Computer Architecture

Ch.1 Introduction & Overview

Introduction and Overview:

- General definition,
- Purpose of Digital Arithmetic and Introduction to computer architecture,
- CPU organization and its parts,
- Sketch CPU organization,
- Definition of the performance factors, reasons for binary arithmetic with computers.
- Review of basic fixed- point number representation systems (non-negative and signed integers), sign detection.

General Definitions

1- Digital arithmetic

Digital arithmetic plays an important role in the design of **general-purpose digital processors** and of **embedded systems for signal processing**, and **communications**.

- ***Digital arithmetic*** includes all aspects of the specification, analysis, and implementation of arithmetic operations (±, ×, ÷, , etc.) in systems above These aspects include number systems, arithmetic algorithms, hardware implementation of arithmetic operators (adders, multipliers, dividers), function implementation, and floating-point arithmetic.

General Definitions

2- Computer Architecture is defined as the functional operation of the individual hardware units in a computer system and the flow of information among the control of those units.

* It describes the design of an electronic computer with its CPU, , memories, and I/O interface.

* **Computer architecture includes three categories :**

– **System H/W Design**: includes all H/W components in the system.

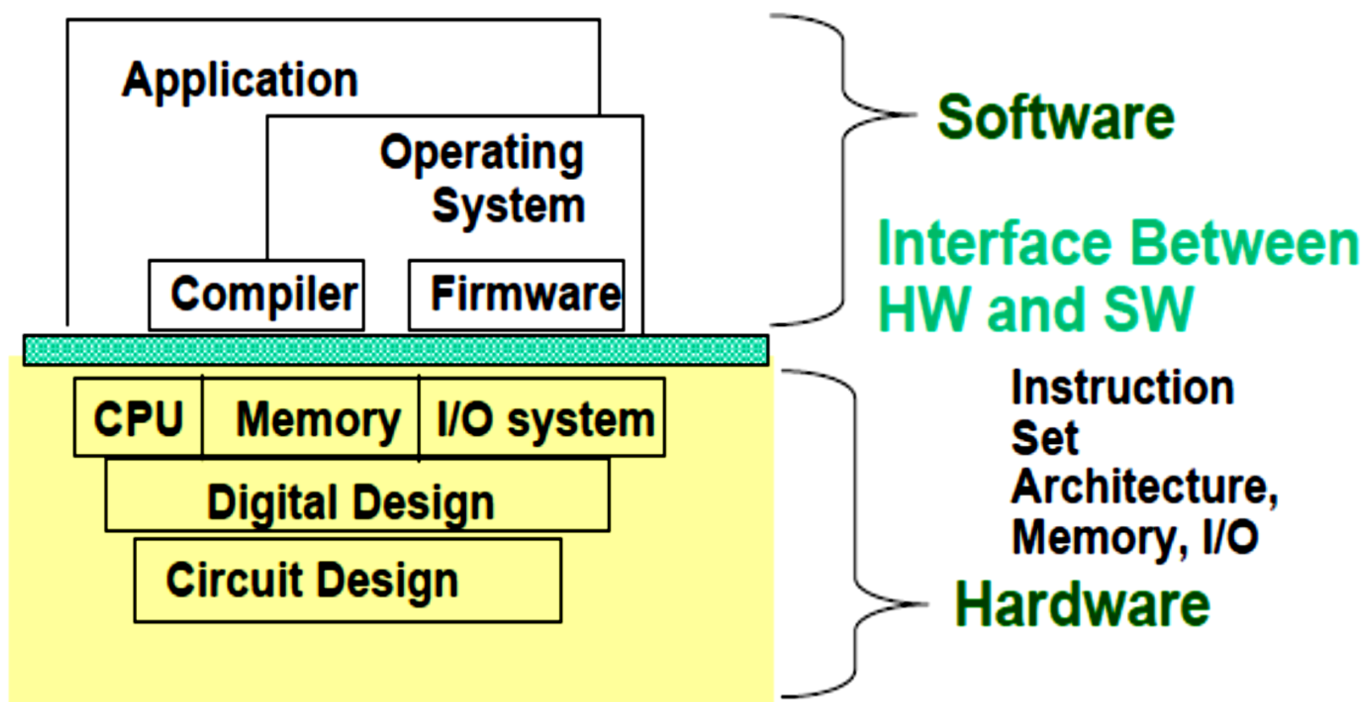
– **Instruction Set Architecture (ISA):** This is the embedded programming language of CPU. It defines the CPU functions and capabilities based on what programming it can perform or process. ISA includes word size , processor register types, memory addressing modes, data formats , and instructions that the programmer use.

