AC 2009-1206: ADVANCED DIGITAL LABORATORY: AN FPGA-BASED REMOTE LABORATORY FOR TEACHING DIGITAL ELECTRONICS

Kayode P. Ayodele, Obafemi Awolowo University, Ile-Ife, Nigeria

Olawale Akinwale, Obafemi Awolowo University, Ile-Ife, Nigeria

Lawrence Kehinde, Texas Southern University

Oladipo O. Osasona, Obafemi Awolowo University, Ile-Ife, Nigeria

E.O.B. ajayi, Obafemi Awolowo University, Ile-Ife, Nigeria

O.O. Akinwunmi, Obafemi Awolowo University, Ile-Ife, Nigeria

Advanced Digital Laboratory: An FPGA-Based Remote Laboratory for Teaching Digital Electronics

Abstract

The experimentation component of most Science and Engineering curricula in Nigeria is inadequate. In Obafemi Awolowo University for example, undergraduate students typically carry out around five assignments related to digital electronics, and there is no treatment whatsoever of Field Programmable Gate Arrays(FPGAs). In the research work being reported, an attempt has been made to develop a remote laboratory though which the number of digital electronics experiments students carry out can be increased.

The remote laboratory, called the Advanced Digital Lab (ADLab), allows students to synthesis digital systems on an FPGA with a hardware description language. To achieve this, a development board with an Altera Cyclone II FPGA is connected to a computer implementing the server tier of the iLab batched architecture. The client through which the remote student interacts with the ADLab is implemented with Java, which allows for a reasonable amount of platform independence.

This paper discusses the software and hardware aspects of the ADLab architecture and gives some insight into some design decisions. The paper also reports that the system is being tested at Obafemi Awolowo University and that student feedback so far indicates high student satisfaction with the remote laboratory.

Keywords: *iLab, ADLab, FPGA, remote laboratory*

I. Experimentation and Remote Laboratories

When applied within an engineering curriculum, experimentation is supposed to achieve specific goals. It allows students to develop skills in any combination of up to 13 distinct categories[1]. Three main elements are required for experimentation in the context of engineering education: the student, the system under test (including associated test equipment), and the laboratory, which is a location or means though which the student can access and manipulate the system under test.

Traditionally, to work on the system under test, students need to be physically present in the laboratory. In recent years however, a set of techniques and tools have made it possible for the student to access laboratory hardware without being at the same physical location or time as the equipment. Such a laboratory in which there is a spatial or temporal displacement between the student and the system under test is generally referred to as a remote laboratory[2]. A remote laboratory is thus a version of a classical *in-situ* laboratory geared towards distance learning environments. Remote laboratories facilitate a flexible learning approach, which is the key to successful hands-on experimentation [3].

Motivations for remote laboratory development include [2, 4, 5] allowing:

- sharing of heavy and expensive instruments and equipments between institutions
- anytime and anywhere lab access

- porting of lab activities to distance learning environments
- resorting to real systems for illustrations, during on-line courses or virtual classrooms
- putting students in front of real situations and allowing them to discover system behaviors, to train at using instruments and to verify scientific theories
- a single remote laboratory to be able to cope with drastically increasing number of students

It should be noted that another approach utilized in laboratory experimentation in recent years is one whereby the system under is implemented test purely in software. That is, the student performs experiments on an interface serving as a metaphor for a back end which is a simulation of some other system. Where such access is local, the setup is referred to as simulation, and if the simulated backend is spatially or temporally displaced from the student, such laboratories are generally referred to as Virtual Laboratories [6] .

II. The iLab Batched Architecture

One of the prominent platforms for remote laboratory development is the iLab architecture, developed at the Massachusetts Institute of Technology (MIT). As this platform has evolved, it has spawned three forks, catering for three different categories of experiments[7].

- *Batched experiments* are those in which the entire course of the experiment can be specified before the experiment begins. MIT's Microelectronics WebLab [2, 5, 7] provides an example. Through the WebLab, students can characterize a variety of semiconductor devices by preparing a test protocol. This is accomplished by using an interactive editor before the semiconductor characterization executes.
- *Interactive experiments* are those in which the user monitors and can control one or more aspects of the experiment during its execution. MIT's online Heat Exchanger [8] provides an example of this. Students can dynamically change the input to heating elements and the action of pumps controlling fluid circulation in the heat exchanger while watching instruments report the changing temperatures. Another example of this is the Obafemi Awolowo University (OAU) Robotic Arm iLab.
- *Sensor experiments* are those in which users monitor or analyze real-time data streams without influencing the the phenomena being measured. MIT's online photovoltaic station is an example of a sensor experiment.

