Introduction to Computer Architecture

Instructor: Nihshanka Debroy
(Notes borrowed from Tammy Bailey and Dr. Forbes)
Computer Systems

• A computer is a system made up of components
  – hardware
    • electronic and mechanical parts
  – software
    • programs and data

• Main hardware components of a computer system
  – processor
  – memory
  – input/output devices
The Big Picture

- Since 1946 all computers have had 5 components

  - The Von Neumann Machine

    - Processor
      - Control
      - Datapath
    - Memory
    - Input
    - Output

- What is computer architecture?
  Computer Architecture = Machine Organization + Instruction Set Architecture + ...
The Instruction Set: a Critical Interface

- What is an example of an Instruction Set architecture?
Fetch, Decode, Execute Cycle

- Computer instructions are stored (as bits) in memory
- A program’s execution is a loop
  - Fetch instruction from memory
  - Decode instruction
  - Execute instruction
- Cycle time
  - Measured in hertz (cycles per second)
  - 2 GHz processor can execute this cycle up to 2 billion times a second
  - Not all cycles are the same though...
Processor

• Central processing unit (CPU)
• “Brain” of the computer
• Contains the circuitry controlling the interpretation and execution of machine instructions
  – arithmetic/logic unit
    • data manipulation
  – control unit
    • coordinates computers activities
• Machine instructions
  – electronic operations encoded as bit patterns
Machine Instructions

• **Data transfer instructions**
  – request the movement or copying of data from one location to another
    • CPU to main memory
    • CPU to input/output devices

• **Arithmetic/logic instructions**
  – basic arithmetic operations, Boolean operations

• **Control instructions**
  – direct the execution of a program rather than the manipulation of data
Registers

- Small amount of memory located directly on the CPU
- Fastest way for the system to manipulate data
- Speed execution of computer programs by providing quick access to data when performing operations
- Temporary holding place for values immediately needed by CPU for computation
  - input for an arithmetic/logic operation is loaded into registers from main memory
  - arithmetic/logic unit in the CPU executes operation
  - output is stored in registers
- Results of computation are written from the registers to main memory upon completion
Main memory

• Primarily **RAM (random access memory)**
  – temporarily stores programs and data when the processor is actively using them
  – interfaces almost directly with the CPU
  – extremely fast (nanoseconds) on modern computers
  – Volatile
  – Can compare to “short-term memory”
Memory operations and ROM

• Read operations
  – get data from memory by requesting the contents of a cell at a particular address

• Write operations
  – store data in memory by requesting information be placed in a cell at a particular address

• ROM (read only memory)
  – non-volatile main memory
  – optimized for performing read operations
  – usually holds the BIOS (program used to boot computer and load the operating system)
Cache

• Stores contents of frequently accessed RAM locations and the addresses where these data items are stored
  – faster to access than main memory since it is usually located on the CPU itself
• Data in cache is likely to be needed soon by CPU
• L1 (level 1) cache
  – closest to the processor, not very large
• L2 (level 2) cache
  – bigger than level 1 cache, buffer between L1 and main memory
Secondary memory

- Permanent memory, mass storage
  - saves programs and data on a long-term basis
    - operating system
    - software required by the user
    - other non-executable data
  - hard disk, floppy disk, CD, DVD
- Read and write operations are slow (milliseconds) compared to main memory
- Inexpensive compared to main memory
Input/Output devices

• Interact primarily with the CPU
• Most systems have a keyboard, mouse (input devices) and monitor (output device)
• May consist of any number of additional devices
  – printer
  – speakers
  – modem or Ethernet card
  – sound and graphics cards
  – digital camera
  – secondary monitor
  – game controllers
High-level programming languages

• Most programs are created using a *high level programming language*
  – closer to human language than machine language (low-level)
  – Java, C, C++, Pascal, FORTRAN
  – easier to read, write, and maintain
• Source programs are translated to executable (machine language) programs by a *compiler*
• Different programming languages require different compilers
• One programming language can have many compilers
  – computer architecture, software package
Source program $\rightarrow$ executable program

- Create source program using a text editor
  - written (typed) instructions in a high-level language
- Save source program on disk
- Compile source program
  - **compiler** translates source program to executable program
  - source program remains unchanged
  - a new executable program is created
  - executable program is saved on disk
- Run executable program
  - copied into main memory and run by processor
Machine Language

- Machine language is the most primitive
  - Everything represented by numbers
  - At hardware level, numbers are in binary
  - Numbers represent instructions (code)
  - AND Numbers represent data
  - Context of use decides whether number is data or instruction

- In practice, seldom program in machine language

- Use a language, very close to machine language called Assembler Language
  - Symbolic in nature (as opposed to numeric)
  - Each instruction number has a mnemonic
  - E.g., 12 is ADD
  - Locations also given names (sometimes variable name)
Simple Program

- Show in assembler rather than machine language
  
  ```
  copy   ax, x  
  add    ax, y  
  copy   z, ax  
  ```

- Implements
  
  ```
  z = x + y;  
  ```

- Remember, really *ALL NUMBERS*

- Could be:
  
  ```
  20 101  
  12 102  
  21 103  
  ```

- If `copy-into = 20, add = 12, and copy-out = 21` and
  
  *x* is stored in **101**, *y* in **102**, and *z* in **103**
Operating Systems
What is an Operating System?

Modern operating systems support:

- Software tools for creating programs
  - libraries, compilers
- Running multiple programs.
  - multiprogramming
- Saving/accessing data.
  - files, virtual memory
- User interaction.
  - window system
- Interaction with other systems.
  - networking
- Core applications programs.
  - client-server
Operating systems

- Most important program that runs on a computer
- Every general-purpose (such as desktop) computer must have OS to run other programs
- Manages the hardware and software resources of the system
- Provides stable and consistent way for applications to deal with the hardware without having to know all the details
- Most common
  - Windows, UNIX, Macintosh
- Many other OS designed for special-purpose applications
  - mainframes, robotics, manufacturing, real-time control systems
Managing hardware and software

- Various programs and input methods compete for the attention of the central processing unit (CPU)
  - programs and methods demand memory, storage and input/output (I/O) bandwidth for their own purposes
- OS makes sure each application gets the necessary resources while cooperating with the other applications
  - economizes capacity of the system to greatest benefit of users and applications.
Memory and storage management

- Each process must have enough memory in which to execute, and it can neither run into the memory space of another process nor be run into by another process.
- The different types of memory in the system must be used properly so that each process can run most effectively.
- Memory types
  - **High-speed cache** - fast, small amounts of memory available to the CPU, where cache controllers predict which data CPU will need next and pull it from main memory.
  - **Main memory** - RAM
  - **Secondary memory** - disk
Classification

- **Multi-user**  
  - allows many different users to take advantage of the computer's resources simultaneously

- **Single-user, multi-tasking**  
  - single user operates many programs at the same time  
  - most desktop and laptop computers today

- **Single-user, single task**  
  - one user can do one thing at a time (e.g. Palm OS)

- **Real time (RTOS)**  
  - particular operation executes in precisely the same amount of time every time it occurs  
  - machinery, scientific instruments and industrial systems
Types of operating systems