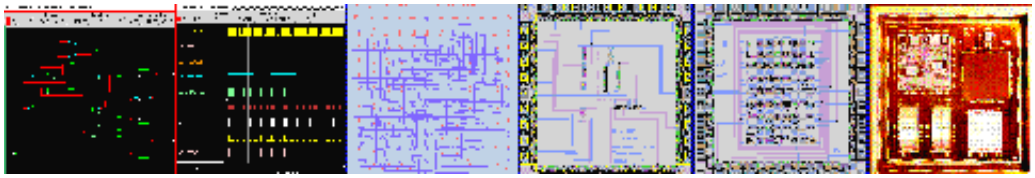
ECE 551 -- Designing Application-Specific Integrated Circuits



ECE 551 (Section# 48023). Tues/Thurs 12:40 p.m. - 1:55 p.m. in 510 Ferris Hall

Don Bouldin, Ph.D.
Prof. of Electrical and Computer Engineering
419 Ferris Hall
1508 Middle Drive
University of Tennessee
Knoxville, TN 37996-2100
TEL: (865)-974-5444
FAX: (865)-974-5483
Email: dbouldin@tennessee.edu
http://www.eecs.utk.edu
http://www.eecs.utk.edu/people/faculty/emeritus/bouldin/
http://web.eecs.utk.edu/~bouldin/courses/551/overview.html

FPGA & ASIC Synthesis

This project-oriented course will present an overview of the design of field-programmable gate arrays (FPGAs) and application-specific integrated circuits (ASICs). Each pair of students will capture a design using a hardware description language (VHDL) and then use synthesis and automatic placement and routing software to implement the design using multiple technologies (Altera and Xilinx). LINUX workstations will be used extensively.

Goals of ECE 551:

- To present an overview of FPGAs and ASICs that are suitable for tasks which cannot be executed efficiently by a general-purpose microprocessor.

- To illustrate capturing a design in a technology-independent means using a mix of levels (behavior and structure)
and then to map the synthesized result into several technologies which can be compared.

- To provide an in-depth project using FPGAs that will involve architectural tradeoffs and simulation.

- To reinforce the lectures and discussions with experience using computer-aided design tools.

- To develop human communication skills via a team project requiring both written and oral reports.

What's New ?

Accessing a Remote Host (Putty & Xming)

Lab Equipment

Our New Text for $99 (John Wiley)

Author's Companion Site for Our New Text

Our Old Text On-Line for Free

Syllabus

Students

Course Overview (pdf file)

Overview Slides (color pdf)

Overview Slides (handout b/w pdf)

Synthesizing Microelectronic Systems (197 KByte pdf)

Homework_1 - LOGIN, EMAIL and WEB PAGE

Homework 2 -- Using ModelSim

Homework 3 -- Using the Spartan3 Demo Board and Xilinx ISE Software

Homework 4 -- Animating Logic Simulations

Homework 5 -- Graphics <---> HDL

Homework_6 -- Targeting Xilinx and Altera; Using Asserts; Coverage

Homework_7 -- FPGA Design Using Graphical Tools

Homework Status

Projects

Project Checkoff Appts

Final Exam

VHDL Examples (our text)

