ECOLE SUPÉRIEURE POLYTECHNIQUE INTERNATIONALE PRIVÉE DE SFAX

Chapter 3

Processor
8086 microprocessor

Dr. Eng. Yasmine KOUBAA
PhD in electrical engineering

Digital circuits and computer architecture

Academic year 2023-204

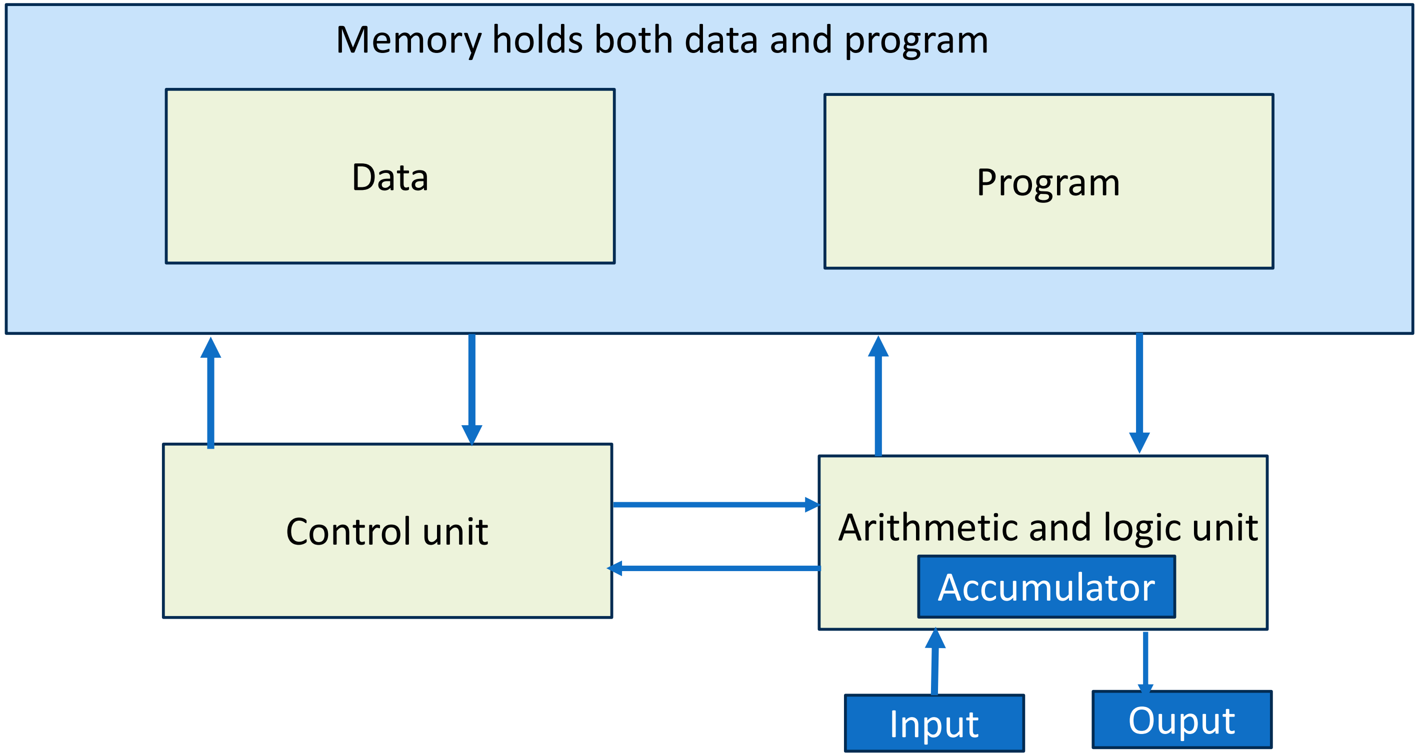
Von New Man Architecture

John von Neumann is a Hungarian American matematician and physist who has made important contributions in computer science. .

The Von New Man Architecture

- In 1945 he described a computer architecture in which the data and the program are both stored in the computer’s memory.
- This novel idea meant that the computer built with this architecture would be much easier to re-program
- This is the fundamental design concept behind all modern computer systems

Von New Man Architecture



Von New Man Architecture

Control Unit
Responsible for decoding the instructions and controlling how data moves around computer system

Arithmetic and Logic Unit
Carries out calculations and logical decisions required by the program instruction (Addition, soustraction,…)

Bus
The wires that carry data around the computer

Registers
Are memory locations with specific purpose

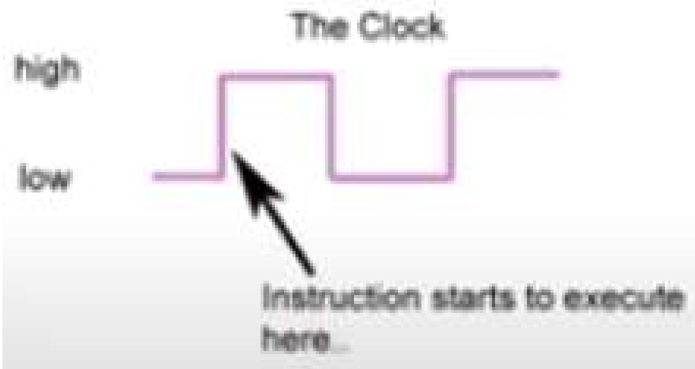
Processor (CPU)

- Central Processing Unit,
- The CPU is seen as the main and most crucial integrated circuit (IC) chip in a computer, as it is responsible for interpreting most of computers commands.
- CPUs will perform most basic arithmetic, logic and I/O operations, as well as allocate commands for other chips and components running in a computer.
- The CPU **fetches instructions from memory (RAM), decodes the instructions and then executes these instructions**. The instructions are provided by a computer program.
- Is made by a million of transistor that combine to build the logic gates to process the data and instructions

Processor (CPU)