ADLab was developed based on the iLab Batched Architecture, which is the iLab software architecture that supports batched experiments(see Figure 1). It resembles the typical three-tier enterprise web application architecture with the following tiers:

- The first tier is the *lab client.* It is usually a rich internet application, running within a web browser.
- The middle tier is the s*ervice broker.* It is a web application which provides the authentication and authorization and administrative functionality. The service broker is a freely available open source web application developed at the Massachusetts Institute of Technology. The lab client communicates solely with the service broker, which forwards experiment specifications sent by the lab client to the third tier.

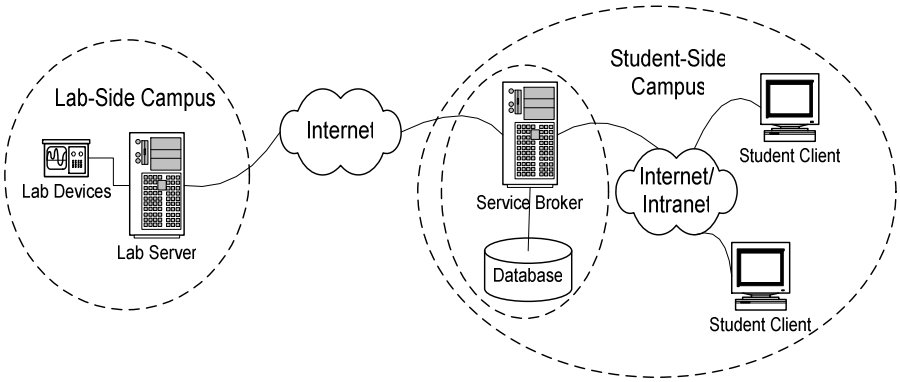

Figure 1: Topology of the Batched Architecture

- The third tier is the *lab server*. It executes the specified experiments and notifies the service broker when the results are ready to be retrieved.

The three tiers of the iLab batched architecture are integrated via platform-agnostic web services. The lab client and the lab server represent the domain- or lab-dependent software modules; while the service broker is a completely generic software module. That is, it is expected that laboratory staff be able to configure a fresh version of the service broker straight from the software distribution to register and cooperate with any lab server that implements the appropriate lab server interface expressed in terms of web service.

It is important to note that the lab server has no knowledge of the users of the system. It only stores experiment specifications and results temporarily. The service broker authenticates users, checks on their authorization to contact a particular lab server, accepts an experiment specification from the user's client, and waits to retrieve the result once the experiment completes. The experiment specification and results are stored on the service broker, which also maintains the link between a user and his experiments. Thus all the resources consumed by a user except for the runtime resources required to execute the experiment can be drawn from a service broker located on the user's campus.

In addition, the service broker knows nothing about the domain dependent nature of the experiments. It forwards an opaque object from the lab server to the user's lab client describing the current lab configuration. When the user submits an experiment specification, it is forwarded to the lab server as another opaque object, and the results are returned as a third. The only part of an experiment that the service broker understands is a metadata description of the experiment that can be used to search for and retrieve old experiments.

III. Basis for Developing ADLab

The digital electronics aspect of the present electronic and electrical engineering curriculum at Obafemi Awolowo University has been deficient for some time. The curriculum has not been reviewed in close to a decade and it does not reflect the rapid changes that have been witnessed in high chip-count digital electronic in the last two decades. Specifically, although the design of application specific ICs are treated under the microelectronics aspect of the curriculum, programmable logic devices are not (figure 2 shows the spectrum of digital

electronic devices). In the laboratory however, there are no modules that deal with CPLDs and FPGA, which have emerged as extremely important tools in digital electronics practice in recent years. Introducing these into the laboratory work of the undergraduate students would normally require changes to the curriculum, a process which takes a long time to complete.

