CS 2318 Summer 2015
Assembly Language
Summer 2018 - Syllabus

Course Catalog Description: (link)

This course covers the organization of digital computers; assembly language programming including addressing, looping, logic, shifting and masking operations, macros, subroutines, co-routines, arithmetic algorithms, and recursion.

Course Objectives:

- Be able to design and develop assembly language level applications
- Gain Familiarity with digital information representation (binary, hex, floating point, Unicode, ...)
- Understanding of how the microarchitecture and instruction set architecture work together with operating system and programming languages to accomplish computer application objectives
- Obtain increased capability in the areas of abstraction, separation of concerns, and recursion

More Details

- Know how program control is achieved in an assembly language: sequencing, alternatives, looping, function call and return.
- Know the role played by the Program (Instruction) Counter.
- Describe the actions taken by a CPU (controller) in the 4 phases of the Von Neumann machine cycle. Know the basic parts of an instruction format and how they are used in the cycle.
- Write assembly code implementing the algorithms described in the course; in particular, be able to write code that uses (some form of) indexing for array or string processing.
- Understand the process of assembling, linking, and executing a program.
- Understand the distinction between a flat space model and a segmented space model. Know the distinction between an assembled address, linked address and actual (run time) address.
- Know the role of the assembler's Location Counter during the assembly process.
- Write structured code in assembly.
- Convert numerical data from one format to another.
- Explain conventional formats for negative integers in fixed length word architectures.
- Trace a simple assembly program using a debugger.
- Debug common errors reported by assemblers and linkers.
- Understand some of the trade-offs between CISC and RISC architectures.
- Know how to interface and link to high level language modules.
- Describe a divide-and-conquer approach to problems.
- Implement a recursive algorithm in assembly language including the use of a stack for local variables, call parameters and return addresses.
- Know how interrupts and traps are handled on at least one computer architecture.
- (Optional) Know the process by which executable code is loaded into memory in at least one Operating System.

- (Optional) Know how simple algorithms are to be implemented on a strictly stack based architecture.

Prerequisite

- C or higher in MATH 2358: Discrete Mathematics I
- C or higher in CS 2308: Foundations of CS2 or concurrent enrollment in this course

Instructor:

Dr. Dan E. Tamir:
dt19@txstate.edu
Office 512-245-7528; Cell: 512-739-9277

Lectures:

MTWRF – 10:00a-11:40a DERR 236

Office Hours:

- **By appointment**; MWF - 11:00a – 1:00p pm; Comal 311F
- **By appointment**; TR - 3:00p – 4:00p; Avery 464-x
- Other times – by appointment

Required Work / Grading

~6 HW assignments
Quizzes
Midterm exam:
Final exam:

Grading

- HW assignments 30%
- Quizzes/attendance 10%
- Midterm: 25%
- Final: 35%

Course Resources

Main channel of communication is TRACS

Required Textbook

- *MIPS Assembly Language Programming* – Britton, R. L., Prentice Hall, 2004.
- C Programming Wikibook – Free Book please load immediately
- Computer Organization and Design Fundamentals by David Tarnoff - Free Book please load immediately
- Chapter two of the Patterson and Hennessy, *Computer Organization and Design; The Hardware/Software Interface*, 3/e or later, Morgan Kaufmann (Elsevier), 2009. Some versions are available on the internet please load immediately.

Supplemental Material

- The SPIM (or XSPIM/QtSPIM) Simulator + Documentation
- Rest of the Patterson book

WWW Resources:

- MIPS Technologies Inc.
- PPC – IBM
- PPC – Freescale
- ARM

Programming Assignments

Programs are submitted on TRACS by the due time/date Please do not submit your assignment via email. If you miss the deadline, then please submit it on TRACS and send me an email notification. Email submission will be deleted

Academic Honesty:

All work submitted for a grade is expected to be your own. As a guideline, you may talk together, but do not write together. Projects may be subject to review through TurnItIn. Students in this class are expected to adhere to the Texas State University Honor Code.

Attendance:

Regular and punctual attendance is required.

Academic Policies:

See Student Handbook and Registrar home page for more information about Texas State Academic Policies including probation, suspension, academic honesty, incompletes, drop, withdraw, and grade changes.

Special Needs:

Students with special needs as documented by the Office of Disability Services should identify themselves at the beginning of the semester.

List of Topics

Date	Topic	Notes
June 4	Introduction	Ch1
June 5	The MIPS Architecture	Ch1/Patterson
June 6	The MIPS Architecture	Ch1/Patterson
June 7	Instructions and Macros	Ch1
June 8	Instructions and Macros	Ch1
June 11	Algorithm Development in Pseudo code	Ch2
June 12	Assembler/Linker/Loader/Directives	Ch2/Appendix B / Patterson
June 13	SPIM - The MIPS simulator	Ch4/ SPIM Manual
June 14	Function Calls	Ch6
June 15	Function Calls	Ch6
June 18	Midterm	
June 19	Efficient Algorithm Development	Ch5
June 20	Function Calls; Arrays and Pointers	Ch6;
June 21	Dynamic Memory Allocation	Ch6
June 22	Number Systems	Ch3/Tarnoff Ch2/CH3
June 25	Number Systems	Ch3/Tarnoff Ch2/CH3
June 26	Number Systems	Ch3/Tarnoff Ch2/CH3
June 27	Reentrant / Recursive Functions	Ch7
June 28	Recursive Functions	Ch7
June 29	Memory Mapped I/O	Ch8
July. 2	Exceptions and Interrupts	Ch9
July. 3	Pipelined Implementation;	Ch10
July. 5	Performance Evaluation	Handout Patterson Ch1
July. 6	Final	11:00-1:30, in class