Digital IC Test Development Engineering Teaching and Learning in Higher Education

Ian Grout
Department of Electronic and Computer Engineering
University of Limerick
Limerick, Ireland
Ian.Grout@ul.ie

***Abstract*—This paper will consider the role of technology in the laboratory for a new course (module) in integrated circuit (IC) test development engineering for fifth year students in a higher education programme of study in electronic and computer engineering. Engineering laboratories form an essential part of the higher education learning experience for students in the engineering disciplines. They allow for theoretical aspects of a subject to be experimented with, and in many cases, ideas suitably visualized. This is particularly important to support student learning and understanding. In the electronic and computer engineering disciplines, laboratories can take several different forms from hands-on computer coding through to physical electronic circuit design, build and test exercises. This paper will discuss the development of module, and the creation of a laboratory arrangement using the field programmable gate array (FPGA) for a range of possible teaching and learning scenarios that will focus on required industrial IC test activities.**

Keywords—IC test, education, laboratory, FPGA

I. Introduction

Engineering laboratories form an essential part of the higher education learning experience for students in the engineering disciplines. They allow for theoretical aspects of a subject to be experimented with and in many cases, ideas suitably visualized through *hands-on* experimentation. This aspect of the education process is particularly important to support student learning and understanding. In the electronic and computer engineering disciplines, laboratories can take a number of different forms from computer software coding through to physical electronic circuit design, build and test exercises. What types of laboratories are incorporated into a programme of teaching and learning, what theoretical aspects are to be considered, and the desired learning outcomes, are in the main driven by the individual academic, provided that the teaching and learning undertaken meet the necessary programme outcome and quality requirements.

Increasingly, the range of different subjects that students may be required to learn in a particular programme of study, be it Bachelors or Masters level, is ever increasing. The need to introduce suitable technology into the classroom to support the learning experience also exists, and this means that the individual academic must select the appropriate technologies to enable the learning experience, and to be manageable within time and financial restrictions. This introduces the need for the individual academic to continually review and understand the range of possible technologies, and be suitably versed in their uses. An individually driven programme of self-learning by the academic is key to these actions.

This paper will consider the role of technology in the laboratory for a new course (module) in integrated circuit (IC) test development [1-3] engineering for fifth year students in electronic and computer engineering. This module has recently been introduced as an elective module for a new five-year Masters programme as part of the extension of an existing four-year Bachelors level programme in Electronic and Computer Engineering [4]. The module is based on providing an introduction to modern electronic circuit and system development that involves an integrated approach to the design, fabrication, and testing of an item from initial design concept through to in-service support. With this in mind, there is therefore a need for engineers to understand and appreciate the different activities in order to be active participants in a multi-disciplinary development team.

This module aims to provide an insight into how electronic circuits and systems are tested at the different stages of the design, evaluation, fabrication (manufacture), and in-service periods in the lifetime of electronic circuits and systems. The experiment content of this new module, both structured laboratory exercises and project work, requires careful consideration to ensure clearly defined and relevant outcomes are realised. A range of different exercises and circuits to test can be envisaged. However, the focus of the laboratories considered in this paper are related to digital IC test development, targeting test program development (the required set of test patterns and their application), design for testability (*DfT*), and built-in self-test (*BIST*).

To solve the test problem, from initial problem definition to final production level test, there are key activities to be undertaken, see Fig. 1, summarised here as:

1. Problem definition.
2. Test development actions.
3. High-volume, automated production test at both wafer and packaged device levels.

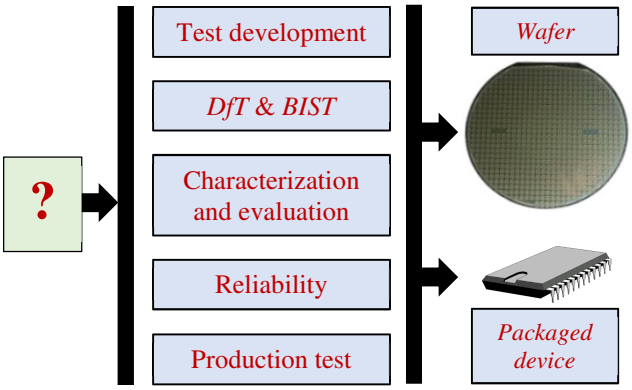

Fig. 1. Key IC test activities.

