Remote Laboratory for e-Learning of Digital Systems Design

Alexander Vera, Jorge Quiñones, Edison Enriquez, Sergio Arenas and Alvaro Bernal

Abstract—Remote Laboratories allow students to practice the experiments related to specific fields. Many research works are still being developed, which encourages the implementation of alternative teaching/learning models. This is a challenge for engineering education because its highly practical contents are difficult for carrying in distance learning. This paper presents an architecture for remote laboratory of Digital Systems Design focused on supporting blended teaching. Learners may access to a remote training board based on a Field Programmable Gate Array (FPGA); then, they may execute a test by a logic analyzer module, and interact with other online users by integrated web 2.0 services. This paper describes both the hardware platform and the software application of the proposed system.

Index Terms—Remote laboratories, Digital Systems, FPGAs, Logic Analyzer

I. INTRODUCTION

In the e-learning methodologies of digital electronic circuits design, both exercises and practices must be integrated. However, an expensive instrumentation is required in the experimentation laboratories for programming, testing and measurement, as essential components for teaching digital circuits. Logic analyzers support these procedures because they are identified as instruments for digital systems testing and diagnosing. Further, commercial development boards based on reprogrammable digital architectures are used for training due to their capabilities to implement and debug digital designs using a PC-based programming tool. Usually, this tool is used to program embedded processors, logic analyzers and other specific cores over a field programmable gate array (FPGA). The remote laboratory’s goal regards to be available both the FPGA and test capabilities to the remote users.

In order to build a development system to support didactical practices for electronic engineering e-learning, this work proposes a low cost remote laboratory for programming and testing digital systems design where the hardware target is a platform with independent programmable resources. Moreover, a designed web-based application allows interaction between users and this target to program and test logical functions according to scheduled laboratory practices. In this paper, the design of both a target platform and a web application for a remote laboratory of digital electronic circuits is presented. The proposed hardware target incorporates a logic analyzer and an FPGA as independent modules to ensure that the main reprogrammable resource for user is fully available. The web-based application involves some features of Web 2.0 technology in order to encourage the active learning, especially for undergraduate students. With these findings, the remote laboratory for digital systems design allows high modularity and easy resources increase.

A state-of-the-art review about logic analyzer architectures and remote laboratories was performed driving this study by higher education. Main contributions proposed remote laboratories for training in general electrical/electronic engineering with hardware/software-based infrastructure allowing to the students the implementation of physical experiments. These projects used computer-based instruments and remotely controllable switch matrices for testing or worked with manual-external instruments. Mostly projects considered client-server architecture, but their hardware resources should include external measurement instruments to complete the design cycle. Moreover, some patents describe logic analyzer architectures to be programmed over logic devices. These architectures are used by both integrated development environments (IDE) and development boards which share the programmable resources because embedded cores are programmed on the same logic device.

This paper also describes the model of proposed remote laboratory for digital circuits e-learning. This approach involves a hardware target which is connected to resources server (PC) via USB and a Web-based application that supports both data and control information based on a database engine. The target consists of the following
subsystems: Timing Logic Analyzer, Sample Memory, Interconnection Interface, FPGA, System Configuration, Microcontroller-based System Manager and Communication Module.

Different Logic Analyzer architectures were studied and a new optimized release for hardware implementation is presented. The logic analyzer was implemented on a reprogrammable digital architecture using Hardware Description Language. Its design was focused for educational field, which suggests reducing costs and achieving modularity. With these findings, this logic analyzer module could be interconnected to other reprogrammable subsystems.

A Real-Time Operating System (RTOS) has been selected for tasks scheduling on the target manager. These tasks are related to the platform specific functions. A FreeRTOS scheduler was configured for both preemptive and cooperative operation to assess to its performance.