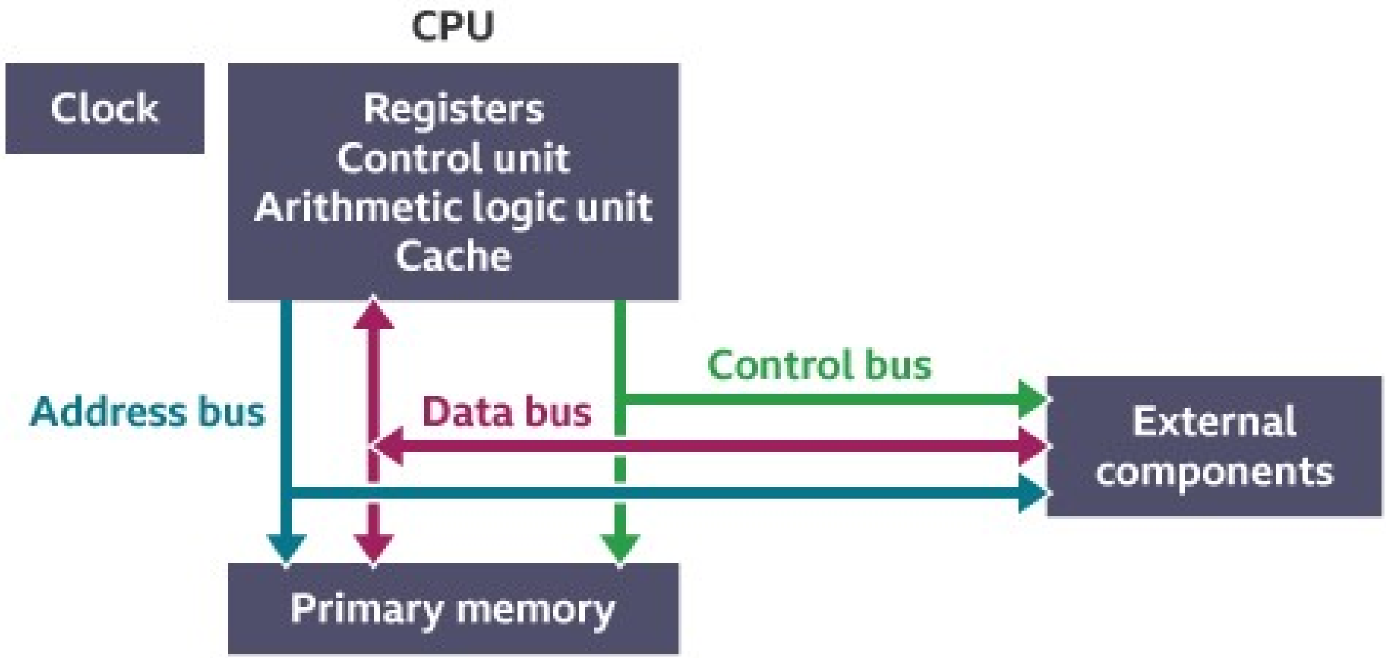
Clock speed

- The clock is a tiny quartz cristal inside the CPU chip that ‘ticks’ at a steady speed.
- The CPU can only do something when the clock ticks. In between each tick it does nothing. **During each tick the CPU process a single instruction**.



Example:
A CPU with a clock speed of **3.2 GHz** executes **3.2 billion cycles per second**.

Inside the CPU



Inside the CPU

Control unit : responsible for decoding the instructions and it sends out signals to control how data moves around the parts of the CPU and memory to execute these instructions

Arithmetic and logic unit: carries out calculations and logical decisions required by the program instruction (for example addition, soustraction and comparision such as equal to, greater than, or less than)

Cache: is a small amount of high-speed random access memory (RAM) built directly within the processor. It is used to temporarily hold data and instructions that the processor is likely to reuse.

Inside the CPU

Clock : The CPU contains a clock which is used to coordinate all of the computer's components.
The clock sends out a regular electrical pulse which **synchronizes** (keeps in time) all the components.

Registers : Registers are memory locations within the CPU that have specific purpose and can be addressed very quicly. (this is because registers are built in the CPU, unlike RAM or HDD

Registers hold data and instructions that the computer is using

- Memory Address Register (MAR)
- Memory Data Register (MDR)
- Accumulator
- Program counter (PC)
- Current Instruction register (CIR)

Inside the CPU

Registers

- **Accumulator** : Stores the results of calculations made by the ALU
- **Program Counter (PC) :** Keeps track of the memory location for the next instruction to deal with. The PC then passes this next address to the Memory Address Register
- **Memory Address Register (MAR) :** The MAR stores the memory location for data or instructions that needs to be fetched from memory or stored into memory
- **Memory Data Register (MDR)** : Register used to store any data or instructions fetched from memory or any data that is to be transfered to stored in memory
- **Current Instruction Register (CIR)** : Register that stores the most recently fetched instruction while it is waiting to be decoded or executed

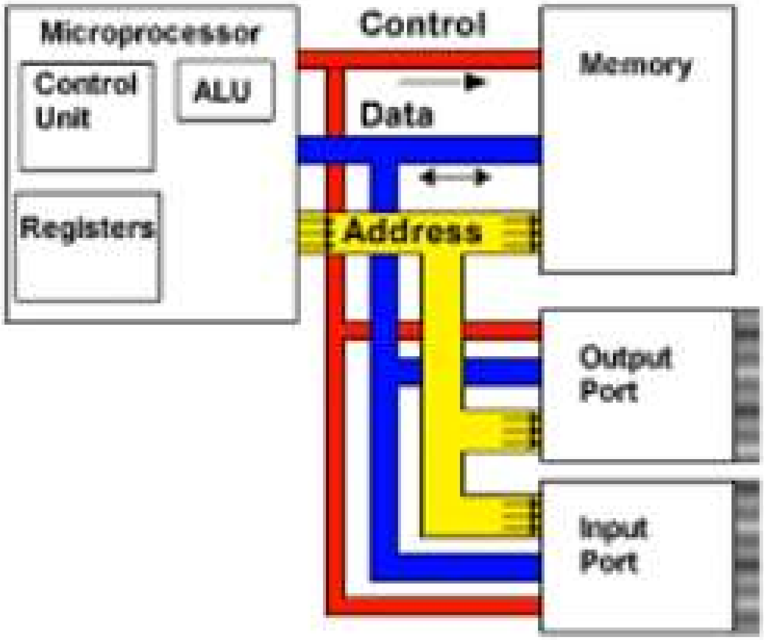
Inside the CPU

Buses

In order to enable data and control signal to move around the CPU and memory there are a number of buses.

A bus is a communication channel through which data can be moved.

There are three main buses inside the computer to consider with the CPU.