VHDL Tutorial Examples

OpenCores

PROTECTED WEBSITE

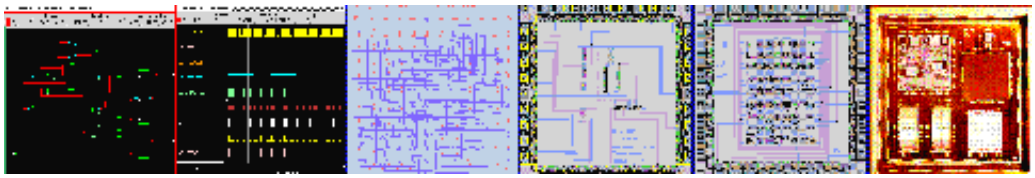
Handouts

---

dbouldin@tennessee.edu

ECE 551 -- Syllabus -- Fall 2011

Revised 03/31/2011 by D. Bouldin



12:40 p.m.-1:55 p.m. in 510 Ferris Hall

Session	Date	Topic
01	08/18/Thu	Course Overview; Microelectronic Systems Design
02	08/23/Tue	HW1; Remote Access; Role of Synthesis
03	08/25/Thu	HDL Examples; HW2 (GUI and CLI)
04	08/30/Tue	VHDL for Combinational Logic
05	09/01/Thu	VHDL for Controllers
06	09/06/Tue	Structural VHDL
07	09/09/Thu	Design Methodology; HW3
08	09/13/Tue	Creating & Integrating IP Blocks
09	09/15/Thu	Validation and Verification
10	09/20/Tue	Asserts; Quality IP Blocks
11	09/22/Thu	Global Design
12	09/27/Tue	No Class (miss# 1)
--	09/29/Thu	No Class (Fall Break)
13	10/04/Tue	Animating Logic Simulations; HW4
14	10/06/Thu	HDL2Graphics; Graphics2HDL; HW5; Testbenches; HW6B; HW6B-Solution
15	10/11/Tue	Project Descriptions and Assignments
16	10/13/Thu	FPGA Floorplans & Interconnect
17	10/18/Tue	Retargeting & Migration; HW6A(altera-fit not working yet)
18	10/20/Thu	Reconfigurable Computing;Text Figs; pong-modified (vhd)
19	10/25/Tue	SOPC; Platform Design
20	10/27/Thu	Model-Based Design; HW7
21	11/01/Tue	VHDL-Verilog Examples
22	11/03/Thu	Need for Test; Failure Analysis
23	11/08/Tue	Designing Testable ICs
24	11/10/Thu	BIST; Boundary Scan

25	11/15/Tue	FPGA Optimizations
26	11/17/Thu	CoreGen; DPRAM; BRAM-Gen; Xilinx-ARM
27	11/22/Tue	No Class (miss# 2)
--	11/24/Thu	No Class (Thanksgiving)
28	11/29/Tue	Team Presentations; Project Checkoffs
29	12/01/Thu	HW & Reports due at noon
30	12/08/Thu	Final Exam (12:30 pm--2:30 pm)???

Grading:

- 05 % -- Project Proposal
- 10 % -- Project Presentation
- 15 % -- Project Report
- 25 % -- Project Demo
- 25 % -- Homework
- 20 % -- Final Exam

Project Report:

A project report consists of a PPT with notes.

Final Exam:

No reference material or scratch paper may be used during the Final Exam which will be two hours in length. It will cover the topics discussed in class.

Homework:

Students may consult others about possible solutions but may not copy from one another directly.

Late Work:

Each assignment is due at the beginning of class. Assigned work that is received late but within the subsequent 24 hours will be discounted 30%. Thereafter, each 24-hour period will necessitate an additional 10% penalty. Saturday and Sunday are considered one period. No work will be accepted after 12/01/Thursday at noon.

Incompletes:

I adhere strictly to this statement from the UTK catalog: "I" indicates that the student has done satisfactory work in the course, but because of circumstances beyond his control has been unable to finish all requirements. It is not to be given to enable a student to do additional work to bring up a deficient grade.

Test Hint:

EXPERIENCES TEACHING SYNTHESIS OF FPGAS AND TESTABLE ASICS

Donald W. Bouldin
Electrical Engineering
University of Tennessee
Knoxville, TN 37996-2100
dbouldin@utk.edu

ABSTRACT

Microelectronic system designers are increasingly capturing their designs using hardware description languages such as VHDL and Verilog. The designs are then most often synthesized into programmable logic devices such as field-programmable gate arrays (FPGAs). This approach places the emphasis on high-level design which reduces time to market by relying on synthesis software and programmable logic to produce working prototypes rapidly. These prototypes may then be altered as requirements change or converted into high-volume mask gate arrays or other application-specific integrated circuits (ASICs) when the demand is known to be sufficient. These ASICs, however, must be designed to be testable to screen out those with manufacturing defects. Hence, scan logic must be inserted, test vectors generated and fault grading performed to ensure a high level of testability. Experiences encountered from teaching a two-semester graduate sequence on these topics will be summarized.