A web application was developed to access from a remote user the platform services. Users are teachers and students, the first ones are usually service managers. The information related to these users is saved on MySQL database. WAMPServer (Windows, Apache, MySQL and PHP - Server) packages were used to develop the web application. In order to manage the server-resident content, the Joomla CMS (Content Management System) was used too. Some java utilities were necessary to carry out the communication process and the graphical user interface.

Remote Laboratories have strengthened the strategies for both blended and project-based learning and together with emerging technologies have contributed to doing collaborative learning. This work presents a contribution to this field benefiting from the social contexts of the web 2.0 services. The paper is organized as follows. Section 2 presents a state-of-the-art review of remote laboratories architectures for engineering. In Section 3 we present a detailed description of the proposed architecture of a remote laboratory for teaching of Digital Electronic Systems, about its hardware subsystems and software description. Finally, we conclude the paper in section 4.

II. REMOTE LABORATORIES ARCHITECTURES

Online learning has increased with the growing demands for access to virtual education environments. Its development helps for approaching to information sources and for complementing the teaching in distance learning. However, Online learning is meaningful when skills are developed by including virtual and/or remote experiences.

Departments of Communications, Informatics and Electronics Engineering have included virtual laboratories applications on their academic programs, as showed by [1][2]. This fact allows the educational training for a greater number of students per class in higher education. An experience-based learning invites to use remote laboratories for improving the specific skills development in these fields.

Both, Experience-Based and Active learning are essential for engineering education, especially for fields as electronics, communications and informatics. Learning strategies and resources should regard several items: first one the loops design, the second one the adaptability to different laboratory practices and finally the encouragement of the team working. Teaching on Digital Systems requires that students simulate digital logic functions, implement these designs and make a validation test.

There are commercial development kits and academic research products to support the training processes for educational. Mostly these products suggest their adaptation to virtual laboratories as general purpose systems, but they require adding both external measurement and testing instruments.

A hardware environment for executing general purpose applications over FPGAs was developed by Universidad Complutense de Madrid in 2005[3]. This work proposes I/O learning by keyboard, VGA output, task scheduling, SDRAM memory, Network and BIOS. I/O resources management is based on task programming per user. Post-implementation tests require the use of external instruments as Logic Analyzers.

The design of a scheduler over FPGA was also proposed in 2006 [4] by Universidad Complutense de Madrid. This work shows the development of both a memory controller SDRAM DDR and a manager application, which get together a multitask environment for dynamically reconfigurable devices.

Simultaneously, a design and manufacturing FPGA-based didactic board was proposed in the same university [5]. This board is aimed to local laboratory practices embodying basic I/O devices as display, Led, buttons, keyboard, PS/2 port, VGA, IDE, buzzer. This system was based on Spartan II Xilinx FPGA and uses a graphics user interface over Windows OS (based on Visual C++).

The works described above were important contributions to the teaching on digital systems because they allow students interact with their designs at the implementation level. However, these projects only support the local execution of both programming and test stages by joining external measurement instruments.

Technical University of Kosice proposed a Remote Laboratory for a FPGA based reconfigurable System Testing [6]. This expensive project uses an Altera FPGA-based development kit, a 68-channels logic analyzer, a 4-channels digital oscilloscope and a vector signal generator. The logic analyzer and the vector signal generator are linked with a server via Ethernet, and the digital oscilloscope is plugged into USB port of the measurement server. Users access the laboratory via Internet by another Web server.

At Northern Illinois University, authors have shown the project named FPGA e-Lab [7], as a technique to access a remote laboratory to design and test. This system consists of a Xilinx Spartan-3E FPGA based development kit, a National Instruments Data Acquisition Card and optionally,
oscilloscope, signal generator and web camera. Users access the laboratory using a remote desktop service which operates under Windows XP OS.

The project RMCLab [8] from the University of Patras is based on client/server architecture and consists of following basic entities: client, instructor client, application server, resources server and its entire laboratory infrastructure, including instruments and hardware modules. This system has a motherboard which drives up 64 expansion cards. Each card consists of a FPGA and auxiliary modules. This hardware is a complete development to remote laboratories support but it demands enough power.