Inside the CPU

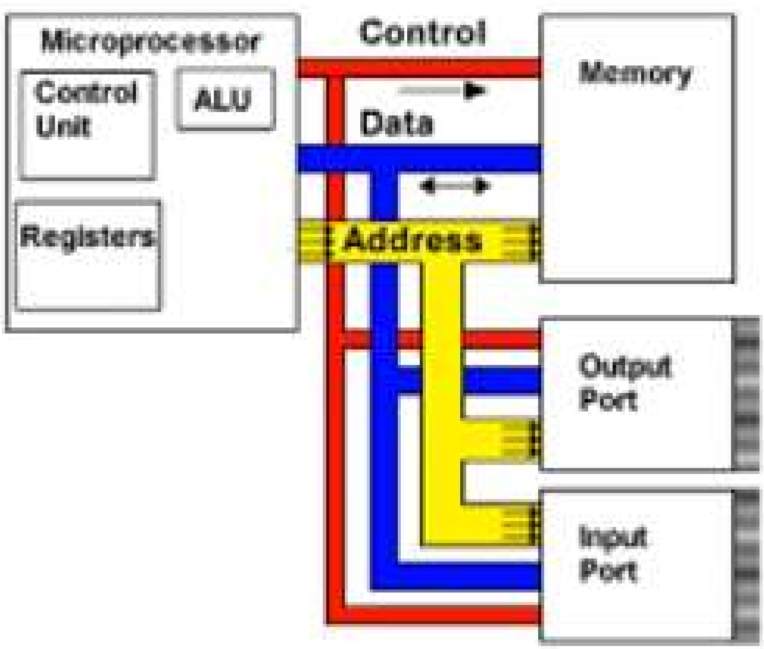
Buses

Control bus: Carries control signals around the CPU and memory indicating whether the operation is a read or a write and ensuring that the operation happens in the right time. The control bus is unidirectional.

Address bus: carries memory addresses from the processor to other components such as primary storage and input/output devices. It is unidirectional, it works only from CPU to memory

Data bus : Carries data between CPU and memory. Data bus is bidirectional

- For a write operation the CPU will put the data on the data bus to be sent to memory.
- For read operation, the data will be taken from a memory block and sent to the CPU.



Summary

- The purpose of CPU is to process data and control other components within the computer system.
- The CPU is located in the mother board.
- The three main parts of CPU are Control Unit, Registers and Arithmetic and Logic Unit.
- The three main buses are Data bus, Address bus and Control bus. They connect parts of the computer system.
- The CPU runs at speed of clock.

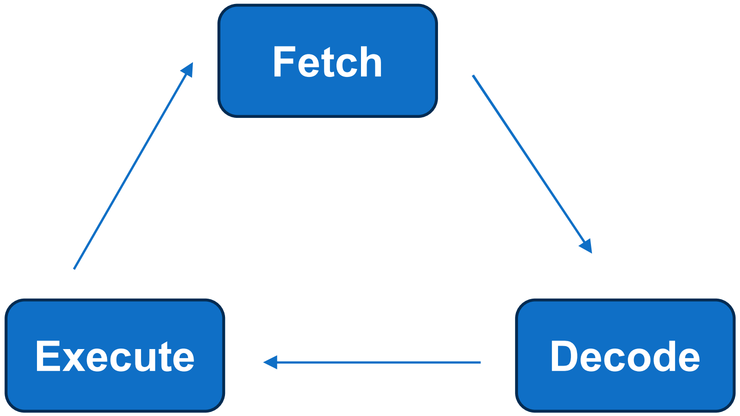
Fetch Decode Execute

Fetch Decode Execute cycle
The processor is continually fetching new instructions from memory, decoding them, then executing them.

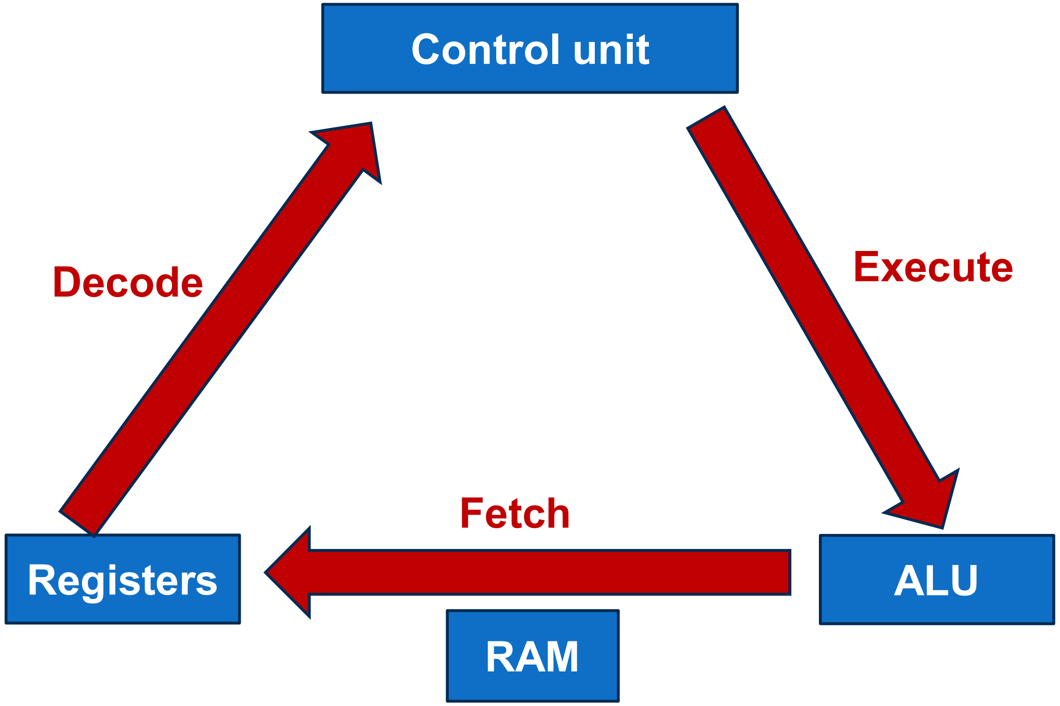
- **Fetch**
The instruction is moved from the memory to the CPU

