# Architecture of Microprocessor – 8086

As 8086 does 2-stage pipelining, its architecture is divided into two units:
1. Bus Interface Unit (BIU)
2. Execution Unit (EU)

## BUS INTERFACE UNIT (BIU)

1. It provides the **interface** of 8086 **to** other devices.
2. It **operates w.r.t. Bus cycles .**
   This means it performs various machine cycles such as Mem Read, IO Write etc to transfer data with Memory and I/O devices.
3. It performs the following functions:
   a) It **generates** the 20-bit **physical address** for memory access.
   b) **Fetches Instruction** from memory.
   c) **Transfers data** to and from the **memory and IO.**
   d) **Supports Pipelining** using the 6-byte instruction queue.

**The main components of the BIU are as follows:**

### a) SEGMENT REGISTERS:

#### 1) CS Register
CS holds the **base** (Segment) **address** for the **Code Segment.**
All programs are stored in the Code Segment.
It is **multiplied by 10H** ($16_d$), to give the **20-bit physical address** of the **Code Segment.**
Eg: If **CS = 4321H** then CS x 10H = **43210H → Starting address** of Code Segment.
CS register cannot be modified by executing any instruction except branch instructions

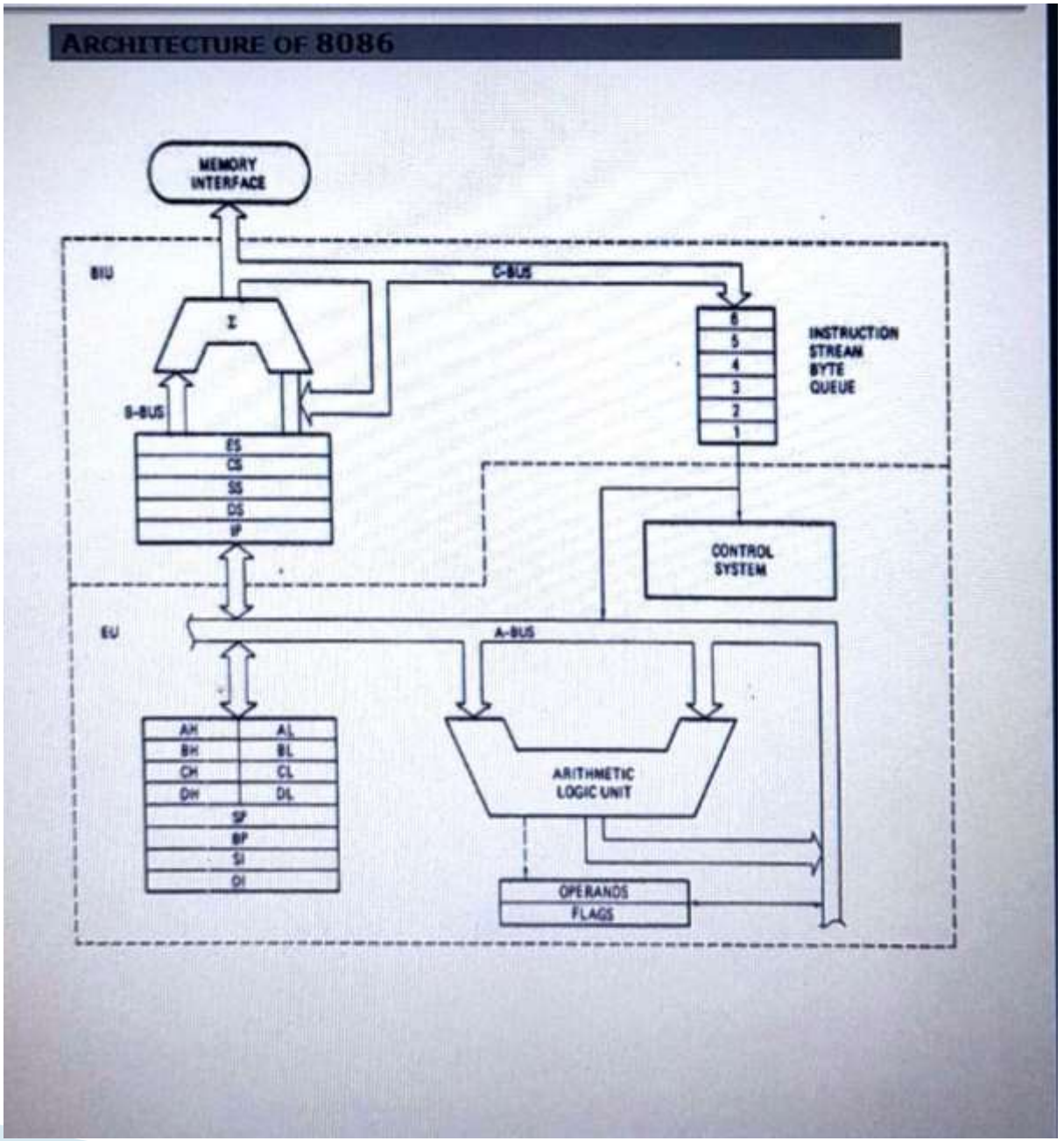
#### 2) DS Register
DS holds the **base** (Segment) **address** for the **Data Segment.**
It is **multiplied by 10H** ($16_d$), to give the **20-bit physical address** of the **Data Segment.**
Eg: If **DS = 4321H** then DS x 10H = **43210H → Starting address** of Data Segment.

