COURSE: EECS 270. **TITLE:** Introduction to Logic Design. **PREREQUISITES:** Engin 101 or EECS 183 or equivalent. **ELECTIVE**

TEXTBOOK: John F. Wakerly, *Digital Design: Principles and Practices*, 3rd ed.

CATALOG DESCRIPTION: Binary and non-binary systems, Boolean algebra digital design techniques, logic gates, logic minimization, standard combinational circuits, sequential circuits, flip-flops, synthesis of synchronous sequential circuits. PLAs, FPGAs, ROMs, RAMs, arithmetic circuits, state-machine design, computer-aided design. Laboratory includes hardware design and CAD experiments.

COURSE OBJECTIVES:

1. To teach students the basics of combinational and sequential logic
2. To provide some hands-on experience with an FPGA-based system.
3. To provide some hand-on experience with computer-aided design software for schematic circuit capture and logic simulation.
4. To prepare the students for advanced courses in logic synthesis and optimization, computer architecture, and VLSI.

COURSE OUTCOMES [Program Outcomes Addressed]

1. Ability to perform simple arithmetic in binary, octal, hexadecimal, BCD number systems [1,11]
2. An ability to manipulate logic expressions using 2-valued Boolean algebra. [1,11]
3. An ability to generate the prime implicants and implicates of logic functions of 6 or fewer variables using graphical (Karnaugh map) and tabular (Quine-McClusky) methods, and to obtain their minimal two-level implementations with and without don't cares. [1,3,11]
4. An ability to analyze and synthesize small multi-level combinational logic circuits containing AND, OR, NOT, NAND, NOR, and XOR gates based on simple delay models. [1,2,3,5,11]
5. Ability to use basic functional & timing (clocking) properties of latches & flip-flops. [1]
6. Ability to analyze synchronous sequential circuits to extract next-state/output functions [1,5,11]
7. An ability to translate a word statement specifying the desired behavior of a simple sequential system into a state table/diagram, to simplify such tables by merging equivalent states, and to design and implement the complete next-state and output logic from such tables. [1,3,5,11]
8. An ability to implement simple digital systems using MSI building blocks (registers, memories, counters, multiplexers, ALUs, etc.) from an algorithmic state machine (ASM) specification. [3,5]
9. Basic knowledge of computer organization, including data path and control sequencing. [1]
10. An ability to build and test simple digital systems using a CAD system [1,2,11]

TOPICS COVERED:

1. Analog vs. digital. Binary numbers and codes. Twos complement.
2. Logic functions. Truth tables. Logic gates (AND, OR, NOT).
3. Implementation issues. CMOS integrated circuits
4. Switch-level models. Boolean algebra. Definitions & theorems.
5. Minterms, maxterms and canonical forms. Two-level circuits.
6. NAND, NOR, and XOR functions, gates, and circuits.
7. Boolean function simplification. Prime implicants.
8. Karnaugh maps. Don't cares. Tabular (Quine-McCluskey) method.
9. Combinational design: adders, code converts, MSI/LSI components.
10. Design with multiplexers, ROMs, PLAs and FPGAs
11. Combinational vs. sequential. Sequential circuit types.
12. Basic latches & flip-flops. Clocking methods. Master-slave & edge-triggered flip-flops.
13. Synchronous sequential circuits. State tables and diagrams.
14. State table simplification. Sequential circuits. Design process.
15. Parallel and shift registers. Counters. RAMs.
16. Basic computer organization. Data path. Control sequencing.
17. Computer-aided design (CAD) techniques. Simulation.
18. Advanced design concepts.

ASSESSMENT (Course outcomes)

1. Biweekly homework [1-10]
2. Hardware & CAD labs [8,10]
3. 3 closed-book exams [1-10]

CLASS/LAB SCHEDULE:

LEC: 3 per week @ 1 hour.

LAB: 1 per week @ 3 hours.

PROGRAM OUTCOMES ADDRESSED: 1,2,3,5,11

PROFESSIONAL COMPONENT ADDRESSED: 13

PREPARED BY: Mark Brehob on Dec. 13, 2004;
modified by Andrew E. Yagle on April 8, 2005