# Lecture Two

# Computer organization & Architecture

Yarmouk University College

Dept. of Cybersecurity Engineering

$2^{nd}$ Stage

By Asst. Lect. Anas Yassin Abbas

# Why Computer Organization and Architecture (COA)?

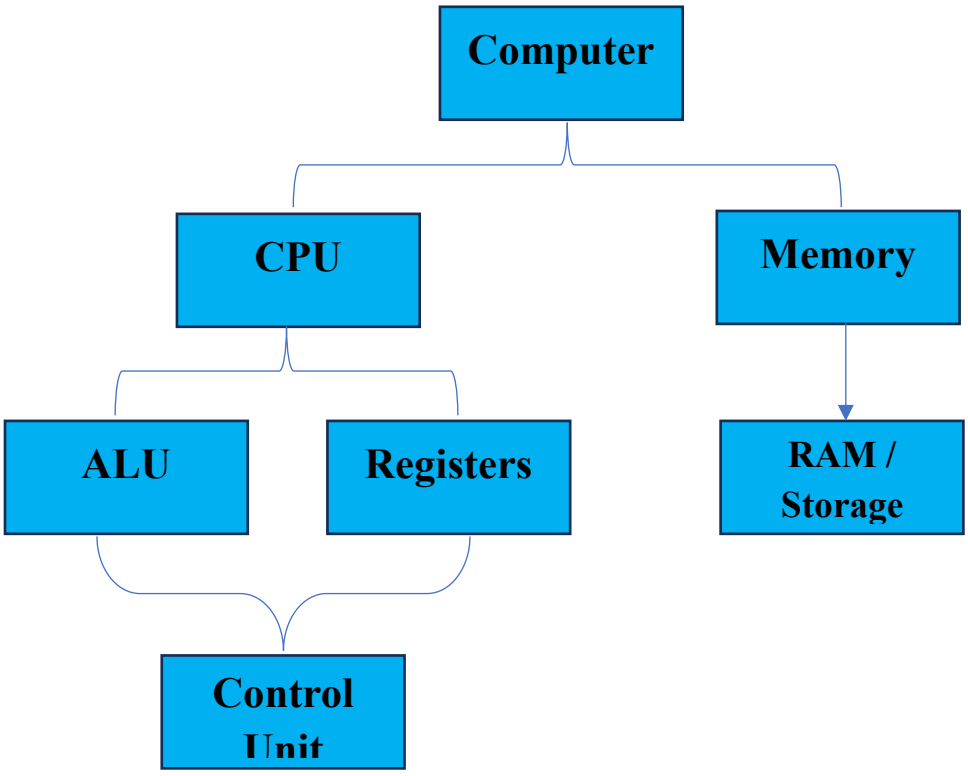
We study Computer Organization and Architecture (COA) to understand how a computer actually works inside.

Computer Architecture tells us what a computer is designed to do. Computer Organization tells us how the hardware is arranged to do it. Assembly language allows us to communicate with that hardware at a low level.

When we use a computer, we normally see:




But internally, many components work together:




COA explains how these components communicate and execute instructions.

# Instructions

We learn how an instruction moves through the processor:

**Fetch**

↓

**Decode**

↓

**Execute**

↓

**Store result**

For example:

MOV AL, 05H

ADD AL, 03H

The CPU must fetch, decode, and execute these instructions using its internal hardware.

# Why is COA important for an engineer?

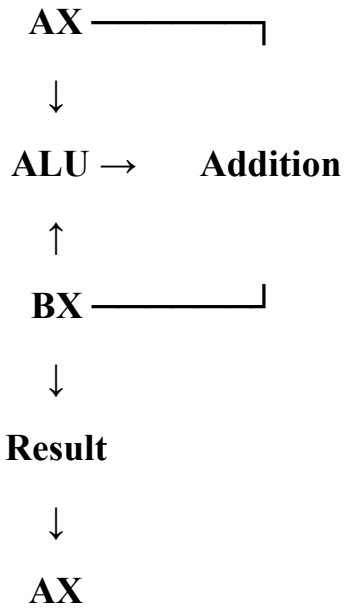
Because software and hardware are **not independent**.

For example, suppose we write:

ADD AX, BX

A programmer sees **an instruction**.