Intrinsic to these activities are electronic hardware design, computer software program development, and documentation.

The paper will be presented as follows. Section I has provided an introduction and rationale for the discussions covered in the paper. Section II will discuss IC test activities that target current and future industrial needs. The focus here will be on digital IC test development from initial test specification development through to production test program creation, and verification. The learning will therefore require specific electronic circuit and computer software design prerequisites. Section III will discuss the role of the laboratory in a new course (module) in test engineering development, and the choices made to select the appropriate mix of teaching and learning styles and technologies. Section IV will present a proposed laboratory platform based on the field programmable array (FPGA) [5, 6] that allows for student access to a hardware emulation of different digital IC test scenarios. This is seen as an alternative to verification through simulation of a software simulation model of the *circuit under test* (*CUT*) that involves hardware emulation. The different possible laboratory scenarios identifying different student access approaches (local and remote), and range of different circuits that can be emulated, will be introduced. The use of a case study design based on the IEEE Std 1149.1 "*IEEE Standard for Test Access Port and Boundary-Scan Architecture*" [7, 8] arrangement is embedded within the FPGA will be presented. Section V will conclude the paper.

II. IC Test Development

In this section, IC test and the different aspects in test development will be discussed. Modern electronic circuits and systems rely on the use of high performance and cost-effective microelectronic circuits (ICs). A typical electronic circuit or system will be developed using a set of ICs, with the ICs varying in performance and signal processing operations that they are required to implement. In general, the types of ICs can be categorized as being analogue, digital, or mixed signal (mixed analogue and digital). The end user of these ICs expects to obtain functioning and reliable devices that they can use without any performance or functional issues. This is particularly important in safety critical applications such as biomedical, automotive, and aerospace. How to provide such functional and reliable ICs is a major concern of the microelectronics industry, and if someone were to look at the lifecycle of an IC from initial concept through to shipment of devices to the customer, there would be a range of different activities that the provider would undertake. Within the microelectronics industry and focusing just on the key engineering disciplines that are required to come together, engineers would work in teams to realise a working IC. Specifically, *design*, *fabrication*, and *test* are the engineering disciplines of primary concern here. The engineers within each discipline would interact with each other in various, and complex, ways during the development of an IC in order to realize an IC design that meets all requirements, and, referring to Fig. 2:

1. **Design engineers** are required to create a circuit design that meets the required specifications. The focus would on to deriving a circuit design that would meet the required circuit design specification with the smallest size circuit possible, and which could be fabricated using the target fabrication process.
2. **Fabrication (process) engineers** are required to fabricate the design to the appropriate quality level. They would take the design created in (1), and fabricate the circuits using the target fabrication process to be available as either a bare die (a known good die (KGD)) or as a packaged device.
3. **Test engineers** are required to develop and implement suitable test programs that will verify the operation of the fabricated designs. There would need to be a working knowledge of the circuit operation as well as the underlying fabrication process.

Underlying these operations are the needs to create and deploy new IC designs in a timely manner that meet the needs of a particular market needs and which are cost-effective.

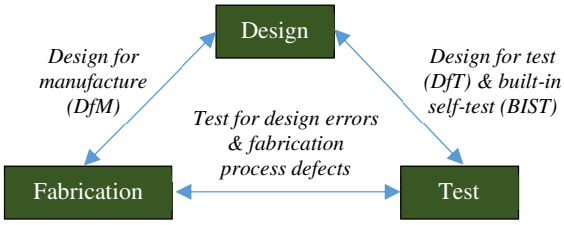

Fig. 2. Design, fabrication, and test interactions.

Increasingly, the role of test is a major concern with the development of increasingly more complex designs, higher operating frequencies, and the use of advanced lower geometry fabrication processes. Such moves in design and fabrication are required to meet current and future market needs but which produce increasingly difficult test challenges to solve. Design, fabrication, and test activities stopped being seen as separate activities years ago, and now are considered as interconnected activities where designs are created with fabrication and test considerations, following what can be referred to as a *DfX* (*Design for X*) approach. For example, the following *DfX* approaches are adopted in different situations:

- Design for testability (*DfT*).
- Design for manufacturability (*DfM*).
- Design for debug (*DfD*).
- Design for assembly (*DfA*).
- Design for reliability (*DfR*).
- Design for yield (*DfY*).

