Introduction to Computer Organization Exam 1

Computer Abstractions and Technology

The computer revolution

Moore's Law chip integration is doubled every **18** (or 80) months

Computer processors become more and more powerful and take less power

Classes of Computers

Personal computers

- General purpose
- Subject to cost/performance tradeoff

Server computers

- Small to building sized

Supercomputers

- Single sytem which can solve a particular application very fast

Embedded computers

- Hidden as components of the system
- Stringent power/performance/cost constraints

The PostPC Era

Personal Mobile Device

- Battery operated (Phones)

Cloud Computing

- Warehouse scale computers (WSC)
- Software as a Service (SaaS)
- Amazon and Google

8 Great Ideas - Architect

- Moore's Law
- Abstraction to simplify design
- Common case fast
- Parallelism
- Pipelining
- Prediction
- Hierarchy of memories
- Dependability

Below Your Program

Applications Software Written in high-level language
System software compiler…
Hardware processor…

Levels of Program Code

High-level –> Assembly language –> Hardware representation (10101001)

Components of a computer

All Computers have the same components

- Input/output
- Processor
- Memory

Inside the Processor (CPU)

Datapath: performs operations on data
Control: sequences datapath, memory
Cache memory: Small fast SRAM memory for immediate access to data

Summary

Abstraction Helps us deal with complexity

Instruction set architecture (ISA) The hardware/software interface

Application binary interface The isa plus the hardware

Safe place for data

Volatile main memory Non permanent

- Loses instruction and data when poweroff
- DRAM

Non-volatile secondary memory

- Magnetic disk
- Solid-state disk
- Optical drive

Networks

- Communication, resource sharing, nonlocal access…

Intel Core i7 Wafer 300mm wafer, 280 chips, 32nm technology. Each chip is 20.7 x 10.5 mm

Relative Performance

Define Performance = 1/Execution Time

> X is **n** times faster than Y

```
Performacncex/performacey = Execution timey / Execution timex = n
```

Example:

```
10s on A, 15s on B
Execution timeb / execution time a = 15s/10s
```

Measuring Execution time

- Elapsed time
- CPU time
  - Number of Executed Instructions

CPU Time formula

```
CPU Time = Instructions/Program * Clockcycles/instruction *
Seconds/Clock cycle
```