But inside the CPU, hardware performs several operations:




So, COA helps you understand what happens between **writing an instruction** and **obtaining the result**.

## What is Assembly Language?

**Assembly language** is a low-level programming language that uses symbolic instructions to control a processor.

Instead of writing binary directly:

**10110000 00000101**

we can write something more understandable:

**MOV AL, 05H**

# Why do we use Assembly?

Assembly is useful when we need direct and precise control over the processor and hardware.

### 1. Understanding the CPU

Assembly shows students how instructions interact with:

**registers**

**ALU**

**memory**

**flags**

**buses**

### 2. Embedded systems

Assembly can be useful in:

**microcontrollers**

**embedded systems**

**real-time systems**

**hardware control**

### 3. Performance-sensitive code

In some situations, low-level code can be useful when precise control over instructions and resources matters.

### 4. Hardware interaction

Assembly can directly work with processor registers and, depending on the system, hardware interfaces.

### 5. Operating systems

Low-level processor operations are important in areas such as:

**boot processes**

**interrupt handling**

**context switching**

**hardware initialization**

### 6. Understanding compiled programs

Assembly is also useful for understanding what happens to high-level code after compilation.

For example, a C statement:

```
x = a + b;
```

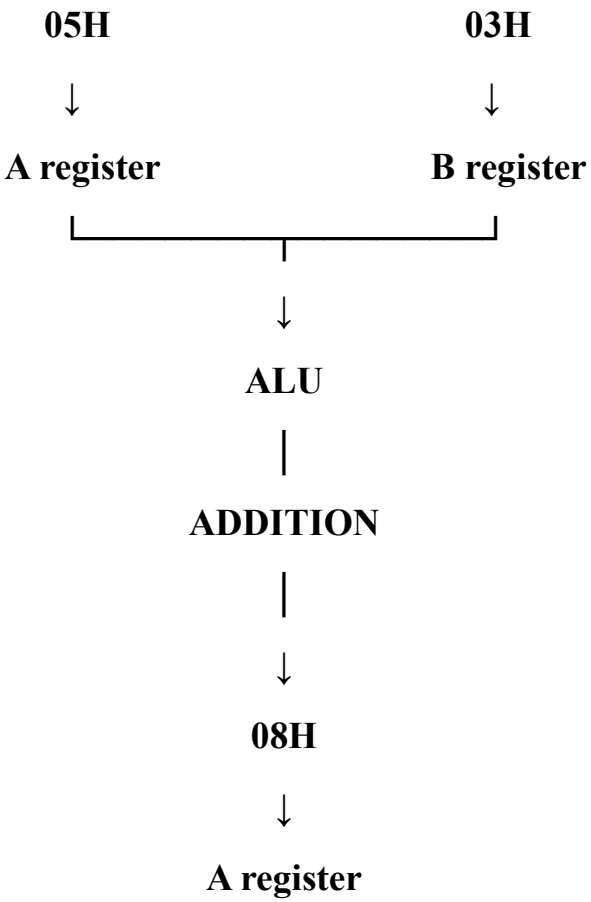
may ultimately involve instructions conceptually similar to:

```
MOV AX, a
ADD AX, b
MOV x, AX
```

MVI A, 05H

MVI B, 03H

ADD B



# General Architecture of Digital Computer

At present there are many types and sizes of computers available. These computers are designed and constructed based on digital and Integrated Circuit (IC) fabrication technology. A digital computer is a machine that can be used to solve problems for people and carrying out the tasks by following the instructions given to it. A sequence of instructions describing how to perform a certain task or job is called a program. There are two basic components of computer system architecture:

a) Computer Hardware

b) Computer Software

## Computer Hardware

General purpose computer as shown in figure blow contains four basic hardware blocks that are:

- arithmetic and logic unit (ALU),
- memory unit (MU),
- input/output unit (IOU), and
- control unit (CU).