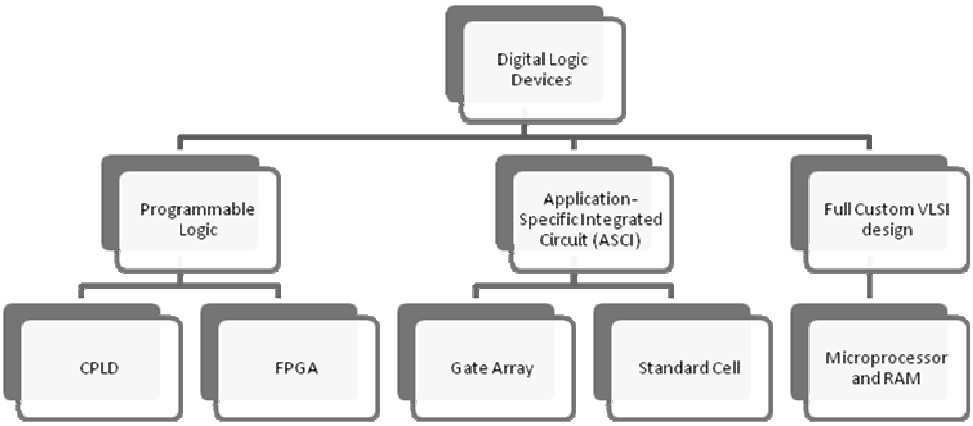

Figure 2: Device Technologies used for Implementing Digital Systems

Therefore, any solution chosen to introduce FPGAs and CPLDs to electronic and electrical engineering students in OAU needed to be effective while recognizing the fact that student's theoretical basis might be shaky because the relevant theory are not yet covered by the curriculum. One approach that was considered was to buy development boards for students to use in the laboratories. However, the poor theoretical basis and lack of understanding of hardware description languages meant that students might find working on such development boards confusing. In addition, the large class sizes meant that a large number of such boards would need to be acquired. Coupled with the fact that their poor bases meant students would spend a lot of time on the boards in guess-work mode, such boards would get damaged very quickly.

Rather than follow that path then, a decision was made to develop a remote laboratory based on a single development board. The feeling was that the remote laboratory offered the best way to allow the students work on the development board while hiding complexities that the student did not need to be exposed to. The remote laboratory would allow the student to be exposed only to the aspects of digital electronics that is applicable to him. Hence for example, a sophomore student would work with a metaphor that exposed simple logic operations of the FPGA, while a final year student would work with a front-end metaphor that showed the backed physical device as an FPGA. Another advantage of implementing the FPGA part of students' laboratory work as a remote laboratory was that this allowed the single board to be used by many students without problems of scheduling and without fears that the board would be damaged.

IV. Overview of ADLab Architecture

ADLab was developed on the iLab batched architecture. As mentioned earlier, the service broker tier of an iLab is a generic application that can be downloaded and reconfigured. Therefore, only two tiers were developed for ADLab:

- A lab client and

- A lab server.

A diagram of the various parts of the system is presented in Figure 3. A browser-based web client forms the front end of the system. This front end interacts with the system under test through the service broker. The backend system is built around an FPGA, and also contains web services, the experiment execution engine, QUARTUS software, and data acquisition (DAQ) software.

