CS 230 – Introduction to Computers and Computer Systems

Module 0 – Introduction

Aakar Gupta
aakar.gupta@uwaterloo.ca

(Slides based on materials prepared by Sandy Graham)

Goals / Overview

Goals

- overview of computer systems
  - from bottom to top
- understand basic challenges & techniques
- understand performance implications

Course Information

- Course materials found on https://www.student.cs.uwaterloo.ca/~cs230/
- Communication
  - Piazza
  - Office Hours
    - IA – Murray Dunne – Thursdays – 3:30-4:30pm (E5-5022)
    - Instructor – Aakar Gupta – Tuesdays – 10-11am (DC2129)
  - Email

Course/Assignment Tools

- student.cs environment
- UNIX tools
- Python
- MIPS assembler – Java based
- MIPS emulator – Java based

- material will be made available as needed
- also: attend the tutorials!

About the Slides

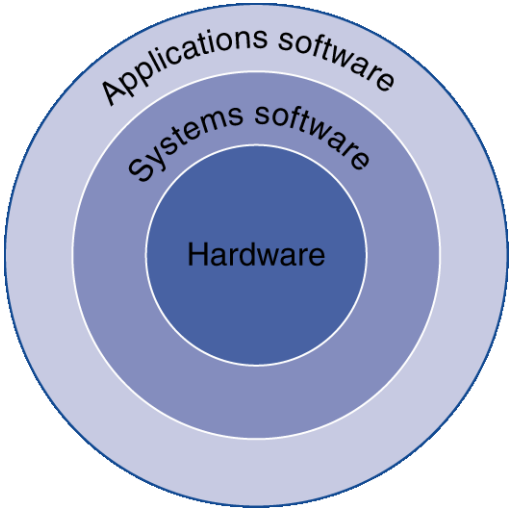
- some material and figures taken from textbook and accompanying slides:

David Patterson and John Hennessy. Computer Organization and Design – The Hardware/Software Interface

- figures taken from other sources are shown with reference
- other material newly developed for this course

Motivation and Background

Below Your Program



- application software
  - high-level language
- system software

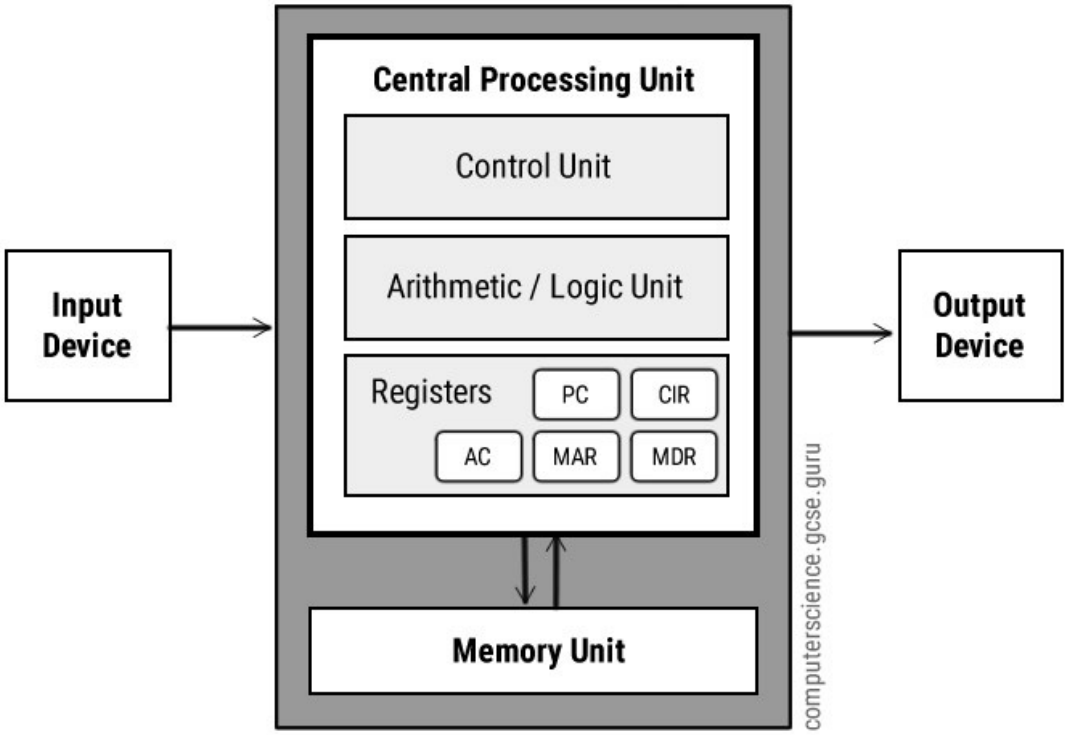
Many types. 2 most central to every computer system today:

  - operating system – windows, linux
  - compilers

- OS: supervising sware that manages processes running and resources (strg & mem), allocating resources to the processes as needed
- Compiler: sware that translates sware written in high-level langs into instructions hware can execute (assembly)

- hardware
  - processor, memory, I/O

Model of a Computer



- DRAM (dynamic RAM)
- Up until the 1940s, only data was stored in memory. Also, prog can be executed from memory.

4 components: input, CPU, memory, output

- von Neumann model
- CPU — Takes and/or stores data that a program needs from memory, runs operations on it, and sends it to output device
  - Control unit
  - Arithmetic/logic unit
- I/O
  - User: mouse, keyboard, monitor
  - Storage: hard drive, optical drives
  - Network: WiFi adapter, router
  - user, storage, network
- Memory
  - **program** & **data** stored in memory

Program

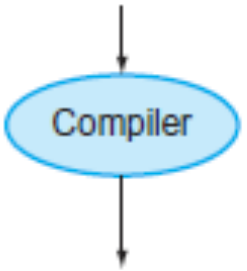
- Sequence of instructions stored in memory
- Instructions are just collections of bits that the computer understands and obeys – *on/off*
  binary
- *Bit – 0 or 1*
- Example:

```
1000110010100000   Machine Language
add A,B            Assemble Language
A+B                High-Level Language
```

used by humans to write code

High-level language program (in C)

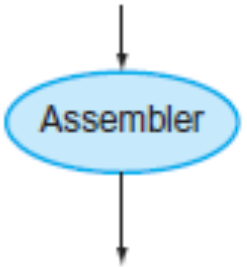
```
swap(int v[], int k)
{int temp;
   temp = v[k];
   v[k] = v[k+1];
   v[k+1] = temp;
}
```



lang understandable by hardware

Assembly language program (for MIPS)

```
swap:
      multi $2, $5,4
      add   $2, $4,$2
      lw    $15, 0($2)
      lw    $16, 4($2)
      sw    $16, 0($2)
      sw    $15, 4($2)
      jr    $31
```



sequence of instructions

Binary machine language program (for MIPS)

```
00000000101000100000000100011000
00000000100001000010000000100001
10001101111000100000000000000000
10001111000010010000000000000100
10101111000010010000000000000000
10101101111000100000000000000100
00000011111000000000000000001000
```

Process

- Program and data “in action”
- Actual execution of the program instructions
- Usually managed by the operating system

Processors (CPUs)

- The hardware that executes the program instructions

Year	Technology	Relative performance/cost
1951	Vacuum tube	1
1965	Transistor	35
1975	Integrated Circuit	900
1995	Very large scale IC (VLSI)	2,400,000
2005	Ultra large scale IC	6,200,000,000

- V Tube vs Transistor: manually vs automatically switched off (trans. can be controlled with electricity)
- IC: multiple trans on a single chip

