8086 Microprocessor

Introduction

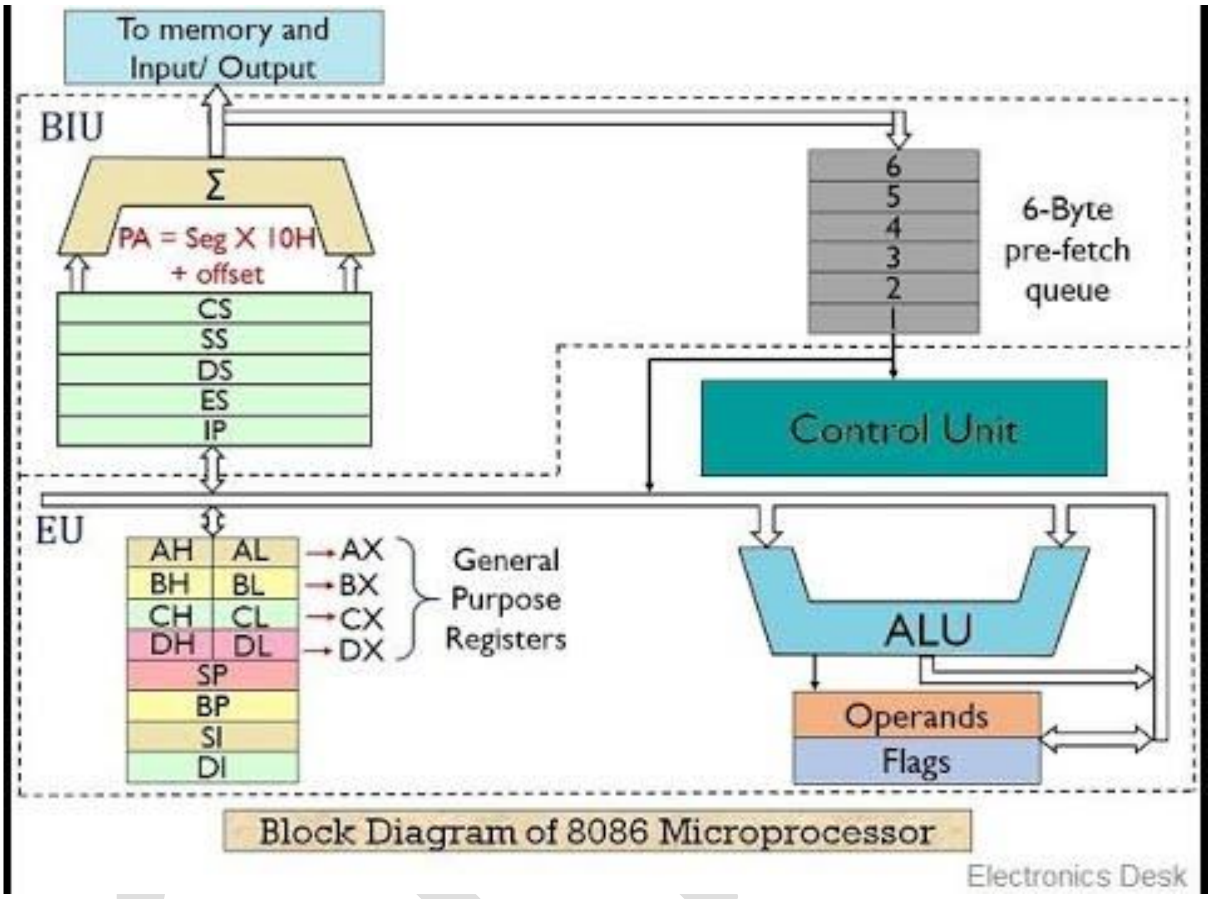
- 8086 Microprocessor is an enhanced version of 8085Microprocessor
- It was designed by Intel in 1976.
- It is a 16-bit Microprocessor having 20 address lines and16 data lines that provides up to 1MB storage.
- Intel 8086 is built on a single semiconductor chip and packaged in a 40-pin IC package.
- It consists of a powerful instruction set, which provides operation like division and multiplication very quickly