Each approach targets different aspects that are deemed to be of importance to the product developer. For example, *DfT* targets the need to test a design with a suitable quality level but in a realistic time, where the design complexity means that the design cannot be effectively tested without the introduction of additional circuitry within the IC to allow access for test purposes to the circuit that would otherwise not be possible. *DfT* is generally considered alongside *built-in self-test* (*BIST*), where:

- ***DfT*** considers the inclusion and use of test circuitry within the IC to allow an external tester access to the internal connections (nodes) and circuitry that would not be otherwise accessible by using the device pins only (the primary input/output (I/O) nodes).
- ***BIST*** considers the inclusion and use of test circuitry within the IC to allow the IC to test itself without the need of an external tester to provide the input stimulus and to monitor the output response of the IC. *BIST* is

a *DfT* technique but is generally considered as a subject by itself.

What aspects of test an individual would be involved in would vary and be dependent on the role within an organization and the test strategies adopted by the organization. However, in general test engineers would be required to be involved in test development activities that cover all aspects of the test process from initial test specification creation through to production test program development and deployment. Both electronic hardware and computer software programming skills would be required in undertaking such roles.

III. Development of a Teaching Module in Test Development Engineering

In this section, a new module in test development engineering is presented. This module has recently been introduced as an elective module for a new five-year Masters programme as part of the extension of an existing four-year Bachelors level programme in Electronic and Computer Engineering [4], and targets fifth year students who would be interested in test activities within the microelectronics industry. The rationale for this module is stated as:

> *"In modern electronic circuit and system development, there is an integrated approach to the design, fabrication, and testing of an item from initial design concept through to in-service support. There is therefore a need for engineers to understand and appreciate the different activities in order to be active participants in a multi-disciplinary development team. This module will provide an insight into how electronic circuits and systems are tested at the different stages of the design, evaluation, fabrication (manufacture) and in-service periods in the lifetime of electronic circuits and systems."*

As such it is required to provide depth as well as breadth to the subject at the appropriate academic level. The module is a one-semester, fifteen-week module consisting of twelve teaching weeks, project work, and an end of semester examination. Within the academic structure of the University of Limerick, the module has been devised with the following considerations:

1. Twelve teaching weeks, with two, one-hour lectures and one two-hour laboratory per week.
2. Space in the schedule for self-study time.
3. Project work that would be undertaken alongside the laboratory sessions.
4. Revision week (week 13).
5. End of semester examination (weeks 14 & 15).
6. Six ECTS (European Credit Transfer System) credit module.

This is a typical module structure and the focus of the module would be on digital IC test development, including design, *DfT*, and *BIST* considerations. However, digital test will be put into context with an introduction to test development activities, industrial requirements, economics, and, analogue and mixed-signal test. Hence, the module learning outcomes would account for the need for theoretical knowledge acquisition and practical skills development based on specified psychomotor, cognitive and affective domains.

IV. Laboratory Platform Based On The FPGA

In this section, the FPGA based laboratory platform developed to support the teaching and learning in the test development module will be introduced and discussed. This is based on the Xilinx Artix-7 FPGA [9], and the Digilent Arty platform [10].

Laboratory experiments can be created using a range of different electronic circuits and software programming languages. The choice of the appropriate set of technologies is not always immediately obvious, and several different approaches can be envisaged that would meet the objectives of the laboratory. In many cases, it is the personal knowledge and interest areas of the relevant academic that define what set of technologies are used. However, there is a significant effort required to set-up and maintain the laboratory.

In the laboratory arrangement to use in this module, the FPGA is used as the core technology that provides for a flexible and programmable laboratory platform to design and develop. The FPGA provides for a powerful alternative to the software programmed processor (microprocessor (µP), digital signal processor (DSP) or microcontroller (µC)) utilizing the hardware resources within the FPGA to develop a hardware only platform, or a hardware-software co-design platform that embeds one or more processors that will operate alongside the embedded hardware within the FPGA. In this laboratory set-up, see Fig. 3, the FPGA is provided in the Arty-35T development board. The Xilinx MicroBlaze processor [11] runs a software program, and this interfaces to experiment hardware within the FPGA that is created using VHDL (VHSIC (Very High Speed Integrated Circuit) Hardware Description Language) [12]. As an alternative, Verilog HDL [13] is also used to describe the hardware instead of VHDL.

