CSCI 370 — Midterm Exam Review

Data Representation
• Concept of words (in regard to architecture)
• Data sizes for x86-64
• Endianness
• Integers
  – Bitwise and Logical Operations, Arithmetic vs. Logical Shift
  – Two’s complement vs. Unsigned
  – Integer ranges
  – “Integer Puzzles”
• Floating Point
  – Binary encoding
  – IEEE 754
    * Normalized vs Denormalized
    * Special values (Infinity, NaN)
    * Ranges
  – Floating point arithmetic errors and precision
• Data Lab

Machine Programming
• CISC vs. RISC -- know what is unique
• Definitions: Instruction Set Architecture (ISA), Microarchitecture (uArch)
• Compilation Flow
• Data types in assembly
• Translation of C-code to x86 (and vice versa)
  – Functions (with parameters and return values)
  – Know what callee and caller saved registers means!
  – Addressing Modes
    * Normal
    * Displacement
    * Complete (your favorite)
    * Address Computation Problems
  – Important Instructions:
    * leaq
    * movq
    * test
    * cmp
  – x86 condition codes
    * Carry, Sign, Zero, Overflow
    * NOTE: do not need to know how flags evaluate to branch conditions
    * Conditional Branching
  – Implementation of different loops (do-while, while, for)
  – Switch Statements (Lec3:end)
    * Jump Tables
• Data layout of arrays and structs
• Alignment
• Stack and Heap (direction of growth, location)
• What is a buffer overflow?
• Return Oriented Programming — what is a "gadget"?
• Understanding the majority of the x86-64 “cheatsheet” might be useful.
Hardware Implementation, HCL, Y86-64, and Pipelines

- Digital Signals (high voltage vs. low voltage, clock)
- Logic Gates
- Combinational vs. Sequential Circuits
- Hardware Control Language (HCL)
  - Data types: bool, int (word)
  - Statements: assignments, boolean expressions, word expressions
  - Boolean expressions:
    * Logic operations
    * Word Comparisons
    * Set Membership
  - Word expressions (Case)
  - Processor State – Registers, memory, PC
- Y86-64
  - Instruction format
  - Register format (4 bits)
  - 5-stage pipeline stages
  - Execution Pattern and Pipeline Registers
- Parallel Implementation
  - Why is parallel pipelining important? What are its limitations?
  - Hazards
    * Data Hazards
    * Control Hazards
    * Stalling – Register Modes (Stall vs Bubble vs Normal)
  - Modifications
    * Pipeline Registers
    * Data Forwarding – Represent Instruction Sequence with and without
    * Pipeline Control Logic – know when to stall/bubble for:
      - Load/Use Hazard
      - Return
      - Mispredicted Branch
  - CPI
    * What does it mean when CPI is approx. = 1? Greater than 1? Less than 1?
    * How does branch misprediction, load-use hazards, and returns affect CPI?
      - What are the penalties for each?