Another reported works [9][10][11] have put forward significant contributions with similar topologies. Most of their hardware platforms involve reconfigurable architectures to program customizable functional blocks; however, these blocks should be supported by developer-provided software libraries. Some architectural models have been adapted to commercial development boards and their respective Integrated Development Environments.

Some different remote laboratory architectures for educational were developed at Universidade Estadual de Campinas. First, a low-cost architecture was proposed by [12][13]. This architecture was based on a microcontroller and involved communication interfaces, instrumentation and I/O. However, a faster architecture was proposed by [14]. This approach presented a functional model in terms of abstraction layers (WebLab Application, Middleware, Network Infrastructure, REDLART), based on the REDLART platform. This hardware platform was based on Virtex II FPGA, both processing and storage cores and I/O interfacing. A set of routines, services and protocols were developed regarding a hardware adaptation but also some software development tools were required as Matlab/Simulink (Mathworks), System Generator (Xilinx) and Project Navigator (Xilinx).

A remote control system based on Java to access via internet was developed at Universidad del Valle in Colombia. This work based on client/server architecture [15] allowed the FPGA configuration from a remote place. An FPGA-based hardware should be plugged to a resources server which contains the main application. This hardware platform [16] consists of two boards: the first one includes the reconfigurable architecture and its I/O interfacing. The second one contains a logic analyzer module implemented over an FPGA, which should be connected to the desired test points using programming tools. The logic analyzer configuration and the FPGA programming were tested via parallel port by a Delphi software application, which resided on resources server. The mentioned application was later improved in [15] where the user interface capabilities was increased using both servlets and applets in Java.

Moreover, user Interfaces for remote laboratories have been built with programming languages as HTML, JAVA or PHP, but some projects have used specialized software environment as LabView (National Instruments), Matlab (Mathworks) and ISE Design Suite (Xilinx). Users can get real-time feedback from experimental place through video streaming by using a local camera. This fact is an advantage for distance learning because each user may access to a remote experimental environment regarding the visual behavior.

III. REMOTE LABORATORIES FOR TEACHING OF DIGITAL SYSTEMS

A Client/Server architecture has been used to build a remote tool for teaching and learning of digital systems. Main contents, procedures, methodologies and practices of these fields are together supported by this proposed system. Both teachers and students may access to the remote laboratory environment for sharing experiences in order to encourage their skills development. With the available services on this system, teachers or instructors may select what knowledge and skills must be achieved by their students on their digital systems course. For instance, teachers may implement online activities to learn the digital design cycle, but students may work in groups by online meetings. So teachers may explore the implementation of strategies for a blended learning, but also gaining teamwork skills.

This proposed system allows to program and test electronic logic functions over a reprogrammable digital device from remote places. This fact enables to students for accessing to laboratory resources from different locations ever they have an internet connection.

A. Overview

A general sketch of the proposed remote laboratory is shown in Figure 1. This system consists of the following: a training board for executing both the programming and test of logic functions, and a network infrastructure to support a set of user services. The shared hardware platform is plugged via USB into a PC, which is a resources server. This computer has a local network connection and supplies the services for management of the training board. An IP camera is also connected to network for providing a visual feedback loop from where the target resides.

A web server with a MySQL database manages the users and access information. This server sends requests to the intranet, according to user specifications. The Graphical User Interface (GUI) and the designed security modules were written with Java and PHP languages. However, a Contents Management System (CMS) was used to reduce design times and facilitate the collaborative creation of the posted contents.

B. Hardware Architecture

A training board is the main hardware target in the proposed system. This is plugged to USB port of the resources server which has the software modules to their control and communication. A web-based application resides into web server for user interfacing, and a router is needed for linking
The shared hardware target consists of the following subsystems: Timing Logic Analyzer and its Sample Memory, FPGA, Interconnection Interface, Configuration Interface, Microcontroller-based System Manager and Communication Module (See Figure 2, Images appendix).

