Introduction to Microprocessors

Definition: Microprocessor is the controlling unit or CPU of a micro-computer, fabricated on a very small chip capable of performing ALU operations and communicating with the external devices connected to it.

Arithmetic Logic Unit

- The microprocessor is one of the most important components of a digital computer.
- It acts as the brain of the computer system.
- As technology has progressed, microprocessors have become faster, smaller and capable of doing more work per clock cycle.
- Sometimes, microprocessor is written as µP.

Computer Architecture

Computer Architecture:

A computer has main three parts. They are – i) I/O devices, ii) Memory, iii) CPU (Central Processing Unit).

I/O devices are the intermediate media between the computer and the user. User can give input to the computer through the input device (e.g. keyboard, mouse etc). Computer shows the output of the operation which was asked by user, on the output devices (e.g. monitor, printer etc).

Memory is the place where computer stores its information and data. Memory is comprised nothing but registers. The memory, in other words registers are usually 4, 8, 16 bits long. That is, memory stores information and data of fixed length.

Von-Neuman architecture

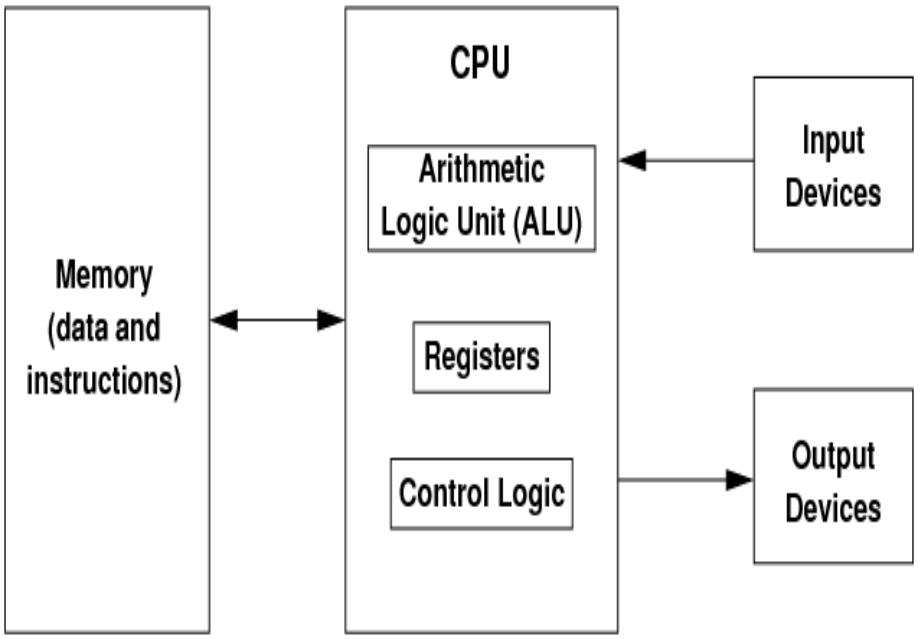


Fig: Main components of a computer

CPU (CENTRAL PROCESSING UNIT):

CPU is the brain of the computer. It executes the instructions which are either given by a user or a system program. In any case, each instruction that the CPU executes is a bit string (i.e. combination of 0's and 1'). The language of 0's and 1's is called the machine language.

A CPU (or microprocessor) communicates with the memory and I/O devices by using signals that travel along a set of wires or connections called **buses.** Buses connect the different components of the computer shown in the figure. Physically buses are simply a group of electrical paths (or wires) which are connected to memory, CPU and I/O devices.

