

BLACKBOARD-BASED DIGITAL HARDWARE DESIGN USING FPGAS

Reza Hashemian¹ and Chandi Pedapati²

¹ Northern Illinois University, DeKalb, IL; Email: reza@ceet.niu.edu

² Northern Illinois University, DeKalb, IL; Email: chandiped@yahoo.com

1. INTRODUCTION

One of the challenges that still remain to deal with in distance learning is how to teach an experimentally intensive course in engineering. An example is a digital hardware design course, taught to the electrical engineering students. Here the students need to have access to both software and hardware tools necessary to practice design, synthesize, implement and test their digital circuits in the classroom environment. The problem can simply be stated as follows: How the students of a hardware design class can continuously interact with the hardware facilities and the equipment without physically moving to the lab? This problem becomes even more challenging when the class is held off campus, where for the most part, the hardware design facilities cannot be moved to the location.

The Internet access seems to be one and the most reasonable means to achieve this goal. In this situation a group of students, or the students with the instructor, can work on a design in a cooperative manner. This situation cannot simply be handled by the conventional methods alone. Consider, for example, the design and construction of digital hardware by using Field Programmable Gate Arrays. This process requires FPGA chips, development boards with power supplies, the design tools, and the instructions for pin assignments. The difficulty is not only preparation of tools and the hardware equipment but the sequence of steps that are needed to be taken. That is, in the sequence of steps, needed for the design and construction, each step is highly dependent on previously completed steps, and any mishandling and errors in the processes must be reported. Testing and debugging the chip is even harder, because it is highly interactive and completely rely on a successful implementation.

One way to approach the problem is to use the Blackboard, as an Internet-based instructional media, and develop techniques that the students could use the local tools to design their projects as much as possible share their designs with other students, and finally report their results to the instructor, all through the Black Board. This is usually known as batch processing. However, the problem gets more complicated when the students need to implement their design and remotely configure and test it. Our proposed method suggests constructing an on-line design platform. An on-line design platform has the following criteria. a) It provides a communication link between the instructor and the students. b) It creates a common library of components for digital hardware designs such that the students and the instructor can share their designs. c) It

contains a hardware platform using FPGAs for prototyping digital designs. d) Finally, it creates an environment such that the I/O interconnections and testing of the hardware can be performed through the Internet.

This new methodology can significantly increase the performance and efficiency in number of applications such as:

1. Teaching experimental and lab-intensive courses through internet.
2. To perform group work on designs, where the partners are distance away.
3. To allow the external reviewers and evaluators to test and verify the hardware.
4. To remotely monitor the performance of systems located in distanced sites.

2. THE PROBLEM STATEMENT

Our particular concern here is to be able to teach courses on digital design using Field Programmable Gate Arrays (FPGAs) for hardware implementation. These courses are normally project oriented, and are totally experimental and hands on. To complete their term projects the students in these classes usually need to go through the following *design steps*:

1. Project description, hardware device specification and preparation.
2. Design entry through schematic capture or Hardware Description Language, such as VHDL, or Verilog HDL. A design net-list is generated in this step.
3. Functional simulation of the design project. This step also needs design tools to functionally simulate and verify the design.
4. Design synthesis. This is a major step in creating an optimal hardware and to create the digital system from the functions or its behavioral description. This is done in several smaller steps started by translating the functional behavior of the design to actual gates that exist within a vendor's library. This step is also tool-based.
5. Design implementation. This is also a crucial step in the process, and it needs i) a library of components, ii) types of FPGA chips, iii) a user's constrained file (.ucf), and iv) software tools to implement the design. Following several steps including pin-assignments, and place and route a configuration file *design.bit* is created for hardware (re)configuration. It is also recommended to verify the design through timing simulation before sending it for hardware configuration. This simulation is similar to that in step 3, except the actual timing delays through the gate paths are present here, and hence, the simulation results are closer to the experimental test results.
6. Hardware configuration. At this step the *design.bit* file, created in step 5, is downloaded into its final destination, an FPGA chip on a hardware board.
7. Finally, the designed chip on the board must be verified and tested by sending input signals to the board and verifying the responses. In case the results do not match with the simulation results the system must allow further verification and debugging.

Steps 2 through 5 require software tools necessary to design, and just like any other programming-intensive course, these steps can simply be performed by using design tools in a local environment. For example, here at NIU predominantly Xilinx ISE tools are used for design

entry, logic synthesis, and design implementation. For simulation purposes ModelSim by Mentor Graphics, Inc, and some other software are used.

However, to fulfill steps 6 and 7 the students need access to certain hardware facilities that allow them to download their design on the FPGA hardware, and do the verification and testing. These facilities are not typically available in the class room, or not at least for each and every student in the class. The problem may then be stated as follows: Following steps 1 through 5 of the design procedure that are locally pursued, how can each student perform steps 6 and 7 of his/her project and get contacted to a remote laboratory facilities and get all results to report his/her progress?