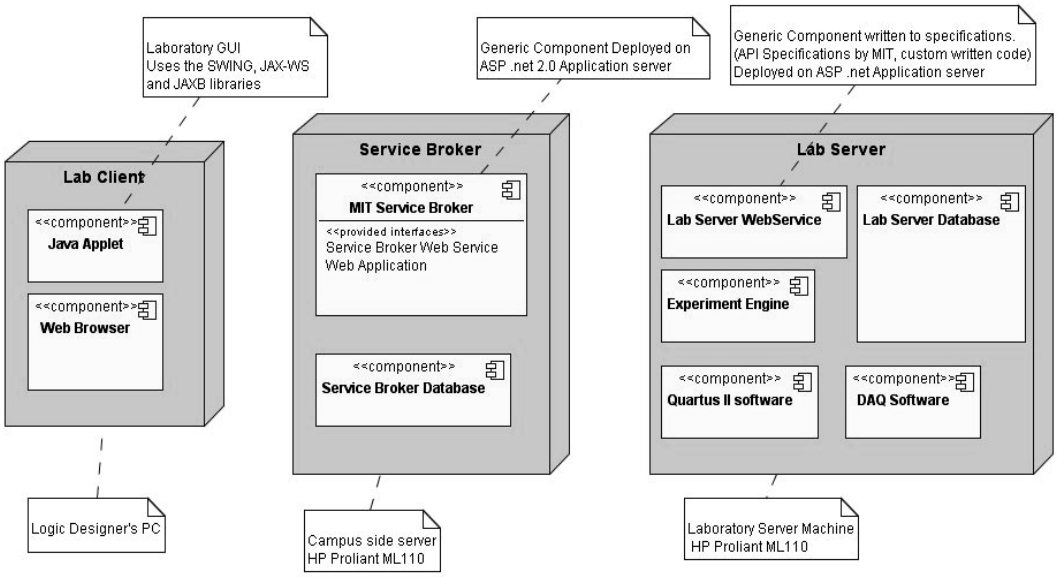

Figure 3: ADLab Architecture Overview

V. The Client

The lab client essentially forms the bulk of the HCI of ADLab. However, there is some interaction with the service broker's web application interface prior to using the lab client. This is because, before using the lab client, the user has to log in to the service broker and initiate a lab session. It is only after this that the client can be launched (also through the service broker). Upon launching the lab client, the user's activities in the experiment are performed through the lab client exclusively.

The lab client was developed as a java applet. It is guaranteed to run on any machine with the java virtual machine installed. While there have been cases where C# WinForm clients have been used in the past, the language has not gained anywhere near the level of platform independence that Java enjoys. The lab client delivers its functionality by the synergistic interaction of custom written code and freely available code libraries. These include (amongst others):

- Java API for XML Binding (JAXB): used to serialize and de-serialize object to and from XML.
- Java API for XML Web Services (JAX-WS): Used to interact with the service broker web service.
- Swing: A class library for building java graphical user interfaces.

Figure 4 shows the main screen of the lab client while Figure 5 shows a UML class diagram of the lab, showing the extent of the user's interaction with the system via the lab client.