8086 is designed to operate in two modes, i.e., Minimum and Maximum mode

Difference between 8085 and 8086 Microprocessor

8085 Microprocessor	**8086 Microprocessor**
It is an 8-bit microprocessor.	It is a 16-bit microprocessor.
It has a 16-bit address line.	It has a 20-bit address line.
It has a 8-bit data bus.	It has a 16-bit data bus.
The memory capacity is 64 KB.	The memory capacity is 1 MB.
The Clock speed of this microprocessor is 3 MHz.	The Clock speed of this microprocessor varies between 5, 8 and 10 MHz for different versions.
It has five flags.	It has nine flags.
8085 microprocessor does not support memory segmentation.	8086 microprocessor supports memory segmentation.
It does not support pipelining.	It supports pipelining.
It is accumulator based processor.	It is general purpose register based processor.
It has no minimum or maximum mode.	It has minimum and maximum modes.
In 8085, only one processor is used.	In 8086, more than one processor is used. An additional external processor can also be employed.
It contains less number of transistors compare to 8086 microprocessor. It contains about 6500 transistor.	It contains more number of transistors compare to 8085 microprocessor. It contains about 29000 in size.
The cost of 8085 is low.	The cost of 8086 is high.

Architecture of 8086



8086 contains two independent functional units: **a Bus Interface Unit (BIU)** and an **Execution Unit (EU)**.

Segmentation and physical address calculation

- The memory in an 8086 based system is organized as segmented memory.
- The 8086 is able to access 1 MB of physical memory.
- Memory is divided into 4 segments.

Segment address- starting address of segment.

Offset address- a location in that segment.

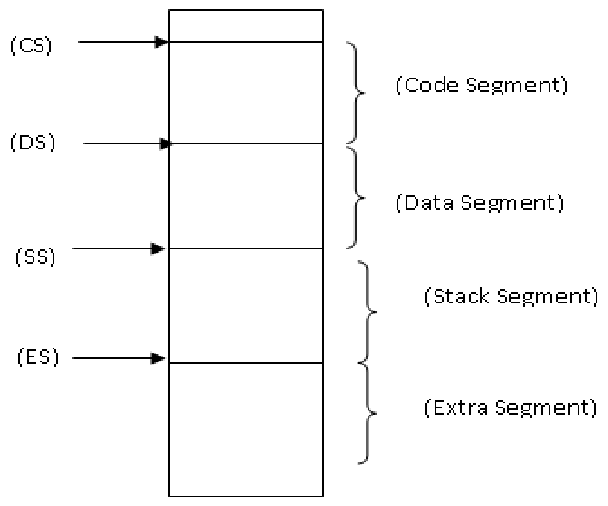


Figure: Memory Segmentation (Code, Data, Stack and Extra Segments)

- To address a specific memory location within a segment, we need an offset address.
- Physical address is calculated as below:
- **Physical address = Segment address * 10H + Offset address.**
- Example
- Segment address – 1000
- Offset address -- 2345H
- Physical address = 1000*10+2345

  =12345H

Bus Interface Unit (BIU)

The segment registers, instruction pointer and 6-byte instruction queue are associated with the bus interface unit (BIU).

The BIU:

- Handles transfer of data and addresses,
- Fetches instruction codes, stores fetched instruction codes in first-in-first-out register set called a **queue**,
- Reads data from memory and I/O devices,
- Writes data to memory and I/O devices,
- It relocates addresses of operands since it gets un-relocated operand addresses from EU. The EU tells the BIU from where to fetch instructions or where to read data

It has the following functional parts:

- **Instruction Queue:** When EU executes instructions, the BIU gets 6-bytes of the next instruction and stores them in the instruction queue and this process is known as instruction pre fetch. This process increases the speed of the processor.
- **Segment Registers:** A segment register contains the addresses of instructions and data in memory which are used by the processor to access memory locations. It points to the starting address of a memory segment currently being used.
  There are 4 segment registers in 8086 as given below:
  - **Code Segment Register (CS):** Code segment of the memory holds instruction codes of a program.
  - **Data Segment Register (DS):** The data, variables and constants given in the program are held in the data segment of the memory.
  - **Stack Segment Register (SS):** Stack segment holds addresses and data of subroutines. It also holds the contents of registers and memory locations given in PUSH instruction.
  - **Extra Segment Register (ES):** Extra segment holds the destination addresses of some data of certain string instructions.
- **Instruction Pointer (IP):** The instruction pointer in the 8086 microprocessor acts as a program counter. It indicates to the address of the next instruction to be executed.