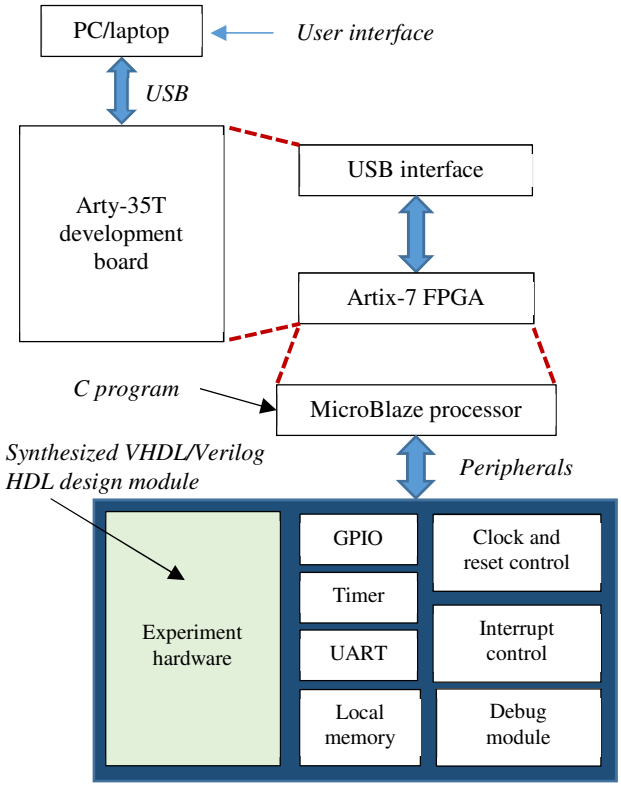

Fig. 3. FPGA based system architecture.