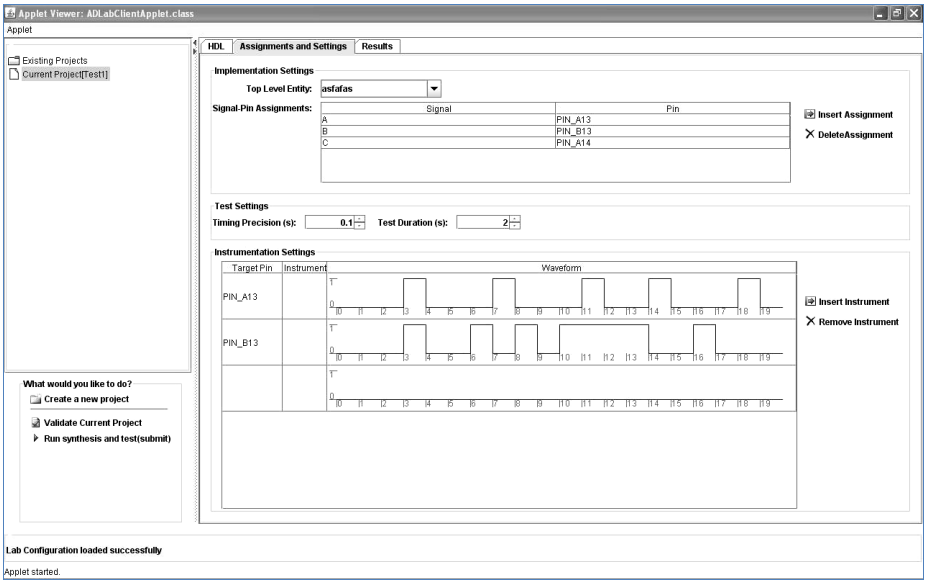

Figure 4: The main screen of the lab client

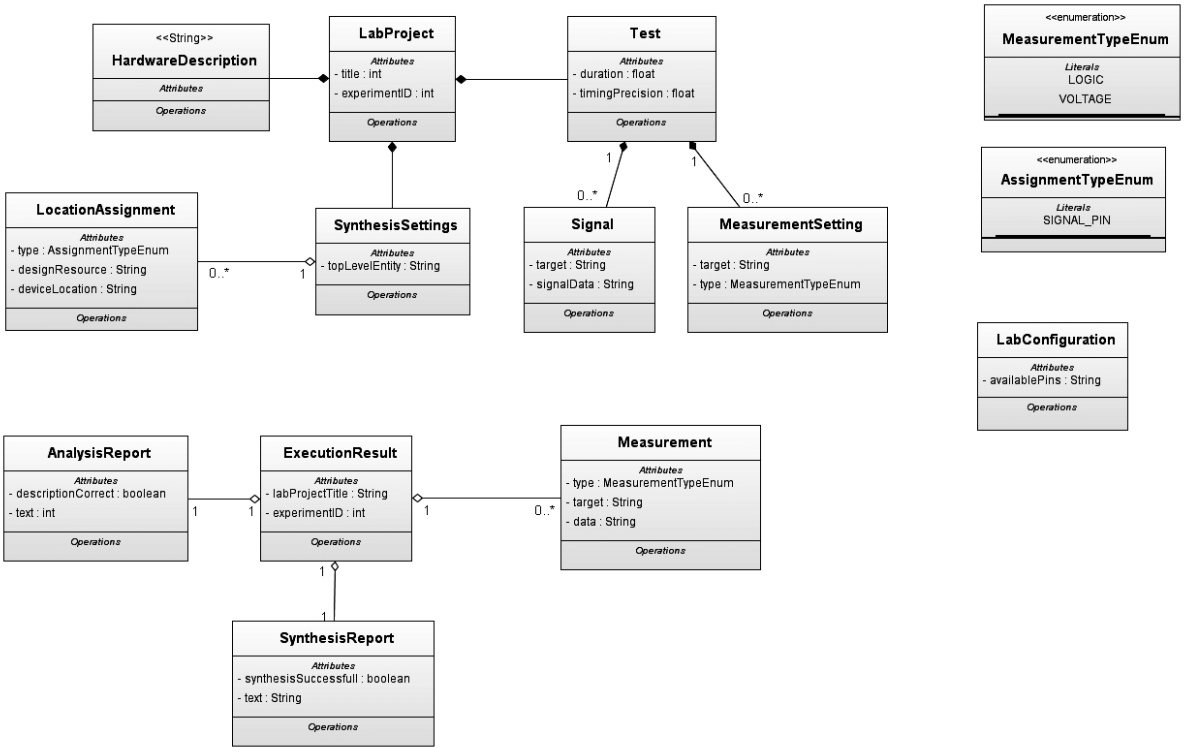

Figure 5: UML Class Diagram of Lab

One important property of the Altera DE1 development board is that it has a number of LEDs and other input/output options by which students can get an idea of what is happening within the programmable logic devices on the board. In developing the client, it was deemed essential to find a means to make this functionality available online. The obvious solution was to use a webcam. However, a single webcam stream requires around 64kbps and with the relatively low bandwidth available at Obafemi Awolowo University (the university has an uplink of 2048kbps), a few ADLab streams might cause congestion on the uplink. To solve this problem, the web video streaming service MOGULUS is used. A single 64kbps stream is broadcast to MOGULUS. A MOGULUS script is them embedded into the client, and this

allows the client display the video feedback from ADLab without causing bandwidth problems.

VI. The Lab Server

The ADLab server tier consists of a number of components:

- A web service
- An experiment engine
- Altera DE1 development board
- Ancillary hardware and software elements

The ADLab ***web service*** is written to specifications which were developed for the iLab architecture. The specification defines an Application Programming Interface (API). The function calls in the API allow for the submission of experiments, and associated functionality (see literature review). Some server code implementing this specification is freely available on the Internet. However, this code was developed specifically for the Weblab and so is not generic. For ADLab, the web service had to be written from scratch. It was developed using the C# programming language, and deployed on the ASP .net application server.

The web service's functionality includes

- Collecting and queuing up experiment requests, for which it uses a database.
- Providing information on the status of queued experiments
- Providing information on the status of the lab server itself
- Validation of experiment specifications before executing them

The ***Experiment Engine*** is a program that checks the experiment queue (database) continually. In the event that an experiment is in the queue, it executes that experiment and stores results in the database. It uses other software and controls the synthesis and DAQ processes for each experiment. The experiment was written in C# and uses the tool command Language (TCL) to automate the use of the QUARTUS II software executables. The experiment engine analyzes hardware descriptions and in the absence of problems, synthesizes hardware on the FPGA. It also uses the DAQ subsystem to take measurements. Figure 6 shows a screen capture of the experiment engine.

The database that serves both the web service and experiment engine contains stored procedures written in transact-SQL. These stored procedures provide an abstraction layer of data access procedures which reduce the regeneration/embedding of boiler plate SQL code in C# code.

