CMSC 313
COMPUTER ORGANIZATION
&
ASSEMBLY LANGUAGE
PROGRAMMING

LECTURE 28, SPRING 2013

ANNOUNCEMENTS

- **Final Exam, Section 01**
  - Tuesday, May 21, 10:30am – 12:30pm, ITE 229
- **Final Exam, Section 02**
  - Tuesday, May 21, 1:00pm – 3:00pm, ITE 229
- **Switching sections must be pre-approved.**
- **Bring 8.5"x11" crib sheet, double sided.**
- **NO ELECTRONICS**
- **NO MAGNIFYING GLASSES!**

FINAL EXAM TOPICS

- Assembly Language Programming
- C Programming
- Digital Logic
- Other Topics

ASSEMBLY LANGUAGE PROGRAMMING

ASSEMBLY LANGUAGE BASICS

- **Base Conversion**
- **Data Representation**
  - negative numbers: 2's complement, 1's complement, signed magnitude
  - ASCII
  - little endian vs big endian
- **Intel CPU**
  - Registers
  - Addressing modes
  - Flags
  - Common instructions

COMMON INSTRUCTIONS

- **Basic Instructions**
  - MOV, ADD, SUB, INC, DEC, NEG
- **Branching Instructions**
  - JMP
  - CMP followed by conditional jump
  - signed vs unsigned conditional jumps (e.g. ja vs jg)
- **Bit Manipulation Instructions**
  - AND, OR, NOT, SHL, SHR, SAL, SAR, ROL, ROR
- **Subroutine Calls**
  - CALL, RET, PUSH, POP

PROGRAMMING IN ASSEMBLY

- **NASM directives**
  - `.data`, `.bss`, `.text` sections
  - `dd`, `dw`, `db`, `resd`, `resw`, `resb` directives
  - `%define`
- **System calls for read & write**
- **Calling C functions from assembly**
- **Writing C functions in assembly**
- **Separate compilation, linking & loading**
- **Interrupts (general principles)**

C PROGRAMMING

BASIC C SYNTAX

- **Functions**
  - local variables
  - function prototypes
  - parameter passing
  - return values
- **Header files**
  - `#include <libfuncs.h>`
  - `#include "mine.h"`
  - Guarding with `#ifndef` ...
- **Separate compilation**

BASIC I/O

- **Input using `scanf()`**
  - `%d, %f, %s`
  - need `&`
  - return value

- **Output using `printf()`**

C TYPES

- **Arrays**
- **Structs**
- **Characters & Strings (null terminated)**
- **`typedef`**

POINTERS

- basic pointer use: * and & operators
- pointers and arrays
- pointers and strings
- pointers to struct
- combinations of pointers, struct and arrays
- pointer arithmetic

MEMORY ALLOCATION

- **allocating memory on the heap**
- **be able to write programs using these:**
  - `sizeof()`
  - `malloc()`
  - `free()`

FUNCTIONS POINTERS

- declaring function pointers (including using `typedef`)
- assigning values to function pointers
- invoking functions using function pointers
- function pointers as actual parameter

DIGITAL LOGIC

BOOLEAN ALGEBRA

- Truth Tables
- AND OR NOT
- Sum of Products (disjunctive normal form)
- Product of Sums (conjunctive normal form)
- Simplification using axioms & theorems of algebra
- Simplification using Karnaugh maps

COMBINATIONAL LOGIC

- CMOS circuits using MOSFET transistors
- combinational vs sequential logic
- logic gates: AND, OR, NOT, XOR (plus ***bubbles***)
- logic components: MUX, DEMUX, DECODER

FLIP FLOPS

- D flip flops
- J-K flip flops
- good flip flops vs ***bad***
- clocks

FINITE STATE MACHINES

- Implemented using flip flops + gates
- State Reduction
- State assignment

OTHER FINAL EXAM TOPICS

OTHER TOPICS

- **Interrupts**
  - What are they? why do we use them?
  - examples
  - typical sequence of events during an interrupt
- **Memory cache**
  - Why?
  - caching policies
- **Virtual memory**
  - Why? what problems are solved
  - hardware assisted (TLB)
  - page tables

NEXT TIME

- **Final Exam**