CISC 3310 Final Exam Review Sheet

Chapter 4: Basics of a Simple Computer

1. Know the basics of CPU and its organization: registers, ALU, control unit, bus, clock, I/O subsystem.
2. Be able to explain how memory is organized, including how it can be addressed (word or byte addressable). Be able to explain what how memory sizes are notated, e.g., 4M x 32.
3. Be able to explain how memory can be constructed from different chips and how an address is divided into parts for addressing those different chips.
4. Be able to calculate the number of bits needed to select a chip, the number of bits needed to give each byte (or word) on the chip a different address, and the number of bits necessary to address an entire chunk of memory, given its size and number of chips.
5. Be able to explain interrupts and how they are processed.
6. Know the architecture of MARIE, including the registers–their purpose and their connection to the bus.
7. Know MARIE'S instruction set, including Load, Store, Add, Subt, Input, Output, and Halt.
8. Be able to explain the two parts of an instruction: the opcode and the operand part.
9. Be able to explain the purpose of RTN and be able to trace or generate RTN (Register Transfer Notation) for MARIE's instruction set.
10. Be able to explain the how the hardware controls execution of a program.
11. Be able to explain and be able to describe the action of the two-pass assembler.
12. Be able to explain the Fetch-Decode-Execute Cycle.
13. Be able to explain the differences and similarities between hardwired and microprogrammed control of instructions, and the fact that they are equivalent. Be able to explain either.
14. Be able to relate the MARIE material to the 80x86 assembly language.

Assembly Language

1. Be able to declare and use dword variables and strings (composed of bytes).
2. Be able to use these arithmetic instructions: mov, add, sub, inc, dec, and neg, for double word integers.
3. Know how to perform arithmetic by using shift operations sal and sar.
4. Be able to use cmp and to perform jumps and create loops using jmp, the relative jump instructions (jl, jle, je, jg, jge, jne, etc.), and loop.
5. Be able to write and use a procedure, including sending parameters, call and, returning a value, save and restore registers using the stack, and using both value and reference parameters.

6. Be able to process an array using lea, index register notation, and base register notation.
7. Be able to perform simple floating point arithmetic using finit, fld, fadd, fsub, fmul, fdiv, and fst.

Chapter 5: Instruction Set Architectures
1. Be able to identify 3 types of architectures, based on storage on the CPU.
2. Be able to explain instruction formats and the decisions made when designing instruction sets: big endian vs. little endian, stacks vs. registers, length of instructions, number of operands. Be able to explain and some advantages and disadvantages of each choice.
3. Be able to explain instruction types (data movement, arithmetic, Boolean, transfer of control, etc.).
4. Be able to explain the different types of addressing: immediate, direct, register, indirect, indexed.
5. Be able to explain the concept of pipelining and how pipelining works.

Chapter 6: Memory
RAM vs ROM. Flash memory. Memory hierarchy: L1 cache, L2 cache, main memory, secondary memory. Cache, cache terminology (hit, miss, etc.). Principle of locality, either spatial, temporal, or sequential. Cache, its purpose, how it works, CAM, associative memory. Cache mapping schemes (not the details of number of bits for tags and words, etc.): Direct mapped, Fully Associative, n-Way Set Associative. Be able to calculate effective access time. Cache write policies.

Chapter 7: I/O Systems
Amdahl's Law: know what it is and means. You don't need to memorize the formula. I/O control: programmed vs interrupt-driven, vs DMA vs channel. I/O on the bus. I/O architectures, serial vs parallel transmission, hard disk technology, CD and DVD to the level discussed in class, RAID: know RAID levels 0 and 1 and the fact that others add error-checking disks.

Chapter 9: Alternative Architectures (if we get to this)
RISC vs. CISC.