CS2422 Assembly Language & System Programming

September 12, 2006

Announcement

• Two sessions for this course:
  – CS2422-01 (張鈞法教授)
  – CS2422-02 (金仲達教授)
Textbooks

- Leland Beck, “System Software”
  - Using the first 4 chapters.
- Other good books:
  - Patterson & Hennesey, “Computer Organization and Design” (not Intel-based)
  - The Art of Assembly (freely available online)

Honor Code

- University code will be followed strictly to handle the cheating in assignments and exams.
- You are allowed (and in fact encouraged) to discuss the assignments, but the work must be your own.
Early Bonus & Late Penalty

• Early bonus: +2% for each day early, up to two days.
• Late penalty: -20% for each day (or partial day) late.

Why Learning Assembly?

• A great way to learn how a computer really works:
  – To talk in the languages of the processors.
  – To see how a computer talks to the other devices.
• To build solid background for other courses:
  – Computer Architecture, Compilers, Operating Systems…etc.
Even More Important Now

• A few examples:
  – The SOC (System-On-Chip) and embedded system trend.
  – The era of ubiquitous computing.
  – For graphics folks: The DirectX and OpenGL shading languages.
• The hardware/software boundary is blurring.

What Exactly Is a PC?

A machine to…
• “…use email and surf the web” (Grandma)
• “…run MS-Office” (said Mom & Dad)
• “…play games and watch movies” (Kids)
• “…to write programs for Linux or Bill Gates’ DOS/Windows” (CS Students)
• “…to show you the BIOS Screen” (EE Students)
Virtual Machine Concept

Section 1.2, Figure 1-1

Inside the case

b. Processor
c. PCI slots (for I/O)
e. Memory slots
Motherboard

Memory

- RAM
- DRAM
- SRAM
- ROM
- Volatile / Non-Volatile
- Magnetic
Processor

Pentium 4
(Prescott 90nm)

You only need switches and wires!

- Relays
- Vacuum tubes
- Transistors
- Integrated Circuits
- VLSI
- Nanotubes?
- Quantum Effect Devices?
5 Classic Computer Components

Instruction and Operands

- Example:
  MOV CX, 25
  ADD AX, CX
- Data are in registers or memory.
- For now, we will focus on CPU, Registers, and Memory only.
- I/O will be discussed later.
Abstraction: C to ASM

\[ x = (a+b) \times b \]

C compiler

Assembly

MOV AX, a
ADD AX, b
MUL c
MOV x, AX

Abstraction: ASM to Binary

Assembly

MOV AX, a
ADD AX, b
MUL c
MOV x, AX

Assembler

Binary

00000000101000010000000000011000
00000000100011100001100000100001
10001100011000100000000000000000
10001100111100100000000000000100
10101100111100100000000000000000
10101100011000100000000000000100
00000011111000000000000000001000
Acknowledgement

• Many slides in this lecture are borrowed from Prof. Gary Bishop’s COMP120 course at UNC-Chapel Hill.