- **Decode**
The instruction is understood by the CPU

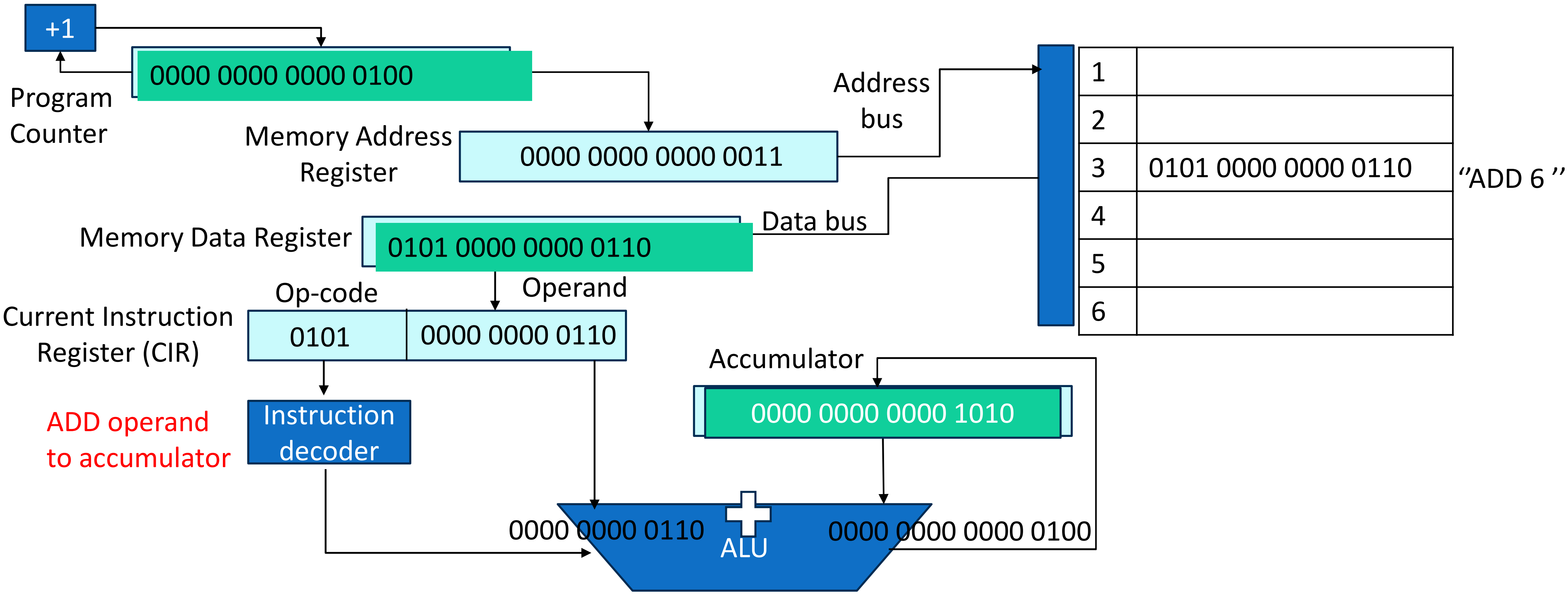
- **Execute**
The instruction is carried out



Fetch Decode Execute



Fetch Decode Execute cycle



Fetch Decode Execute

Fetch

- The address to be fetched is moved from the PC (Program Counter) and placed in MAR (Memory Address Register)
- The instruction is transferred from memory to the MDR (Memory Data Register)
- Next, the instruction in the MDR is coppied into CIR (Current Instruction Register)
- The PC (Program Counter) is incremented by 1

Fetch Decode Execute

Decode

- The control unit reads the content of CIR.
- It checks that is a valid instruction i.e, it is a part of its instruction set (Each type of CPU has its own instruction set that the control unit understand)
- If the instruction is not valid, the program will crash.

Fetch Decode Execute

Execute

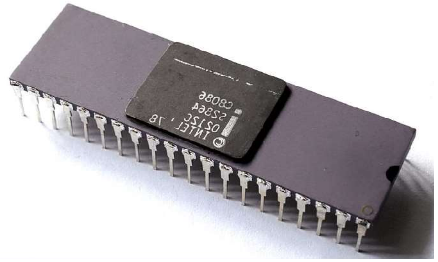
- The instruction is carried out by the CPU
- This process may use the accumulator and ALU and may fetch additional data from the memory

History of microprocessor

Processor	No of bits	Clock Speed (HZ)	Year of introduction
Intel 4004	4	750 k	1971
8080	8	2M	1974
8085	8	3M	1976
8086	16	5, 8 or 10M	1978
8088	16	5, 8 or 10M	1979
Pentium	32	66M	1993
Pentium II	32	233 to 500M	1997
Pentium III	32	500M to 1.4G	1999
Pentuim IV	32	1,2 to 3 G	2000
Core 2 duo	64	1,2 to 3 G	2006
I3, i4 and i7	64	2,4G to 3,6G	2010

Why studying 8086 microprocessor ?

- The 8086 microprocessor is the basic of the current processors
- Each successive processor gradually adds new instructions and features
- New processors supports the instruction set of the previous processor
- This is called upward compatibility which
- Allows a program written for 8086 to work on a new computer with a Pentium processor
- So, studying 8086 will give us a simple introduction to the entire family of the Intel 80X86 processors.



8086 Microprocessor

It was designed by Intel in 1976

It is the first member of the x86 family of microprocessors(8086, 80186, 80286, 80386, 80486, Pentuim,…), which includes many popular CPUs used in personal computers

It is a **16-bit** Microprocessor having **20 address lines** and **16 data lines** that provides **up to 1MB storage**