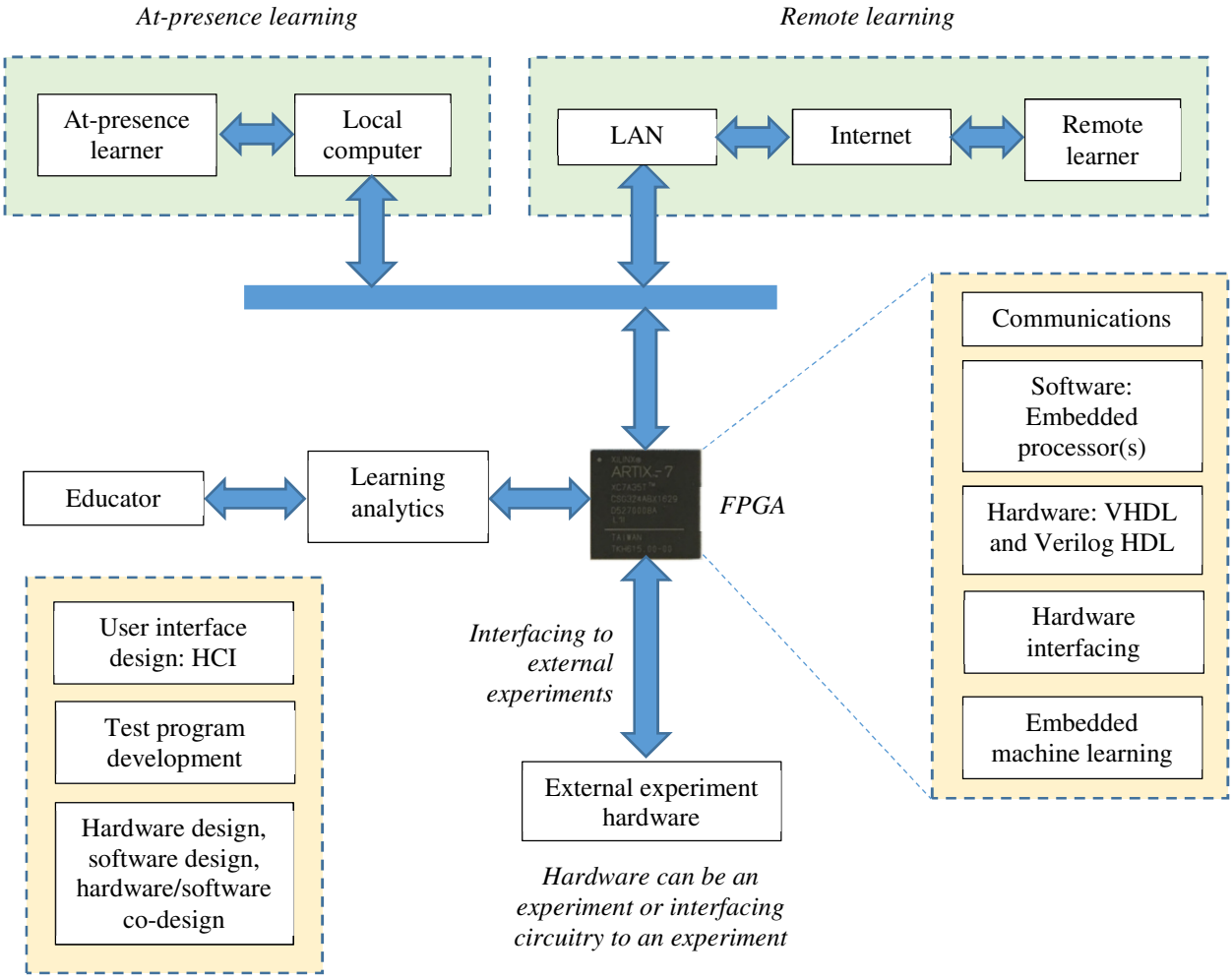

Fig. 4. Potential uses of the FPGA in test engineering laboratories.

The circuit design within the FPGA is based on the use of the Xilinx MicroBlaze 32-bit RISC (reduced instruction set computer) architecture soft processor core (version 11.0) [11] to provide the user and experiment interfacing, and a hardware module that implements the experiment (*core to test*) as shown in Figs. 4 and 5. Fig. 5 shows the top-level block diagram created using Xilinx Vivado 2019.2 [14]. The MicroBlaze processor runs a compiled C program created using Xilinx Vitis 2019.2 [15]. The processor communicates with the connected computer using a UART/USB (universal asynchronous receiver transmitter/universal serial bus) arrangement with the UART embedded as a processor peripheral within the FPGA. The peripherals are connected to the processor via the AXI (**A**dvanced e**X**tensible **I**nterface) bus [16]. The processor operates on a master clock frequency of 100 MHz. Development of the FPGA based system required design decisions in order to determine which operations were to be performed in software, and which operations were to be performed in hardware. The processor allows interfacing to the external computer (user), and the experiment hardware (internal and external), as well as the ability to implement a range of possible learning analytics operations that would be based on data collection and sorting, and analysis using supervised and unsupervised machine learning algorithms (e.g., [17]). For example, where each learner ran their test program on the FPGA, the FPGA could automatically log and locally store the user identification code (ID, i.e., a unique ID (UID)) along with the time/date and statistics on the running of the test program. This data could then be accessed at a suitable time to analyze the user operations. These are readily implemented and modified in software using C or C++ programming.

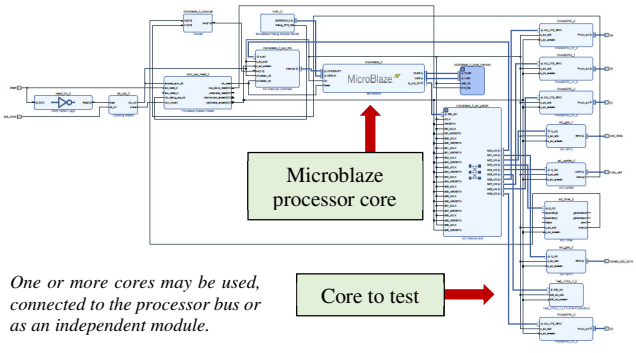

Fig. 5. MicroBlaze block design in Vivado 2019.2.

The *circuit under test* (*core to test*) was implemented in hardware, and any changes in the hardware would require re-synthesis of the design and reconfiguring the FPGA. This means that the *circuit under test* was a hardware emulation of a physical system rather than a simulation model. The hardware approach would also model a final system implementation more closely than a software model built into the processor program.

