Assembly Language Lecture Series:

CISC vs RISC

Sensei RL Uy, College of Computer Studies,
De La Salle University, Manila, Philippines

Copyright Notice

Talking Points

CISC

Complex Instruction Set Computing (CISC)

- **Minimize** the number of instructions per program BUT **sacrificing** the number of cycles per instruction

CISC: Characteristics

- Has a lot of instructions. Some of the instructions are **complex**.
    - LOOP L1 is a complex instructions. It performs decrement, comparison and branch operations in one instruction.

CISC: Characteristics

- Supports **many** types of **addressing modes**
  - Register-register, register-memory, index, scaled-index byte addressing mode

CISC: Characteristics

- Allows **register** and **memory operands** at the same time
  - **ADD [BETA], RAX instruction** means fetch the data from memory location BETA; add the data to the internal register RAX and the result is stored back to memory location BETA

CISC: Characteristics

- **Flag register** use as **condition code**
  - Carry flag, zero flag, overflow flag and other condition code flags are stored in the flag register

CISC: Example CISC Assembly Program

Increment the value of memory location ALPHA 5 times

```
   MOV RCX, 0x0000000000000005
L1:INC byte [ALPHA]
   LOOP L1
```

RISC: Characteristics

- **Small** and **simplified** set of instruction set architecture (ISA)
    - Less common instructions implemented as solutions

Example:

- CISC: MOV RAX, 0x0000000000000004
- RISC: ADDI x5, x0, 0x00000004

RISC: Characteristics

- **Load-store architecture**
  - Operands that are in the memory must be loaded first in the registers before any arithmetic and logic execution could proceed and the result is stored back to the memory

Example:

- **CISC:**
```
ADD [BETA], RAX
```
- **RISC:**
```
LA x7, BETA              ; x7 points to BETA (equivalent to LEA x7, [BETA])
LW x18, (x7)             ; load data to x18 (equivalent to mov x18, [x7])
ADD x20, x18, x19        ; assume x19 is the "rax"
SW x20, (x7)             ; store result back to x20 (equivalent to mov x20, [x7])
```

RISC: Characteristics

- **Has more registers than CISC**
  - **RISC-V,** a RISC-based architecture**,** has **32** (programmer usable) **integer registers**
  - **x86-64,** a CISC-based architecture, has **15** (programmer usable) **integer registers**

RISC: Characteristics

- Each RISC instruction is encoded with **fixed number of bits**
  - Each RISC-V instruction is **32-bit** (4 bytes) in size

RISC: Characteristics

- Each RISC instruction executes in **equal** and **few clock cycles**
    - Each RISC-V instruction executes in **5** clock cycles

RISC vs CISC

Complex Instruction Set Computing (CISC)

- **Minimize** the number of instructions per program BUT **sacrificing** the number of cycles per instruction

Reduced Instruction Set Computing (RISC)

- **Reduce** number of cycles per instruction BUT **increase** number of instructions per program
- Uses **pipelining**

RISC: Example RISC Assembly Program

Increment the value of memory location ALPHA 5 times

```
        LA x5, ALPHA                        ; x5 points to ALPHA
        LW x11, (x5)                        ; load data to x11
        ADDI x10, x0, 0x00000005            ; x10 = 5 (# of loop)
L1:     ADDI x11, x11, 0x00000001           ; increment data
        ADDI x10, x10, 0xFFFFFFFF           ; decrement counter
        BNEZ x10, L1                        ; while x10 !=0
        SW x11, (x5)                        ; store result to memory
```

7 instructions vs 3 instructions for x86-64 (CISC) assembly