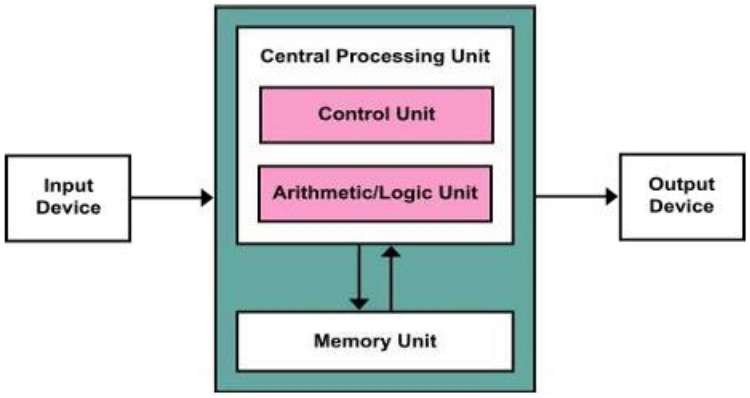



Figure (1): General purpose computer organization

Input/output (I/O) devices input and output data into and out of the memory unit. In some systems, I/O devices send and receive data into and from the ALU rather than the MU. Programs reside in the memory unit. The ALU processes the data taken from the memory unit (or the ALU) and stores the processed data back in the memory unit (or the ALU). The control unit coordinates the activities of the other three units. It retrieves instructions from programs resident in the MU, decodes these instructions, and directs the ALU to perform corresponding processing steps. It also supervises I/O operations.

## Component 1: Arithmetic and Logic Unit (ALU)

The ALU unit functions are performing arithmetical operations, i.e. addition, subtraction, multiplication, division as well as logical operations. The control unit gives the instruction to the ALU which operations they have to perform and where it supplied (store purpose), it is directed to perform the operations.

## Component 2: Memory Unit (MU)

The memory unit or storage section of the computer consist the devices used to store the data or information during the process. Memory unit is used to hold intermediate and final result of computer program. There are various types of memory which are used in computer system.

## Component 3: Input / Output Unit (IOU)

The input unit contains the hardware devices those are used to enter the data in to computer system, Keyboard and mouse are most common devices. The output contains the hardware devices those are used to output the data from the computer system, Monitor and printer are most common output devices. Nowadays so many other I/O devices are used as shown below.

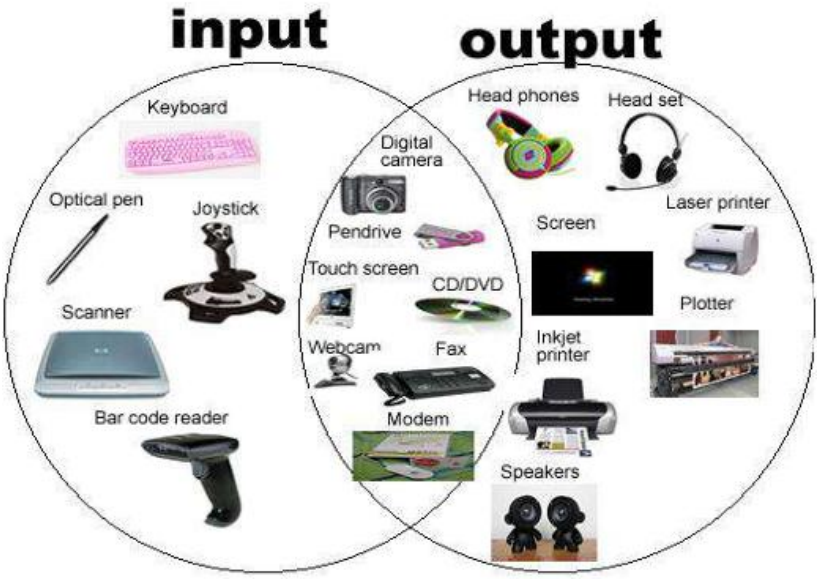



Figure (2): Input / Output hardware devices

## Component 4: Control Unit (CU)

The control unit maintains the sequence of the operation, controlling the actions of all other units it can perform the instruction which constitute the program and direct as per the operation perform by machine.