- **Microarchitecture:** **Sometimes called Computer Organization.**

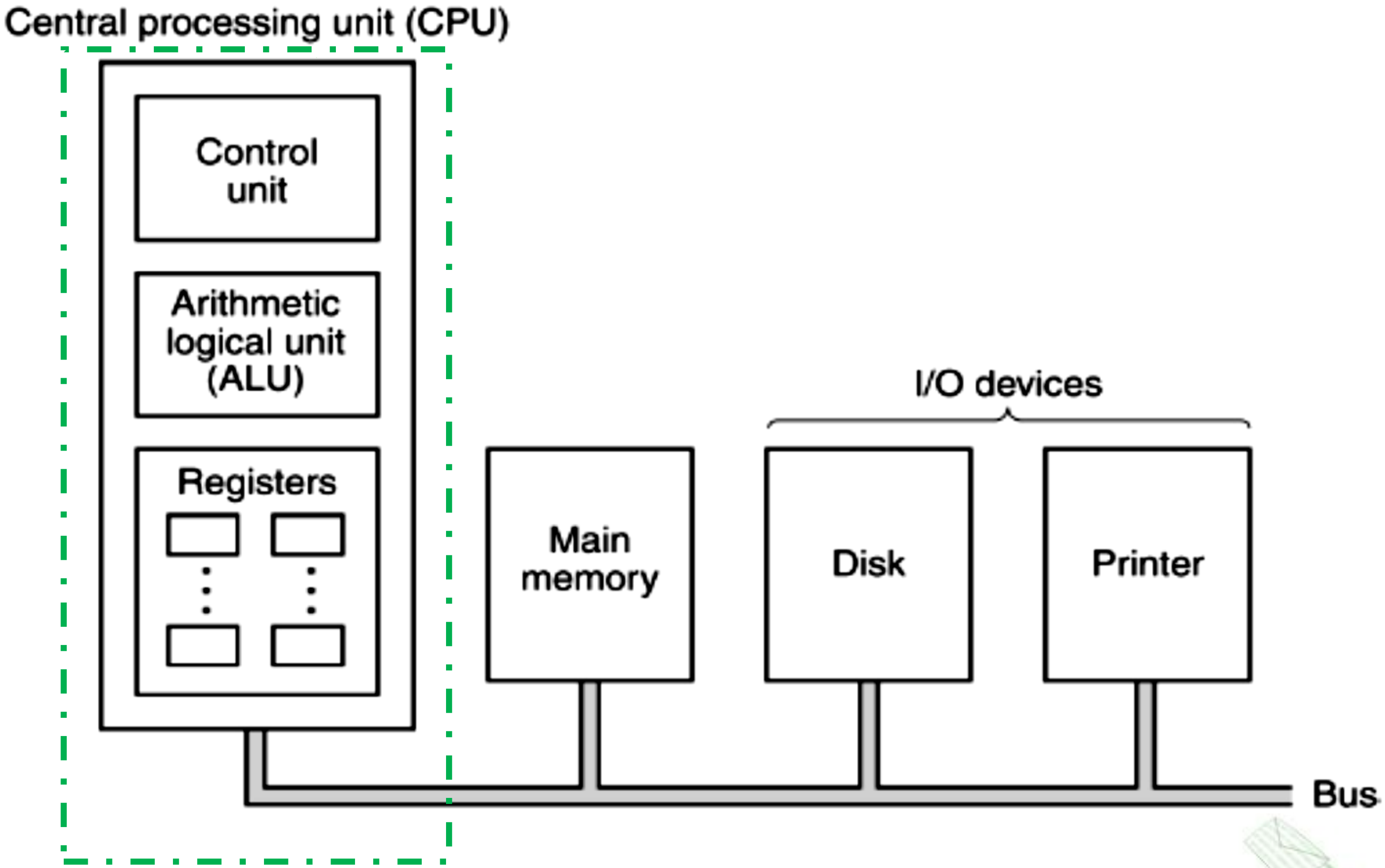
It defines the datapaths, storage elements, as well as how they should be implemented in ISA.

What is Computer Architecture?

- Coordination of levels of abstraction



CPU Organization and its Parts



The organization of a simple computer with one CPU and two I/O devices

CPU Organization and its Parts

The CPU contains three elements:

- **Register File**:includes (General Purpose Registers, and Dedicated Registers).

- **Execution Units** : In old computers called ALU (Arithmetic Logic Unit).

- **Control Unit.**

Definition of the Performance Factors

Q: When we say one computer has better performance than another , what do we mean?

Ans: **Computer systems are machines designed to perform information processing and computation. Their performance** **is typically measured by how much information processing they can accomplish per unit time.**

Definition of the Performance Factors (continue)

There are two main performance factors:

1- Response Time (or Execution Time)

Is the time between the start and completion of a task.

i.e CPU execution time = The time CPU spends computing for this task

CPU time = User CPU time + System CPU time

***User CPU time** : is the time spent to execute a user program (The time spent running the application program itself).