3. USING THE INTERNET ACCESS

A new methodology is proposed here to use the Internet access for teaching experimentally intensive courses in engineering. In particular, we are going to offer a methodology to use Blackboard as an internet-base instructional forum and develop techniques to design and construct digital hardware by using Field Programmable Gate Arrays. Evidently, certain steps of the design are conveniently carried out through the Blackboard tools. These steps include design exchange, group discussions, problem solving and reporting. This is particularly true when the hardware design tools are locally available and only the results need to be reported through the Internet (known as, batch processing). However, in cases of highly interactive processing through the internet, where a group of students, or the students with the instructor, are working on a design the situation cannot simply be handled by the conventional methods. The objective here is to handle the cases that the hardware facilities are remotely located elsewhere, and the hardware implementation, configuration and testing needs to be performed through the Internet, with or without an on site technician's assistance.

Soft I/O Interface System

Figure 1 shows a schematic diagram of a *soft I/O interface system*. This system is to remotely perform both software and hardware design procedure through the Internet. The system is intended to be interactive.

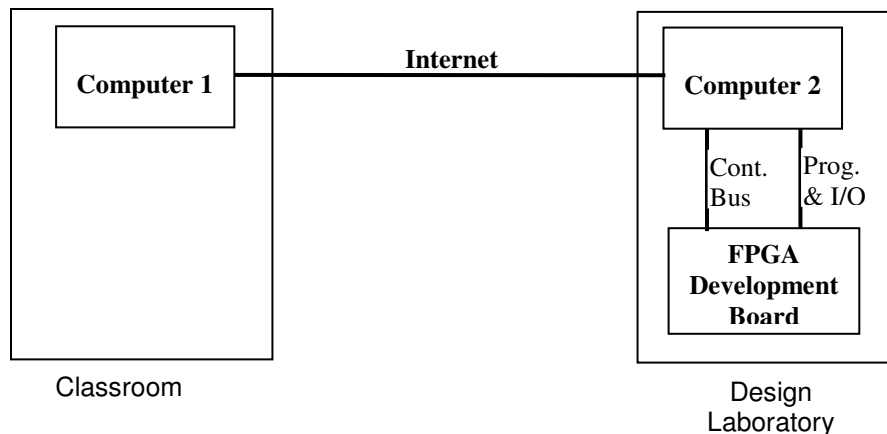


Figure 1: Schematic diagram of a soft I/O interface system connecting the classroom to the lab facilities.

Both Computers 1 and 2 are capable to run the design tools, and basically perform all design steps from 1 to 5, independently. The advantage that Computer 2, sitting next to the development board in the Lab, has over Computer 1 is its proximity to the FPGA board and the test facilities. This allows the designer to do the hardware portion as well as the software portion of the project at the same location.

The other possibility, that is more of our concern, is to proceed to perform the design on Computer 1. Most parts of the design, from step 1 to step 5, are locally accomplished in this scenario. However, to perform the hardware implementation, testing and I/O interfacing, i.e., steps 6 and 7, the ideal solution is to approach Computer 2 in the Design Lab, and through it perform the hardware reconfiguration and test on the FPGA chip. One possible solution in this situation is to use the Internet connection to transfer I/O files, programs, and the instructional codes between two computers, as shown in Fig. 1. After receiving the program and the instructions Computer 2 separates the control signals from the program and I/O files and sends them in to two different channels to the FPGA board. For example, it is possible to use either a parallel port or a USB terminal to (JTAG) configure the hardware and to perform I/O interconnection, and to use a serial port to handle the control (instructional) signals.

The followings are the *design stages* that a designer (a student or the instructor), in the classroom, must get through to accomplish his/her design and the hardware implementation (steps 6 and 7, as mentioned earlier).

1. The design is carried out from the design step 1 to 5 on Computer 1 with design tools being available to the designer on site, and a *design.bit* file is created as a result.
2. With a command given by the designer, Computer 1 connects to Computer 2 through the Internet. Computer 1 stays at the transmitting mode while Computer 2 goes to the receiving mode. The *design.bit* file is sent to Computer 2. While at the receiving mode no other computer can access Computer 2.
3. Computer 1, still being in the transmitting mode, sends a *power on* signal to Computer 2 to turn on the FPGA board in the lab. Upon receiving the signal, Computer 2 asserts a signal to the board through the *control bus*, and turns the FPGA board on. Note that this step needs some prior preparation, to be able to establish the signal through Computer 2 to the FPGA board.
4. Next, after receiving a download signal from Computer 1, Computer 2 goes into a complete cycle of hardware reconfiguration (use of iPACT from Xilinx, for example), and downloads the *design.bit* file into the FPGA. This completes the step 6 of the design procedure.
5. Similar procedures can be taken to verify the hardware and test the design on the board through the Internet access. However, this step can get somewhat involved if the objective is a fully automated and unassisted operation. In our proposal we offer two different routes to fulfill this step:
 - *Assisted Design*: Have a Lab assistant, and have him/her to perform the reconfiguration, hardware verification and test on the chip in the Design Lab, and send the results and reports to Computer 1 through the Internet.
 - *Unassisted Design*: Design and establish an instructional-based system and programming such that the designer sitting by Computer 1 could remotely run the

necessary steps in the Design Lab and have the results (outputs, and reports) sent back to him/her through the Internet.

