2)CPU:Bus Interface Unit and Execution Unit

The internal function of 8086 processor are partitioned logically into processing units ,Bus Interface Unit(BIU) and Execution Unit (EU).general block diagram of 8086 processor is shown in figure (4).

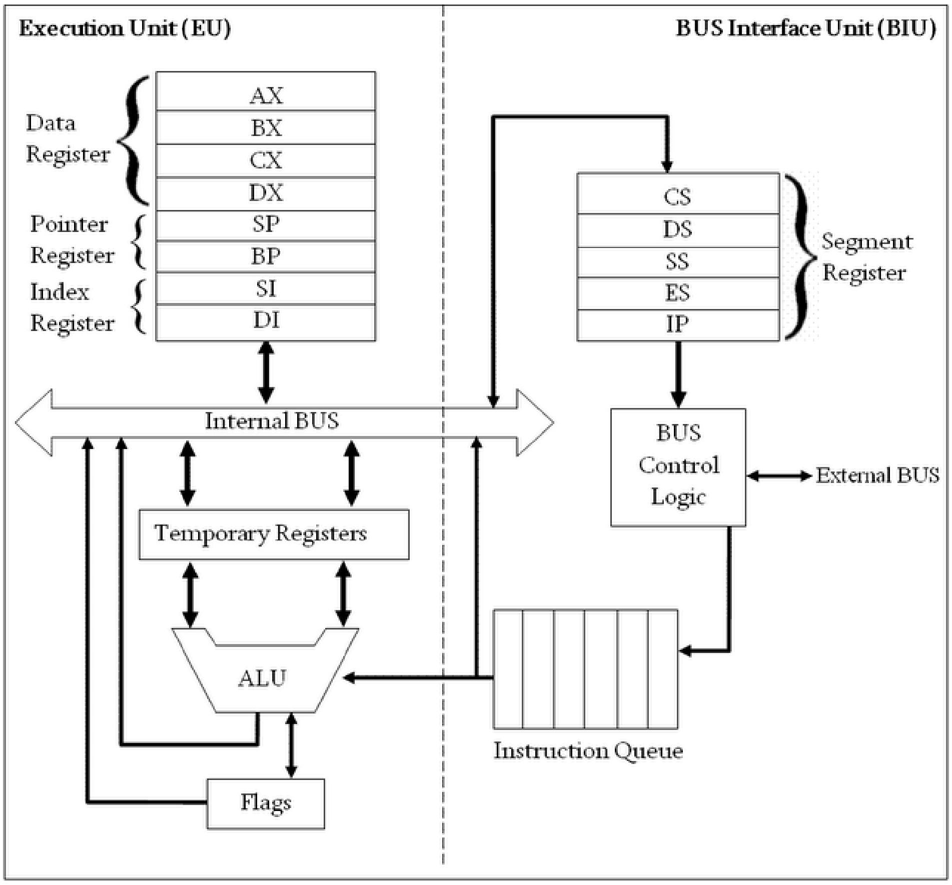

Figure (4): Bus Interface Unit(BIU) and Execution Unit (EU).

Execution Unit (EU) : Execution unit receives program instruction codes and data from the BIU, executes them and stores The results in the general registers. It can also store the data in a memory location or send them to an I/O device by passing the data back to the BIU. This unit, EU, has no connection with the system Buses. It receives and outputs all its data through BIU.

Bus Interface Unit : As the EU has no connection with the system Busses, this job is done by BIU. BIU and EU are connected with an internal bus. BIU connects EU with the memory or I/O circuits. It is responsible for transmitting data, addresses and control signal on the busses.

EU is used mainly to execute instructions. It contains a circuit called the arithmetic and logic unit (ALU). The data for operations are stored in circuit called Registers. The EU has eight registers for storing data; their names are AX, BX, CX, DX, SI, DI, BP, SP and FLAGS register. The EU accepts instructions and data that have been fetched by the BIU and then processes the information. Data processed by the EU can be transmitted to the memory or peripheral devices through the BIU. EU has no direct connection with the outside world and relies solely on the BIU to feed it with instruction and data. It is here that instructions are received, decoded, and executed from the instruction queue portion of BIU. The instructions are taken from the top of the instruction queue on the first-in, first-out, or FIFO, basis.

ALU (Arithmetic & Logic Unit) : This unit can perform various arithmetic and logical operation, if required, based on the instruction to be executed. It can perform arithmetical operations, such as add, subtract, increment, decrement, convert byte/word and compare etc and logical operations, such as AND, OR, exclusive OR, shift/rotate and test etc.

- **Index Registers**

1. **SP (Stack Pointer):** This is stack pointer register pointing to program stack. It is used in conjunction with SS for accessing the stack segment.
2. **BP (Base Pointer):** This is base pointer register pointing to data in stack segment. Unlike SP, we can use BP to access data in the other segments.
3. **SI (Source Index):** This is source index register which is used to point to memory locations in the data segment addressed by DS. By incrementing the contents of SI one can easily access consecutive memory locations.
4. **DI (Destination Index):** This is destination index register performs the same function as SI. There is a class of instructions called string operations.

- **Segment Registers :** BIU has 4 segment busses, CS, DS, SS, ES. These all 4 segment registers holds the addresses of instructions and data in memory. These values are used by the processor to access memory locations. It also contains 1 pointer register IP. IP contains the address of the next instruction to execute by the EU.

1- CS (Code Segment): The code segment register holds the base location of all executable instructions (code) in a program.

2- DS (Data Segment): the data segment register is the default base location for variables. The CPU calculates their location using the segment value in DS.

3- SS (Stack Segment): the stack segment register contain the base location of the stack.

4- ES (Extra Segment): The extra segment register is an additional base location for memory variables.

……………………..

XCHG instruction

```
XCHG    R., R.

XCHG    R., [add.]

XCHG    [add.], R.
```

EX: write an 8086 assembly language program to exchange two byte number stored in memory location started at 3000 with two byte number stored in memory location started at 4000 .

Answer:

```
Mov   AX, [3000]

XCHG  AX, [4000]

Mov    [3000],  AX
```

Increment and Decrement

INC : to increment (add) 1 value

DEC : to decrement(sub) 1 value

```
EX:   MOV  AX, 45D1H

      INC AX ;  AX = 45D0H

      MOV DH, 8FH

      DEC DH ; DH= 90H
```

Q) Execute this four steps using XCHG, INC ,and DEC instruction.

1. Load 12cdh into BX

2.copy 16-bit from BX to AX

3. Load 10110111b to address memory contain in the BX reg.

4. Replace the data between AX and [12C00H]