Moore's Law

Created by cofounder of Intel; more of a prediction than a law

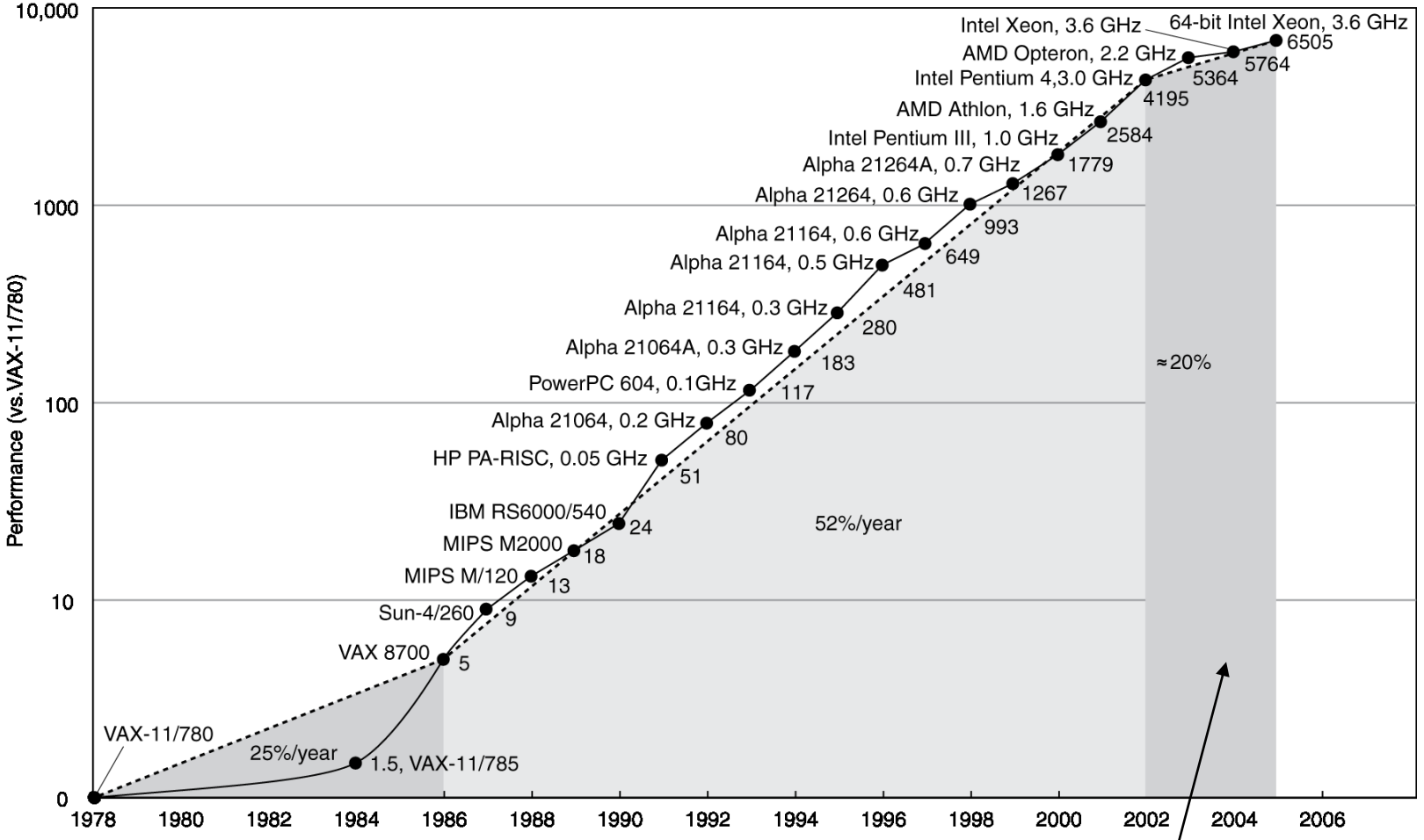
Transistor density: the # of transistors that can be put on a single chip. Higher -> faster comp response

- transistor density doubles every two years
  - every year 1959-1975 underestimation; stayed true until 2010
- in the past
  - transistor density translated into processing power
  - almost double speed every 2 years...
  - Reduces response time, increases performance
- not applicable anymore

Transistors are now so small (1/10th of a human's hair) that they can't get any smaller, and hence we cannot fit any more transistors on a single chip

Uniprocessor improvements have slowed down as a result; Intel I7 has been std for years

Uniprocessor Performance



Constrained by power, instruction-level parallelism, memory latency

Multiprocessors

Solution to problem of how to increase performance in computers

- multicore microprocessors
  - more than one processor per chip
- requires explicitly parallel programming
  - compare with instruction level parallelism (hidden)
- hard to do

- Processors in a m-processor run explicitly different programs, hence parallelism is needed
- Hard to do as there is no optimal way to distribute + run processes in parallel

Performance

How to improve it?