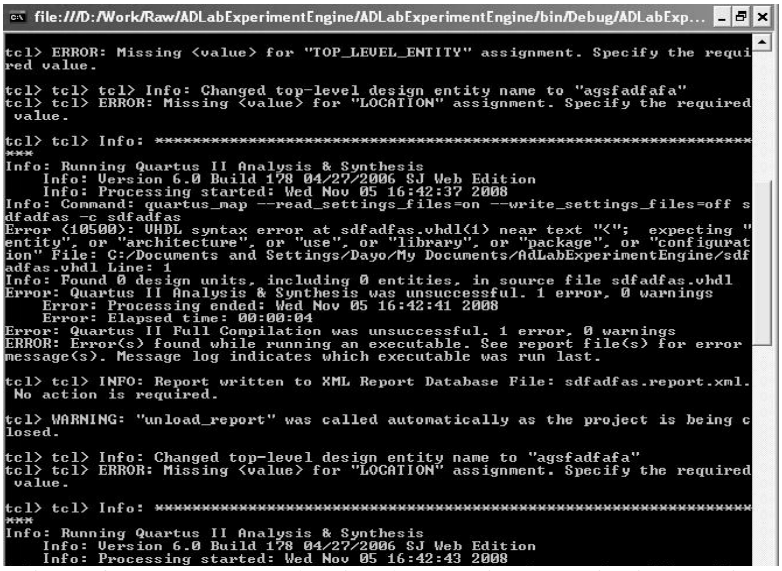

Figure 6: Screen capture showing the Experiment Engine in operation

The DE1 board features a state-of-the-art Cyclone® II 2C20 FPGA in a 484-pin package. The original purpose of the board was to serve as a vehicle for learning about digital logic, computer organization, and FPGAs. The board offers a rich set of features that make it suitable for use in a laboratory environment for university and college courses, for a variety of design projects, as well as for the development of sophisticated digital systems. All important components on the board are connected to pins of this chip, allowing Users to control all aspects of the board's operation.

The pins in the Expansion headers (figure 7) are used to supply or measure signals on the FPGA board. The USB blaster port is used to connect the DE1 board to the lab server. It is through this port that the FPGA is programmed.

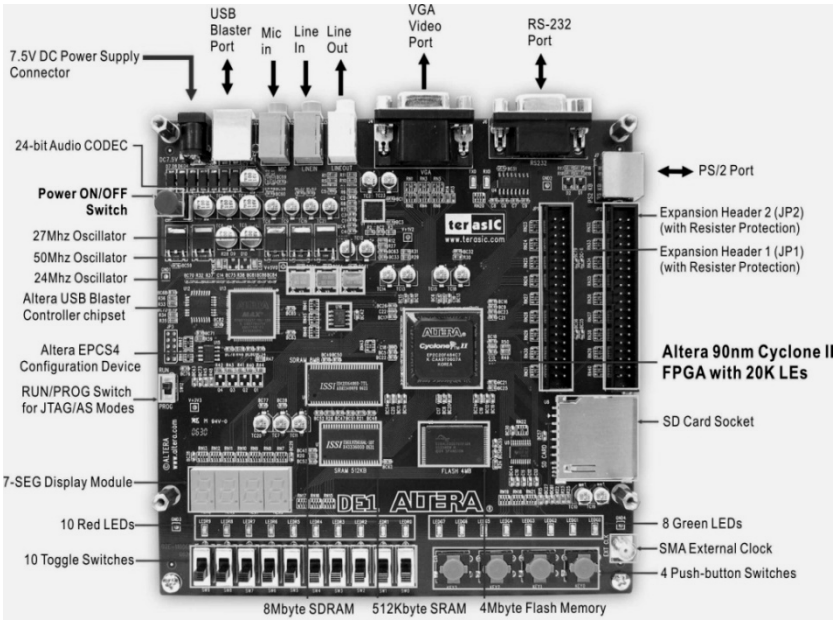

Figure 7: Altera DE1 FPGA development board

Ancillary Hardware and Software

- ***ASP.Net application server***: A combination of IIS and the .Net framework is an application server. It is to this application server that the service broker is deployed on the service broker machine. The ASP .net application server is the deployment environment for the lab server web service.
- ***Database Management Software (DBMS)***: This is used to manage and secure the databases used by software components. Microsoft's SQL Server 2005 is used on the lab server and on the service broker. The service broker software stores user identities, authorization, experiments, experiment results and other data in a database hosted on the service broker machine. The lab server also store's service broker credentials, experiment records and other data in a database hosted on the lab server machine.
- ***Altera's QUARTUS II software***: The QUARTUS II software provides functionality for verifying VHDL code, synthesizing hardware and generating reports about the analysis and synthesis. All experiment engine interactions (asides from measurements) with the FPGA board are done through the QUARTUS software.
- ***Data Acquisition (DAQ) Subsystem*:** The DAQ subsystem consists of a National Instruments (NI) USB 6251 multifunction DAQ. NI also provides the *DAQmx* software for programmatic control of all its DAQ cards. The experiment engine utilized *DAQmx* to control the DAQ card, and it is by this means that signals on the DE1 board are measured.

