Chapter 1 : Introduction to computer programming

Dr Kafi Rahman

email: kafi@truman.edu

1.1

Why Program?

Why Program?

- Computer – programmable machine designed to follow instructions
- Program – instructions in computer memory to make it do something
- Programmer – person who writes instructions (programs) to make computer perform a task
- So, without programmers, no programs; without programs, a computer cannot do anything

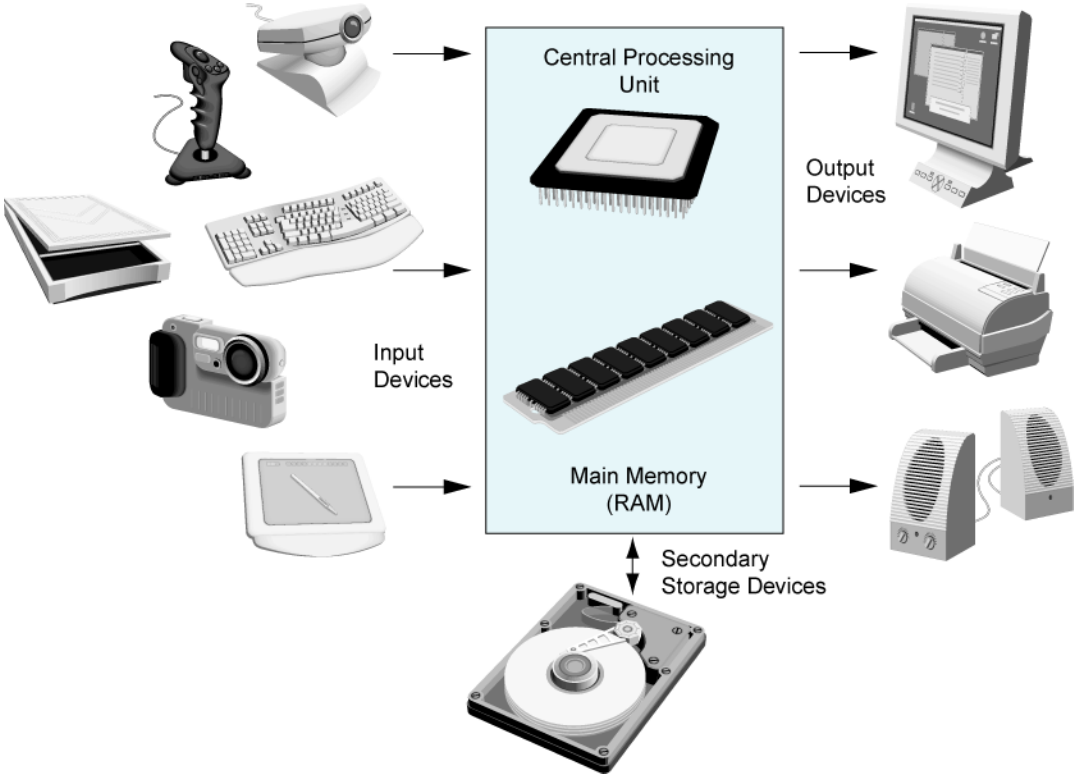
1.2

Computer Systems: Hardware and Software

Main Hardware Component Categories:

- Central Processing Unit (CPU)
- Main Memory
- Secondary Memory / Storage
- Input Devices
- Output Devices