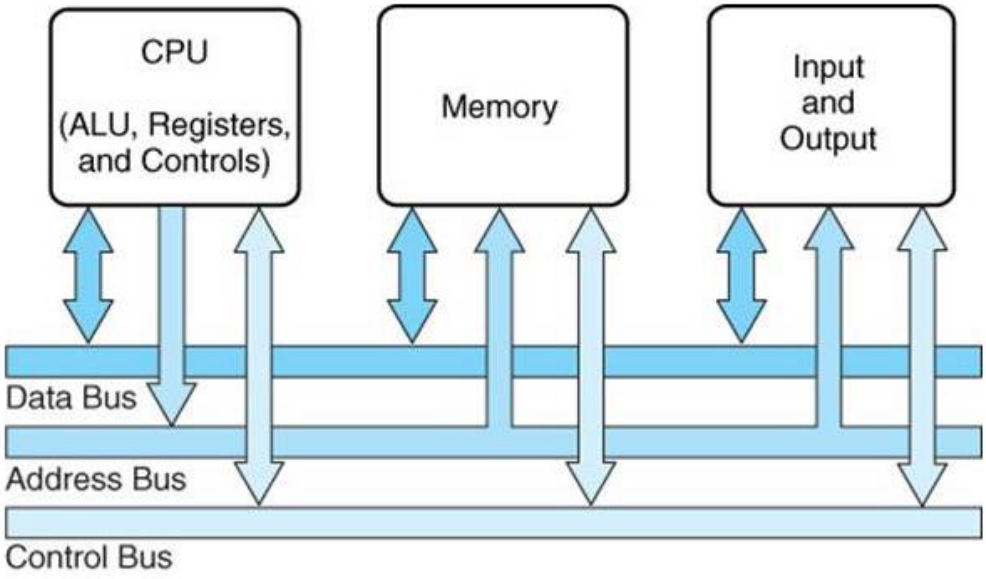
There are three types of buses. They are – i) Data Bus, ii) Address Bus & iii) Control Bus.

bidirectional unidirectional bidirectional

Role of buses - if user wants to know (read) the content of a specific memory location, then microprocessor will place the address of that memory location on the **address bus**. A control signal is sent to the memory by microprocessor to perform read operation on the **control bus**. Finally, microprocessor read the memory location and sends the data to the output device via

data bus.

Data bus is a bidirectional path i.e. information on the data bus can travel to and from microprocessor, memory and I/O devices. But address bus is unidirectional. Information travels only one direction i.e. from microprocessor to memory and microprocessor to I/O devices. The directions of buses are shown:



MEMORY:

Memory consists of registers. Computer has two types of memory – i) RAM (random access memory), ii) ROM (read only memory).

volatile

non-volatile

The difference between them can be understood from their name. RAM locations can be both read and written whereas ROM location can be read only. This is because the contents of ROM, once initialized, cannot be changed. In addition to it, the contents of RAM are lost after power is cut i.e. it has temporary memory. So anything valuable in RAM must be saved on a disk or printed out. But the contents (data and instructions) of ROM are retained even after the power is off. The instructions responsible for stating-up program and self-testing the computer when it is turned on, are written in ROM by the manufacture company. These programs are known as

in 2^n,
#address bus = n

firmware.

Significance of 1 MB memory – 1 MB = 1024×1024 B = 1048576 B. That is 1 MB memory has 1048576 locations where it can store information. Again, 2^{20} = 1048576 that is a 20 – bit number has different 1048576 values. From the above two points, we can say that 20 – bit address can be used to address 1 MB memory. 8086 microprocessor uses 20 – bit for address.

if a microprocessor can transfer 8 bit data with 64 kB = 64*1024 bit = 2^16 bit memory, then it is a 8 bit microprocessor with 16 bit address bus.

Historical Background of Intel Microprocessors

- **Intel 4004**

- Year of introduction 1971
- 4-bit microprocessor data bus 4 bit
- 4 KB main memory 2^10 * 2^2 = 2^12 address bus
- 45 instructions
- PMOS technology was first programmable device which was used in calculators

- **Intel 8008**

- Year of introduction 1972
- 8-bit version of 4004
- 16 KB main memory
- 48 instructions
- PMOS technology
- Slow

Intel 8080

Year of introduction 1973

- 8-bit microprocessor
- 64 KB main memory

- 2 microseconds clock cycle time
- 500,000 instructions/sec
- 10X faster than 8008
- NMOS technology
- Drawback was that it needed three power supplies.
- Small computers (Microcomputers) were designed in mid 1970's using 8080 as CPU.

Intel 8085

- Year of introduction 1975
- 8-bit microprocessor-upgraded version of 8080
- 64 KB main memory
- 1.3 microseconds clock cycle time
- 246 instructions
- Intel sold 100 million copies of this 8-bit microprocessor
- uses only one +5v power supply.

