

## **CSC 214 - COMPUTER ORGANIZATION**

**CREDIT HOURS:** 3

**PREREQUISITES:** CSC 202

**GRADE REMINDER:** Must have a grade of C or better in each prerequisite course.

### **CATALOG DESCRIPTION**

Binary representation of data and instructions, arithmetic and logical data operations, hardware structures, memory organization and addressing of data and data structures. Machine language and assembly language instructions and programming, hardware/software interface, and selected programming techniques.

### **PURPOSE OF COURSE**

To understand the basic design and operation of a computer system by progressing from designing low level logic circuits to programming in low and high level languages. Upon completing of this course, students should have a complete understanding of the role played by each major component of the system, including hardware, the instruction set, language translators, and the operating system.

### **EDUCATIONAL OBJECTIVES**

Upon successful completion of the course, students should be able to:

1. Demonstrate knowledge and computational ability in numeric representation systems (binary, hexadecimal).
2. Determine design of combinatorial logic structures using the basic electronic building blocks.
3. Evaluate physical memory and memory organization, data storage, and memory access (addressing - machine and symbolic).
4. Demonstrate an understanding of the standard models of computers including the instruction fetch cycle and the physical components involved in this process; memory, CPU, I/O.
5. Demonstrate skills in problem analysis and program design.
6. Evaluate logic and arithmetic operations and conditions.
7. Develop decision and iteration control constructs in a low-level language.
8. Demonstrate skills in machine language programming and recognize how the hardware components are controlled.
9. Demonstrate an understanding of symbolic assembly language programming and the process of translation into machine language.
10. Understand high level language programming by looking at the compiled program as it is translated into assembly language and into machine language.
11. Solve a series of problems involving numeric (integer and floating point), character, and string data including data conversion techniques.
12. Create appropriate test data and apply debugging strategies.
13. Utilize operating system input and output routines.

## CONTENT

## Hours

Bits, Data Types, and Operations . . . . .	6
Bits and the concept of a data type	
Binary-Hex-Decimal conversions	
Bit operations; addition and subtractions, sign extension, overflow	
Logical operations, AND, OR, NOT, XOR	
Integer and floating point data types, ASCII codes	
Digital Logic Structures . . . . .	9
Logic gates	
Boolean algebra and DeMorgan's Law	
Latches, registers, and combinatorial logic structures	
Memory design; address space and addressability	
Program Execution and Programming . . . . .	3
Basic components of memory, arithmetic logic unit, control unit, and I/O strategies	
Instructions and the instruction cycle	
Sequential, conditional, and iterative control constructs	
Problem solving and decomposition; debugging operations and strategies	
Machine Language Concepts and Programming . . . . .	9
Instruction Set Architecture	
Memory organization, addressing modes	
Operate, data movement, and control instructions	
Registers, data types, and condition codes	
Assembly Language Concepts and Programming . . . . .	9
Instructions and directives	
The two pass translation process: creating the symbol table, generating machine instructions	
Creating an object file and an executable image; multiple object files	
Subroutine calls and returns, activation records, and the system stack	
Data type conversion; ASCII to binary, binary to ASCII	
Input and Output in low-level programming . . . . .	3
Input from the keyboard	
Output to the monitor	
Interrupts and other output strategies	
High-level Language Utilization of Hardware Components . . . . .	3
Translating high-level language programs and analyzing machine language compiler output	
Data types and variables, global and local scope	
The symbol table and allocating space for variables; literals, constants, and symbolic values	
Expressions and statements; arithmetic and logical operators	
Control structures, data structures, and pointers	
Functions and parameter passing	

Exams (plus final) .....	3
	TOTAL 45

**REFERENCES**

Patt and Patel, Introduction to Computing Systems Second Edition, McGraw-Hill, 2004.

Petzold, Code, Microsoft Press, 1999.

Tannenbaum, Andrew S., Structured Computer Organization, 4th Ed., Prentice Hall, 1999.

Null and Lobur, Essentials of Computer Organization and Architecture, Jones & Bartlett, 2003.