Timing Logic Analyzer

The Timing Logic Analyzer is a 32-channels module for testing signals which are defined by user. Its architecture was described using VHDL (a Hardware Description Language) to be further implemented into a CPLD.

The Logic Analyzer architecture is based on a finite-state machine and other interfacing modules (see Figure 3, Images appendix). This state machine has six states: “Reset Setup”, “Stopped”, “Running”, “Triggered”, “Acq Complete” and “Uploading Data”. The “Reset Setup” state occurs when the logic analyzer is not set and the user is logged off. The “Stopped” state occurs when the scheduled laboratory user is online and logic analyzer is set, but user has not started the acquisition process. The “Running” state occurs once the user delivers a “start acquisition” command and the “Triggered” state occurs once the set trigger condition is met. When the number of samples programmed by the user has been completed, then the state machine enters the “Acq Complete” state. While data is uploading to resources server, then the “Uploading Data” state is activated, and when this uploading process finishes then the state machine returns to “Stopped” state.

The Logic Analyzer has also modules for sample memory addressing, trigger selection and stimulus generation. However, a command interpreter supports the control and interaction with the logic analyzer. These actions are related to set, start, status inquiry, and stop this functionality.

Sample Memory

The sample memory for logic analyzer is a SRAM module that stores up to 4M 32-bit words. Its control and addressing are driven by the logic analyzer module for reading and writing processes. This memory block is used only for storing sampled data. The memory writing process is active when the logic analyzer is in either state “Running” or “Triggered”. Whether the logic analyzer is in “Uploading Data” state, a memory reading process is ongoing to carry out sampled data to resources server via USB.

FPGA and Interfaces

A Spartan-6 Xilinx FPGA is used in the hardware platform for programming the electronic logic function defined by the user. This device (XC6SLX45T) has 43661 logic elements and 190 I/O pins, which is inexpensive and usable by the student user.

Spartan-6 FPGA series support several configuration modes, but this work uses only two modes: Slave serial and SPI configuration. First one is used for configuring it from a microcontroller unit, which must manage all the configuration sequence up to send the bit stream by I2C interface [17]. SPI configuration is used for optional programming from a 16 Mbit SPI-flash memory. These configuration modes are for FPGA programming from a user project. It may be programmed for temporary use, as when user uses the first mode, but also for a long time use by flash memory configuration. This permanent configuration may be used by teachers to implement strategies where students have to analyze specific logic functions by programming a predefined project.

The training board has also basic I/O interfaces as leds, dipswitches, push buttons, LCD and 7-segment displays. These I/O devices are mapped to the same bank on the FPGA. However, a direct interfacing is required between fixed 32-pin of the FPGA and the input channels of the logic analyzer module. Users should know this connections map prior to FPGA configuration in order to select the correct test pads. Also, the training board has expansion connectors for connecting external modules in local practices or when teacher needs to plan specific practices interfacing with external hardware.

Microcontroller-based System Manager

The microcontroller unit is an 8-bit MCU from Microchip. It has a USB link with the resources server for accessing the management unit of the training board. This server sends commands and data from the user requests to the microcontroller unit, which interprets this information and generates both the control and data signals of the logic analyzer and FPGA modules.

The uploading data process is also controlled by the microcontroller, which reads the sample memory. It is synchronized with the logic analyzer module which addresses the SRAM. The read sampled data are sent to resources server through the embedded USB module on microcontroller.

A real-time kernel for embedded systems needs to reside on the microcontroller unit for scheduling all demanded tasks. Then a FreeRTOS kernel code is used in order to manage the hardware platform resources. A C-compiler (C18) must be
integrated with the MPLAB IDE to configure, simulate and program this RTOS system over selected microcontroller.

The training board demands six tasks to be managed by a preemptive FreeRTOS scheduler. The USB communication and instructions decoding tasks have the highest priority, because they are the main system interface to interaction between the resources server and the hardware platform. All user requests can be driven from the webserver to the training board through the resources server interface by sending commands.

