E85: Digital Electronics & Computer Engineering

Harris **Fall 2019**

Syllabus

Teaching Team

Professors:	David Harris	Parsons 2374
	Yizhe Chang	
Lab Assistants:	Noah Boorstin	
	Pinky King	
	Peter Johnson	

Schedule

Lecture:	MW 8:10 Shan 3481
Office Hours:	Chang: M 7:30-9:30 pm (with TBΠ in Platt), Sunday 2-6 (Digital Lab)
	Harris: M 11-12, Tu 1:15-2, Wed 10-11, Thurs 1:15-2:30 Parsons 2374 or Digital Lab
Grutor Lab Hours:	Saturday 12-2, Sunday 12-4 in Parsons B183 (Digital Lab)
TBΠ Tutor Hours:	Monday 7-8, Tuesday 6-8 pm, Platt

Feel free to stop by even if I do not have official office hours. One of the main reasons that I teach at Harvey Mudd is that I value working with students 1-on-1 and in small groups.

Text

You will get the most out of class if you do the reading before lecture. See the schedule for recommended reading. Copies of the textbook are available in the lab and in the Engineering lounge.

Harris & Harris, *Digital Design and Computer Architecture, ARM Ed.*, Morgan Kaufmann 2016.

Electronic Communication

Class web page:	http://pages.hmc.edu/harris/class/e85
Class email list:	eng-85-l@g.hmc.edu

You also will need a Harvey Mudd College computer to complete your labs. If you are not a HMC student, email me your full name and school affiliation and I will request an account for you.

Course Objectives

Digital systems have revolutionized our world. From television to cell phones to GPS to warfare to medicine to automobiles, computers and digital processing have reshaped the way we live and work. Computers are also a vital part of daily practice in every field of science and engineering.

Previous generations of engineers learned the "nuts and bolts" of the profession by doing things like disassembling and rebuilding engines. As technology has advanced, cars have become too complicated for the layperson to work on. Ironically, the same advances have made computers much easier to build. While most fields of engineering require extensive mathematics and complicated analysis of even rather simple components, digital systems merely require counting from 0 to 1. Their challenge, instead, is in combining many simple building blocks into a complex whole. Field programmable gate arrays (FPGAs), containing the equivalent of thousands or millions of logic gates, make it possible to build these complex systems in the lab without the tedium of manually connecting components. In this class, you will build your own microprocessor and test it on a FPGA. In the process, you will master the art and science of digital design. You will learn to speak to and control processors in their native tongue, assembly language. And you will put all the pieces together to demystify how a computer works.

As you probably know, very few complex systems work the first time you put them together. Engineers must become good at systematically and efficiently debugging their creations. One of the course objectives that can be frustrating but vitally important is to learn to teach yourself professional-strength computer-aided design tools and to use these tools to debug systems.

By the end of this course, you should be able to:

- Build digital systems at all levels of abstraction from transistors through circuits, logic, microarchitecture, architecture, and C culminating with implementing and programming a microprocessor soft core on a field programmable gate array.
- Manage complexity using the digital abstraction, data types, static and dynamic disciplines, and hierarchical design.
- Design and implement combinational and sequential digital circuits using schematics and hardware description languages.
- Program a commercial microcontroller in C and assembly language and use it in a physical system.
- Begin the practice of implementing and debugging digital systems with appropriate lab techniques including breadboarding, interpreting datasheets, and using field-programmable gate arrays and microcontroller boards, simulators, debuggers, and test-and-measurement equipment.

Grading E85

Labs:	30%
Problem Sets:	20%
Midterm:	17%
Final:	33%

E85A

Labs:	30%
Problem Sets:	20%
Midterm:	50%

Lab 11 is the capstone of the labs, in which you design and simulate a microprocessor, drawing on most of the skills you have acquired over the semester. **You must turn in a working Lab 11 to pass E85**.