VII. Pedagogical Value of ADLab

ADLab presently allows students to create a VDHL listing and to use this listing for synthesis of digital circuits on the Cyclone® II 2C20 FPGA. Synthesis settings including the pin signal assignments, selection of top-level entity and any other relevant settings that are required for the synthesis process are also specified. All this data is used by the execution engine to synthesis the 2C20 through QUARTUS.

As part of the exercise, students specify details of tests to be carried out, including test duration timing precision, all signals and measurements required. These tests are carried out through the NI DAQ card. Two categories of results are available to the student. For synthesis that uses the optoelectronic devices on the DE1 board as output, the student is provided with video feedback through a MOGULUS broadcast stream. Figure 8 shows a view from the webcam. Apart from video feedback through the webcam, more detailed textual results are also available and these include the results of the lab server's activities like VHDL analysis report, Synthesis report and measurements taken at FPGA pins as specified in the Lab Project.

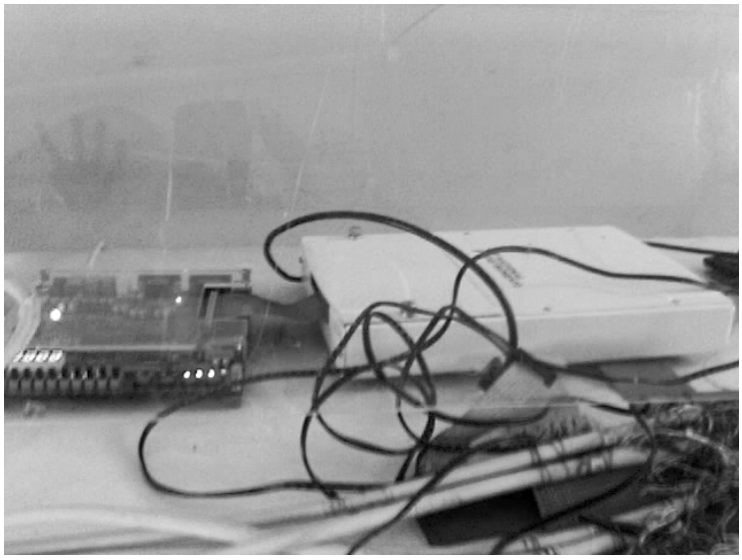

Figure 8: Webcam image showing the DE1 Board

VIII. Testing and Results

The ADLab is presently undergoing tests and was used in a credit awarding course for the first time in the first semester of the 2008/2009 academic session. Twenty five students offering the course EEE407 ("pulse and digital techniques") were randomly chosen and given an introduction to FPGAs and VHDL. They were then asked to carry out an experiment that required building simple logic structures with VHDL. Assessment of the laboratory followed the same process used in assessing the OAU OpLab[9]. Students were presented with 15 statements and asked to show their level of agreement or disagreement with each statement by assigning a number between 1 and 5 as follows:

1=I strongly agree
2=I agree
3=I do not know
4=I disagree
5=I strongly disagree

The fifteen statements were:

1. This exercise has been useful
2. Virtual labs can never be as effective as real labs
3. Virtual labs will eventually replace real labs
4. Using the lab was enjoyable
5. The interface is intuitive
6. There aren't enough parameter variations to make this worthwhile
7. Every course should have a few iLabs associated with it
8. Using the lab was **not** intellectually stimulating
9. The lab was slow
10. The interface is confusing
11. I cannot relate this to real life
12. I believe I was actually working with real devices through the internet
13. The lab has no relevance to the coursework
14. Using this lab has made me think about and understand some things. I would not have been able to do that from just our lectures or textbooks.
15. This lab should be used next year

As expected, the student's responses showed generally positive views about the laboratory (Figure 9). We however recognize that too much cannot be read into the results, considering

