths, it provides a soft landing and flattens the learning curve for students. Students may access these advanced learning tools upon grasping the fundamental concepts. The concept of Virtual Instrumentation is critical to teaching and learning of electronic embedded systems. The combination of PSoC and LabVIEW ecosystems provides an unmatched synergy for embedded system design. A design can be initiated on PSoC Creator and the signals acquired through LabVIEW and eventually displayed online. The use of PSoC environments for design, in general, needs to be tied to a cloud system so that students can access and perform experiments online. Furthermore, a cloud management system should be implemented. The recent outbreak of COVID-19 has made online laboratories an unavoidable venture for learning institutions. PSoC and LabVIEW ecosystems are an excellent platform for learning fundamental and advanced digital, analog, and virtual instrumentation concepts.

Acknowledgements. We would like to express our great appreciation to NI Corporation for availing a free LabVIEW license. Our gratitude also goes to Cypress Semiconductor Corporation for providing us with a free PSoC 6 BLE Prototyping Board (CY8CPROTO-063-BLE), thus facilitating this study and making it possible. Their generosity and collaboration were greatly appreciated. Special mentioned to Jessica Lewis-Realtor for providing free proofreading and editing services.

References

1. Deal, L.: The Technological Advancements in Embedded Systems. https://www.redlinegroup.com/insights/the-technological-advancements-in-embedded-systems-74615914431. Accessed 05 May 2020
2. Bukowiec, A., et al.: New Trends in Digital Systems Design. VDI Verlag Publishing (2014). ISBN 978-3-18-383610-9
3. United Nations, Investment, Enterprise and Development Commission Multi-Year Expert Meeting on Investment, Innovation and Entrepreneurship for Productive Capacity-building and Sustainable Development, Sixth session Geneva, 2 and 3 July 2018 Item 3 of the provisional agenda (2018)
4. OmniSci, Embedded System. https://www.omnisci.com/technical-glossary/embedded-systems. Accessed 06 May 2020
5. Ananda, M., Andrew, D.M., Alexander, A.K.: Intelligent online interface to digital electronics laboratory with automatic circuit validation. In: Michael, F.A., Kalyan, R.B. (eds) Proceedings of the 16th International Conference on Remote Engineering and Virtual Instrumentation, vol. 80, pp. 3–18. Springer (2019)
6. Cypress Semiconductor, Universal Digital Block (UDB) Editor Guide, Document Number 001-94131, Rev. **, 9/24/2014
7. Hanson, K.: Digital Electronics Part III: Finite State Machines©2011-2015. https://www.physics.wisc.edu/undergrads/courses/current/623/lecture_notes/digital/digital-03.html. Accessed 09 May 2020
8. National Instruments, Importing HDL Code into FPGA VIs Using the HDL Interface Node. https://www.ni.com/ro-ro/support/documentation/supplemental/06/importing-hdl-code-into-fpga-vis-using-the-hdl-interface-node.html. Accessed 10 May 2020
9. Badiger, R.: Getting Started with PSoC® Analog Coprocessor, © Cypress Semiconductor Corporation, 2016–2017
10. Ravulavaru, A.: Enterprise Internet of Things Handbook. Packt Publishing, Birmingham (2018). ISBN 9781788838399