Lower priority tasks can be temporarily created to save memory locations. The instruction decoding task can make this creation because has decoded what user is requesting. This task creation process is shown in Figure 4.

![Fig. 4. Task Creation Process](image)

**C. Software Description**

There are two types of applications to develop on this system: first a web application to graphical user interface and second, a background application to the communication with the training board. The general procedure for interaction between user and system is described by the flow diagram shown in Figure 5. Images appendix.

According to Netcraft and W3Techs statistics, Apache is the most popular web server (65.6% usage), followed by Microsoft IIS (18.0% usage) [18]. An Apache server was installed on the laboratory architecture because it is also modular, multiplatform and supports languages as PHP, Phython and Perl [19]. Then, the WampServer package is selected because it is delivered with the latest releases of Apache, MySQL and PHP [20].

The user records are managed from a MySQL database engine which saves user id, passwords, names, user type, registered courses, scheduled activities, last IP addresses, email, social networks id, Nicknames and other temporary information to statistical ends. However, a Joomla Content Management System (CMS) was selected to organize the content to present the user and to expedite both the web applications development and the database management.

A table for user registration has been created and then, an access form request login and password to the user. The application identifies if the logged user is administrator (teacher) or standard user (student) and then shows the corresponding information. For example, only admin users may add users (see Figure 6, Images appendix), configure laboratory practices, modify practices scheduling and enable online assessments and self-assessments.

The standard users of this remote laboratory may access to manage the training board only when they have scheduled an appointment. Whether user is not scheduled, he may go online and uses only enabled social services as chat, forums, view Facebook activity, write Facebook comments into laboratory platform, and click “Like”. All users may request the video streaming service which was added using Joomla forms, but the scheduled user has the highest priority to access this service. These tools are services that support the strategies of collaborative learning because encourage the teamwork when teacher thinks appropriate.

When students have logged and enter to online-laboratory workspace, the software tool waits for the following fields:

- **FPGA configuration**: Device for programming (FPGA, Flash-Memory), Bitstream source (uploaded file, Flash-Memory)
- **Logic Analyzer Setup**: Trigger (pre-post trigger, edge, level, falling, rising), number of samples, sampling frequency, status info, type of stimulus (clock, low level, high level), channels selection (8, 16, 32)
- **Start Programming**, Start Logic Analyzer sampling, and **Stop Acquisition with Data Downloading** (Graphics visualization for user analysis) (See Figure 7)

Additional activities can be uploaded by the admin user and displayed to standard user during practice execution.

**D. Scope and Limitations**

This proposed system allows the scheduled access to the reconfigurable hardware resource, one user at a time. However, many students may be logged simultaneously to share experiences, files and video using the Web 2.0 resources.

The contents on the website could be managed by the course tutor who guides the learning process. Students would be enabled to upload only the allowed data, which are homeworks, activities, configuration files and social networks activities.

The training board has an user available FPGA and a logic analyzer, which are configurable via USB. This logic analyzer does not demand resources of FPGA and its channels number can be set up to 32. These sample channels have been laid out on the Printed Circuit Board (PCB), so it does not require test probes.

Open source tools supported the development of software application. Thus, many contributions can be made to improve it and arranging for other courses and levels, so as it can run over different platforms.

**IV. CONCLUSION**

A remote laboratory for teaching of Digital Systems Design is a proposed system to support the adoption of alternative strategies for teaching and learning on this field. Teachers can
try to gradually implement strategies as blended learning, problem-based learning and doing collaborative learning on this course, thanks to the integration of the web 2.0 services. The concepts and tendencies about this web evolution were analyzed by [21] and considered in this proposed academic application.