Execution Unit (EU)

- The **EU** receives opcode of an instruction from the queue, decodes it and then executes it. While Execution, unit decodes or executes an instruction, then the BIU fetches instruction codes from the memory and stores them in the queue.
- The BIU and EU operate in parallel independently. This makes processing faster.
- General purpose registers, stack pointer, base pointer and index registers, ALU, flag registers (FLAGS), instruction decoder and timing and control unit constitute execution unit (EU). Let's discuss them:
- **General Purpose Registers:** There are four 16-bit general purpose registers: AX (Accumulator Register), BX (Base Register), CX (Counter) and DX. Each of these 16-bit registers are further subdivided into 8-bit registers as shown below:

16-bit registers	**8-bit high-order registers**	**8-bit low-order registers**
AX	AH	AL
BX	BH	BL
CX	CH	CL
DX	DH	DL

- **Index Register:** The following four registers are in the group of pointer and index registers:
    - Stack Pointer (SP) , Base Pointer (BP) ,Source Index (SI) , Destination Index (DI)
- **ALU:** It handles all arithmetic and logical operations. Such as addition, subtraction, multiplication, division, AND, OR, NOT operations.
- **Flag Register:** It is a 16?bit register which exactly behaves like a flip-flop, means it changes states according to the result stored in the accumulator. It has 9 flags and they are divided into 2 groups i.e. conditional and control flags.
    - **Conditional Flags:** This flag represents the result of the last arithmetic or logical instruction executed. Conditional flags are:
        - Carry Flag ,Auxiliary Flag, Parity Flag, Zero Flag, ,Overflow Flag
    - **Control Flags:** It controls the operations of the execution unit. Control flags are:

Trap Flag, Interrupt Flag, Direction Flag

PIPELINING

- While the EU is decoding an instruction or executing an instruction, which does not require use of the buses, the BIU fetches up to six instruction bytes for the following instructions.
- The BIU stores these pre-fetched bytes in a first-in-first-out register set called a queue.
- Fetching the next instruction while the current instruction executes is called pipelining.

Flags in 8086

Conditional Flags: This flag represents the result of the last arithmetic or logical instruction executed. Conditional flags are :

- **Carry Flag**
- **Auxiliary Flag**
- **Parity Flag**
- **Zero Flag**
- **Sign Flag**
- **Overflow Flag**

SF – Sign Flag : This flag is set, when the result of any computation is negative.

ZF-Zero Flag: This flag is set, if the result of the computation or comparison performed by the previous instruction is zero.

PF-Parity Flag : This flag is set to 1, if the result contains even number of 1's.

CF- Carry Flag : This flag is set, when there is a carry out of MSB in case of addition or a borrow in case of subtraction.

AF-Auxialary Carry Flag : This is set, if there is a carry from the lowest nibble, i.e, , bit three during addition, or borrow for the lowest nibble, i.e, bit three, during subtraction .

OF- Over flow Flag : This flag is set, if an overflow occurs, i.e, if the result of a signed operation is large enough to accommodate in a destination register.

Control Flags: It controls the operations of the execution unit.

Control flags are

- **Trap Flag**
- **Interrupt Flag**
- **Direction Flag**
- **TF-Trap Flag :** If this flag is set, the processor enters the single step execution mode.
- **IF- Interrupt Flag :** If this flag is set, the maskable interrupts are recognized by the CPU, otherwise they are ignored.
- **D-Direction Flag :** This is used by string manipulation instructions. If this flag bit is '0', the string is processed beginning from the lowest address to the highest address, i.e, auto incrementing mode. Otherwise, the string is processed from the highest address towards the lowest address,i.e, auto decrementing mode.

Addressing modes of 8086

- Addressing mode indicates a way of locating data or operands.
- The addressing modes describe the type of operands and the way they are accessed for executing an instruction.

1.Immediate: In this type of addressing , immediate data is a part of instruction and appears in the form of successive byte or bytes.