## Computer Software

Without software, most hardware would sit there doing nothing or perform specific tasks. To make most hardware run we need to use software, and the task here is to select the correct type of software for each job. The two main classifications of software that all programs fit under are:

- System software
- Application software

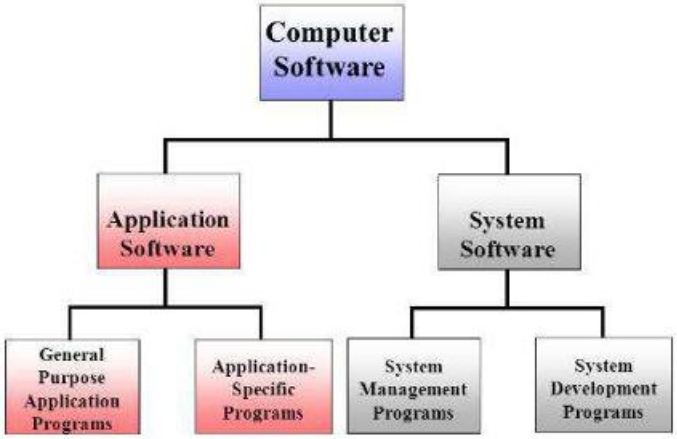



Figure (3): Basic classification of computer software

## System software

software designed to operate the computer hardware and to provide a platform for running application software. Modern computers are complex machines involving many different parts. To keep it running well you will need system software. System software will handle the smooth running of all the components of the computer as well as providing general functionality for other programs to use, tools to speed up the computer, tools to develop new software and programs to keep you safe from attacks. There are several different types of system software that we will look at in more detail very shortly:

Operating Systems are a collection of programs that make the computer hardware conveniently available to the user and also hide the complexities of the computer's operation. The Operating System (such as Windows 7 or

Linux) interprets commands issued by application software (e.g. word processor and spreadsheets). The Operating System is also an interface between the application software and computer. Without the operating system, the application programs would be unable to communicate with the computer.

## Utility programs

are small, powerful programs with a limited capability, they are usually operated by the user to maintain a smooth running of the computer system. Various examples include file management, diagnosing problems and finding out information about the computer etc. Notable examples of utility programs include copy, paste, delete, file searching, disk defragmenter, disk cleanup. However, there are also other types that can be separately installable from the Operating System.

## Library programs

are a compiled collection of subroutines (e.g. libraries make many functions and procedures available when you write a program)

## Translator software (Assembler, Compiler, Interpreter)

 Assembler translates assembly language programs into machine code (A binary code that a machine can understand).

 Compiler translates high level language code into object code (which is the machine language of the target machine).

 Interpreter analyses and executes a high-level language program a line at a time. Execution will be slower than for the equivalent compiled code as the source code is analyzed line by line.

## Application software

software designed to help the user to perform specific tasks Application software is designed for people like me and you to perform tasks that we consider useful. This might be the ability of a scientist to work out statistical information using a set of results, or someone who wants to play the latest

computer game. There are several categories of Application software that we'll look into shortly:

- General purpose application software.
- Special purpose application software.
- Bespoke application software

## 8085 Microprocessor Architecture

Intel 8085, an 8-bit NMOS microprocessor is available in the form of 40 Pin dual in line IC package. It is fabricated on a single LSI chip. It operates on +5 V d.c. supply. The clock speed used in this microprocessor is about 3 MHZ. General Purpose 8-bit microprocessor is capable of addressing up to 64 K bytes (i.e. 216 = 65536 bytes) of memory. 8085 Microprocessor can fetch instructions from a memory, decode and execute them i.e. performs certain arithmetic and logical operations, accept data from input device, and send results to output devices.

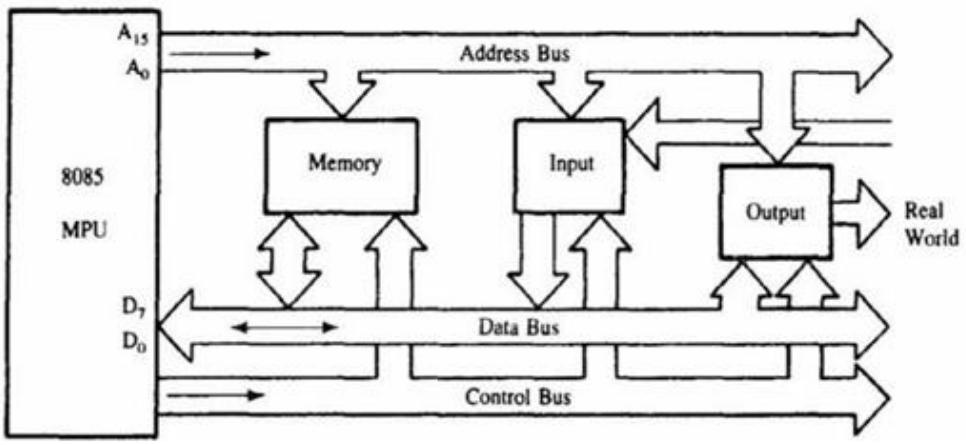



Figure (4): General Architecture of 8085 Microprocessor

In a computer system ALU and Control Unit are combined in one unit called Central Processing Unit (CPU). The CPU is analogous to the human brain as all the decisions as per the instructions are made by this unit. All other parts