The FPGA and experiment would be accessed by the learner (user) either in a local learning mode (*at-presence*) or in a remote learning mode. The local learning mode would require access to the physical experiment hardware whereas a remote learning mode would require internet access and access to a remote laboratory [18] housing the physical experiment hardware:

1. ***Local***: The experiment hardware is local to the user. The *at-presence learner* interacts with the FPGA directly, and possibly additional external circuitry, via a computer software application. The FPGA provides the user communications hardware/software and internal circuitry to test.

2. ***Remote***: The experiment is remotely located from the user. The *remote learner* interacts with the FPGA indirectly via a network or internet connection. The FPGA forms a remote laboratory and allows multiple users to access the single hardware resource.

For example, in a *local* learning mode, the FPGA would be connected directly to the user computer, and the user would either access a provided software application or write their own software application to apply the test stimulus and read/analyze the test results. Different programming/scripting languages could be used, and either a graphical user interface (GUI) or command line (text) based approach could be adopted. Fig. 6 shows an example command line-based approach that reads and displays the input and output signals as binary values using a Python [19] script.

The idea behind the use of the PC (personal computer) side interface would be as follows:

1. The user interprets the available information on the *circuit under test*, and develops a suitable test procedure that defines the sequence of actions to follow.

2. Using either an existing PC side software application or an application that is written by the user following a predefined communications protocol, the test procedure is translated to a test program.

3. The test program is run by transmitting the test data and control signals, and reading the circuit response via the USB port.

4. The user saves and analyzes the results to determine the operation of the *circuit under test*.

In Fig. 6, the idea behind the *look and feel* of the interface is to be text based only to allow the learner to develop a test program/script at a lower level, and hence a greater understanding of the individual steps required in running the test than a GUI based approach might allow for. Hence, the target learner would be an individual who would need to learn how to develop test programs at the code level for activities ranging from design debugging through evaluation and characterization, to production test.

In Fig. 6, the test data is provided in a human readable form and would require further post-processing (text string manipulation) in order to extract the required information. This could be presented to the user in different formats such as raw text, within a spreadsheet, test output as predefined pass-fail values, and logic timing diagrams. This approach would allow for the laboratory developer to create different teaching and learning scenarios within the laboratory that could be used for different target learners. For example, the laboratory could be:

1. More hardware oriented, with the learner considering the operation of pre-defined hardware in terms of logical behaviour and signal timing.

2. Understanding the standard document(s), and interpreting the document(s) to develop test hardware and test programs.

3. Developing functional test programs or structured test programs. In a structured test program, the test program developer would aim to develop tests that look for specific faults, such as the stuck-at-fault or bridging fault, rather than looking for correct functionality. Designs with programmable faults could be included in the design.

4. More software oriented in developing suitable applications that fit the needs of the user, and which enable the learning of software design techniques as well as different programming/scripting languages.

5. Developing new design cores to test, and extending the provided test logic to incorporate more aspects of the standard document, more instructions, and enabling extensions to the set-up with more complex designs and other *DfT* approaches such as internal scan path.

```
Command Prompt
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000001 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000001 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000001 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000001 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000011100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001100001
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001000011
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10010001000011
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10011001000011
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001000111
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001000111

---------- Boundary scan register test 3 (SAMPLE/PRELOAD) ----------

---------- Boundary scan register test 4 (BYPASS) ----------

---------- Final values ----------
>> a=0 b=0 cin=0 ; Cout Sum(3) Sum(2) Sum(1) Sum(0) TDO = 000000 = 0 ; 10000001000111
-- ------------------------------
End script run Sun Apr 12 09:11:16 2020
```

Fig. 6. Example test I/O using a Python script and Windows Command Prompt interface.

V. Conclusions

This paper has introduced and discussed the design of a new teaching module in IC test development. The rationale for the module was identified and the structure of the teaching and learning discussed. The role of the laboratory in this type of teaching and learning was developed and, for this module, a laboratory arrangement using the FPGA was elaborated. The laboratory is based on the Xilinx Artix-7 FPGA and utilizes the hardware resources within the FPGA to provide for a flexible hardware platform to implement a range of different teaching and learning scenarios.

