An Introductory Digital-Logic Design Laboratory

Daniel J. Tylavsky (tylavsky@asu.edu)
Department of Electrical Engineering
Arizona State University

Abstract

A series of digital-logic design laboratory experiments have been created for a first course in digital logic design. These laboratory experiments are aimed primarily at first and second year electrical engineering and computer science/engineering students. The laboratory exercises include a set of six hardware laboratory experiments, and eight digital-logic simulation experiments. To receive a copy of the digital-design experiments discussed in this paper, send a request to Dr.Dans@ieee.org

I. Hardware Laboratory Experiments

The objective of the hardware laboratory design is to start students with basic experiments that emphasize common laboratory measuring and debugging techniques. Later more sophisticated experiments emphasize the design skills students have acquired in the lecture portion of the class as shown in Table 1.

All of the hardware labs emphasize hardware realizations. Later labs include some use of digital-logic simulations to simulate circuits students build using TTL hardware. The objective of this mix is to allow students to discover, on their own, the role that simulation plays in the prototyping of complex engineering systems. There are more tasks included in the experiments than students can complete in the laboratory time allotted for most college courses. Selecting a subset of cohesive tasks that vary from semester to semester allows students to rely on their own understanding of the material rather than that of students from previous semesters. The order of the experiments is chosen to be in synchronism with the order of topics covered by most textbooks on introductory digital-logic design.

In the capstone design project, students are assigned to produce two designs that meet a given functional specification and pick the better of the two designs using their own metric. It is part of their task to define what "better" means and to describe in their report how one of their designs is better than the other. They then develop a LogicWorks™ simulation to “proof” their concept and demonstrate this simulation to a teaching assistant. The “proofed” concept is then built in the hardware lab and again demonstrated to a teaching assistant. A final report on their capstone design project is a requirement. This project and report allows students to demonstrate three things. They demonstrate their knowledge of synchronous machine construction by designing (typically) a Mealy and Moore machine that meets the problem statement. They demonstrate their knowledge of building and debugging circuitry, using both hardware and a hardware simulator. They demonstrate in their final report the communication skills they have developed during the semester.

Digital-Logic Simulation Experiments

To complement the hardware experiments, we created a set of digital-logic simulation experiments. We felt that students would feel a sense of accomplishment and satisfaction if they could build a simulation of a microprocessor, program it's instruction set, and execute a program on it. This microprocessor uses 4-bit words, instructions, and a 4-bit address bus. This microprocessor is built using LogicWorks™ software through five successive laboratory exercises during the semester. Each simulation assignment uses only elementary logic gates (AND, OR, NOT, NAND, etc.) to build adders, decoders, multiplexers etc., which are then modularized. (Emphasis is place on building all devices using primitive gates so that students cannot help but understand that computers are built primarily using logic gates.) The microprocessor is then constructed by connecting each module appropriately. The titles, desired outcomes, and pedagogical emphasis of each lab experiment are listed in Table 2. One of three versions of the experiments is assigned each semester. This allows students to rely on their own understanding of the material rather than that of students from previous semesters

The digital-logic-simulation and hardware-lab exercises are conducted in alternating weeks. This schedule allows students to build a half adder using TTL hardware one week, then to build its simulation the next week (or vice versa.) By alternately using TTL hardware and then circuit-simulation software to build devices, students learn the role of circuit simulation in the design of complex engineering systems. The digital-logic-simulation labs are designed to be in synchronism with the sequence and pace of lectures supported by most introductory digital-design textbooks.

Report Writing Guidelines

It is our experience that first and second year engineering students have difficulty writing effective laboratory reports. The lab manual has a section on report writing guidelines that help the student understand the context of report writing, the major sections needed for a complete report, how to handle references and direct quotations, and how to handle section numbering.

Assessment Guidelines and EC 2000

Each laboratory assignment has a list of objectives and concomitant learning outcomes. These outcomes are assessed using two tables contained in the last pages of the laboratory assignment. The first table lists each task and each facet of report writing. The maximum possible points awarded for each line item are filled in by the instructor and made available to the students a priori so that they know how their report will be graded. The second table is a self-assessment worksheet for students to fill in. Both sheets are assessment vehicles that may be used as supporting evidence of having met learning objectives for EC 2000 accreditation visits.

Conclusion

We have found from using and refining this laboratory experience over the last 8 years, that students do indeed get a sense of accomplishment from completing the laboratory experiments and that industry is impressed with what students learn from their experience.

Table 1. Hardware Experiments: Main Performance Outcomes and Pedagogical Emphases

Lab Title	Outcome: At the completion of the exercise the student will be able to:	Emphases
Lab 0: Using a Prototype Board, Logic Probe & Voltmeter	Use a breadboard, logic probe and voltmeter.	Electrical Measurement
Lab 1: Debugging a Half And Full Adder	Build a half and full adder and debug combinational logic.	Electrical Measurement, Debugging
Lab 2: TTL Characteristics, Three-State Buffers, Open-Collector Buffers	Use three-state and open-collector buffers.	Electrical Characteristics, Debugging
Lab 3: Latches, Flip-Flops, Registers and Counters	Build and debug latches, counters, and registers.	Sequential Circuit Performance, Debugging
Lab 4: Vending Machine Design	Build a Vending Machine Controller using MSI circuits. Use cut-and-try design principals.	Use of MSI circuits, Principals of Design, Role of Circuit Simulation in Design
Lab 5: Capstone Design Project	Design, simulate and build an arbitrarily complex synchronous sequential machine.	Principals of Design, Classical Design Procedures, Circuit Simulation

Table 2. Digital-Logic Simulation Experiments: Main Outcomes and Pedagogical Emphases

Lab Title	Outcome: At the completion of the exercise the student will be able to:	Emphases
Lab 0: Hardware Simulator Operation (supports LogicWorks™4 hardware simulator)	Use LogicWorks™ to create and debug a circuit module, a communication bus, and use the hex keypad and hex display.	Tutorial approach to the use of LogicWorks™
Lab 1: Adder, Incrementer and Two's Complementer	Build and debug a multi-bit full adder, incrementer, and 2's complement circuit.	Building and Debugging Combinational Logic Circuits
Lab 2: 4-Bit Adder, Multiplexer and Demultiplexer	Build and debug a circuit simulation of multi-bit full adder, mux, and demux.	Building and Debugging Combinational Logic Circuits
Lab 3: The ALU and Decoder	Build, and control 4-bit ALU & decoder	Design and operation of an ALU
Lab 4: The Brainless Microprocessor, ROM and RAM	Act as a controller for a microprocessor comprised of ROM, RAM, ALU, program counter, memory address register, and accumulator.	Controlling the interaction of the major components of a microprocessor.
Lab 5: The Complete Microprocessor with ROM Controller	Build, and debug a controller for a microprocessor. Develop an instruction set. Program the microprocessor and execute the program.	Design of a ROM based synchronous machine as a microprocessor controller. Construction of an instruction set.
Lab 6: Adding Jump-Instruction Capability to Your Microprocessor (Advanced Lab)	Describe the architectural elements and functional blocks needed to implement a jump procedure.	Microprocessor addressing architecture.
Lab 7: Adding Status Logic and Branch Instructions to Your Microprocessor (Advanced Lab)	Describe the architectural elements and functional blocks needed to implement a conditional jump procedure.	Microprocessor addressing architecture. Role of a status register and status logic.