Solutions to the labs and problem sets from previous semesters are undoubtedly floating around campus and on the web. You may **not** refer to solutions while doing the assignments; they must be your own work. Many of the labs build on previous labs. If you do not turn in a lab, you may refer to the solutions handed out to work through the lab you missed to learn the skills needed for a subsequent lab. However, you may not simply copy another student's files.

Labs and homework are due by the end of class. You may have a one-week extension on one assignment of your choice (except Lab 11) and are responsible for tracking this yourself under the honor code (no need to notify the instructor; just turn it in with the following week's assignment). Your lowest problem set and lab grade will each be dropped. Please ration your extension and drops carefully lest you find yourself ill at the end of the semester and out of options. The class moves quickly and it is difficult catching up if you fall behind. Contact the Associate Dean of Academic Affairs if you have a protracted emergency.

You are welcome to discuss labs and problem sets with other students or with the instructor or lab assistants or tutors **after** you have made an effort by yourself. However, you must turn in your own work, not work identical to that of another student. For labs, asking classmates or tutors for help when you are stuck on a specific issue is encouraged (especially on difficulties with the tools and equipment), but sitting at adjacent computers and working through the lab together in lock-step is specifically prohibited. Pair/group programming is also prohibited. Be sure to credit at the top of your assignment anyone with whom you discussed ideas. **It is an honor code violation to simply copy someone else's work.**

Readings for each lecture are listed on the schedule below. Many students say they have found the readings valuable and enjoyable. You'll get the most out of the class if you read the sections in advance of the lecture and come with questions, and then reread as necessary when you work your problem sets and labs.

Tentative Schedule

Lecture	Date	Topics	Readings	Assignment
0	9/4	Introduction: digital abstraction, numbers	1.1-1.5	
1	9/9	Logic gates, Static discipline, transistors	1.6-1.9, A1-A7	
10	9/11	Combinational logic design	2.1-2.8	PS 1 due
11	9/16	Timing, sequential circuits	2.9-2.10, 3.1-3.2	Lab 1 due Digital Circuits
100	9/18	Finite state machines	3.3-3.4	PS 2 due
101	9/23	Dynamic discipline, metastability	3.5-3.7	Lab 2 due Comb Logic
110	9/25	Hardware description languages: Verilog	4.1-4.3	PS 3 due
111	9/30	Verilog, Part II	4.4-4.10	Lab 3 due Structural FSM
1000	10/2	Arithmetic circuits	5.1-5.2	PS 4 due
1001	10/7	Fixed and floating-point number systems	5.3	Lab 4 due Behavioral FSM
1010	10/9	Sequential building blocks, arrays	5.4-5.7	PS 5 due
1011	10/14	Catchup / Midterm Review		Lab 5 due Building blocks
	10/16	Midterm		
	10/22	HAPPY FALL BREAK!		
1100	10/24	C Programming		
1101	10/28	C Programming	C.1-C.7	
1110	10/30	Microcontrollers: Memory-mapped I/O	C.8-C.11	
1111	11/4	Parallel & serial interfacing, ADCs	9.1-9.3.3	Lab 6 due C Programming
10000	11/6	I/O libraries and examples	9.3-9.4	PS6 due
10001	11/11	ARM assembly language		Lab 7 due C I/O
10010	11/13	Function calls, machine language	6.1-6.3.6	PS 7 due
10011	11/18	Single-cycle processor datapath	6.3.7-6.9	Lab 8 due C Peripherals
10100	11/20	Single-cycle processor control, Verilog	7.1-7.3.1	PS 8 due
10101	11/25	Multicycle processor	7.3, 7.6	Lab 9 due Assembly
	11/27	HAPPY THANKSGIVING!	7.4	
10110	12/2	Pipelining	7.5.1-2	PS 9 due
10111	12/4	Advanced architecture: a sampler	7.7	Lab 10 due Multicycle Control
11000	12/9	Case study: ARM processors	6.7, 8.7, 8.5	PS 10 due
11001	12/11	Class summary and review		Lab 11 due Multicycle CPU