1. INTRODUCTION

Microelectronic system courses generally emphasize both fundamental issues and technology-dependent skills. The fundamentals are basic concepts which are likely to be applicable for several years after graduation, whereas the technology-dependent skills may last only a few years. Therefore, it is prudent to teach state-of-the-art skills that enable graduates to contribute soon after employment begins.

Since microelectronic system designers in industry are increasingly using synthesis to produce systems based on FPGAs and/or ASICs, a two-course sequence has been developed and taught at the University of Tennessee to provide students with exposure to this style of design. This paper summarizes the objectives of this sequence, the prerequisities, an outline of the lecture topics, descriptions of the laboratory assignments and software/hardware support. The extensive use of email and the world-wide web is described along with experiences learned from teaching this sequence over a ten-year period.

2. COURSE OBJECTIVES

The philosophy guiding this two-semester sequence corresponds to the vision espoused by those attending the NSF-sponsored 1993 Workshop on Rapid Prototyping of Microelectronic Systems for Universities: "Educate students who can use the paradigm of design, simulate, design-for-test, build and test (as opposed to just design, build and test) to create microelectronic systems, not just integrated circuits, of sufficient quality that the global competitiveness of U.S. industry will be continued and enhanced." [1].

Hence, the objectives of these courses [2] are:

- To present fundamental techniques of microelectronic systems design including design synthesis, simulation, testing, prototyping and measurement.
- To reinforce the lectures and discussions with laboratory experience using state-of-the-art computer-aided design tools.
- To develop human communication skills via a team project requiring both written and oral reports.
- To illustrate capturing a design in a technology-independent means using a mix of levels (behavior and structure) and then to map the synthesized result into several technologies which can be compared.
- To provide in-depth projects using FPGAs and testable ASICs that involve architectural tradeoffs, simulation and prototyping.

3. COURSE CONTENT

This sequence is intended for senior undergraduate or beginning graduate students in electrical engineering who have completed introductory digital logic design and microprocessor interfacing courses. Each student is expected to spend about ten hours per week on this sequence with 2.5 hours of lectures per week and the remaining time consisting of laboratory and project assignments performed in an open lab.

Lecture topics include:

- Design Methodology
- Design Capture
- Role of Synthesis
- Hardware Description Languages
- Partitioning System Functions and I/O
- VHDL for Combinational Logic
- VHDL for Controllers
- Role of Simulation
- Technology Choices
- Technology-Independent Design
- FPGA Floorplans and Interconnect
- Unique Attributes of FPGAs
- Choosing an FPGA
- Physical Placement and Routing

0-8186-7996-4/97 $10.00 © 1997 IEEE

- Post-Layout Timing Simulation
- ASIC Testing
- Fault Grading
- Technology Comparisons
- Rapid Prototyping
- Interaction with CAD Suppliers
- Interaction with ASIC Foundries
- Technical Report Writing
- Group Presentations

The laboratory and project assignments for the first semester include:

- LOGIN, UNIX, File Editing and Email
- Schematic Entry and Simulation (Viewlogic)
- Capturing Structure Using VHDL; Downloading
- Capturing Behavior Using VHDL
- Capturing Behavior Using ABEL
- FPGA Placement and Routing (Xilinx, Actel and Altera)
- Post-layout Simulation
- Retargeting
- Project Proposal, Presentation, Demo (Xilinx) and Report
- Final Exam

The laboratory and project assignments for the second semester include:

- ASIC Design Validation
- ASIC Design Synthesis and Simulation
- Making an ASIC Design Testable (Synopsys)
- Physical Placement and Routing (Epoch)
- Post-Layout Timing Simulation
- Reconfigurable Computing (EVC Demo)
- Retargeting
- Project Proposal, Presentation and Report
- Final Exam

4. LABORATORY SUPPORT

In keeping with the goal of providing students with exposure and experience with state-of-the-art CAD tools, we have joined the university programs of several CAD vendors. Thus, the software used in this course sequence includes synthesis tools from Viewlogic and Synopsys and physical placement and routing tools from Xilinx, Actel, Altera and Cascade Design Automation (Epoch).