It has **multiplexed** address and data bus AD0-AD15 & A16-A19

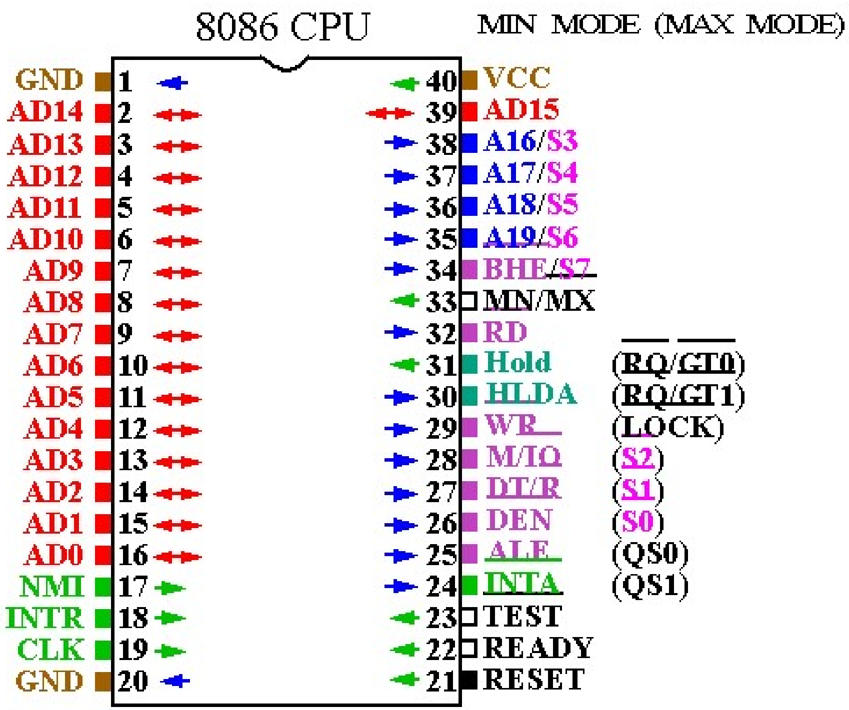
It is fully compatible with 8088:
- Instruction Set Architecture (ISA) is identical
- The only difference is in the data bus (only 8 bits for 8088)

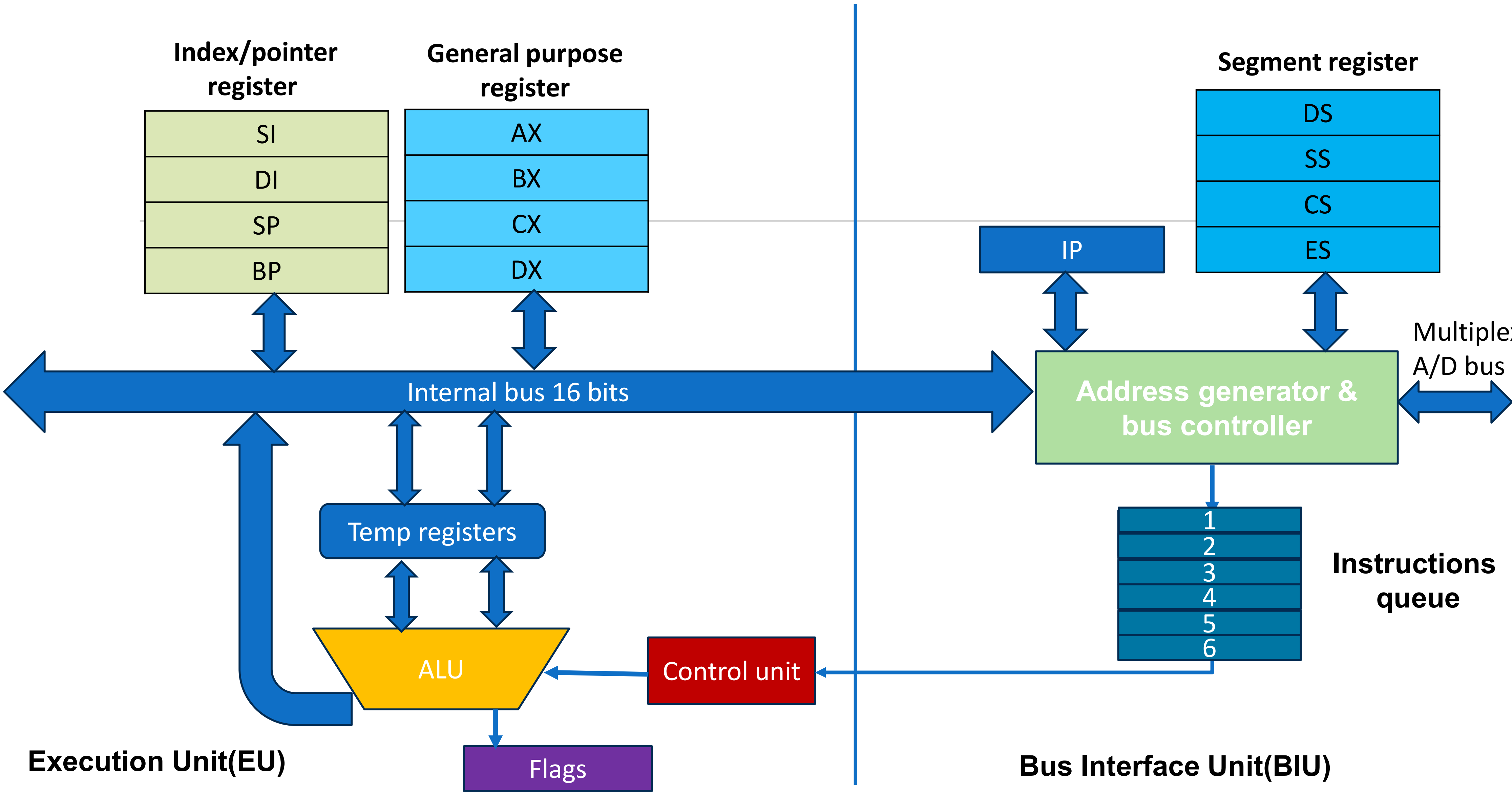
The 8088's programs will run a littele slower than 8086's programs because it needs to exchange 16 bit words in two stages

8086 Microprocessor

Operates in various clock frequencies (5, 8 or 10 MHz)

Introduced in DIP form (Dual In-line package) 40-pin.




Index/pointer register
SI
DI
SP
BP
General purpose register
AX
BX
CX
DX
Segment register
DS
SS
CS
ES
IP
Multiplex
A/D bus
Internal bus 16 bits
Address generator & bus controller
Temp registers
1
2
3
4
5
6
Instructions queue
ALU
Control unit
Flags
Execution Unit(EU)
Bus Interface Unit(BIU)