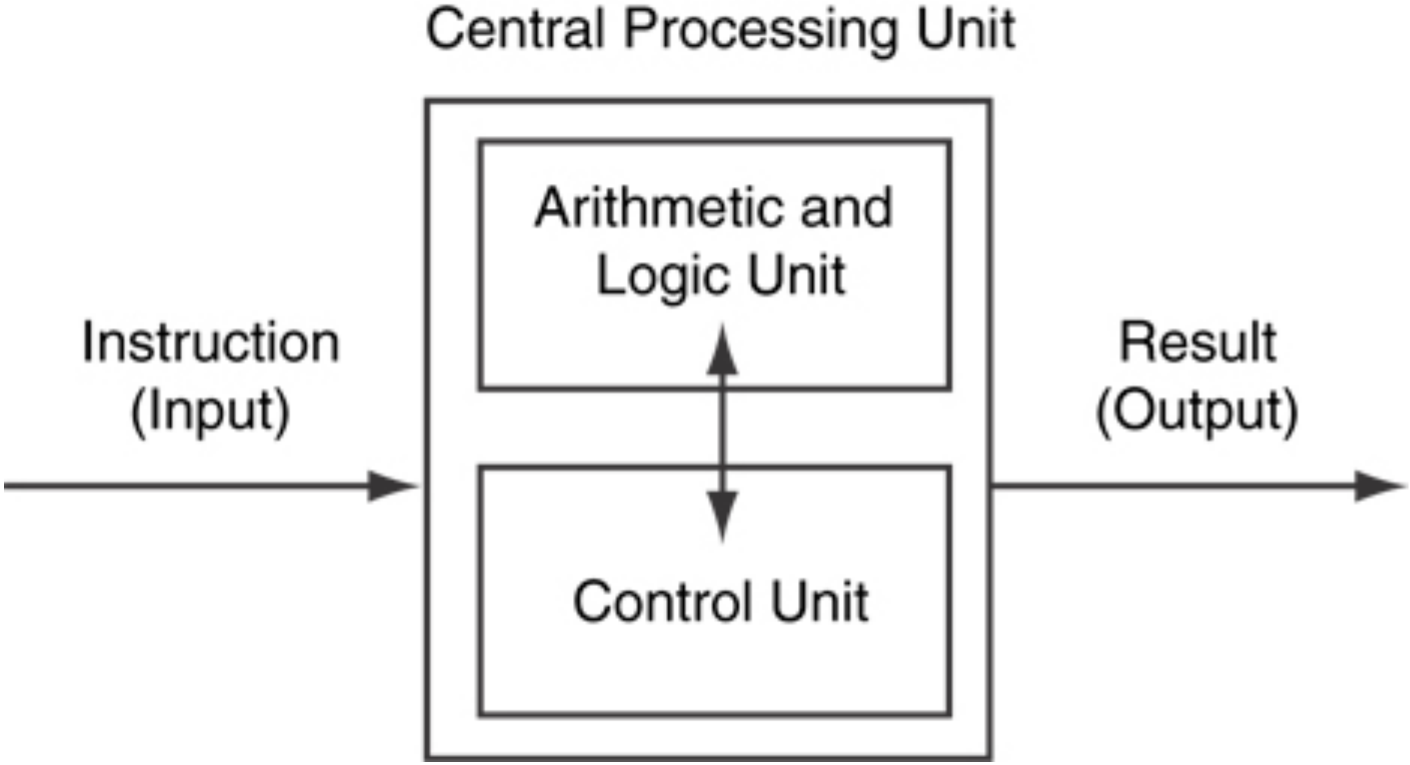
Main Hardware Component Categories



Central Processing Unit (CPU)

- Comprised of:
  - Control Unit
    - Retrieves and decodes program instructions
    - Coordinates activities of all other parts of computer
  - Arithmetic & Logic Unit
    - Hardware optimized for high-speed numeric calculation
    - Hardware designed for true/false, yes/no decisions

CPU Organization



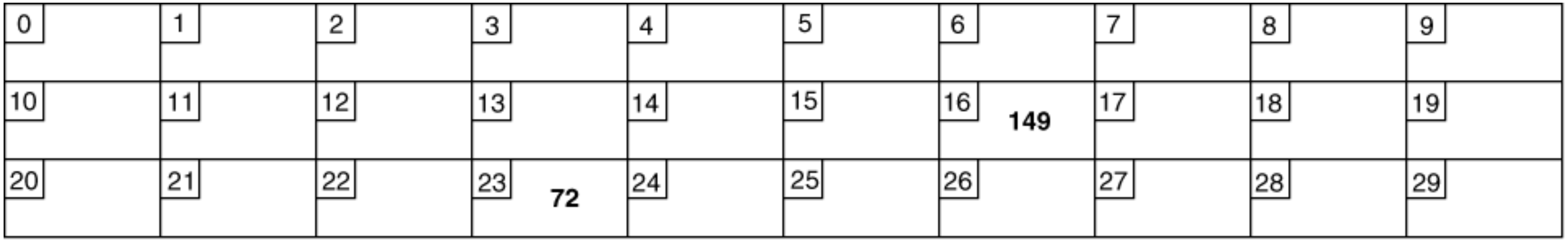
Main Memory

- It is volatile. Main memory is erased when program terminates or computer is turned off
- Also called Random Access Memory (RAM)
- Organized as follows:
  - bit: smallest piece of memory. Has values 0 (off, false) or 1 (on, true)
  - byte: 8 consecutive bits. Bytes have addresses.

Main Memory

- Addresses – Each byte in memory is identified by a unique number known as an address.

Main Memory



0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16 149	17	18	19
20	21	22	23 72	24	25	26	27	28	29

- When we create variables in a program, the variable name is then associated with a memory address.
  - When we use a variable in our program, the name then is used to locate the value that has been stored in that variable.
  - int age = 72;
    - For example, the age variable has the address 23 and at that address the value 72 has been stored.

Secondary Storage

- Non-volatile: data retained when program is not running or computer is turned off
- Comes in a variety of media:
  - magnetic: traditional hard drives that use a moveable mechanical arm to read/write
  - solid-state: data stored in chips, no moving parts
  - optical: CD-ROM, DVD
  - Flash drives, connected to the USB port

Input Devices

- Devices that send information to the computer from outside
- Many devices can provide input:
  - Keyboard, mouse, touchscreen, scanner, digital camera, microphone
  - Disk drives, CD drives, and DVD drives

Software-Programs That Run on a Computer

- Categories of software:
  - System software: programs that manage the computer hardware and the programs that run on them.
    - Examples: operating systems, utility programs, software development tools
  - Application software: programs that provide services to the user.
    - Examples : word processing, games, programs to solve specific problems

1.3

Programs and Programming Languages

Programs and Programming Languages

- A program is a set of instructions that the computer follows to perform a task

- In order to solve a computer problem, we start with an algorithm, which is a set of well-defined steps.
  - We then write instructions for each of these steps to create a computer program.

