CMPE012 – Computer Engineering 12 (and Lab)

Computing Systems and Assembly Language Programming

Fall 2015
Additional Seats

- Additional Section is opening up for another 35 seats
- Will occur at 2PM today
- Section is Ming Ong Mon/Fri 12-2PM
About the Instructor

- UCSC Alumni
  - B.S in Computer and Electrical Engineering
  - M.S. in Computer Engineering
- Heavily involved in Mechatronics (the robot class) since undergraduate
- Graduate work was designing a Mountain Lion tracking collar
The Course

• Introduction to computer systems and assembly language and how computers compute in hardware and software.

• Topics include digital logic, number systems, data structures, compiling/assembly process, basics of system software, and computer architecture.

• May include C language.

• Prerequisite(s): course 3 or 8, or Computer Science 10 or 12A or 5C or 5J, or 5P, or Biomolecular Engineering 60, or suitable programming experience; previous or concurrent enrollment in course 12L required.
The Team

• Instructor
  – Max Dunne (mdunne@soe.ucsc.edu)

• Teaching Assistants
  – Megan Boivin < mcboivin@soe.ucsc.edu>
  – Kendal Lewis < klewis@soe.ucsc.edu>
  – Sina Hassani < shassani@soe.ucsc.edu>
  – Priyesh Vakayil < pvakayil@ucsc.edu>

• Tutors: Still being figured out
Our online presence

Our online presence...

• Ecommons [http://ecommons.ucsc.edu](http://ecommons.ucsc.edu)
  – Slides
  – Announcements
  – Assignments - Getting and submitting (labs only)
  – Calendar

• Online forum (Piazza)
  – Main forum for communication
The textbook

Patt, Patel

Introduction to Computing Systems:
From Bits and Gates to C and Beyond

McGraw-Hill, Second Ed.
What we will cover in this class

Part 1: Introduction to Logic Design
- Logic Gates and Functions (Ch 3)
- Integer Number Representation (Ch 2)
- Computing Systems & Abstraction (Ch 1)
What we will cover in this class

Part 2: The LC-3 computer system

- LC-3 Architecture (Ch 4)
- LC-3 Instruction Set Architecture (Ch 5)
- LC-3 Machine Language Programming (Ch 6)
- LC-3 Assembly Language Programming (Ch 7)
- LC-3 Input and Output (Ch 8)
- LC-3 TRAPS and subroutines (Ch 9)
- LC-3 Stack (Ch 10)
What we will cover in this class

Part 3: The PIC32 (Arduino) microcontroller (not in book)

- Microcontrollers and embedded systems
- PIC32 microcontroller
- MIPS assembly
- PIC32 I/O and interrupts
- MPLAB IDE (integrated Development Env.)
- PicKit3 Debugger
What we will cover in this class

Part 4: Fractional numbers

• Fixed and floating-point numbers and arithmetic
  – IEEE 754 Floating-Point
• Floating Point Arithmetic
  – Addition
  – Subtraction
  – Multiplication
  – Not division
What we will cover in this class

Part 5: The C programming language
• Intro to the C programming language (Ch 11)
• C variables (Ch 12)
• C control structures (Ch 13)
• C functions (Ch 14)
• C pointers and arrays (Ch 15, 16) - maybe

Taught in more detail in CMPE13

We will use C as our reference high-level programming language, with examples throughout. You will get an overview in C, but focus will be on other concepts.
What we will NOT cover in this class

- C recursive functions (Ch 17)
- Hardware design (CMPE 100/110)
- Extensive C coding (CMPE 13)
- Software engineering (many)
- Algorithms (CS 101)

This class is intended to be a bottom to top overview of computer systems. Other classes will cover material in greater details.
Required skills to pass the course.

1. Number representations, including
   a. arbitrary base conversion       b. binary, hex, decimal, 2’s C
   c. bitwise operators              d. Binary fixed point numbers
   e. single-precision floating-point format

2. Binary Arithmetic, including
   a. Signed magnitude add/sub       b. Unsigned add/sub/mul
   c. Two’s compliment add/sub/mul   d. IEEE floating point add/sub/mul

3. Computing Systems
   a. Basic logic gates (and, or, not, xor)
   b. Determining the function of simple combinational circuits
   c. Adder and mux logic blocks

4. Assembly language programming
   a. Arithmetic and bitwise operations   b. Procedure calls
   c. Stack & memory operations
   d. Assembly implementation of C control structures

5. An understanding of acceptable and unacceptable collaboration, the need to ensure permission to collaborate in a class, and an automatic urge to acknowledge collaborators and others who have assisted in a project.
Extended course description

Core topics:

1. Assembly language programming including
   a. Arithmetic and bitwise operations
   b. Arrays, stacks,
   c. Procedure calls
   d. Addressing modes
   e. Both CISC and RISC architectures
2. An understanding of basic computing systems including
   a. Basic logic gates and/or/xor/not
   b. Basic logic blocks (adder, mux)
   c. Registers, memory, CPU, I/O
   d. Steps to execute an instruction
   e. Data structures
3. Binary arithmetic
   a. Signed magnitude add/sub
   b. Unsigned add/sub/mul/div
   c. Two's compliment add/sub/mul
   d. Floating point add/sub/mul
4. Number representations, including
   a. Arbitrary base conversion
   b. Binary, hex, decimal, 2s C
   c. Bitwise operators
   d. Binary fixed point numbers
   e. Arbitrary bases (e.g., 3, 60)
   f. Biased representation
   g. IEEE Floating point format
5. An understanding of basic system software including
   a. Assembly and compilation
   b. Loading and linking
   c. The basic functions of the operating system
6. Interrupts and I/O
   a. Causes of interrupts
   b. Interrupt service routines
   c. Memory mapped I/O
Lab Kit

- Digilent Uno32
  - Arduino-compatible
- Digilent IO Shield
- PICKit3 Debugger
- Plastic case+cables+header
- Costs from BELS
  - ~$30 to rent for a quarter
  - ~$150 to buy
- Used in CMPE 13 and 118 too!

Get from BELS later in the quarter
Course work

• Class (CMPE12) Requirements
  – Attending lectures is highly recommended
  – Doing the weekly homework
    • Posted online, turned into class on Thursdays
    • Must be hand written, must be your OWN work
  – Quizzes + Midterm + Final

• Lab (CMPE12L) Requirements
  – Going to your lab section meetings
  – Weekly/Semi-weekly lab assignments
    • Posted online, submitted online
    • Lab report as PDF
Lab work

• Part 1: Logic design with Multimedia Logic
  – Simulated logic gates

• Part 2: Programming assignments in LC-3
  – Simulated architecture

• Part 3: Programming assignments in PIC32/Arduino
  – Real hardware kit
Lab rules

• Each lab assignment consists of two things
  – Lab work (code, design file, etc.)
  – Lab write-up
• Lab assignment score = code + write-up
• Assignments are turned in through eCommons
• **Must demo lab to TA/tutor in last lab before deadline**
• Must submit both the lab code and the write-up
• Make sure to make the deadline, since even 1 second late will stamp your assignment as...

LATE
Lab rules

- On time labs
  - Eligible for 100% of points, including extra credit
  - Eligible for 100% of points on the write-up
- Late labs
  - Lose all extra credit
  - Every day late, max score decreases by 10%
- Ex: Lab worth 40 points + 5 extra credit + 10 write-up
  - 1 day late: 0 extra credit, max points are 36 and 9
  - 2 days late: 0 extra credit, max points are 32 and 8
  - 3 days late: 0 extra credit, max points are 28 and 7
  - Etc
  - Always turn in something
Get your lab checked off

- Go through the grading checklist in lab with the TA/tutor **in person, in your lab**
- Lock in your score with your TA/tutor
- Do the lab write-up
- Submit the lab assignment by the deadline
  - Both the lab file(s) and the write-up file
  - Submission into eCommons is what really counts
- Check off is required
- Only the submitted files will be considered “official”, regardless of what you showed in class
Check off is before the deadline?

- Yes
- You MUST get a check off BEFORE the deadline during your lab sessions
  - Getting check off AFTER your last lab for the week will be considered late
  - Contact me before the deadline if you need an extension
- Final write-up and file submission deadline is a day or so after check off
No time? Submit it late

- Lab re-submission is allowed before deadline
- If you submit late, upload it to eCommons
  - You can still resubmit, but it will be marked later each time
- Get a check off in your next lab or TA office hours
  - They will use your copy from eCommons
  - They will go by the time stamp in eCommons
  - They might not provide help, depends on if other students working on next lab or homeworks
- Final lab assignment cannot be late, period
Attendance

- Highly recommended for the class
  - How will I know if you cut?
  - Pop quizzes
  - Announcements

- Mandatory for the lab
  - In lab, we may take roll
  - You may only attend your registered lab section

- Attending both lab sections are required
  - First and second day each week
  - Unless told otherwise
Grades for CMPE12/L

- 35% Labs
- 15% Homework
- 5% Quizzes
- 20% Midterm
- 25% Final

- Things you don’t turn in score as a 0
- Cannot make up missed exams
- Some grade curve is done
- Same Grade for the Lab as the class
Grade Tips

- Lab work reinforces lecture material
- Lecture must be understood to do labs
- Good homework and lab scores mean you will get good exam scores
  - Assuming you did the work yourself
  - Please work with integrity
  - Get help when you do not understand!!!
Special needs

• Students with special needs should refer to the Disability Resource Center (DRC)
• Notify me within first week of class
• We will accommodate your needs
• Confirm with me and SOE office at least 1 week before each exam so we may try to accommodate your needs
Academic Dishonesty (Cheating)

- Cheating is presenting someone else’s work as your own
- All code turned in will be run against a code-checker
- Anyone caught cheating will immediately fail the class and the lab, and be reported to their college
- Copying each other’s code is never acceptable.