REFERENCES

[1] S. L. Hurst, *VLSI Testing, digital and mixed analogue/digital techniques*, IEE Circuits, Devices and Systems Series 9. United Kingdom: The Institution of Electrical Engineers, 1998, ISBN 0-85296-901-5

[2] R. Rajsuman, *System-on-a-Chip: Design and Test*. United States of America: Artech House Publishers, 2000, ISBN-13: 978-1580531078, ISBN-10: 1580531075

[3] V. Agrawal and M. Bushnell, *Essentials of Electronic Testing for Digital, Memory and Mixed-Signal VLSI Circuits*. United States of America: Springer-Verlag NY Inc., 2013, ISBN-10-1475781423, ISBN-13-9781475781427

[4] University of Limerick, Department of Electronic and Computer Engineering, LM118, Bachelor/Master of Engineering in Electronic and Computer Engineering. Accessed on: April 18 2021. [Online]. https://ece.ul.ie/wp-content/uploads/2019/10/lm118_outline_descriptors.pdf

[5] Xilinx, Field Programmable Gate Array (FPGA). Accessed on: April 18 2021. [Online]. https://www.xilinx.com/products/silicon-devices/fpga/what-is-an-fpga.html

[6] Intel®, Intel® FPGAs Resource Center, "What is an FPGA?". Accessed on: April 18 2021. [Online]. https://www.intel.com/content/www/us/en/products/programmable/fpga/new-to-fpgas/resource-center/overview.html

[7] The Institute of Electrical and Electronics Engineers (IEEE), IEEE Std 1149.1™-2013 (Revision of IEEE Std 1149.1-2001), IEEE Standard for Test Access Port and Boundary-Scan Architecture, 13th May 2013

[8] Kenneth P. Parker, *The Boundary-Scan Handbook*, Second Edition, Analog and Digital. United States of America: Kluwer Academic Publishers, 2000, ISBN 0-7923-8277-3

[9] Xilinx, Artix-7. Accessed on: April 18 2021. [Online].https://www.xilinx.com/products/silicon-devices/fpga/artix-7.html

[10] Digilent, Arty Family. Accessed on: April 18 2021. [Online]. https://store.digilentinc.com/arty

[11] Xilinx, MicroBlaze Soft Processor Core. Accessed on: April 18 2021. [Online]. https://www.xilinx.com/products/design-tools/microblaze.html

[12] Institute of Electrical and Electronics Engineers, IEEE Std 1076™-2008 (Revision of IEEE Std 1076-2002) - IEEE Standard VHDL Language Reference Manual, 26th January 2009, ISBN: 978-0-7381-6853-1, ISBN: 978-0-7381-6854-8, DOI: 10.1109/IEEESTD.2009.4772740

[13] Institute of Electrical and Electronics Engineers, IEEE Std 1364™-2005 (Revision of IEEE Std 1364-2001), IEEE Standard for Verilog® Hardware Description Language, 7th April 2006, ISBN 0-7381-4850-4, ISBN 0-7381-4851-2, DOI: 10.1109/IEEESTD.2006.99495

[14] Xilinx, Vivado Design Suite HLx Editions. Accessed on: April 18 2021. [Online]. https://www.xilinx.com/products/design-tools/vivado.html

[15] Xilinx, Vitis Platform. Accessed on: April 18 2021. [Online]. https://www.xilinx.com/products/design-tools/vitis/vitis-platform.html

[16] Xilinx, AXI Reference Guide, UG761 (v13.4) January 18, 2012. Accessed on: April 18 2021. [Online]. https://www.xilinx.com/support/documentation/ip_documentation/axi_ref_guide/v13_4/ug761_axi_reference_guide.pdf

[17] Ahmad Shawahna, Sadiq M. Sait and Aiman El-Maleh, “FPGA-Based Accelerators of Deep Learning Networks for Learning and Classification: A Review", IEEE Access, pp 7823 - 7859, 28th December 2018

[18] Ian Grout and Abu Khari bin A’ain, "Introductory Laboratories in Semiconductor Devices using the Digilent Analog Discovery”, Proceedings of 2015 12th International Conference on Remote Engineering and Virtual Instrumentation (REV), 25th - 27th February 2015, DOI: 10.1109/REV.2015.7087278

[19] Python Software Foundation, Python. Accessed on: April 18 2021. [Online]. https://www.python.org/