The highly social tendencies have now transcended to both internet and education fields, where a lot of content is being produced by users. Collaboration and user generated content is possible by the availability of data for developers and network support to teamwork. With the remote application for laboratory, together cognitive, communicative and propositional skills may be encouraged. The sharing online support and the included services allow an entertained learning. Learners on Digital Systems can access to a training board with support for implement logic functions, test them and discuss it. The programmable device is fully available and the integrated logic analyzer is a real instrument, which is ideal for different levels of learning. This instrument was designed with 32 channels, and an average sample time of 10 ns was obtained from simulation results. Its design suggests reducing costs and achieving modularity. This logic analyzer module was interconnected to a Spartan-6 FPGA, which has DSP48A1 Slices, RAM Blocks, 43K logic cells, and 190 user I/O. Thus medium complexity digital designs can be supported by this hardware platform. Both USB and network connectivity guarantees a fast interaction between resources server, training board and users.

Finally, the training board could be used by more than one student simultaneously; however, it requires a dynamic configuration interface for the FPGA and some additional tasks to the RTOS. Thus, multiple users can simultaneously access to this resource for programming, test and debug functions.

ACKNOWLEDGMENT
This work was supported by RENATA, MEN-Colombia, and Universidad del Valle.

REFERENCES


Alexander Vera T., was born in 1979, Colombia. In 2003 he received his B.S. in Electronic Engineering from Universidad del Valle, Colombia. He received his Specialist degree in Radio communications from Universidad del Quindío, Colombia in 2008. Currently, he is Ph.D candidate in Electronic Engineering at Universidad del Valle.

From 2001 to 2005 he worked as teacher at Escuela Americana de Negocios, Centro Colombiano de Estudios Profesionales and Institucion Educativa Guillermo Valencia. In February 2006, he joined the Electronic Engineering Department at Universidad del Quindío as professor, where he joined also the GIDSPRO research group as member. He teaches Architecture of microprocessors, Digital Systems and Interfaces, and Microcontroller-based systems, which are his areas of interests. Now, he is a member of two research projects at Digital Architectures and Microelectronic Group from the Universidad del Valle.

Jorge Quiñones, was born in 1987, Colombia. In 2008 he received his Technologist degree in Electronics from Universidad del Valle, Colombia. Currently, he is working toward the B.S in Electronic Engineering at Universidad del Valle.
In September 2011, he joined the Digital Architectures and Microelectronic Group from the Universidad del Valle as researcher. His areas of interests are the computer programming and the programmable architectures.

**Edison Enriquez.** was born in 1990, Colombia. Currently, he is working toward the B.S in Electronic Engineering at Universidad del Valle, Colombia.

In September 2011, he joined the Digital Architectures and Microelectronic Group from the Universidad del Valle as researcher. His areas of interests are the programmable architectures, Microcontroller-based Systems and embedded Systems.

**Sergio Arenas.** was born in 1991, Colombia. Currently, he is working toward the B.S in Electronic Engineering at Universidad del Valle, Colombia.

In September 2011, he joined the Digital Architectures and Microelectronic Group from the Universidad del Valle as researcher. His areas of interests are the programmable architectures, Digital Systems Design, and Hardware Description Languages.

**Álvaro Bernal.** received the B.S. degree in electrical engineering from Universidad del Valle, Colombia, and the M.Sc. degree in electrical engineering majoring in VLSI circuit design from Sao Paulo University, Brazil in 1992. In 1999 he received a Ph.D. degree emphasized in microelectronic from Institute National Polytechnique, Grenoble, France.

In 1993, he joined the Electrical and Electronic School of the Universidad del Valle where he teaches CMOS VLSI Design, Physics of Semiconductor and Electronic Devices. His research interests are in the areas of digital circuits design, low-power digital CMOS and embedded systems. Currently, is director of the Digital Architectures and Microelectronic Group.
IMAGES APPENDIX

Fig. 2. Block Diagram of Training Board

Fig. 3. State Diagram for Logic Analyzer Module
Fig. 5. General Procedure for User/System Interaction

Fig. 6. "Add User" Interface for Admin User
Fig. 7. Screenshot of Signal Visualization