- Don’t do it—not worth it.
DO NOT CHEAT!
Wait list and Attendance

- There are waits lists for the class and the lab
  - Priority if given on need, not position
  - Course is offered every quarter, sometimes in the summer

- Sign up for new section at 2pm
Practice Quiz

Put registered name and student ID at top

Write neatly!!

Answer the following questions to see if you are ready for CMPE12

1. \(2 + (-1) = \)

2. \(-3 - (-4) = \)

3. \(1.25 \times 4 = \)

4. What gets printed?

\[\begin{align*}
x &= 2; \\
\text{if } (x == 3) \\
\text{puts “bob”;} \\
\text{else} \\
\text{puts “steve”;}
\end{align*}\]
Week #1

- Labs start next week
History of Computers
The History of Computers

The history of computers is interesting (or should be if you are in this class) and relevant to our professional lives.
The First Computer Hardware

Charles Babbage, born 1791
- Father of the computer
  - 1830 **Difference engine** - used mechanical power
    - calculated mathematical tables
    - smallest imperfections caused errors
    - Funded by the British government
- Funding was pulled, even his colleagues thought it wouldn't work
  - conceived of **analytical engine** to perform many types of calculations
  - son built a model of the machine
  - working version finally built 1991
The First Programmer

Ada, the countess of Lovelace

- Mother of computer programming – the first programmer!
- A gifted mathematician.
- She helped develop instructions for computations on the analytical engine.
- Saw Babbage's theoretical approach as workable.
History of Computers

The First Electrical computer

1890 Herman Hollerith
- Able to count the census in 6 weeks rather than 7 years
  - Used Jacquard’s punch cards
    - Sorted into bins
    - Count number of cards
- Developed in 1800 by a French silk weaver
- Electrical power
- Tabulating machine company merged into IBM in 1924
History of Computers

Aiken, Zuse, Atanasoff, Berry

- 1936 - Harvard graduate student Howard Aiken began thinking of modern equivalent of analytical engine...
- 1939 Germany - Konrad Zuse completed first programmable, general-purpose calculating device to solve mathematical problems
  - Paper was in short supply during war, used film tape
- 1939 - Iowa State Professor John Atanasoff developed the first electronic digital computer, the Atanasoff-Berry Computer (ABC)
  - Above is a picture of Berry
1944 Harvard professor Howard Aiken completed the Mark I
- Assistant Grace Hopper
  - Developed compiler for the computer
- 8 feet high, 55 feet long steel and glass
- used noisy electromechanical relays
- 5-6 times faster than a person
- not very efficient
Enter data into computer using paper tape
First Computer “Bug”

Found on the 9th of September, 1945, by Grace Murray Hopper while she was working on the Harvard University Mark II Aiken Relay Calculator (a primitive computer).

Coined term “debug”.

Maxwell James Dunne – Fall 2015
History of Computers

ENIAC, UNIVAC by John Machly & J. Presper

WWII - ENIAC Electronic Numerical Integrator and Computer
- based on the ABC
- machine to calculate trajectory table, new guns
- First general-purpose computer

June 14, 1951
- UNIVAC 1 - Universal Automatic Computer
- First general purpose commercial computer

Maxwell James Dunne – Fall 2015
Four generations of computers

1. 1951-1958 Vacuum Tube
   - about the size of light bulbs
   - thousands of them
   - is the bug a problem with tube or program?
   - machine code and punch cards

2. 1959-1964 Transistor
   - transfers electronic signals across \textit{resistor}
   - assembly language
   - 1954 - FORTRAN

\textbf{Transistor}
\textit{mighty mite of electronics}
Four generations of computers

3. 1965-1970 Integrated Circuit
   - complete electronic circuit on a small chip of silicon
   - silicon is a semiconductor - will transmit electrical signal when specific chemical impurities are introduced to lattice structure.
   - IBM 360 series of IBM
   - first time small and medium businesses could afford a computer.
   - unbundle software - sell software separately
   - birth of software industry

4. 1971-PRESENT Microprocessor (VLSI)
   - extension of third generation
   - get specialized chips for memory and logic
The Next Generation?

• I think the next generation is upon us and you are seeing it in your daily lives. I call it the SOC – System-on-a-Chip generation.
  – Put everything on a single chip
    • CPU
    • GPU
    • Memory (or at least part of it)
    • IO
  – Enables very low power with high performance
    • Smart phones, tablets, etc.
  – Also hear it called Ubiquitous Computing
    • Computing everywhere - Internet of Things (IoT)
History Summary

- Knowing something about the evolution of computers is helpful to understanding why things are the way they are now
- Computing devices have been around a long time
- Digital computers are fairly new
- Rate of improvement and growth is amazing, Moore’s Law