**ECE627: Register-transfer-level Digital Systems**

Instructor: Nachiket Kapre (nachiket@uwaterloo.ca)

**Motivation**

The design and engineering of computing chips is an exciting, and valuable skill that spans multiple engineering disciplines including software engineering, parallel programming, verification and reasoning, compilation techniques, down to physical/electrical effects. This course will help opens the door to careers in richly rewarding chip design industry. With the impending demise of Mooreâ€™s Law, now more than ever, careful bottom-up rethinking of the hardware stacks are necessary. Many chip startups today are developing new hardware for machine learning/AI, database acceleration, virtualized network functions, and other domains. The next decade or so will see exciting opportunities for overhauling our computing infrastructure. 627 will be vital to navigate these trying times.

**Content**

What are the key concepts we must know to design correct, high-performance digital chips? How do we capture the architecture of these chips in a programming language, how do we verify correctness of these designs, and how do we translate these descriptions down to silicon? In ECE627, you will learn RTL design and analysis concepts that are positioned above ECE637: Design of VLSI MOS Integrated Circuits, and below ECE621: Computer Organization. We will cover the entire design flow and implementation on FPGAs as part of this course. We will use state-of-the-art industry-standard tools such as Modelsim, and Xilinx Vivado and complement them with fun open-source tools such as yosys and verilator. There will be a significant project component (series of labs), series of quizzes (almost every week), and a final exam.

**Objectives**

1. The design of hardware components requires reasoning about a unique form of parallelism and concurrency and operations at a cycle-by-cycle level of abstraction. Hardware description languages such as Verilog, and VHDL provide a programming environment that makes it possible to express this form of parallelism. We will understand the fundamental building blocks of the programming languages necessary to describe hardware.
2. There are commonly used paradigms, and design patterns that we must known when organizing the RTL code. Most digital designs must describe interaction protocols with the outside world, manage state, and capture datapath operation behavior of the digital design. We will learn how to break down a complex design into these components, and learn how to reason about correctness, and understand their translation to hardware primitives. We will use tools such as Modelsim (verification) along with Vivado (synthesis).
3. It is important to analyze the design for goodness metrics such as resource usage, performance, and power consumption. Most digital design are constrained to fit specific cost targets, and the analysis tools help guide the design process with appropriate feedback. We will learn how to interpret the results of the FPGA CAD tools and incorporate that feedback to redesign our RTL. These compilation iterations are fundamental to digital design and a good designer knows how to direct the tools and generate the desired hardware.
4. A class project will be used to tie together all the key design lessons we learn in this course. The project will be broken down into a series of labs; each lab is a phase/milestones over the course of the term to guide student progress. We will be developing a matrix multiplier as part of the project.

**Assessment Criteria**

*Quizzes* (20% credit): This will be a short in-class quiz each week to evaluate your understanding of the core concepts of RTL design. There will be a set of 8 quizzes spread throughout the term. The quiz will cover a topic from the previous week (or weeks). Each quiz will be worth exactly the same number of points. There is no midterm.

*Labs+Project* (30% credit): Students will work on a prescribed class project through a series of labs and submit a design + report (2 pages). There will be 4 labs spread throughout the term. Lab 1 is worth 5%, Lab 2 is worth 7.5%, Lab 3 is worth 7.5%, and Lab 4 is worth 10%. Taken together, the labs are worth 30% of the total grade.

*Final exam* (50% credit): Students will have a final exam for the course that will cover RTL design and analysis topics. The final exam will drawn from questions in the quizzes for ideas and inspiration.

**Learning Outcomes**

At the end of the course, students should be able to do the following.

1. Describe hardware components at the RTL-level, verify these designs through simulation, and map these design to an FPGA fabric.
2. Learn how to use, configure, and understand FPGA CAD tools for specific problem requirements.
3. Analyze key metrics of an RTL design. Perform engineering tradeoffs in design for area, performance, and power.
4. Operate an FPGA board, and debug designs in real hardware. Implement a sophisticated project on the board and demonstrate correct operation.
5. Document the class project in the language of a 4-page technical report.

**Pre-Requisites**

Undergraduate course on digital hardware design with VHDL or Verilog. These may be waived at the discretion of the instructor through prior arrangement. If you feel you deserve a waiver, please talk to me.

**Co-Requisites (not mandatory)**

If students are interested in mastering hardware design at all levels of abstraction beyond RTL design, they are encouraged to take **ECE637**, and **ECE621**. If they want to know about FPGA design, they should take **ECE720T6** (if offered).

**Syllabus**

| **Topic** |
| --- |
| Introduction to RTL Design |
| Simulation of RTL Designs |
| Synthesizing RTL Designs |
| FPGA Architecture Fundamentals |
| ASIC Design Basics |
| State Machine Design |
| Datapath Design |
| Managing Memories |
| Timing Analysis |
| Power Analysis |
| High-Level Synthesis |
| Formal Verification |

**Plagiarism Policy**

We will enforce a strict anti-plagiarism policy in accordance with [Policy 71](https://uwaterloo.ca/secretariat/policies-procedures-guidelines/policy-71). There will be no tolerance for copying of lab solutions, and will result in immediate triggering on Policy 71 provisions.