are also controlled by this unit. A microprocessor is an integrated circuit designed for use as Central Processing Unit of a computer. The CPU is the primary and central player in communicating with devices such as memory, input and output. However, the timing of communication process is controlled by the group of circuits called control unit.

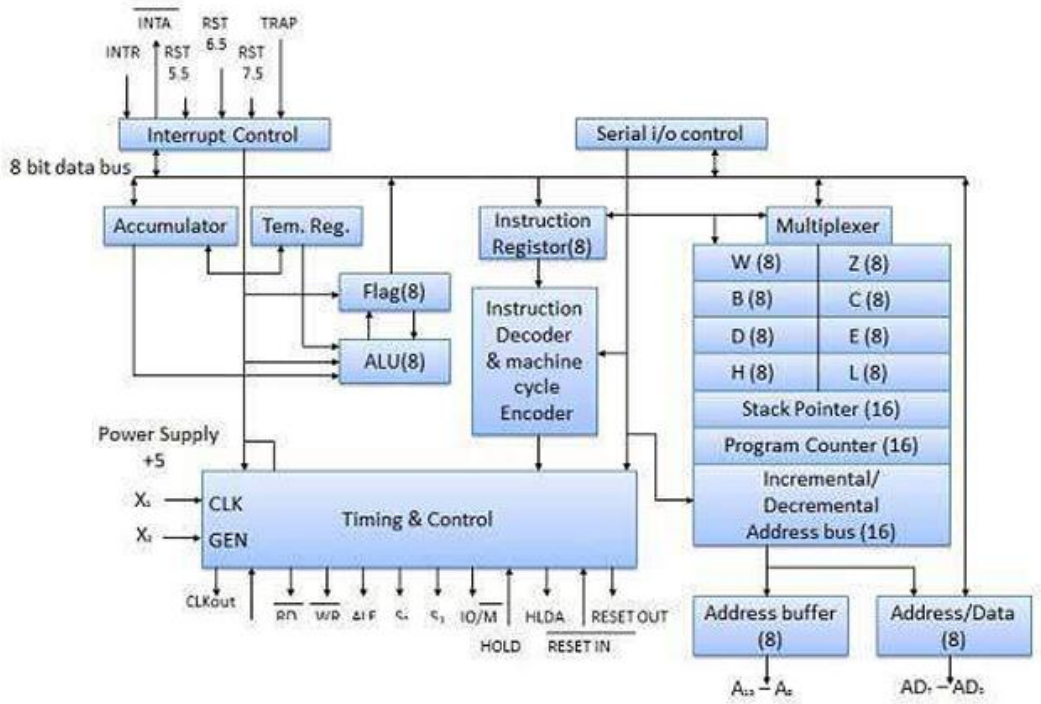
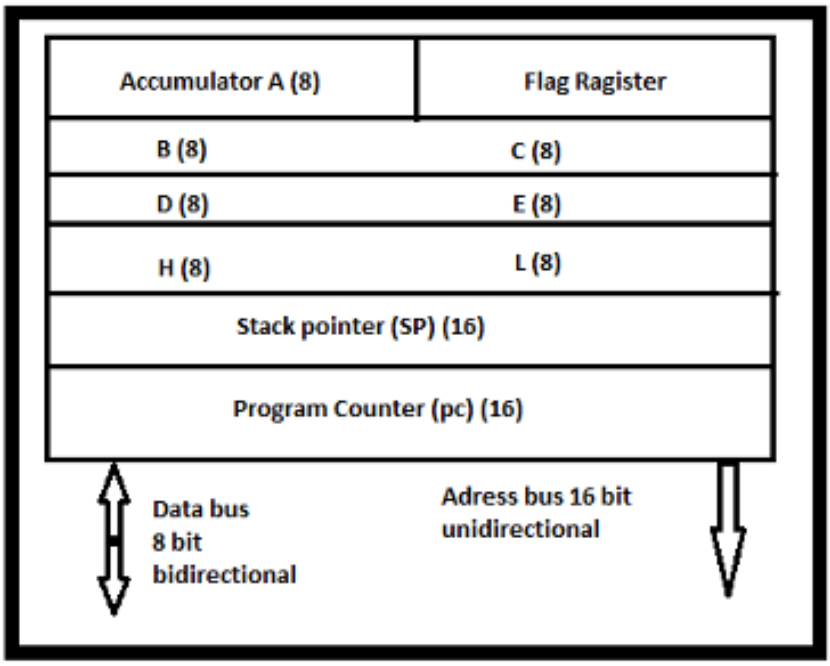



