COMPUTER ORGANIZATION & ASSEMBLY LANGUAGE
Syllabus - Spring 2024
CSC 230 SECTION 16766
EEE 230 SECTION 16767

Instructor: William Meacham Phone: 480 423-6664
Email: William.Meacham@Scottsdalecc.Edu

Office Hours:

Class Format and Time

Title	Section	Days	Time	Location	Format	Credit Hours
CSC230 EEE230	16766 16767	TR	10:00 - 11:50	Live-Online	Live-Online	4

Description: Assembly language programming including input/output (I/O) programming and exception/interrupt handling. Register-level computer organization, I/O interfaces, assemblers, and linkers. Processor organization and design, data path, control, pipelining, and input/output. Memory organization with cache and virtual memory. Prerequisites: A grade of C or better in (CSC100 or CSC110) and CSC/EEE120, or permission of Instructor.

Official Course Objectives

1) Identify the five major components of a computer system and describe how the components interact and the effect on the system as a whole.
2) Describe data representation, the instruction set, addressing modes, and register organization.
3) Describe execution of instructions and hardware components used in each step.
4) Manipulate data to perform arithmetic operations and floating point.
5) Measure, report, and summarize performance of a computer and describe the major factors that determine it.
6) Describe three approaches to processor implementations: single-cycle, multi-cycle, and superscalar.
7) Describe how pipelining enhances processor performance.
8) Describe data path, control, and branch hazards.
9) Describe virtual memory and cache organization structures.
10) Describe data storage, networks, and peripherals.
11) Describe what busses are and their role in connecting the major system components.
12) Describe how the processor identifies different sources of interrupts and exceptions, and invokes the corresponding handler to deal with the interrupt or exception.
13) Use all registers in assembly language programs.
14) Determine appropriate registers to use with specific assembly language commands.
15) Use addressing modes to move information and perform computations.
16) Use conditional and unconditional branches to create loop, if-then, and case branching structures.
17) Write call procedures with parameter passing.
18) Use input/output interrupts to control a hardware device.

COURSE OUTLINE

I. Five classic components of a computer:
   A. Data Path
   B. Control
   C. Memory
   D. Input
   E. Output

II. Instructions: language of the computer
   A. Operations and operands of computer hardware
   B. Instructions in the computer and logical operations and instructions for making decisions
   C. Addressing for 32-bit immediates and addresses
   D. Translating and starting a program and compiler optimization and function
   E. Implementing an object-oriented language and arrays versus pointers

III. Arithmetic for computers
   A. Signed and unsigned numbers
   B. Addition, subtraction, multiplication, division
   C. Floating point

IV. Assessing and understanding performance
   A. CPU performance and its factors
   B. Evaluating performance

V. Processor: datapath and control
   A. Logic design conventions and building a datapath
   B. A multicycle implementation and exceptions
   C. Microprogramming: simplifying control design
   D. Introduction to digital design using a hardware design language
   E. The organization of a microprocessor implementation

VI. Enhancing performance with pipelining
   A. A pipelined datapath and pipelined control
   B. Data hazards, forwarding, and stalls
   C. Branch hazards and using a hardware description language to describe and model a pipeline
   D. Exceptions

VII. Large and fast: exploiting memory hierarchy
   A. Caches and measuring and improving cache performance
   B. Virtual memory and common framework for memory hierarchies

VIII. Storage, networks, and peripherals
   A. Disk storage, dependability, and networks
   B. Buses and other communications between processors, memory, and I/O devices
   C. Interfacing I/O devices to the processor, memory, and operating system
   D. I/O performance measures and designing an I/O system

LMS: Course Materials, grades, assignments, slides, etc, will be via Canvas.

Textbook (Required): zyBooks Edition of "Patterson & Hennessy, Revised Printing Computer Organization & Design" The text will be accessed directly through Canvas. See the instructions on Canvas.

Textbook (Required): Introduction To MIPS Assembly Language Programming, Charles W. Kann, Gettysburg College. Free to download. It will also be available within Canvas.

Software: MARS Simulator MARS MIPS simulator - Missouri State University

Attendance:

CSC230 and EEE230 will be delivered in a Live-Online Format. The live-online class will open up at 9:55 p.m. and students are expected to be online by 9:58. Otherwise, they will be counted as late. Attendance is required. To be counted as present, you are expected to be prepared, be on time, participate in group/class discussions, and stay the full length of class. If you have a legitimate reason to be late or leave class early, let me know before the start of class. You may be withdrawn from the course if you miss more than three classes.

You are responsible for all missed notes, concepts and assignments if you miss class. Check with your classmates for anything you may have missed.

In Class Work/Homework/Quizzes: (4% of your grade). There will be in-class work, homework and Quizzes throughout the course.

- In-Class work will consist of programs written in class and will be collected/uploaded (electronically) periodically.
- Homework will be turned in as a pdf electronically and must be completed on engineering paper.
- Quizzes will be taken online.

zyBooks Participation Problems/zyBooks Challenge Problems: (10% of your grade). zyBooks Participation and Challenge Problems will also be completed online. It is important that you only access these problems through Canvas so your grades are recorded.

Programming Assignments and zyBook Labs: (40% of your grade). Programming Assignments and Hardware Labs will be posted on Canvas. They will be graded according to the Programming Assignment Guidelines and the Rubric provided. These guidelines will detail rules for writing your program, formatting requirements and submission requirements (how you send your code to me).

Due Dates: Due dates for each assignment (HW, Programs, Labs) are on Canvas.