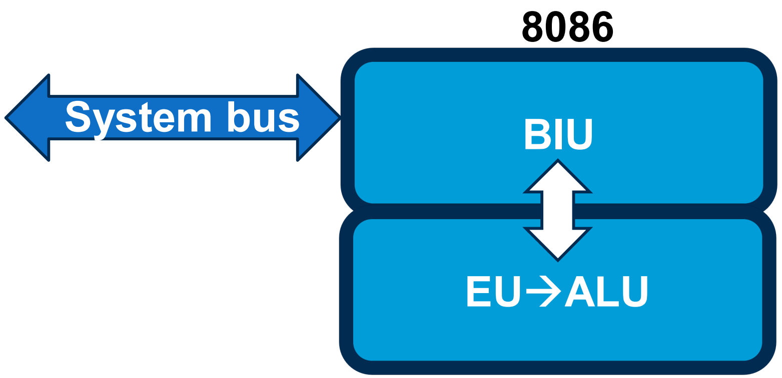
Architecture of 8086 microprocessor

8086 Processor → **Bus Interface Unit(BIU)**

8086 Processor → **Execution Unit(EU)**

Dividing the work between these two units' speeds up processing

I/O devices
External Memory



Architecture of 8086 microprocessor

Bus Interface Unit (BIU)

- The BIU handles all transfers of data and addresses on the buses for the execution unit.
- Generates address of instruction in memory
- Stores instruction in queue
- Establishes communications with the system bus

The Execution Unit (EU)

- The EU has a 16-bit arithmetic logic unit (ALU) which can add, subtract, AND, OR, XOR, increment, decrement, complement or shift binary numbers.
- The main functions of EU are:
  - Decoding of Instructions
  - Execution of instructions

Architecture of 8086 microprocessor

Steps

- EU extracts instructions from top of queue in BIU
- Decode the instructions
- Generates operands
- Perform the operation specified by the instruction on operands

Architecture of 8086 microprocessor

Why is the architecture divided into two sections ?

It is because of **piplining**

Both units operate simultaniously

Fetching the next instruction while the current instruction executes

- While the EU is decoding an instruction or executing an instruction, the BIU fetches up to six instruction bytes for the following instructions.

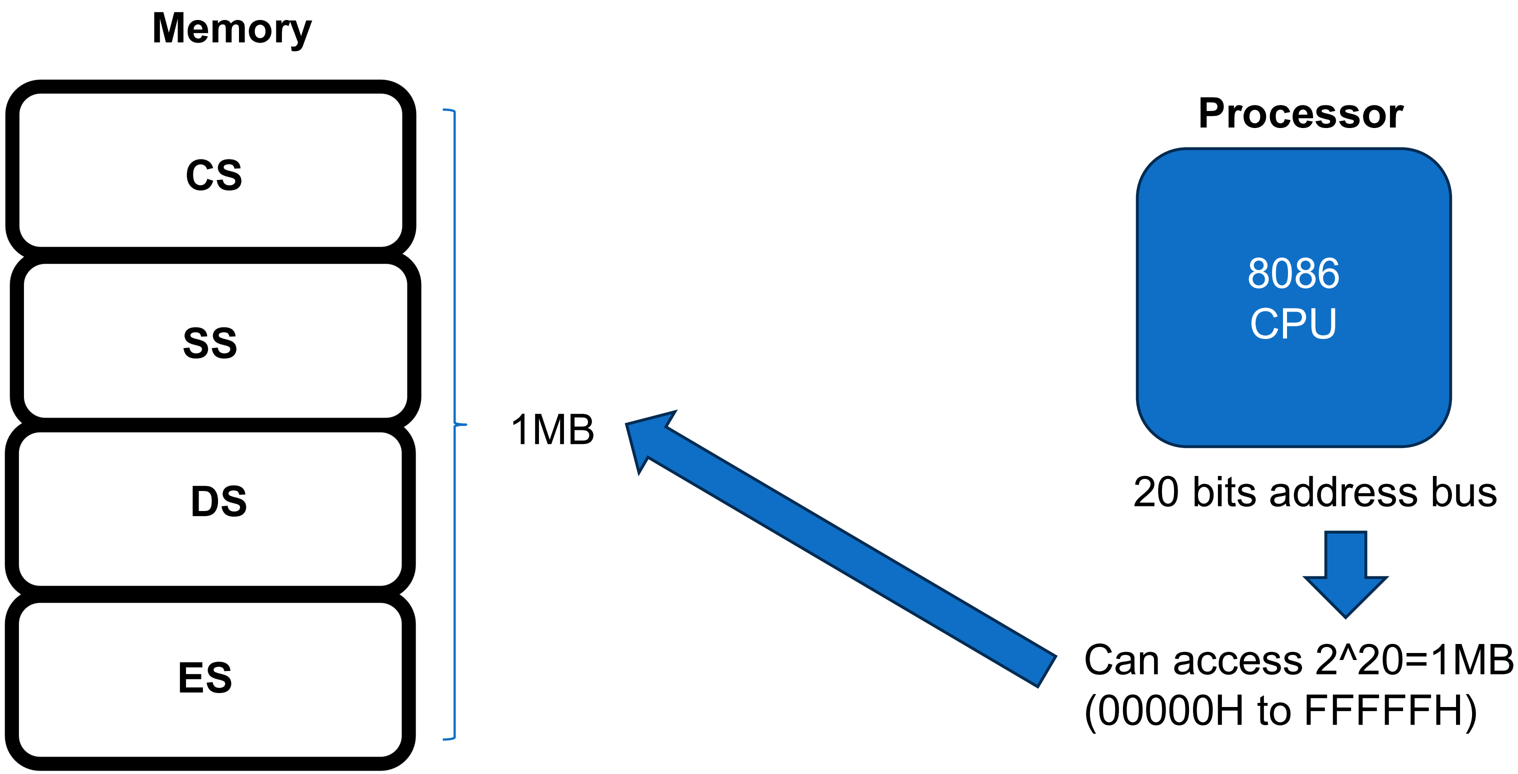
EU continiously active

In this way we are saving a lot of time

Memory segmentation

- The 8086 microprocessor has a **segmented memory architecture**, which means that memory is divided into segments that are addressed using both a **segment register** and an **offset**.
- The segment register points to the start of a segment,
- The offset specifies the location of a specific byte within the segment.
- This allows the 8086 microprocessor to access large amounts of memory, while still using a 16-bit data bus.