Intel 8086/8088

- Year of introduction 1978 for 8086 and 1979 for 8088
- 16-bit microprocessors
- Data bus width of 8086 is 16 bit and 8 bit for 8088
- 1 MB main memory address bus = 20 bit
- 400 nanoseconds clock cycle time
- 6 byte instruction cache for 8086 and 4 byte for 8088
- Other improvements included more registers and additional instructions

 In 1981 IBM decided to use 8088 in its personal computer

Architecture of 8086 Microprocessor

8086 Microprocessor:

We will learn about Intel 8086 microprocessor as an example of CPU.

8086 is a 16 – bit microprocessor. That is - all the registers of 8086 is 16 - bit long. It has 20 – bit address bus. So it can access 2^{20} different memory locations. 2^{20} is called 1 MB. In other words, 8086 has 1 MB memory. So the problem is – the address is too big (20 – bit) to fit into the 16 – bit long register. This problem has been solved by segmented the memory logically.

Memory segmentation:

Memory of 8086 is logically segmented into four segments – i) Code Segment, ii) Data Segment, iii) Stack Segment and iv) Extra Segment.

Each memory segment is a block of 2^{16} (or 64 KB) consecutive memory bytes. Each segment is identified by a **segment number** (stored in the segment registers which is discussed later) starting with 0000h. As the segment number is 16 – bit long, so the highest segment number can be FFFFh.

To access each memory location within the segment, an **offset** is given to them.

A 20 – bit physical address is written in the form Segment:Offset. This is known as **logical address**.

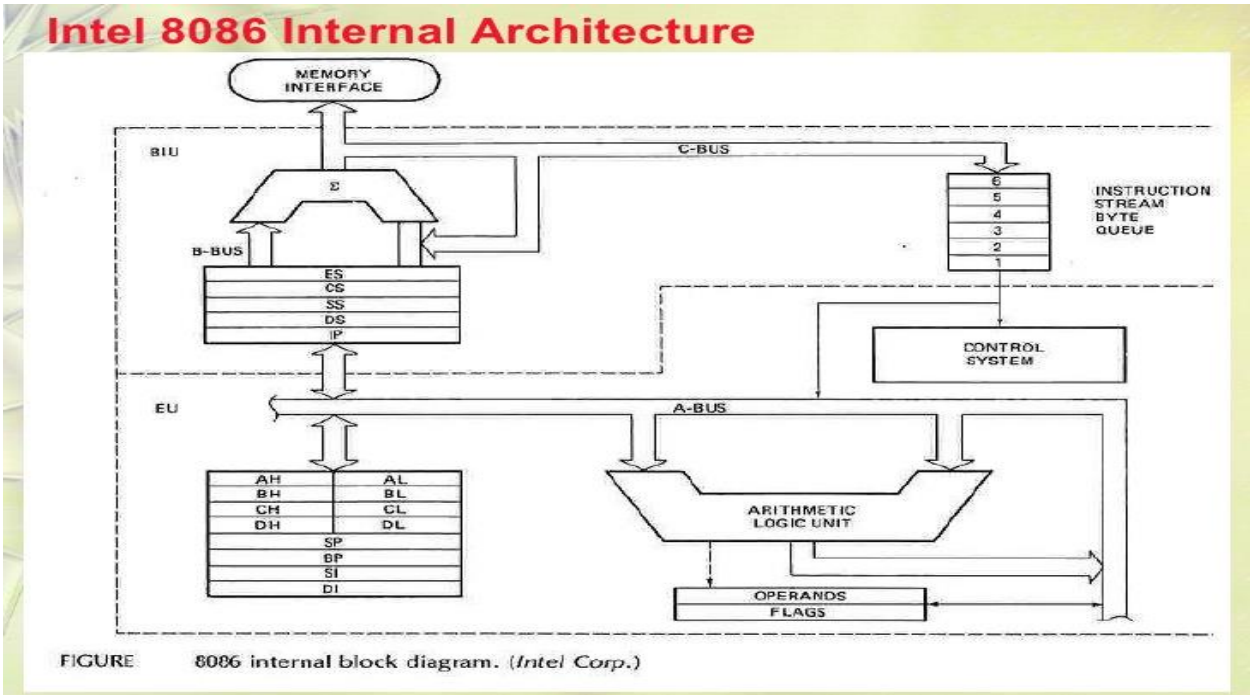
Logical address can be converted to physical address using the following formula:

✓ Physical Address = Segment × 10h + Offset.

