Course Numbers: ECE/CS 5740/6740
Lectures: Mon, Wed, Fri: 10:45 - 11:35am, WEB L 120.
Office & Hours:
   MEB 4512, tel: 587-7617
   Office hours: TBD
Credit: 3 credit hours.
Grading policy:

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homeworks</td>
<td>25%</td>
</tr>
<tr>
<td>Mid-Term Exam</td>
<td>25%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>25%</td>
</tr>
<tr>
<td>Final Project</td>
<td>25%</td>
</tr>
</tbody>
</table>

Course Materials: Textbook, Lecture notes, published papers, thesis, etc. We will use the book titled Logic Synthesis and Verification Algorithms, authored by Gary Hachtel and Fabio Somenzi, as the textbook for the course. Since the course will cover a wide (and selective) range of topics, there is no single textbook which covers all of them. For the topics that are not (properly) covered in this book, I will give you notes/papers for reading. (These topics include Boolean Decomposition, Retiming and Resynthesis, some symbolic encoding problems, and some recent advances in Logic Synthesis). I will also direct you to most of the literature that is available on the web so you can download electronic copies for yourself. Lecture notes can be downloaded from the class webpage that I have developed. The URL is

http://www.ece.utah.edu/~kalla/index_5740.html

Listed below is a set of reference books, you may find them in the library.

- *Switching and Finite Automata Theory*. Author: Kohavi. McGraw-Hill. An excellent text to understand switching theory, with fantastic coverage of sequential machine operations. Covers Finite State Machines (FSM) like no other. Unfortunately, I think it is out of print. Should be in the library.

Homeworks: Please pay due attention and importance to the homeworks as they will be quite thorough in covering the material. My intention is not to make you slog through 20 pages of Boolean algebra or algorithm design for each homework. All I want you to do is to understand the practical application of the basic theoretical concepts, their power and their limitations. In addition to problem solving, I would want you to get hands-on experience with some CAD tools and implementation. So the HWs will contain some CAD-TOOL use and some programming assignments.

Exams: We’ll have two exams & a final project.

Final Project - A Team-Effort: Students are required to work on a project. A team of 2 students should get together and decide upon a project that they would want to do. Because there aren’t too many students in the class, I would also allow individual student-projects. Within the first 4-5 weeks, you should discuss with me what project you would be interested in, and we will do a feasibility analysis. Ideally, the project should be a theoretical study + implementation of a logic optimization problem. I may also allow some experimental analysis of a particular problem, a thorough theoretical study of a problem (a review paper), etc.

Prerequisites: Basic Digital Logic Design concepts, and some knowledge of fundamental computer algorithms: sorting, searching, graph algorithms, etc. Knowledge of any advanced programming language (C, C++, etc.).

Some of you may not have had formal course-work in algorithms, but advanced knowledge of algorithms is not a strict requirement. We are going to mostly employ fundamental/basic algorithms to solve the optimization problems. So long as you know the following, you’re going to be alright.

1. Difference between tractable and intractable problems.
2. Basic data structures: arrays, queues, lists, symbol tables, etc.
3. Fundamental Graph traversal: shortest path, longest path, breadth-first search, depth-first search, sorting, etc.

If you have absolutely no idea of what the above means, pick up any algorithms book and you can very easily understand these topics. Talk to me and I’ll guide you through it.

What will you learn in this course? Logic Synthesis tools have become so common that almost all of you may have used them at some point in your course work/workplace. In fact, logic synthesis has matured to a point where it is now-a-days just a “push-button” process. But what happens when you push that button on the dialog box? That is what this course is all about.

First of all, this course will give you a good dose of switching circuits and Boolean function manipulation. We will study many different operations on Boolean functions, and observe how these operations in the Boolean domain reflect on digital circuit structure. This part is going to be the mathematical introduction to switching algebra. One important issue that we will analyze is that of Boolean function representation. There are many different representations for Boolean functions and each has its own power and limitations. Some representations are most suited for a particular application, while others are not. After you complete this course, you would have a fair idea of what type of representations should we use for various applications/problems.

After covering switching theory fundamentals, we will see how these concepts can be employed to minimize two-level SOP forms. Then we will study the multi-level logic optimization techniques. Both algebraic and Boolean models will be considered. Then, we will study the sequential circuit optimization problems in detail. We will cover
most modern, contemporary techniques in logic synthesis.

This is not a crash-course in Logic Optimization, nor should it be considered a short-cut to building an impressive resume. This is going to be a rigorous course, and I hope all of us will enjoy it. This course is an important curriculum component in computer engineering, VLSI design, CAD/Design Automation.

**Ethics:** Work/study together with your colleagues, but do all your homeworks yourself. Attend classes regularly, participate in class, make learning fun for yourself, and make teaching fun for me.