ECE 572: ADVANCED LOGIC SYNTHESIS

PROFESSOR: Marek A. Perkowski

CLASS LOCATION: Sixth Avenue Building (SAB), room 212, Mondays and Wednesdays 6:00 pm - 7:30pm. Additional, non-mandatory, recitations on Fridays, room CECS 54, 6:00 pm - 8:00 pm. These meetings are devoted to help you to learn solve problems.

GRADING POLICY

The class will be graded as follows:

1. Homeworks = TOTAL OF % 25
2. Midterm Exam = % 15
3. Project = % 40
4. Final Examination = 20 %

I am a strong believer in practical projects. Such projects should be state-of-the-art and relevant to current research and development. Therefore the project component of the grade is high. It can be even higher for well-qualified and willing students.

Projects for Ph.D. and M.S. students with theses will be more complex and/or ambitious. For Ph.D students, the projects are on a level high enough to warrant collaboration with faculty and more advanced students on publishing papers in conferences and journals.

It is assumed that every student who takes this class has had some exposure to computer programming. It has been however my observation in the past that very few students are able to write recursive, backtracking and searching non-numeric programs that are a base of modern EDA tools. Therefore I spend some time to teach LISP which is easier than C or Visual C++, being another languages of choices for the class.

TEXTBOOK

Gary D. Hachtel and Fabio Somenzio ``Logic Synthesis and Verification Algorithms". Kluwer.

It is important to know that the main class material are slides available on my web page.

TOPICS TO BE COVERED

It is the goal of this class to cover all the following topics. However, because the class is changing every year because of projects and new material added and old removed, the order of presentation may change.

1. **Introduction to binary logic and digital design:** . Review of Boolean Algebra, flip-flops, maps, SOP minimization and basic procedures.
2. **Realization Technologies of Digital Logic:** PAL, PLA, ROM, PLD, CPLD, FPGA, FPAA, FPIC. Custom Design, ASIC, Standard Cell, PGA. Requirements for design: power, area, speed, testability.
3. **Boolean, Multiple-Valued and Fuzzy Functions and Relations:** Characterization. Special Types of functions (symmetric, unate, etc.). Operators on Data.
4. **Programming Language LISP:** basic data structures (atoms, pairs, lists, trees), conditionals, expressions, functional programming, recursion, advanced recursion, data-driven programming, use of predicates, representation of logic data, realization of typical EDA algorithms using recursion and rule-based programming.

5. **Representation of Boolean and Multiple-Valued Functions:** Cube Calculus, Binary Decision Diagrams, Spectra, Spectral Decision Diagrams, rough partitions, Labeled Rough Partitions, netlists.
6. **Basic Combinatorial Algorithms:** Set Covering (Unate Covering), Covering/Closure (Binate Covering), Graph Coloring, Maximum Clique, Shortest Path, Longest Path, Maximum Flow, etc.
7. **Two Level Minimization:** Two-level minimization is still a very important topic and a fundament of modern logic synthesis. We will cover primes and their generation and various approaches to covering. In addition to standard SOP minimization we discuss ESOP and CDEC minimization as well.
8. **Minimization of functions with few levels.** TANT and three-level circuits, four-level circuits. AND/OR Circuits with limited numbers of levels.
9. **Factorization methods for multi-level circuit minimization:** .
10. **Spectra of functions:** Walsh, Hadamard, Haar, Fourier, Reed-Muller, Linearly Independent, Arithmetic. Their uses in logic design and digital signal processing.
11. **Technology Mapping and regular structures:** relation between logic synthesis and physical design.
12. **Functional Decomposition:** Ashenhurst, Roth-Karp, Curtis, generalized Ashenhurst/Curtis, decomposition of multi-valued relations, bi-decomposition, BDD-based decompositions, other approaches to decomposition of functions.
13. **Finite State Machines:** Examples, types, minimization of number of states, state assignment, generalized encoding problems, retiming. Discussion of special types of machines. Use of various types of memories. Practical design using EDA tools and hardware description languages.
14. **Low Power Design:** Many of the above problems will be repeated with low power in mind.
15. **Reversible Logic and Quantum Logic.** Principles, gates, circuits.
16. **Implicit Methods:** Many of the above algorithms will be presented using implicit rather explicit representations. This leads to very efficient algorithms.
17. **Overview of recent methods and theories:** DNA computing and biomolecular applications.