Let the physical address of a memory location is 179B8h. If the start of the code segment is 1234h and the offset is 5678h, 179B8h can be presented by segment:offset.

Organization of 8086 Microprocessor:

8086 is organized as two separate blocks – i) BIU and ii) EU. The block diagram of 8086 is shown below:



EU (Execution Unit):

EU executes instructions from the instruction queue. It decodes instruction fetched by the BIU. Then it generates the control signal and executes instruction. It has a circuit called arithmetic logic unit (ALU).

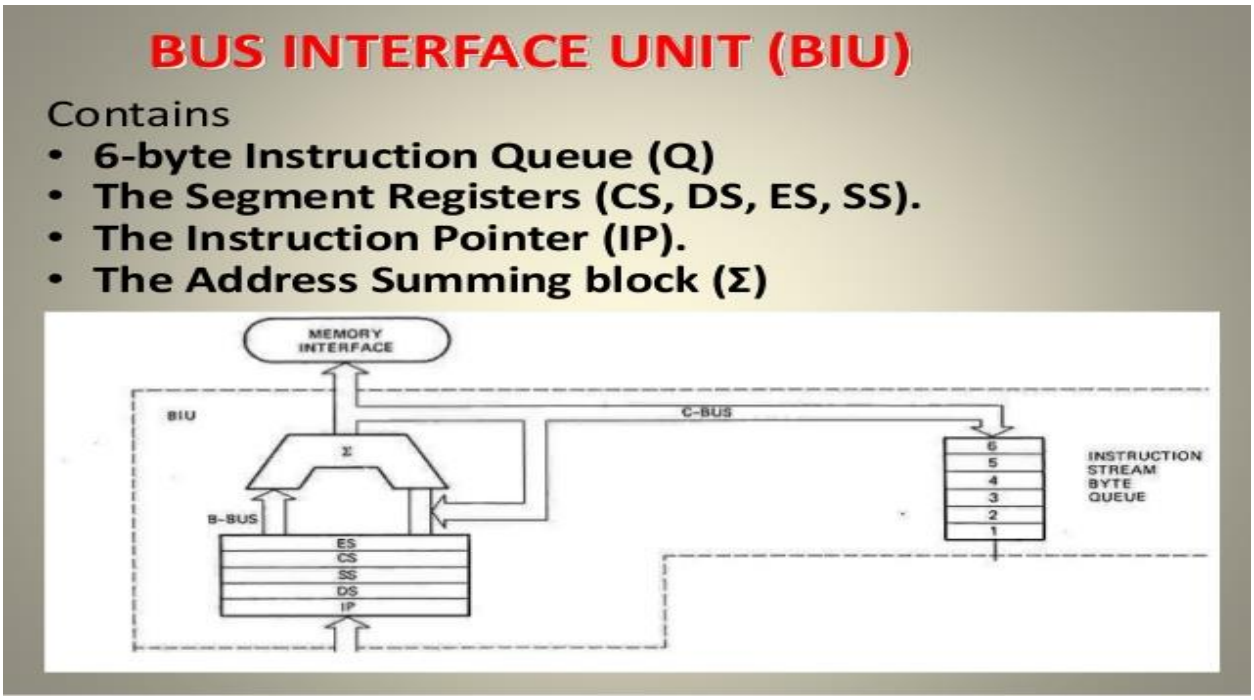
The ALU can perform arithmetic (+, -, ×, ÷) and logic (AND, OR, NOT) operations. The data for the arithmetic and logic operations are stored in registers. EU has eight registers for storing data. They are AX, BX, CX, DX, SI, DI, BP and SP. The high and low byte of the data registers can be accessed separately. The high byte of AX is called AH and the low byte is AL. Similarly, the high and low bytes of BX, CX and DX are BH and BL, CH and CL, DH and DL respectively. This arrangement provides more registers to use when the data are of byte – size.