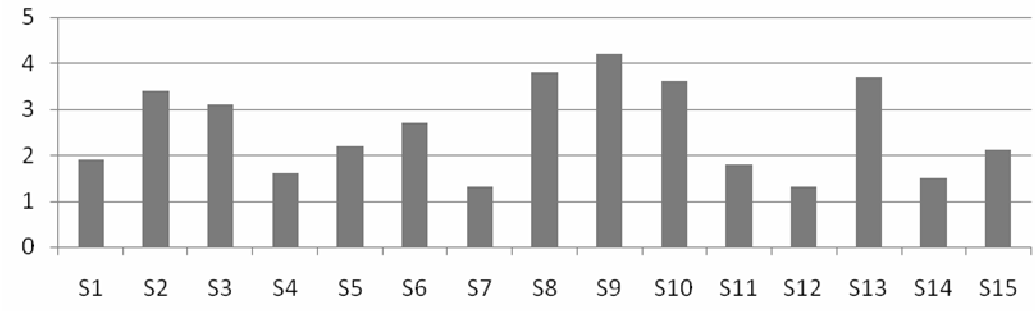

Figure 9: Means of students' ranking of assessment statements

the fact that such a small sample size was used. We expect that in the next few semesters, as larger numbers of students use the system, a better picture of the lab's performance will be obtained. In spite of this however, it is still interesting to note that students felt the lab was effective (statement 14). The fact that they were not properly introduced to FPGAs and VHDL before the test was reflected in their ranking of statement 11 (they generally could not relate the lab to real life). Statement 7 shows that students of Obafemi Awolowo University seem to be getting more comfortable with the idea of using remote laboratories.

Another way of assessing the effectiveness of laboratory experiments is by assessing the performance of students in related parts of their coursework. Even though it was unlikely that any visible difference in the students' performance could be noted after such a short exercise, we still compared the performance of students who took the ADLab experiment to the class averages. No statistically significant differences were noted, but this is an approach that we will like to explore further for assessing labs in the future.

IX. Conclusion

An iLab has been developed and deployed at Obafemi Awolowo University to allow students carry out experiments on programmable logic devices. The lab is built around an Altera DE1 development board and uses a Java applet client. The lab has been tested and has been used in a credit-awarding course once. From the positive responses of the initial testers, it is believed that the system would be very useful in augmenting the digital electronics part of the electronic and electrical engineering curriculum of the university.

Bibliography

1. Feisel, L., and Peterson, G.D. (2002): A Colloquy on Learning Objectives for Engineering Educational Laboratories, Proceedings of the 2002asee Annual Conference And Exposition, Montreal, Ontario.
2. Del Alamo, J. A., L. Brooks, C. Mclean, J. Hardison, G. Mishuris, V. Chang, And L. Hui. (2002):The MIT Microelectronics Weblab: A Web-Enabled Remote Laboratory For Microelectronics Device Characterization. World Congress On Networked Learning In A Global Environment, Berlin (Germany), May 2002.
3. Gillet, D., Salzmann, C., Longchamp, R., & Bonvin, D. (1997). Telepresence: An Opportunity to Develop Practical Experimentation in Automatic Control Education. European Control Conference. Brussels.

4. Benmohamed, H., Leleve, A., & Prevot, P. (2004). Remote Laboratories: New Technology and Standard Based Architecture.
5. Del Alamo, J.A., Chang, V., Hardison, J., Zych, D., and Hui, L.(2003): An Online Microelectronics Device Characterization Laboratory with a Circuit-like User Interface, Proceedings of the International Conference on Engineering Education, Valencia, Spain.
6. Nedic,Z., Machotka, J., and Nafalski, A(2004): Remote Laboratories Versus Virtual And Real Laboratories, 33rd ASEE/IEEE Frontiers In Education Conference, Boulder, Colorado.
7. Harward, J., Del-Alamo, J., Choudhary, V., DeLong, K., Hardison, J., Lerman, S., et al. (2004). iLab: A Scalable Architecture for Sharing Online Experiments. International Conference on Engineering Education.
8. Colton, C.K., Knight, M., Khan,R.A., Ibrahim, S., and West, R. (2004): A Web-Accessible Heat Exchanger Experiment, in: INNOVATIONS 2004: World Innovations in Engineering Education and Research. Ed. Aung,W., Altenkirch, R., Cermak, T., King, R.W., and Ruiz,L.M. pp 93-106. Begell House Publishing, Arlington, VA.
9. Ayodele, K. P., Kehinde, L.O., Jonah, O., Ilori, O., Ajayi, E.O.B., Osasona, O.O. (2008): Development Of An Operational Amplifier Virtual Laboratory Based On ILab Architecture And NI Elvis. Proceedings of the ASEE Annual Conference and Exposition, June 2008, Pittsburgh.