Machine Language

- Although the previous algorithm defines the steps for calculating the gross pay, it is not ready to be executed on the computer.
- The computer only executes machine language instructions

Machine Language

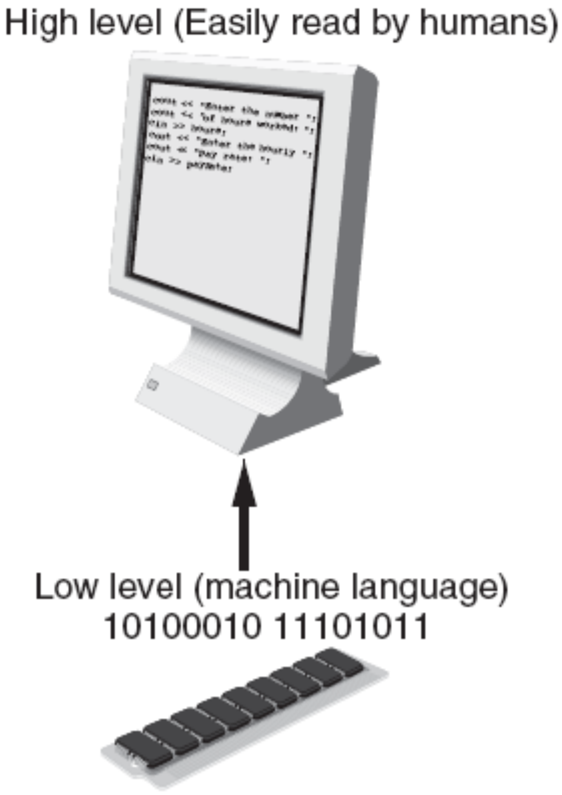
- Machine language instructions are binary numbers, such as

  10111010000000101

- Rather than writing programs in machine language, programmers use easy to read/write programming languages.

Programs and Programming Languages

- Types of languages:
  - Low-level: used for communication with computer hardware directly. Often written in binary machine code (0’ s/1’ s) directly.
  - High-level: closer to human language


High level (Easily read by humans)
Low level (machine language)
10100010 11101011

```
cout << "Enter the number ";
cout << "of hours worked: ";
cin >> hours;
cout << "Enter the hourly ";
cout << "pay rate: ";
cin >> payRate;
```

Some Well-Known Programming Languages

C++

BASIC

Ruby

Java

FORTRAN

Visual Basic

C#

COBOL

JavaScript

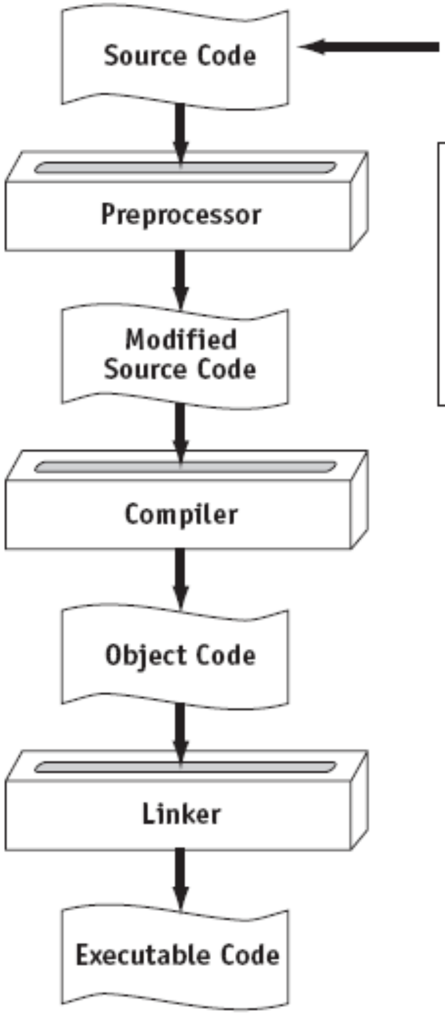
C

Python

From a High-Level Program to an Executable File

- Create file containing the program with a text editor.
- Run preprocessor to convert source file directives to source code program statements.
- Run compiler to convert source program into machine instructions.
- Run linker to connect hardware-specific code to machine instructions, producing an executable file.
- Steps b – d are often performed by a single command or button click.
- Errors detected at any step will prevent execution of following steps.

From a High-Level Program to an Executable File



Source code is entered with a text editor by the programmer.

```
#include <iostream>
using namespace std;

int main()
{
   cout<<"Hello World\n";
   return 0;
}
```

Questions?