#### 3) SS Register
SS holds the **base** (Segment) **address** for the **Stack Segment.**
It is **multiplied by 10H** ($16_d$), to give the **20-bit physical address** of the **Stack Segment.**
Eg: If **SS = 4321H** then SS x 10H = **43210H → Starting address** of Stack Segment.

#### 4) ES Register
ES holds the **base** (Segment) **address** for the **Extra Segment.**
It is **multiplied by 10H** ($16_d$), to give the **20-bit physical address** of the **Extra Segment.**
Eg: If **ES = 4321H** then ES x 10H = **43210H → Starting address** of Extra Segment.

### b) Instruction Pointer (IP register)
It is a **16-bit register.**
It **holds offset of** the **next instruction** in the **Code Segment.**

# ARCHITECTURE OF 8086



## Execution Unit (EU)

1. It **fetches** instructions **from** the **Queue in BIU, decodes** and **executes them.**
2. It performs **arithmetic, logic** and **internal data transfer** operations.
3. It sends request signals to the BIU to access the external module.
4. It **operates w.r.t. T-States** (clock cycles).

**The main components of the EU are as follows:**

### a) General Purpose Registers

8086 has four 16-bit general-purpose registers **AX, BX, CX** and **DX.** These are **available** to the programmer, for storing values during programs. Each of these can be **divided** into two **8-bit registers** such as AH, AL; BH, BL; etc. Beside their general use, these registers also have some **specific functions.**

***AX** Register (16-Bits)*

It holds operands and results during **multiplication** and **division** operations.
**All IO data transfers** using IN and OUT instructions use A reg (AL/AH or AX).
It functions as accumulator during **string operations.**

***BX** Register (16-Bits)*

**Holds** the **memory address** (offset address), in **Indirect Addressing modes.**

***CX** Register (16-Bits)*

Holds **count** for instructions like: **Loop, Rotate, Shift** and **String** Operations.

***DX** Register (16-Bits)*

It is used with AX to hold **32 bit** values during **Multiplication** and **Division.**
It is used to **hold** the **address** of the **IO Port** in **indirect IO addressing** mode.

### b) Special Purpose Registers

*Stack Pointer (**SP** 16-Bits)*

It is holds **offset address of** the **top of the Stack. Stack is a set of memory locations operating in LIFO manner. Stack is present in the memory in Stack Segment.**
SP is used with the SS Reg to calculate physical address for the Stack Segment. It used during instructions like PUSH, POP, CALL, RET etc. During PUSH instruction, SP is decremented by 2 and during POP it is incremented by 2.

*Base Pointer (**BP** 16-Bits)*

BP can hold **offset address of** any location in the **stack segment.**
It is used to access random locations of the stack.

*Source Index (**SI** 16-Bits)*

It is normally used to hold the **offset address** for **Data segment** but can also be used for other segments using Segment Overriding. It holds **offset address** of **source data** in Data Seg, during **String Operations.**

*Destination Index (**DI** 16-Bits)*

It is normally used to hold the **offset address** for **Extra segment** but can also be used for other segments using Segment Overriding. It holds **offset address** of **destination** in Extra Seg, during **String Operations.**

**c) ALU** (16-Bits)

It has a **16-bit ALU**. It performs 8 and 16-bit arithmetic and logic operations.

**d) Operand Register**

It is a 16-bit register used by the control register to hold the operands temporarily.
It is **not available** to the Programmer.

**e) Instruction Register and Instruction Decoder** (Present inside the Control Unit)

The **EU fetches an opcode from** the **queue into** the **Instruction Register.** The **Instruction Decoder decodes** it and sends the information to the control circuit for execution.

**f) Flag Register (16-Bits)**

It has **9 Flags.**
These flags are of two types: **6-Status** (Condition) Flags and **3-Control** Flags.
**Status flags** are affected by the ALU, after every arithmetic or logic operation. They give the **status of the current result.**
The **Control flags** are used to control certain operations.
They are changed by the programmer.