This completes the step 7 of the design verification and test.

6. Now, the entire system resets, and it gets ready for the next design order.

4. A CASE STUDY

To demonstrate the procedure an example course, **Digital Hardware Design Platform**, is developed using the Internet access and the Blackboard instructional tool. A web site has been developed for the course and it has been linked to the NIU Blackboard instructional tool (at http://webcourses.niu.edu/bin/common/content.pl?action=LINK&render_type=DEFAULT&file_id=_336776_1). Figure 2 shows the banner of the site, and a front web page is given in Appendix A.



Figure 2: The banner of the web site the course **Digital Hardware Design Platform**.

The web site performs the following features:

- It contains all regular course materials such as the syllabus, course coverage, project lists, and other announcements.
- It introduces the main providing companies of the FPGA chips, the Design software, and other tools and materials used in the course.
- It introduces the design, simulation and implementation tools available for students to do their project design. Documentations such as tools manuals, tutorials, and component data sheets are provided for students.
- Students are guided to search for other sources and related sites for example of designs, and connect to design libraries of their interest.
- The students are given step by step instructions to go through their design project and follow the design stage 1 on their site (class, lab, or even in a remote site). They are also instructed to go through stages 2 to 6 in the design stages when a Lab assistant is available carry to on the implementation and test in the Design laboratory (see Fig. 1).

5. CONCLUSION

A new method is proposed to use Internet access for teaching experimentally intensive courses in engineering. By using the Blackboard instructional tool it has been shown that a step by step design procedure can be carried out through the Internet access to assist students of a hardware design course to remotely control and perform their design, including hardware construction and test. The later part could be done with a lab assistant in the Design Laboratory. The students are

able to locally go through all stages of their design including design entry, simulation, synthesis and implementation. But to gain experience with hardware reconfiguration and testing the students can use the Internet access to remotely communicate with the Design Lab. A site is developed for a course in hardware design using FPGAs that can be accessed by the students. Presently, the *assisted design* procedure is completed and it is practiced.

REFERENCES

- [1] Ciletti, Micheal D. Advanced Digital Design with Verilog HDL. Upper Saddle River, New Jersey: Prentice Hall Inc. 2002.
- [2] Mano, Moris M. Digital Design. Upper Saddle River, New Jersey: Prentice Hall Inc. 2002.
- [3] Palnitkar, Samir. VerilogHDL. Mountain View, California: SunSoft Press, 1996.
- [4] Web Authoring FAQ: Hyperlinks. Darin McGrew. Fall 2004. Web Design Group.
<<http://www.htmlhelp.com/faq/html/links.html#named-anchor>>.
- [5] Product and Services, Xilinx, Inc. 2005, <http://www.xilinx.com/xds/>

Appendix A



Hardware Design: Programmable Logic

- INTRODUCTION**
- [Purpose](#)
- [Background](#)
- [Prerequisite](#)
- [Applications](#)
- OVERVIEW**
- SAMPLE**

Programmable Logic Design with Verilog HDL and FPGA

Purpose of this Site

Consider this site an aid to guide you through designing your own programmable logic device.

Background

Digital design of programmable logic has become increasingly valuable in the past 25 years. Before the development of these automated processes there was SSI (early 60's), small-scale chip integration, with very small gate counts. Next in the evolution was medium scale integration, MSI, with gate counts in the hundreds. Circuit design continued to increase in complexity with LSI, large scale integration, with gate counts in the thousands. In response to the increasing difficulty of manually verifying functionality automated circuit and logic simulation was developed. However, circuits were still tested on bread boards, and layouts were still done by hand. At the onset of VLSI, very large scale integration, involving hundreds of thousands and millions of transistors, verification of circuits on bread boards were simply not possible. Logic simulators were then created to verify circuits before they were fabricated on chips. HDLs, the standard language to describe digital circuits as well as simulators, such as VHDL and Verilog HDL, were developed. The FPGA chips were then created, which are reconfigurable logic devices, making development and implementation of electronic systems fast and readily available.



an FPGA board

To continue you will need to study

- Combinational Logic: [Click here for a quick REVIEW](#)
- Verilog HDL, or VHDL code: [Click here for Sutherland Reference Guide](#)
- Software
 - Xilinx