Figure (5): 8085 Microprocessor Block Diagram

The main functional components of 8085 microprocessor are as given below:

- Register Section
- Arithmetic and Logic Unit
- Timing and Control Section
- Interrupt Control
- Serial Input / Output Control

## Register Section




The 8085 microprocessor contains eight addressable 8-bit registers namely:

- A (Accumulator) register
- F Flag register (Flag flip-flops)
- B register
- C register
- D register
- E register
- H register
- L register

Out of these registers B, C, D, E, H and L registers are 8-bit general purpose registers. These registers can either be used as single register store 8-bit data for processing or a combination of two registers as 16-bit register pair. The valid register pairs are B-C, D-E or H-L register pairs. The higher order byte of 16-bit data is stored in first register (B in B-C register pair), and low order byte in the second register (C in B-C register pair).

The H-L register pair can also be used for register indirect addressing; since this register pair can also function as data pointer. The large number of general-purpose registers gives more flexibility and ease in the programming. However, the general-purpose registers are limited as registers occupy more space on the silicon chip.

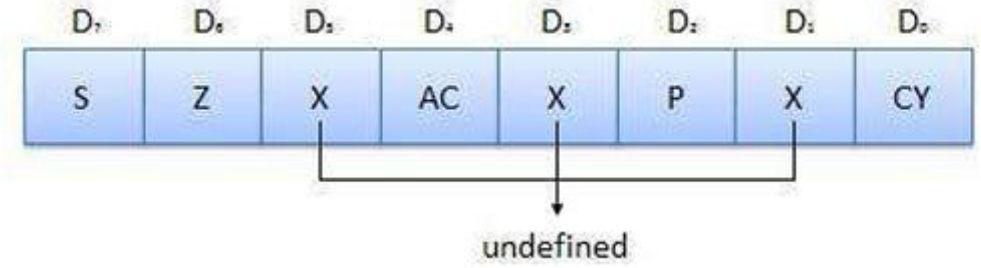
Beside these general-purpose registers, the 8085 has remaining two 8-bit registers Accumulator (A) and Flag (F) as special purpose registers and two 16-bit registers namely Program counter (PC) and stack pointer (SP).

## Accumulator (A)

is an 8-bit register which stores the results of arithmetical and logical operations. It is also used to receive data from input port to microprocessor and to send data to output port from microprocessor. It is referred as register A in the program.

## Flag Register (F)

is also an 8-bit register which indicate status of the accumulator after any arithmetical or logical operation. Out of 8 bits, five are defined as flags. These flags are given below.



 **Sign Flag (S)** It indicates the sign of the number. If the number is negative then sign flag is set (changed to 1) and if positive then reset (changed to 0).

 **Zero Flag (Z)** If after any arithmetical or logical operation, all the bits in accumulator is zero then zero flag is set otherwise reset.

 **Auxiliary Carry Flag (AC)** If carry is generated from D3 to D4 in the accumulator after any operation then auxiliary carry flag is set otherwise reset. This flag is used internally by microprocessor for BCD operations and cannot be used by the programmer.

 **Parity Flag (P)** If after any arithmetical or logical operation number of 1's in accumulator is even, then parity flag is set, otherwise reset.

 **Carry Flag (CY)** The size of accumulator is 8 bits. If after any operation 9th bit is generated then carry flag is set, otherwise reset.