- faster processor
- more processors (needs parallelization)
- better software (algorithms)

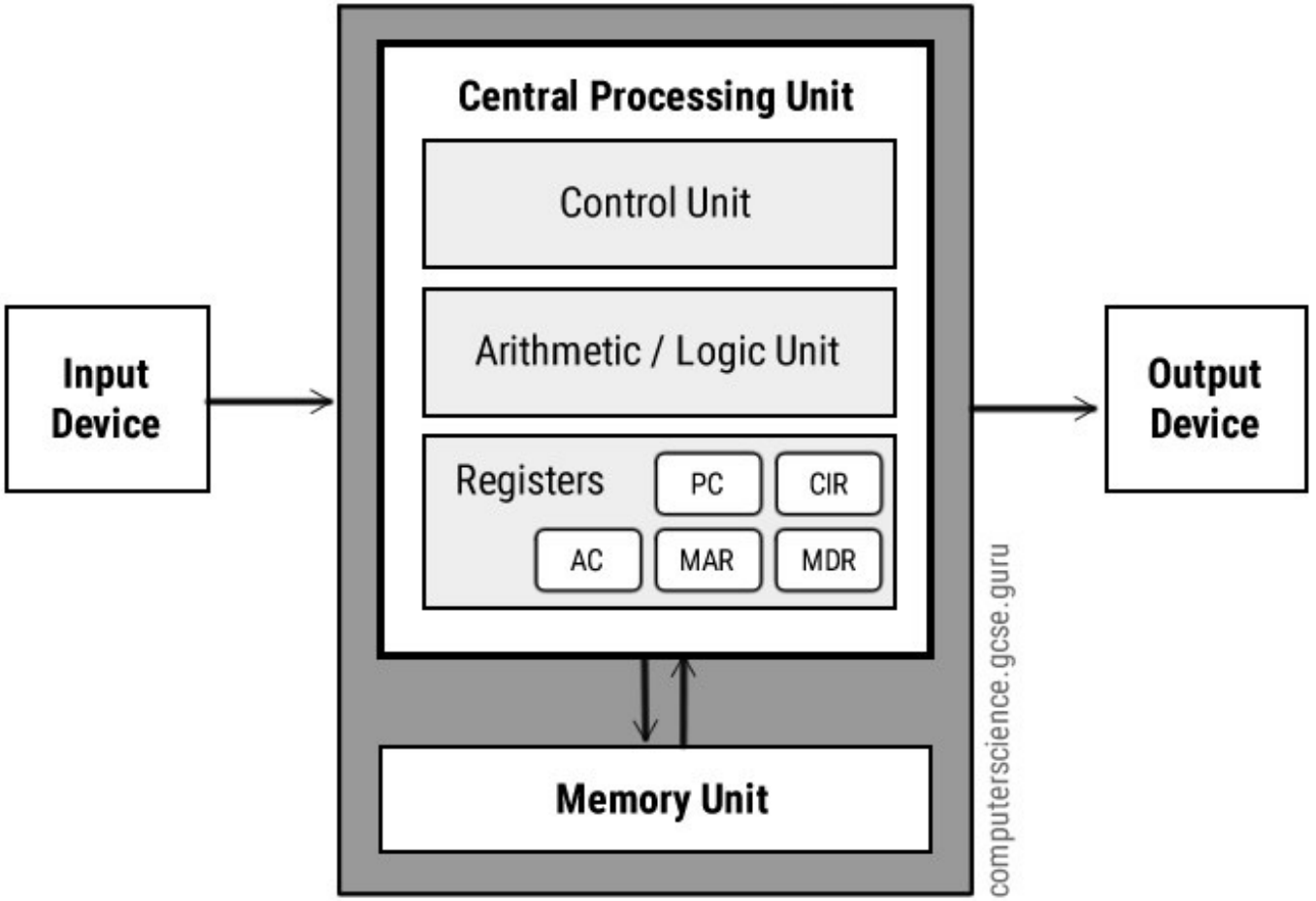
- Better software: writing a program w/ fewer instructions and/or one that has lower memory usage/consumption, i.e. a more efficient and/or less memory-intensive program

Efficiency Matters

- More processors require more power
- Network-centric computing, Internet

-> large data centers

- hardware cheap, but
  - power consumption -> heat
  - heat -> cooling -> more power consumption
  - money and environment costs
- software performance
  - Minimize instructions
  - Minimize memory



- Program
- Data

Data & Registers

Data refers to:

- Data needed by the running programs
  - Usually refers to RAM – Random Access Memory

- Registers are high speed storage areas in the CPU. All data must be stored in a register before it can be processed.

Trade-Offs

- almost everything in CS is a trade-off
  - very few absolute truths
- “fast, good, or cheap – pick two”

Cheap in terms of resource consumption (storage and memory)

CS 230

Course Topics / Modules

- arithmetic, hardware, data
- assembly language
- system internals
- build and runtime
- multiprocessing
- operating systems (if time allows)