Memory segmentation

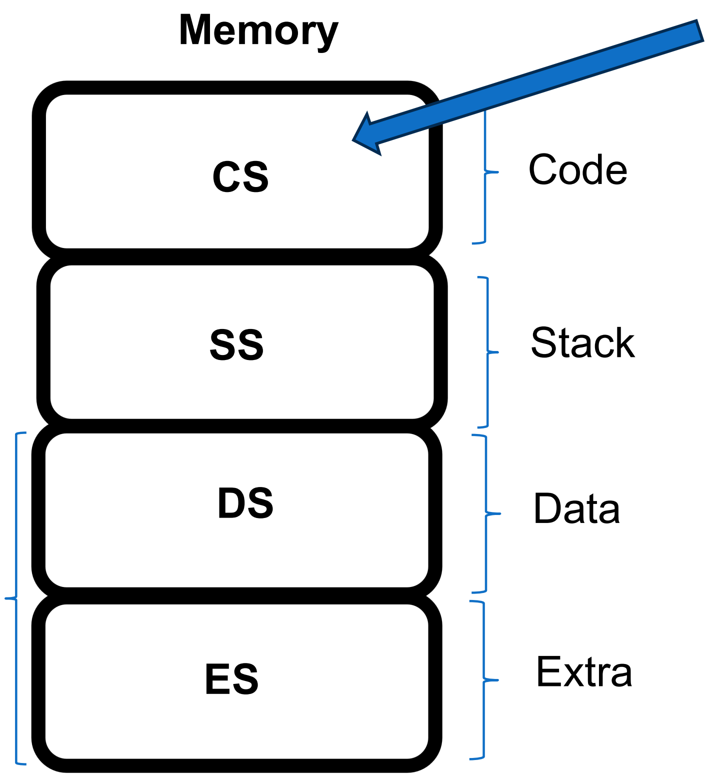


Memory segmentation

Contains all the program

Contains all the entire stack

Contains random data

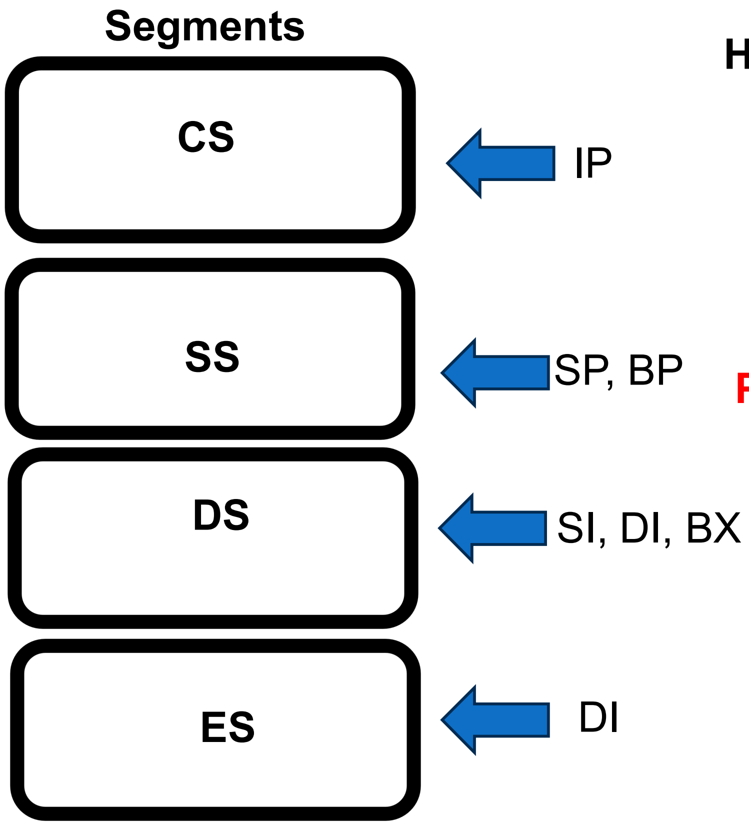


Any locations in the memory we want to access has its own unique address:

1- a segment address

2- an offset address

Memory segmentation



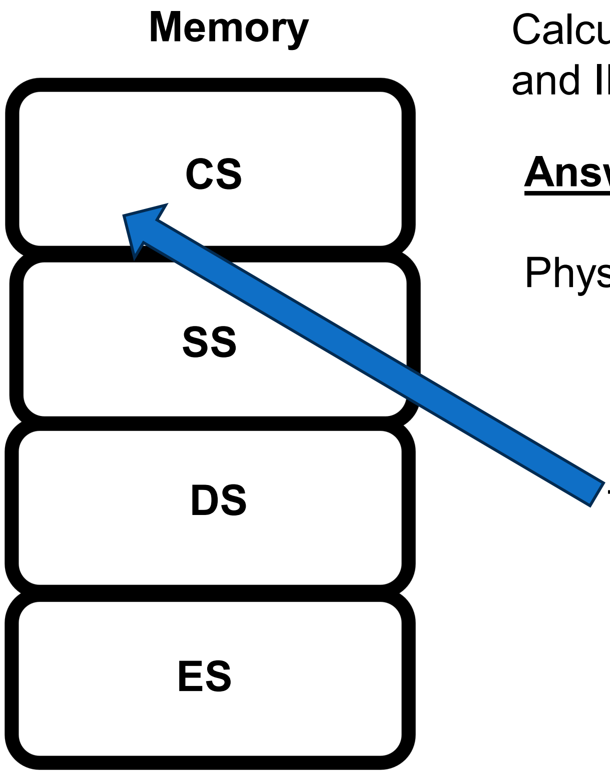
How do we calculate the physical address ?

Based on segment address and offset address :

Physical address = (Segment address*10H)+Offset address

The size of physical address is 20 bits

Memory segmentation



Calculate the physical address of a location whose CS is 1000H and IP is 3008H ?

Answer:

Physical address = (Segment address*10H)+Offset address
= (1000H*10H)+3008H
=13008H

Therefore, 20 bits address of the location is **13008H**

Register organization:

- 8086 has a powerful set of 14 registers known as general purpose registers and special purpose registers.

- All of them are **16-bit** registers.

- **General purpose registers**:
    o These registers can be used as either 8-bit registers or 16-bit registers.
    o They may be either used for holding data, variables and intermediate results temporarily or for other purposes like a counter or for storing offset address for some particular addressing modes etc.

- **Special purpose registers:**

These registers are used as segment registers, pointers, index registers or as offset storage registers for particular addressing modes

Register organization:

General Data Registers:

- The registers AX, BX, CX and DX are the general purpose 16-bit **registers** and they are used in all arithmetic and logical operation.
- AX is used as 16-bit accumulator. The lower 8-bit is designated as AL and higher 8-bit is designated as AH. AL can be used as an 8-bit accumulator for 8-bit operation.
- All data register can be used as either 16 bit or 8 bit. BX is a 16 bit register, but BL indicates the lower 8-bit of BX and BH indicates the higher 8-bit of BX
- The register BX is used as offset storage for forming physical address in case of certain addressing modes.
- The register CX is used default counter in case of string and loop instructions.
- DX register is a general purpose register which may be used as an implicit operand or destination in case of a few instructions.

