Instruction-set Design Issues: what is the ML instruction format(s)

ML instruction

| Opcode | Dest. Operand | Source Operand 1 | ... |

1) Which instructions to include:
   - How many?
   - Complexity - simple “ADD R1, R2, R3”
     complex e.g., VAX
     “MATCHC substrLength, substr, strLength, str”
     looks for a substring within a string

2) Which built-in data types: integer, floating point, character, etc.

3) Instruction format:
   - Length (fixed, variable)
   - number of address (2, 3, etc)
   - field sizes

4) Number of registers

5) Addressing modes supported - how are the memory addresses of variables/data determining
Reduced Instruction Set Computers (RISC)

Two approaches to instruction set design:
1) CISC (Complex Instruction Set Computer) e.g., VAX
   1960’s: Make assembly language (AL) as much like high-level language (HLL) as possible to reduce the “semantic gap” between AL and HLL

Alleged Reasons:
- reduce compiler complexity and aid assembly language programming - compilers not too good at the time (e.g., they did not allocate registers very efficiently)
- reduce the code size - (memory limited at this time)
- improve code efficiency - complex sequence of instructions implemented in microcode (e.g., VAX “MATCHC substrLength, substr, strLength, str” that looks for a substring within a string)

Characteristics of CISC:
- high-level like AL instructions
- variable format and number of cycles
- many addressing modes (VAX 10 addressing modes)

Problems with CISC:
- complex hardware needed to implement more and complex instructions which slows the execution of simpler instructions
- compiler can rarely figure out when to use complex instructions (verified by studies of programs)
- variability in instruction format and instruction execution time made CISC hard to pipeline

Lecture 6 - 2
2) RISC (1980’s) Addresses these problems to improve speed by making instruction pipelining efficient.

(Tables 13.1 and 13.7 - characteristics of some CISC and RISC processors)

General Characteristics of RISC:
- emphasis on optimizing instruction pipeline
  a) one instruction completion per cycle
  b) register-to-register operations
  c) simple addressing modes
  d) simple, fixed-length instruction formats
- limited and simple instruction set and addressing modes
- large number of registers or use of compiler technology to optimize register usage
- hardwired control unit