- Homework and zyBooks Participation/Challenge Problems will not be accepted late.
- Programming Assignments and zyBook Labs turned in within 2 days of the due date will be penalized 10%. Programming Assignments and zyBook Labs turned in within 2 weeks of the due date will be penalized 50%. Submissions turned in later than two weeks late or after the final exam will not be accepted.

Exams: (36% of your grade). There will be three exams during the semester, each worth 12%. All will be given via Lockdown Browser. There will not be a cumulative final exam. You are expected to take the test at the scheduled time. If an emergency arises that is **beyond your control**, you must contact me **on or before** the day of a test to arrange for a make-up test. Make-up tests are possible only in special circumstances, at the instructor's discretion. All make-ups, if granted, must be **completed** before the exams are returned to the class. Failure to adhere to this policy may result in a grade of zero for the test missed. You are only allowed **one** make-up test per semester.

Lockdown Browser: Lockdown Browser: The three tests will be taken using Lockdown Browser. Lockdown Browser is free to download, but is not available on a Chromebook. Lockdown Browser

also requires that you have a webcam and microphone. See the Lockdown Browser folder in Canvas for additional details.

Final Project: (10% of your grade). There will be a final project where students will work in groups of two. Requirements for the project will be posted on Canvas. Projects will be demonstrated on the last day of class and will not be accepted late.

Office Hours: I will be online Tuesdays and Thursdays. I will also open up this class at 9:30 a.m. Please feel free to drop in at any time. For meetings via ZOOM outside of those hours, please schedule an appointment using your school Google Calendar. I am very flexible on times we meet, including over the weekend. My only request is that if you make an appointment, please be sure to not miss it.

Mandatory Check-In Meetings: You will be required to meet with me for at least one mandatory check-in meetings during the semester. One will be after exam 1 as an opportunity to discuss your academic performance, offer insightful feedback, and address any other related issues during this time. You will need to schedule your virtual or in person meeting using the Google Calendar in your student account. It can be at any time I am shown as available on the calendar.

The other two can be any time during the semester. It could be to work together on an assignment or to discuss topics in the class. For students attending In-Person, these two meetings will not be required.

Speaking of Ethics… I encourage students to help each other as needed and to get help from our Tutors often. But please make sure you write your own code on all programming assignments for this class. You will have to write similar code on all of the tests, so it is important that you have worked through the code. Students handing in code they did not write will have their grade for the course lowered 10% depending on the complexity of the assignment. Students handing in code downloaded from sites such as CourseHero or any AI generated code will be given an immediate "F" for the class.

Grading:

Point Allocation	
40%	Programming Assignments and Labs
10%	zyBooks Participation and Challenge Problems
10%	Final Project
4%	In Class Work/Quizzes/Homework
36%	Exams

Grading Scale	
A	90% - 100%
B	80% - 89%
C	70% - 79%
D	60% - 69%
F	0% - 59%

Week	Topics (Reading)
Module 0 Review from CSC120	Module 0 – Course Content – Review Video's (on Canvas)
Module 1 1/30 and 2/1	Syllabus and Course Structure 1.1 - MIPS Architecture Overview (Ch 1.1 – 1.5, 2.1) 1.2 - Numbers in Different Systems Thru ASCII/Unicode (2.4, 3.5) 1.3 - Intro to MARS and MIPS – First Program (2.1, 2.2, 2.3, Appendix A.7.1)
2/6 and 2/8	1.4 - MIPS Program Elements (2.3, Appendix A.7.1) 1.5 - Integer Arithmetic (2.1, 2.2, 2.5, 3.1, 3.2, 3.3, 3.4, 3.5) 1.6 - MIPS Data Segment, Data Types, and Data Transfer (2.3)
Module 2 2/13 and 2/15	2.1 – Translating Assembly Language into Machine Code (2.5) 2.2 - Logical Operators and Bitwise Operators in MARS and MIPS (2.6) 2.3 - Syscall's in depth and making life easier 2.4 – Bitmap Display!!
2/20 and 2/22	2.5 - Boolean Expressions and If Statements (2.7) 2.6 - Loops (2.7) 2.7 – Keypad
Module 3 2/27 and 2/29	3.1 - Simple and Reentrant Subprograms (2.8, 2.9, A.7.6) 3.2 - Translating Branches and Jumps into Machine Code (2.10)
3/5 and 3/7	3.3 - Translating and Starting a Program (2.12) 2.7 - Continued MMIO
Module 4 3/19 and 3/21	3.4 - Arrays (2.13, 2.14) 3.5 - Dynamic Memory Allocation
Module's 4 & 5 3/26 and 3/28	3.6 - Exceptions (Appendix A.7.7) 3.7 – Etch a Sketch 4.1 – Compiling C and Java Putting it all together – (2.15 and Appendix A)
4/2 and 4/4	4.2 – Addition and Subtraction 4.3 - Multiplication and Division 4.4 – MIPS Performance, CPI
4/9 and 4/11	5.1 - DataPath and Pipelining (4.1, 4.2) 5.2 - Building a Datapath (4.3) 5.3 - A Simple Implementation Scheme (4.4)
Modules 5 4/16 and 4/18	5.4 - Multicycle Implementation (4.5) 5.5 - Overview of Pipelining (4.6)
4/23 and 4/24	5.6 - Pipelining, Datapath and Control 5.7 - Hazards and Exceptions 6.1 - Memory hierarchies; cache basics
4/30 and 5/2	6.2 - Cache Basics Pt 2 6.3 – Improving Cache Performance 6.4 - Virtual Memory
5/7 and 5/9	Lab Time
5/12	Test 3 and Final Project Display

Disclaimer: Students are responsible for the information contained in this syllabus. The information in this syllabus is subject to change based on the discretion of the instructor.