CSCI 0039 - INTRODUCTION TO COMPUTER ARCHITECTURE AND ASSEMBLY LANGUAGE

Catalog Description

Prerequisite: Completion of CSCI 10 with grade of "C" or better
Hours: 72 (54 lecture, 18 laboratory)
Description: Assembly language programming techniques and introductory computer architecture concepts. Topics include addressing modes; pseudo operations; stack processing; subroutine linkage; arithmetic and logical operations; input and output; digital logic. Programs are designed, coded, tested, and debugged. (C-ID COMP 142) (CSU, UC)

Course Student Learning Outcomes

- CSLO #1: Write simple assembly language program segments.
- CSLO #2: Implement a machine level language program based on the given high level language program.
- CSLO #3: Explain the internal representation of integer, floating point and non-numeric representation of data.
- CSLO #4: Explain addressing modes used to access data.
- CSLO #5: Use fundamental logic gates to design and implement combinational and sequential digital circuits.

Effective Term

Fall 2020

Course Type

Credit - Degree-applicable

Contact Hours

72

Outside of Class Hours

90

Total Student Learning Hours

162

Course Objectives

Lecture Objectives:
1. Describe the programmer's model of the target computer.
2. Describe the operation and use of an assembler.
3. Pseudocode algorithms that perform multi-byte arithmetic.
4. Pseudocode algorithms that perform code conversion.
5. Pseudocode algorithms that make use of the input-output devices, such as the USART, on the microcontroller.
6. Demonstrate the ability to convert high-level pseudocode algorithms to assembly language.
7. Compare and contrast RISC and CISC architectures.
8. Compare and contrast microcontrollers and microprocessors.
Laboratory Objectives:
1. Demonstrate the ability to use an assembly language integrated development environment.
2. Write assembly language programs that use multi-byte arithmetic and memory load-store operations.
3. Write assembly language programs that make use of LEDs as output devices and switches as input devices.
4. Write assembly language programs that perform code conversion.
5. Write assembly language programs that make use of common peripherals including the USART, timer-counter, analog to digital converter, pulse-width modulator, and EEPROM.
6. Write assembly language programs that interface to external peripherals using the SPI or I2C bus.
7. Write assembly language programs that interface assembly language modules to a high-level language program.
8. Write assembly language programs that demonstrate fundamental operating system concepts.
9. Design combinational logic circuits.

General Education Information

- Approved College Associate Degree GE Applicability
- CSU GE Applicability (Recommended-requires CSU approval)
- Cal-GETC Applicability (Recommended - Requires External Approval)
- IGETC Applicability (Recommended-requires CSU/UC approval)

Articulation Information

- CSU Transferable
- UC Transferable

Methods of Evaluation

- Objective Examinations
  - Example: 1. Convert the binary number 01001010 to decimal. 2. Which registers in the arm Cortex-M are used for the stack pointer, link register, and program counter?
- Problem Solving Examinations
  - Example: The following assignment is evaluated based on program functionality (50%), appropriate division of labor in subprograms (30%), and quality of documentation (20%): Write an arm assembly language program that implements the four fundamental 64-bit math operations. Each binary operator subprogram should accept its operands though the R0 and R1 registers. Assume the operands are stored in little-endian notation. Rubric Grading.
- Skill Demonstrations
  - Example: The following assignment is evaluated based on program functionality (50%), object-oriented design techniques (30%), and quality of documentation (20%): Write a program that counts in binary from 00000000 to 11111111. Your program should display each number on the LEDs connected to port B of the microcontroller. You will need to implement a 0.5 s time delay subprogram that is called after each number is displayed. Rubric Grading.

Repeatable

No

Methods of Instruction

- Laboratory
- Lecture/Discussion
- Distance Learning

Lab:

1. Following an instructor discussion on the if/else in assembly language, students will be given a problem to be solved using if/else in assembly language. (Laboratory Objective 7)

Lecture:

1. When teaching the topic of addressing modes, the faculty uses primarily lecture. Each addressing mode is discussed in turn and multiple examples of each mode are written on the white board. Special attention is paid to indexed addressing and its use. Determining the length of a null terminated string is used an example of where indexed addressing is necessary. Students are assigned multiple labs that demonstrates their knowledge of addressing modes. (Lecture Objective 1)

Distance Learning

1. Online lecture and discussion about microcontrollers and microprocessors, followed by students comparing and contrasting microcontrollers and microprocessors in a report that is posted for other students to review and provide comments.(Lecture Objective 8)

Typical Out of Class Assignments
Reading Assignments

1. Read Chapter Three in Upton, Electronic Memory (pages 47-91). Create a table showing the various types of memory, a short description of each type, and the access time in nanoseconds and be prepared to discuss in class. 2. Read Chapter four in Upton, System on a Chip (SOC) and be prepared to discuss in class.

Writing, Problem Solving or Performance

1. Implement the binary to ASCII decimal algorithm discussed in class. Use register R16 to pass the binary number into the subprogram, and a four byte memory buffer to store the result. Test Data Pass each of the following numbers to your binary to ASCII decimal subprogram: 0b00000001 0b11111111 0b10100101 0b00000000 0b11001100 You must call the subprogram once for each test datum. So there should be 5 calls to your subprogram in your main program. Make sure you clear the memory buffer before each call. 2. Write a pair of programs that count and display the number of times the switch connected to the low order bit of Port D is pressed. The first program accepts the switch input without debounce, and the second program adds a debounce delay. Recall that when a switch is pressed, the corresponding bit of the input port goes to logic-level 0.

Other (Term projects, research papers, portfolios, etc.)

Required Materials

- Learning Computer Architecture the the Raspberry Pi
  - Author: Eben Upton, Jeff Duntemann, Ralph Roberts, Tim Mamtora, Ben Everard
  - Publisher: Wiley
  - Publication Date: 2016
  - Text Edition: 1st
  - Classic Textbook?: No
  - OER Link:
  - OER:
- Modern Assembly Language Programming with the ARM Processor
  - Author: Larry D. Pyeatt
  - Publisher: Newnes
  - Publication Date: 2016
  - Text Edition: 1st
  - Classic Textbook?: No
  - OER Link:
  - OER:
- Raspberry Pi Assembly Language Programming: ARM Processor Coding
  - Author: Stephen Smith
  - Publisher: Apress
  - Publication Date: 2019
  - Text Edition: 1st
  - Classic Textbook?: No
  - OER Link:
  - OER:

Other materials and-or supplies required of students that contribute to the cost of the course.