ECE 464/520 Digital ASIC Design
Summer 2014
Course Overview & Policies

Instructor: Professor Paul D. Franzon, Ph.D.
Office: EGRC 443, (919) 515-7351
E-mail: paulf@ncsu.edu
Home page: www.ece.ncsu.edu/erl/faculty/paulf.html

Introduction: Digital ASICs are at the core of ALL modern Information Technology. They rely on silicon chip technology to enable society to build complex systems including wireless devices, tablets, computers, network routers, etc. Due to the rapid rate of improvement of the underlying silicon technology, these chips are constantly being redesigned. Thus digital ASIC design companies are strong employers all around the world.

This is a course that teaches you how to design a digital ASIC (standard cell or FPGA) using the Verilog Hardware Description Language. At the end of this course you will have an understanding of what ASICs are, and how they are designed and verified and you will have demonstrated those skills via a project.

Class Web Site: The class web site can be found on Moodle.
https://moodle1415-courses.wolfware.ncsu.edu/course/view.php?id=534

Class Format and Schedule: This will be conducted as a combined EOL and pseudo-residential class. All students will be taking the online only course, whether you are enrolled in ECE 464-051, -651, or ECE 520-051, -651. The Lab TA will be available to EOL student via Elluminate (Blackboard Collaborate), and physically to residential students via appointment. The instructor will be available by appointment. Unlike the Spring Version of ECE 464/520-001, there will be no physical meetings of the class.

The course will be run over a 10 week period. The course is broken into a number of modules, one module for each major topic to be covered. Each module consists of one or more short (generally about 20 – 30 minute) videos, that together with the notes, convey content. At the end of each video, an online quiz will be available to review major points within the video. At the end of each module, a quiz is available that is really a “mini-exam” for that module. In addition, there are design exercises and homeworks which will be turned in. Finally there is a midterm and final exam.

A course schedule will be published. Every week you will be expected to complete one larger, or two shorter, modules including the associated quizzes. The quizzes for each week’s module closes on Wednesday of the week following the week in which the module starts. It closes at 2355 (5 minutes before midnight) New York time. E.g. The quizzes for the first module (Revision) will close Wednesday of week 2. After the module closes, the answers will be available and no further submissions can be considered. Note that “SAVING” an answer set does not submit it. You have to hit the “SUBMIT” button.

Communications: Students are expected to use the Forums for questions and general communications. Please take the effort to subscribe to these, read them and answer questions.

Textbooks & Notes:
Purchase is Optional.

- Course notes, etc. on class web page.
- References:
Prerequisite: It is assumed that you have taken and passed an Undergraduate logic design course. At NCSU, this is ECE 212. This course should have covered logic gates, combinational and sequential logic, and logic minimization. Hopefully it also covered timing diagrams and finite state machines. Though I do have a revision module in the class, it is not intended to substitute for a complete course. However, I do review timing diagrams and finite state machines in some detail. If you wish to brush up on the course background in detail, any book with the terms “logic design” or “digital design” in the title should suffice. I assume NO KNOWLEDGE of Verilog or any other Hardware Description Language. While knowledge of C is helpful, it is not assumed.

Course Objectives
1. To prepare the student to be an entry-level industrial standard cell ASIC or FPGA designer.
2. To give the student an understanding of issues and tools related to ASIC/FPGA design and implementation, including timing, performance and power optimization, verification and manufacturing test.

Course Outcomes
1. Students will be able to design and synthesize a complex digital functional block, containing over 1,000 gates, using Verilog HDL and Synopsys Design Compiler.
2. Students will demonstrate an understanding of how to optimize the performance, area, and power of a complex digital functional block, and the tradeoffs between these.
3. Students will demonstrate an understanding of issues involved in ASIC design, including technology choice, design management, tool-flow, verification, debug and test, as well as the impact of technology scaling on ASIC design.

Course Syllabus
0. Revision of Digital Design
1. Introduction to ASIC design
2. Timing design
3. Design of digital hardware using Verilog HDL I
4. Design of digital hardware using Verilog HDL II
5. Design of Finite State Machines
6. Design of complex systems.
7. Managing hierarchy.
8. Verification.
10. Low Power Design.
11. Introduction to FPGAs
12. Closing Comments

Computer Aided Design Tools
You will be expected to use the Mentor Modelsim simulator and Synopsys Design Compiler. Students will be set up for remote access to these tools on campus. However, you might find it useful to download and use the Mentor Modelsim PE Student Edition (http://model.com/content/modelsim-pe-student-edition-hdl-simulation) and use that instead of the on-campus version. Based on prior experience, this is easier than using oncampus tools remotely. Of course residential students can use these tools on Linux machines in Unity labs. You might also consider downloading the free “web”edition of Altera Quartus for synthesis and Modelsim-Altera for simulation. These are available from www.altera.com. (Click on “downloads” on the top bar.) A full tutorial will be provided as part of the course. Note, these tools only run on windows7 and linux. While you are expected to use Synopsys Design Compiler, Quartus might be useful for you for early debug of your designs, as you can run that locally. Both Linux and Mocrosoft7 versions are available.

Student Evaluation

<table>
<thead>
<tr>
<th>Item</th>
<th>Grade Contribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intra-module quizzes</td>
<td>5%</td>
</tr>
<tr>
<td>End of module quizzes</td>
<td>15%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>15%</td>
</tr>
<tr>
<td>Homeworks</td>
<td>10%</td>
</tr>
<tr>
<td>Project</td>
<td>25%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>30%</td>
</tr>
</tbody>
</table>
Instructor Research Interests
- Application specific processors and sensor systems. Current projects are in applications of 3DICs, sensor system design, and secure IP design.
- Interconnect, including transceivers, electronic packaging, on-chip interconnect, and between-chip interconnect.
- Nanocomputing – how to build the computers that will eventually displace or complement CMOS.

Exams
Exams will be conducted online. You will be given a 24 hour window in which to complete the exam.

Students with disabilities
Reasonable accommodations will be made for students with verifiable disabilities. In order to take advantage of available accommodations, students must register with Disability Services for Students at 1900 Student Health Center, Campus Box 7509, 515-7653. [http://www.ncsu.edu/provost/offices/affirm_action/dss/](http://www.ncsu.edu/provost/offices/affirm_action/dss/) For more information on NC State’s policy on working with students with disabilities, please see [http://www.ncsu.edu/provost/hat/current/appendix/appen_k.html](http://www.ncsu.edu/provost/hat/current/appendix/appen_k.html)

Academic integrity
All the provisions of the [code of academic integrity](http://www.ncsu.edu/provost/offices/affirm_action/dss/) apply to this course. In addition, it is my understanding and expectation that your submission of any test or assignment means that you neither gave nor received unauthorized aid. **My policy for homeworks and projects is that while you are free to collaborate, sharing of design data, specifically Verilog code, is expressly forbidden. If you collaborate on a design problem, I still expect you to turn in individually developed code.**