For prototyping of FPGAs, we use demonstration boards supplied by Xilinx and the Engineer's Virtual Computer (EVC) supplied by the Virtual Computer Corporation. All of these units contain reconfigurable logic so the equipment is available for reuse each year. We have 12 Xilinx boards to support 12 two-person projects. We have only one EVC but it is easily shared by all of the students in the class since it is accessible from any workstation.

For ASIC prototyping, we originally submitted designs to MOSIS but have stopped doing this since we almost always found no design errors. The use of NSF funds for this purpose seemed superfluous. We do, however, continue to submit to MOSIS manual layout designs from another course not described here.

We use a cluster of six UNIX workstations (Sparcs) for this sequence. This provides a ratio of 4 persons per seat, which has proved to be very adequate. In fact, the machines are shared by another course which has 8-16 students performing custom layouts. Each workstation has 64 MBytes of internal RAM and a 19-inch color monitor. Since several of the CAD tool packages require 1-3 GBytes of disk storage for the executables and we have several toolsets, we have allocated 12 GBytes for tools and 2 GBytes total for all (not each) user files. For microelectronic systems research projects, we have additional machines and storage capacity so that the machines for this course sequence are not overloaded.

5. EXPERIENCES

Except for the initial handout in the first course, all syllabi, assignments and tutorials are disseminated via the the world-wide web [2]. Students generally utilize a split-screen approach with one window containing tutorial instructions and a second window exercising the appropriate CAD tool needed for the design. Email is used for lecture followup questions and answers and to report completion of assignments. These practices have facilitated communication and enhanced learning.

During the first semester, Viewlogic is used as the initial schematic capture and VHDL synthesis and simulation toolset because of its simplicity. During the second semester, Synopsys is used because of its ability to insert scan logic, automatically generate manufacturing test vectors and to perform fault grading for ASICs. Projects during the first semester are performed by two-person teams. During the second semester, some moderately complex designs are performed on an individual basis with large designs being conducted by four-person teams.

During the first semester, only the basics of the VHDL language are introduced. These include structural techniques which use "components" much like schematics and behaviorial constructs such as "case" and "if-then-else" statements. Thus, only a few of the language concepts need to be mastered but the students are being prepared for the second semester in which more advanced features are presented.

A variety of projects have been undertaken. Generally, each contains a mixture of computation and control. Emphasis is placed on **finishing** all of the project design steps (synthesis, simulation, testing, prototyping and measurement) and not on the complexity of the project itself. Students are encouraged to complete every step for a base level of functionality before adding additional features.

In summary, this sequence has met its goals of providing exposure to state-of-the-art tools and design techniques. However, it is a continuing struggle to achieve this high level of quality.

REFERENCES

[1] Bouldin, D. (Editor), "Report of the 1993 Workshop on Rapid Prototyping of Microelectronic Systems for Universities", University of Tennessee Report, April 1994, http://microsys6.engr.utk.edu/ece/nsf_ic93.html.

[2] Bouldin, D., "Designing Microelectronic Systems", http://microsys6.engr.utk.edu/ece/bouldin_courses.

