Lecture 23
July 04

Instructions Set Arch

- Register set
- Other choices/assembly instructions look like?

Operand Addressing

- How many operands can an instruction have?
- Three-address instructions
    - An instruction can specify two sources & one destination address
    - E.g. ADD A,B,C M[A] ←M[B] + M[C]
- Two-address instructions
    - Two operands specified
        - Specified a source and a sources/destination
            - I.e. One sources is implicitly used as the destination
        - E.g.
        - MOVE R1, A R1 ← M[A]
        - ADD R1, R2 R1←R1+R2

One-address instructions

- The source/destination is specified implicitly
    - Usually an "accumulator"
    - E.g.
    - LD A ACC ← M[A]
    - ADD B ACC←ACC +M[B]
    - ST R2 R2 ←ACC
- Zero-address instructions
    - Operands are all specified implicitly
        - The top elements of a stack
    - E.g. ADD: take the top 2 stack elements add them & push the result
        - TOS← TOS +TOS_{-1}
    - Generally need load/store operations with one operand
        - E.g.
        - LD A TOS←M[A]
        - ST B M[B] ←TOS

Address Architecture

- The type of operands used might be restricted
- Memory to memory architecture
    - no general-purpose registers
    - all operands in memory
    - e.g.
        - ADD A,B,C M[A] ←M[B]
        - ADD D,E M[D] ← M[D] + M[E]
    - Problem: takes a long time to do memory operations with each instructions

Register-to-register (or load store) architecture

- No memory access in calculation instructions
  - Only registers
- Only load & store operations can access memory.
- Need a large register file to hold values
- E.G.
  - LD R2, A   R2 ← M[A]
  - ADD R8, R0 R31   R8 ← R0+R31
  - ST B, R4   M[B] ← R4

Register memory architectures

- Allows one memory access per instruction
- E.G.
  - ADD R1, A   R1 ← R1+M[A]
  - ADD R0, R1, A   R0 ← R1 + M[A]

Single accumulator architecture

- Single register: ACC
- Generally single operand instructions
- The other operands must come from memory
  - E.g.
  - ADD A   ACC ← ACC+M[A]
  - ST B   M[B] ← ACC

Stack architecture

- All operands access the stack
  - Except load/store
- Stack could be implemented with register or memory

Addressing Modes

- The “addressing mode of an instruction determines how the operand is interpreted
- The address that is used after translation is the “effective address”
- E.g. If the programmer specifies
  - “load 184” is it
    R184 or M[184] or 184?
  - ….