Ex. MOV AX,0005H

In the above example ,0005H is the immediate data.

The immediate data may be 8-bit or 16-bit in size.

2.Direct addressing: In the direct addressing mode a 16-bit memory address (offset) is directly specified in the instruction as a part of it.

EX. **MOV AX, [5000H]**

3.Register Addressing: In register addressing mode, the data is stored in a register and is referred using the particular register.

All the registers, except IP , may be used in this mode.

Ex: **MOV BX,AX**

4. Register Indirect Addressing mode: Sometimes, the address of the memory location, which contains data or operand is determined in an indirect way, using the offset register. This mode of addressing is known as register indirect mode.

In this addressing mode, the offset address of data is in either BX or SI or DI register.

EX: **MOV AX,[BX]**

5.Indexed Addressing mode: In this addressing mode, offset of the operand is stored in one of the index registers.

EX: **MOV AX [SI]**

6.Register Relative : In this addressing mode,the data is available at an effective address formed by adding an 8-bit or 16-bit displacement with the content of any one of the registers BX,BP,SI and DI in the default (either DS or ES) segment.

EX: **MOV AX,50H[BX]**

7.Based Indexed : The effective address of data is formed, in this addressing mode, by adding content of a base register (any one of BX or BP) to the content of an index register (any one of SI or DI)

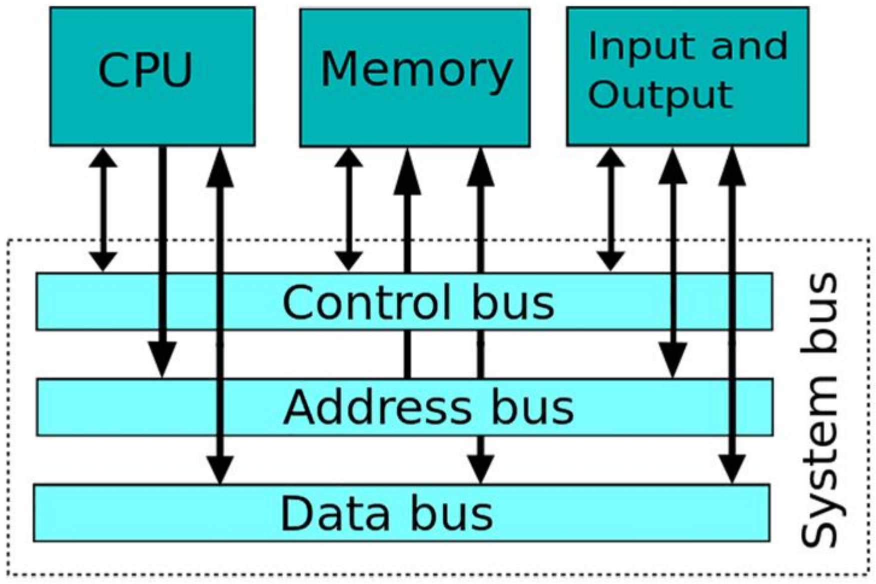
EX: **MOV AX,[BX][SI]**

8.Relative Based Indexed: The effective address is formed by adding an 8-bit or 16-bit displacement with the sum of the contents of any one of the base register (BX or BP) and any one of the index register .

EX: **MOV AX, 50H[BX][SI]**

Here , 50H is an immediate displacement , BX is base register and SI is an index register the effective address of data is computed as 10H*DS+[BX]+[SI]+50H

BUSES IN 8086



- Central processing unit or CPU controls the operation of the computer.
- Input/output or I/O section allows the computer to take in data from the outside world or send data to the outside world.
- Purpose of memory is storage.

ADDRESS BUS: On these lines the CPU sends out the address of the memory location that is to be written to or read from.

- The number of address lines determines the number of memory locations that the CPU can address.
- If the CPU has n address lines then it can directly address 2^n memory locations.

DATA BUS: CPU can read data from memory or from a port as well as send data out on these lines to memory location or to a port.

CONTROL BUS: Control bus signals are memory read, memory write, I/O read and I/O write.

- To read a byte of data from a memory location, for example, the CPU sends out the address of the desired byte on the address bus and then sends out a memory read signal on the control bus.