Miami Dade College

| GENERAL INFORMATION | |
|---|---|
| Course Prefix/Number: CET2123C | Course Title: **Microprocessors** |
| Number of Credits: 4 credits | |
| Degree Type | ☐ B.A. ☐ B.S. ☐ B.A.S ☒ A.A. ☐ A.S. ☐ A.A.S. ☐ C.C.C. ☐ A.T.C. ☐ V.C.C |
| Date Submitted/Revised: 5/4/12 | Effective Year/Term: 2012-1 |
| ☐ New Course Competency ☒ Revised Course Competency | |
| Course Description (limit to 50 words or less): Students will learn to apply digital principles to the understanding of microprocessor parameters and characteristics (addressing range and models, instruction set, architecture, input/output, interrupts, and programming). Laboratory fee. ( 2 hr. lecture 4 hr. lab ) | |
| Prerequisite(s): CET1110C, and COP2270 | Corequisite(s): EET1141C |

**Competencies:**

**Competency 1:**
The student will demonstrate an understanding of the history of microprocessors and major recent computer developments by:

1. Outlining the history of computers.
2. Explaining the role of computers in business, scientific usage, control systems, and the military.
3. Defining and explaining microprocessors, floppy disc memories, large scale integration and microprogramming.
4. Recognizing terminology used in technical literature and in industry.

**Competency 2:**
The student will demonstrate an understanding of the fundamental hardware circuitry and architecture of modern digital computers by:

1. Defining list components and peripherals of a typical personal computer.
2. Defining assembly language and high level language and explaining the advantages of an assembly language over high level languages.
3. Explaining the function of each component: microprocessor, memory, and input/output (I/O) devices and their line of communication (buses).

**Competency 3:**
The student will demonstrate an understanding of the data flow inside the microprocessor by:

1. Explaining the terms low level and high level languages.
2. Defining operating system terms and commands and explaining their functionality, including STO, HALT, and FLAG.
3. Explaining chip design terminology and functionality, including Small Scale Integration (SSI),

Revision Date:
Approved By Curriculum Report: 92 Reviewed By Director of Academic Programs Date: _____

Medium Scale Integration (MSI), large Scale Integration (LSI), bit, byte.
4. Explaining the terms ASCII, word, instruction, software, and hardware.
5. Explaining the logic instructions, and recognizing the flags that are set or reset for given data conditions.

**Competency 4:**
The student will demonstrate an understanding of the microprocessor architecture, microcomputer systems, and memory interfacing by:

1. Defining the address bus, the data bus, and the control bus and explaining their functions in reference to the 8085 microprocessor.
2. Explaining the functions Reset, Interrupt, Wait, and Hold.
3. Explaining memory organization and memory map, and explaining how memory addresses are assigned to a memory chip.
4. Recognizing the functions of various pins of the 8085 microprocessor.
5. Listing the various internal units that make up the 8085 architecture, and explaining their functions in decoding and executing an instruction.
6. Drawing the block diagram of an 8085-based microcomputer.

**Competency 5:**
The student will demonstrate an understanding of the basic software techniques, including both machine and assembly languages by:

1. Explaining the functions of data transfer (copy) instructions and how the contents of the source register and the destination register are affected.
2. Explaining the functions of the machine control instructions HLT and NOP.
3. Writing a program in 8085 mnemonics to illustrate an application of data copy instructions, and translating those mnemonics manually in their Hex codes.

**Competency 6:**
The student will demonstrate a basic understanding of the fundamental principles of digital computers and computer circuitry by:

1. Identifying the basic components of digital computers, including, Input devices, Control element, Storage, Arithmetic element, Output devices.
2. Explaining the central processing units (CPU) operation and processes.
3. Demonstrating the use of software to examine the operation of the CPU.
4. Analyzing BUS concepts.
5. Identifying and analyzing addressing concepts.
6. Writing, assembling, executing, and debugging assembly language instructions and programs
7. Identifying the various types of RAM and ROM memories and explaining how they interface to the microprocessor.
8. Interfacing input and output devices with the microprocessor.

**Competency 7:**
The student will demonstrate an understanding of digital logic by:

1. Defining and using the instruction word.
2. Charting the flow of a basic program.
3. Writing a program in machine language, in assembly language, and in compiler language.

4. Using unconditional and conditional Jump instructions in programs.

**Competency 8:**
The student will demonstrate how to apply the basic rules of arithmetic for positional significance number systems to those systems normally used in the computer field by:

1. Defining the characteristics of a positional significance number system with any radix.
2. Converting between binary, octal, decimal, and hexadecimal number systems.
3. Performing simple arithmetic operations in the binary and hexadecimal number systems.
4. Identifying the characteristics of common computer information codes.

**Competency 9:**
The student will demonstrate an understanding of the functions and inter-relationships of the elements that comprise a microprocessor based computer by:

1. Drawing the block diagram and describing the basic architecture of a microcomputer.
2. Identifying and giving functional descriptions of data, address, and control buses.
3. Describing the internal architecture of the 8085 microprocessor.
4. Explaining the function and operation of each register in the 8085 microprocessor
5. Describing the sequence of operations in the execution of a microprocessor instruction.

**Competency 10:**
The student will demonstrate how to use assembly language mnemonics to write programs and how to code assembly language instructions in binary by:

1. Writing data-handling and arithmetic instructions
2. Writing logic instructions.
3. Creating condition code testing and branching instructions.
4. Programming registers and stacks operations.
5. Programming communication between the computer and I/O devices.