- **AX (Accumulator Register):** It is used in all arithmetic and logical operation. In multiplication and division operations, one of the numbers involved must be in AX or AL.
- **BX (Base Register):** It is an address register. It points in a specific memory location in data segment.
- **CX (count register):** it is used as loop counter in program loop construction. In string operation, it also serves as counter. lower half CL is used a shift count.
- **DX (data register):** it is used in division and multiplication operation and also in I/O operation.
- SI, DI, SP and BP are the index and pointer registers. They contain the offset of the logical address of memory.
- **SI (Source Index):** it is used to point memory locations in the data segment.
- **DI (Destination Index):** it performs the same function as SI. It addresses in the extra segment.
- **SP (Stack Pointer):** it points to a specific memory location in stack segment.
- **BP (Base Pointer):** it is also used to access the stack segment. But it points to any data on the stack unlike SP. It can be used to access other segments also.

BIU (Bus Interface Unit):

It facilitates communication between the EU and the memory or I/O circuits. It handles all the data and addresses on the buses for EU. Bus operation includes instruction fetching, reading and writing operands for the memory and calculating the addresses of the memory location. BIU has four segment registers and one pointer register. They are – CS, DS, SS, ES and IP. Instruction queue is also a part of BIU. There is also a part in BIU called ‘address compute engine’ which

converts the logical address into the physical address (denoted by 'Σ' in the following figure).



Instruction queue is an internal memory holding instruction. In 8086 microprocessor, it contains six consecutive instructions. It is included in BIU to speed up microprocessor. EU is much faster than peripheral part i.e. memory (RAM/ROM). That is why, architect in advance, stores specific instructions in the instruction queue that would be executed in future, while EU is executing the current instruction. This technique is called '**pipelining**'. So pipelining is a technique wherein BIU pre-fetches instructions for EU.

Segment Register:

Segment registers are used to address memory location and I/O space. We have seen that the memory of 8086 is logically segmented into four parts – Code Segment, Data Segment, Stack Segment and Extra Segment. Segment registers hold the starting addresses of the four memory segments. To access each and individual location in memory, we need offset registers. Index and Pointer registers are used to hold offset.

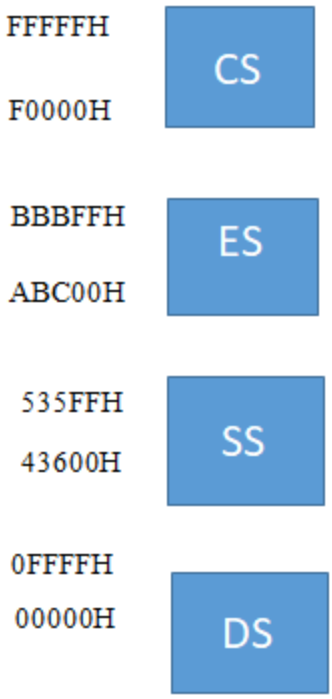
- **CS:** It contains the starting address of the code segment. This address plus the offset value contained in the IP indicates the address of an instruction to be fetched or executed.

- **DS:** It contains the starting address of the data segment. This address plus the offset value refers to a specific location in the data segment.
- **SS:** It contains the starting address of the stack segment. This address plus the offset value contained in the SP is used for stack operation.
- **ES:** It contains the starting address of the stack segment. It is used by some string operation. String instructions always use ES and DI registers to calculate the physical address for the destination.

Each segment register is 16 – bit long. That is, each can address 2^{16} = 65536 different memory location. 2^{16} is called 64 KB. So each segment is 64 KB long.

You are the programmer who will decide how to map the 1 MB into different segments. Here is an example:

Mapping of the 1 MB memory of 8086: (starting and ending address of the four segments are shown)



Different Registers holding the address of segments:

Segment and index & pointer registers are used in the following way to hold segment and offset respectively:

- **CS:IP** represents the physical address of code segment.
- **SS:SP & SS:BP** represent the physical address of stack segment.
- **DS:BX & DS:SI** represent the physical address of data segment.
- **DS:DI** is used for other than string operation.
- **ES:DI** is used for string operation to represent the physical address inside extra segment.