Definition of the Performance Factors (continue)

***System CPU time:** time spent in the OS, performing tasks the program requested of the OS (i.e is the time when the application calls operating system code).

The Speed of CPU = 1/ Execution Time ← **Frequency (MHz or GHz)**

2- Throughput: The total amount of work done in a given time (i.e No. of tasks per unit time) measured in MIPS

MIPS: Million Instruction Per Second

Definition of the Performance Factors (continue)

Note: For best CPU Performance, we should have lower execution time (or High Speed) and High throughput

.

3- Cost: includes not just processor cost , but other components (e.g., memories)

4- Power Consumption (lower power consumption is preferred).

Reasons for Binary Arithmetic with Computers

Binary system is used by almost all modern computers. Binary arithmetic are important because they simplify the design of computer components more than decimal system.

Basic Fixed- Point Number Representation (Integer Representation) Systems (non-negative and signed integers)

To Perform arithmetic and logic operations on fixed point numbers, a specific representation is required. In **digital representation,** such a number is represented by an ordered n-tuple. Each of the elements of the n-tuple is called a **Digit.** The n-tuple is called a digit vector. While the number of digits (n) is called the precision of the system representation.

Examples:

1- Binary (also called Radix-2) representation :

let X= (1110 0010 1000 0001)

X= (1110 0010 1000 0001)

- Each bit in the binary number X is a **digit**
- X: called **digit Vector of n= 16 (1-bit) digits**
- The precision of the 16-bit number (X) is **half precision**

2- Radix-4 Representation: Let Y=(2210 1132 2311 1102)

- Each element in Y called **digit** , (each digit in radix-4 consists of two binary bits).
- Y: called **digit Vector of n= 16 (2-bit) digits.**
- The precision of the 16-bit number (Y) is **Single precision** (why the number Y single precision ,not half precision?)

A) Non-negative Binary Integer Number Representation

The digit vector that represents an integer number (X) is denoted by:

$$X = (x_{n-1}\ x_{n-2}\ x_{n-3}\ \ldots\ldots\ x_1\ x_0)_2$$

A non- negative binary representation of a number (X), means that this number is positive or simply the number (X) is without a sign (i.e. absolute value of X).

Example:

$$X = (11010001)_2 \equiv +255_{10}$$

B) Signed Binary Integer Number Representation

Two representations are used to represent the signed integer numbers:

- **Sign & Magnitude Representation of a Number.**
- **2's Complement Representation of a Number**

- Sign & Magnitude (SM) Representation of a Number.

In a sign & magnitude representation system, a number (B) is represented by a pair of (b_s , b_m)

Where

b_s: represents the sign bit (if $b_s = 0$,the number B is positive , else if $b_s = 1$, then B is negative)

B) Signed Binary Integer Number Representation (continue)

Example:

Represent the following numbers in SM binary representation:

- **A= 55_{10}**
- **B= - 69_{10}**

A= 55_{10} =$(11011)_2$

in SM binary representation A =(0 11011 $)_2$

a_s a_m

- **B= - 69_{10} ⟹ b_s= 1 , b_m = + 69_{10} = 1000101_2**
- **Thus in SM representation B = (11000101)**

B) Signed Binary Integer Number Representation (continue)

- 2's Complement Representation of a Number

In the 2's complement system representation, there is no separation between the representation of the sign and the representation of the magnitude. In this representation, the MSB of the number represent the sign of the number.

Example:

Let us represent the same numbers but in 2's complement binary representation: **$A = 55_{10}$, $B = -69_{10}$**

$A = 55_{10}$ $=(11011)_2$. A is positive numbers, thus its 2's complement representation is $A = (011011\ )_2$

$B = -69_{10}$ $b_s = 1$, $b_m = +69_{10} = 1000101_2$

Thus in SM representation B = (11000101)

- 2’s Complement Representation of a Number (continue)

*** B= - 69_{10}**

B = + 69_{10} = 01000101_2

Take the 2’s complement of +69 to get -69. This is done by converting each bit of B (i.e. 1’s complement of B), then add 1 to the LSB of the 1’s complement of B.

$$\bar{B} = 10111010$$

$$\begin{array}{r} 1 \quad + \\ \hline \end{array}$$

B = 10111011

Sign Detection

There are two algorithms to detect the sign of a number, depending on the representation type of a signed number:

*If the number (X) is represented in a SM representation , the sign detection is trivial since there is a sign bit in the representation ***(i.e. the sign of X is x_s).***

* For 2's complement representation of a signed number, a sign detection will take it later.

Thank You..