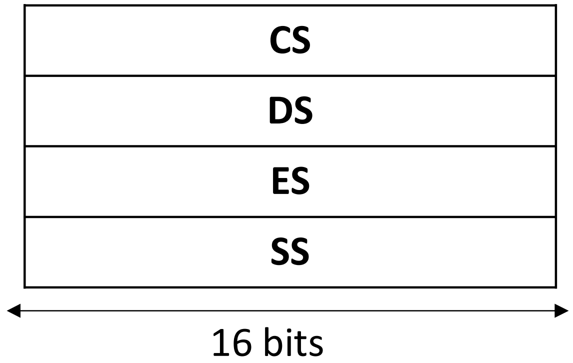
8 bits	8 bits	
AH	**AL**	AX
BH	**BL**	BX
CH	**CL**	CX
DH	**DL**	DX

16 bits

Register organization:

Segment Registers:

- There are 4 segment registers:
  - o Code Segment Register(CS)
  - o Data Segment Register(DS)
  - o Extra Segment Register(ES)
  - o Stack Segment Register(SS)

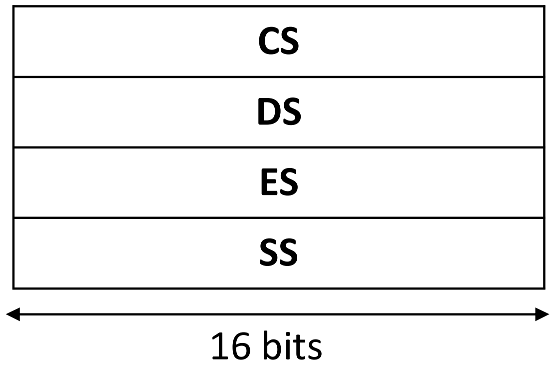


The 8086 architecture uses the concept of segmented memory. 8086 able to address a memory capacity of 1 megabyte and it is byte organized..

Register organization:

Segment Registers:

- Code segment register (CS): is used for addressing memory location in the code segment of the memory, where the executable program is stored.
- Data segment register (DS): points to the data segment of the memory where the data is stored.
- Extra Segment Register (ES) : also refers to a segment in the memory which is another data segment in the memory.
- Stack Segment Register (SS): is used for addressing stack segment of the memory. The stack segment is that segment of memory which is used to store stack data



Register organization:

Pointer and Index Registers:

These 16 bit (addressing) registers can be used to address an operand within a segment of 64KB (2^16)

SP
BP
SI
DI

16 bits

SP : Stack pointer

- Used to access the stack
- Points to the top of stack
- By default, contains offset within the stack segment. **SS**

BP : Base pointer

- Contains offset within the data segment **SS**
- Can be used as general purpose

Register organization:

SI : Source Index

- General purpose
- Used to store the offset of source data in data segment **DS**.
- Addressing as an index register
- Used by some movement instruction as index of Source operand

SP
BP
SI
DI

16 bits

DI : Destination Index

- Addressing as an index register
- Used to store the offset of destination in data or extra segment **ES**.
- Used by some movement instruction as index of Destination operand

Register organization:

Instruction pointer and flag registers

IP
FLAGS

IP : Instruction pointer:

- store memory location of next instruction to be executed

Flags

*	*	*	*	O	D	I	T	S	Z	*	A	*	P	*	C

- **C: Carry**

Set 1: unsigned arithmetic result is out of range

- **P: Parity**

Indicates that the number of 1 is an even number

Register organization:

*	*	*	*	O	D	I	T	S	Z	*	A	*	P	*	C

- **A: Arithmetic retained/Auxilary carry**

Set to 1 when there is a carry from bit 3 to bit 4
When the sum of two low parts digits exceeds 'F'

- **Z: Zero**

Indicates that the result of arithmetic or logic operation is zero

- **S: Sign**

Set to 1 when the result is negative

- **T: Trap**

Put the CPU in step mode which is used for debugging

Register organization:

*	*	*	*	O	D	I	T	S	Z	*	A	*	P	*	C

- **I: Interrupt**

I=0, interruption is enabled
I=1, interruption is disabled

- **D: Direction**

Sets the direction of self increment/decrement in sring manipulation
D=0, increment
D=1, decrement

- **O: Overflow**

Indicates the overflow when working with signed numbers

Assembly language:

- An assembly language is a type of low-level programming language that is intended to communicate directly with a computer's hardware.
- Each family of processors has its own set of instructions for handling various operations such as getting input from keyboard, displaying information on screen and performing various other jobs. These set of instructions are called 'machine language instructions'.
- Two different approaches :

- Reduced Instruction Set Computers (RISC), is a type of computer architecture that uses a small but highly optimised set of instructions.

- Complex Instruction Set Computers (CISC) is a type of computer architecture that can operate with single instructions which execute several low-level operations (e.g. load, arithmetic, store).

Set of instructions:

The 8086 microprocessor supports 8 types of instructions :

1.Data Transfer Instructions.(MOV, PUSH, POP,…)

2.Arithmetic Instructions.(ADD, SUB, DIV,INC, …)

3.Bit Manipulation Instructions.(NOT, AND, OR,…)

4.String Instructions (REP)

5.Program Execution Transfer Instructions (RET, JMP,)

6.Processor Control Instructions.

7.Iteration Control Instructions.

8.Interrupt Instructions.

Assembly language:

Syntax of an instruction

Label	OP-CODE	{OPERANDS}	;Comments
Lab:	MOV	AX, 500H	;Put 500H in AX

Destination Source

- ✓ OP-CODE : identifies the operation (eg, MOV ADD SUB,JMP,…)
- ✓ OPERANDS : Specify the data required by the operation
- ✓ Comments : explain the program’s purpose, begin with semicolon
- ✓ Lablel : marks the address of an instruction

Assembly language:

Example:

```
s=0;
For (i=1;i<=10;i++)
    s=s+i
```

```
       MOV AX, 0
       MOV CX, 1
Label: ADD AX, CX
       INC CX
       CMP CX, 10
       JNE label
```