COE 205 Syllabus – Fall 2005
Computer Organization & Assembly Language

Computer Engineering Department
College of Computer Sciences & Engineering
King Fahd University of Petroleum & Minerals

Professor:	Muhamed Mudawar, Room 22/328, Phone 4642
Office Hours:	SMW 10:00 – 12:00 noon or by appointment
Course URL:	http://www.ccse.kfupm.edu.sa/~mudawar/coe205/051/
Email:	mudawar@ccse.kfupm.edu.sa

Catalog Description

Introduction to computer organization. Octal and hexadecimal number systems, ASCII codes. Assembly language programming, instruction formats and types, memory and I/O instructions, arithmetic instructions, addressing modes, stack operations, and interrupts. ALU design, RTL, microprogramming, and hardwired control design. Practice of assembly language programming.

Prerequisites: COE 200, ICS 201.

Note: COE 205 is equivalent to ICS 232. Students can take credit for only one of them.

Textbooks and References

1. Sivarama P. Dandamudi, *Introduction to Assembly Language Programming: From 8086 to Pentium Processors*, Springer, 1998. ISBN: 0-387-98530-1.
2. Vincent P. Heuring & Harry F. Jordan, *Computer Systems Design and Architecture*, Addison Wesley, 1997. ISBN: 0-8053-4330-X.
3. Online Material: http://assembly.pc.ccse.kfupm.edu.sa

Course Objectives

1. Proficiency in assembly language programming, particularly for the Intel 80x86 processor family. Ability to analyze, design, implement, debug, and test assembly language programs.
   - Knowledge of the Intel x86 16-bit and 32-bit architectures.
   - Knowledge of assembly language directives and instructions, instruction formats, and addressing modes.
   - Knowledge of Pentium instructions: data movement, arithmetic, bit manipulation, compare, jump, loop, stack instructions, call, return, and interrupt instructions.
   - Ability to write control structures and algorithms in assembly-language.
   - Ability to handle arrays and data structures in assembly-language.
   - Ability to write procedures, pass parameters and return results through registers and on the stack.
   - Ability to use and to develop interrupt service routines. Knowledge of hardware, software, and processor interrupts.
2. Knowledge of the organization of a computer and its components. Understanding the design of the data path and the control unit of a simple CPU, and the fetching and execution control sequences of simple instructions at the register transfer level.

- Knowledge of the various components of a computer system: CPU, memory, busses, and I/O devices.
- Knowledge of the alternative implementations of the data path inside the CPU.
- Ability to develop the fetching and execution control sequences of an instruction at the register transfer level.
- Understanding the design of the control unit and the generation of the control signals using hardwired and microprogrammed implementations.

Course Outline

1. Introduction to Computer Organization & Assembly Language – **1 week (2½ lecture hours)**
   - Assembly language versus high level language programming
   - Basic Components: processor, memory, bus, input and output devices
2. Review of Internal Data Representation – **1 week**
   - Unsigned & signed integer representation
   - Integer arithmetic, carry, and overflow
   - Character representation
3. Assembly Language Concepts **– 2 weeks**
   - Assembly language statements, directives versus instructions, program structure
   - Variable and array declarations, constant definitions
   - Input and output using interrupt 21H
   - Memory segmentation: logical and physical addresses
   - Intel x86 16-bit and 32-bit registers and architectures
   - Instruction formats, opcodes and operands
   - Addressing modes
4. Assembly Language Instructions **– 2 weeks**
   - Data movement instructions
   - Arithmetic instructions and flags
   - Logical and bit manipulating operations
   - Compare, jump, and loop instructions
5. Assembly Language Programming **– 2 weeks**
   - Implementing conditional statements and loops in assembly language
   - Pentium Memory Addressing Modes
   - Traversing and processing arrays
   - String processing
6. Procedures and the Stack – **1½ weeks**
   - Pentium implementation of the stack
   - Stack operations, calling and returning from a procedure
   - Writing procedures, Parameter passing, Local variables
   - Macros (self learning)
7. Interrupts and I/O – **1½ weeks**
   - Software, hardware, and processor interrupts
   - Interrupt processing
   - Peripheral support chips
   - Direct Memory Access (DMA)
8. Processor and Control Unit Design **– 4 weeks**
   - Register transfer
   - Data-path design
   - 1-bus, 2-bus, and 3-bus organizations, performance considerations
   - Fetch and execute control sequences at the register transfer level

- Hardwired control unit design
- Horizontal and vertical microprogramming, microprogrammed control unit design
- Instructions sets and formats, examples of instruction set design and architecture.

Grading Policy

Laboratory	20%
Assignments and Quizzes	25%
Two Major Exams	35% (20% is given to the best)
Final Exam	20%

- Assignments should be submitted at the beginning of class time in the specified due date.
- Late programming assignments are accepted, but will be penalized for each late day, up to a maximum of five late days.

Academic Honesty Policy

***Assignments* -** All assignments must be done individually and must be **your own work**. This includes written and programming assignments. Copying assignments in whole or in parts, altering the work of others, or allowing other people to do your own work and submitting it for grading are clear violations of academic honesty. Also, you must refrain from giving your work to other students, especially if they were your friends. Allowing your friends to copy your work will not help them, but will put you in trouble. Detected copies are given zeros to all involved students including those who did the original work. A warning is also sent. If repeated cheating cases are detected on assignments then a failing grade will result in the course.

***Seeking Help* -** Sometimes you are stuck in an assignment at a given problem and you do not know how to solve it. You tried your best, but still cannot figure out the solution. How do you seek help to solve a problem? There are many ways. You may go first to your professor and seek help from him. Alternatively, you may seek help from a graduate assistant. You can even seek help from senior students, who have previously taken the course and are well acquainted with its subjects, as long as they outline to you the concepts and ideas, but do not provide you with a detailed solution. You can then develop your own detailed solution after having obtained a clear understanding of the problem and an idea about its solution.

***Exams* -** Using unauthorized information or notes on an examination, peeking at others work, or altering a graded exam to claim more grades are severe violations of academic honesty. Remember that if you cheat, you are cheating no one but yourself. Detected situations will result in failing grades in the course, and depending on the severity of the situation, some cases may possibly end up in suspension from the university.