Homework_1 - LOGIN, EMAIL and WEB PAGE

Revised 19 May 2011 by D. Bouldin

- **Enter 501 Ferris and use your ECE login/passwd on one of the arc.eecs.utk.edu (0-19) machines or setup your PC to login remotedly.**

Login to any of the following machines:

ada7.eecs.utk.edu
ada8.eecs.utk.edu
ada9.eecs.utk.edu

I. UNIX FUNDAMENTALS

- **To change directory to an existing directory: *cd dir1***
- **To change directory to the next level up the tree: *cd ..***
- **To list the contents of the current directory: *ls***
- **To make a new subdirectory: *mkdir dir3***
- **To move an old filename to a new filename: *mv old_file new_file***
- **To move file3 down into dir2: *mv file3 dir2***
- **To move file3 up one level of the directory tree: *mv file3 ..***
- **To copy file1 from directory /usr/cad to the current directory denoted by ".": *cp /usr/cad/file1 .***
- **To remove a file: *rm file1***
- **To remove a subdirectory and recursively everything below it in the tree: *rm -r dir2***
- **To view more of a file one page at a time: *more file***

Press *spacebar* to continue to the next page.
Press *q* to quit.
Press *b* to go back a page.

Commonly Used UNIX Commands

II. E-MAIL FORWARDING & FUNDAMENTALS

I recommend you create a .forward file in your root directory to point to the machine you like to use to read your email.
For example,

.forward

dbouldin@utk.edu

III. EMAIL YOUR PARAGRAPH and CREATE a .PLAN FILE

- **Use an editor (gedit, vi or emacs; not WORD) to write a 100-word paragraph on why you are taking this course and what you expect to gain from it.**

- **Email a typo-free version of your personal paragraph to "dbouldin@tennessee.edu".**

- **Use an editor to create a ".plan" file that you can place in your root directory containing the following information.**

Name:
Address:
Phone Number:
Course:
Anticipated Graduation Term:
M.S. or Ph.D. Thesis Topic:
M.S. or Ph.D. Thesis Advisor:
Assistantship:
Employer:

Then, whenever anyone types "finger yourusername", this information can be accessed. Try typing "finger bouldin".

IV. MAKE YOUR OWN WEB PAGES

First, go to your root directory and type:

mkdir webhome
cd webhome
cp ~bouldin/webhome/551.html .

Then edit 551.html as required (note 551/651).
Your page should include items like those in your .plan file
and in your Homework 1 paragraph on why you are taking this course.
Please keep everything on this page formal. You may link your home
page to other pages and then put whatever you wish.

Ask another student in the class to check what you have done by accessing:

http://web.eecs.utk.edu/~username/551.html

where "username" is your own.

For example:

http://web.eecs.utk.edu/~bouldin/551.html

Scott Fields

You should also create a protected website that requires a username and password to access.
It is intended for material that only you and I need to see and not other students.
To create yours,
(1) Edit your public page to point to the private page
cd ~your_username/webhome
gedit 551.html
INSERT: a href="protected/551/index.html">PROTECTED WEBSITE
(2) Create a protected subdirectory and one below that for for the private material
mkdir protected
chmod 705 protected
cd protected
(3) In the protected subdirectory, create .htaccess and .htpasswd files
CONTENTS of .htaccess:

```
AuthType Basic
AuthName authorization
AuthUserFile /home/your_username/webhome/protected/.htpasswd
AuthGroupFile /dev/null


require user your_username
require user bouldin
```

CONTENTS of .htpasswd:

```
bouldin:byaShdv/TCtEE
your_username:goxLFUKzdrvsk
```

(4) Edit "your_username" to be your own and replace the temporary password I have for you with one of your choosing.

To generate a password, type:

perl -e 'print crypt("govolsgo","go");'

The following will be printed on the screen:

goxLFUKzdrvsk

In general, you type: perl -e 'print crypt("password","salt");'
where password is obvious and salt is a 2 character string.
mkdir 551
(5) Move to that subdirectory and create a index.html file
cd 551
gedit index.html
(6) Put restricted material in the subdirectory and add links in the index.html file.
(7) Test the access by using Firefox or Internet-Explorer.
Open the public page. Click on the PROTECTED link.
A pop-up dialog box should ask you for your username and password.

---

dbouldin@tennessee.edu