Architecture of 8086 Microprocessor:

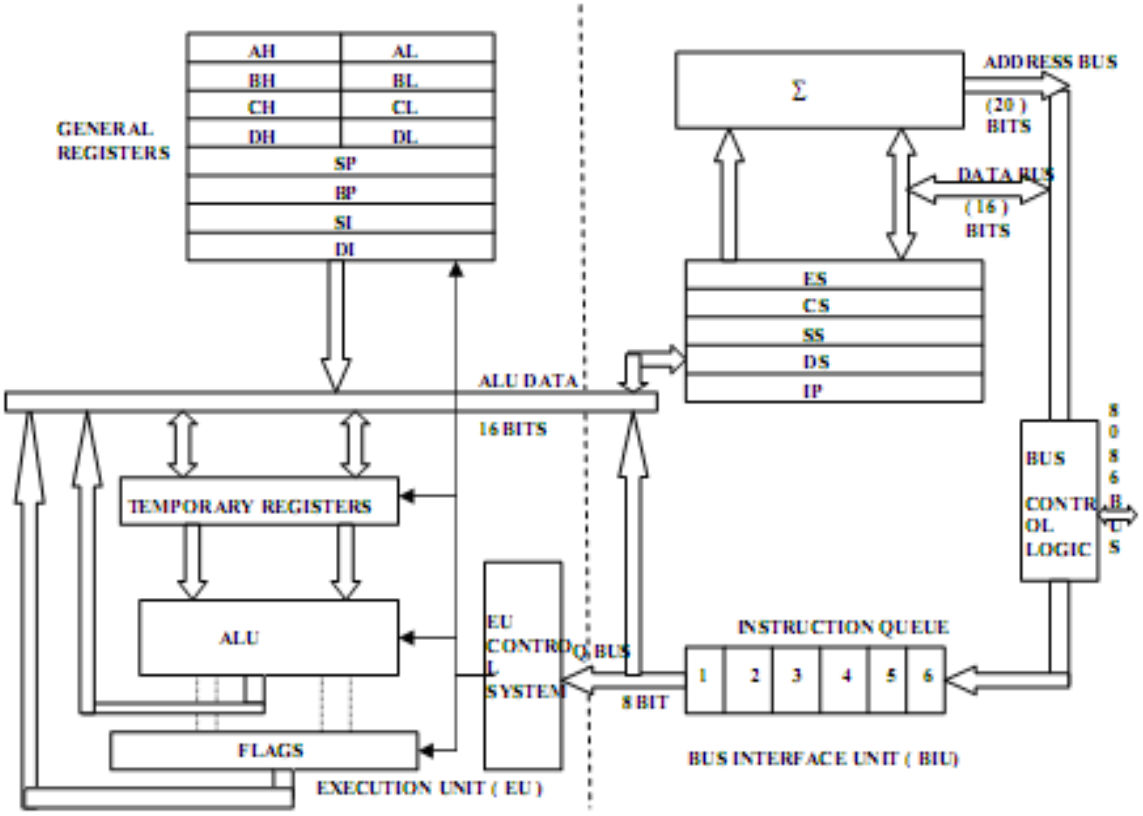
The micro-architecture of a processor is its internal architecture. The micro-architecture of the 8086 microprocessors employs parallel processing; they are implemented with several simultaneously operating processing units. Figure-2 shows the block diagram of 8086. The 8086 CPU is divided into two independent functional units:

1. Bus Interface Unit (BIU)
2. Execution Unit (EU)

Each unit has dedicated functions and both operate at the same time. In essence, this parallel processing effectively makes the fetch and execution of instructions independent operations. This results in efficient use of the system bus and higher performance for 8086 microcomputer systems.

The differences between an 8088 microprocessor and an 8086 microprocessor are:

- In the 8088, the BIU data bus path is 8 bits wide versus the 8086's 16-bit data bus.
- Another difference is that the 8088 instruction queue is four bytes long instead of six bytes in 8086.



Block Diagram of 8086

Bus Interface Unit (BIU)

The BIU contains the segment registers, the instruction pointer, the address generation adder, bus control logic, and an instruction queue. The functions of BIU are :

- Manage the bus control unit , segment registers and instruction queue.
- Control the buses that transfer data to the EU , to memory and to external input /output devices , whereas the segment registers control memory addressing.
- Fetch the instructions or data from memory.
- Write the data to memory.
- Write the data to the port.
- Read data from the port.

In simple words, the BIU handles all transfers of data and addresses on the buses for the execution unit.

Execution Unit (EU)

The Execution Unit (EU) is responsible for decoding and executing instructions. The EU contains an Arithmetic and Logical Unit (ALU) , a Control Unit (CU) and a number of registers.

The EU receives program instruction codes and data from the BIU, executes these instructions, and store the results in the general registers. By passing the data back to the BIU, data can also be stored in a memory location or written to an output device. Note that the EU has no connection to the system buses. It receives and outputs all its data thru the BIU.

In other words, the functions of execution unit are:

- To tell BIU where to fetch the instructions or data from.
- To decode the instructions.
- To execute the instructions.

Instruction Queue(IQ)

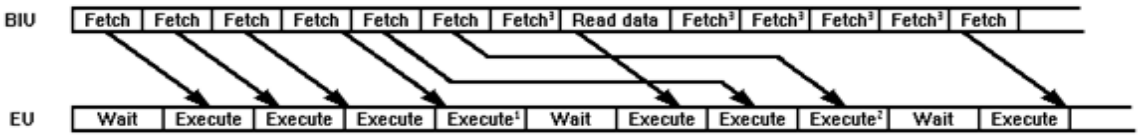
The functions of IQ are:

1. To increase the execution speed, BIU fetches as many as six instruction bytes ahead to time from memory.
2. All six bytes are then held in First In First Out (FIFO) 6 byte register called instruction queue.
3. Then all bytes have to be given to EU one by one.
4. This prefetching operation of BIU may be in parallel with execution operation of EU, which improves the speed execution of the instruction.

8086 has Pipelining Architecture

- 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 (FIFO) register set called a queue.
- When the EU is ready for its next instruction from the queue in the BIU. This is much faster than sending out an address to the system memory and waiting for memory to send back the next instruction byte or bytes.
- Fetching the next instruction while the current instruction executes is called pipelining. The advantage of this pipelined architecture is that the EU can execute instructions almost continually instead of having to wait for the BIU to fetch a new instruction

FETCH AND EXECUTE CYCLE



[1]This instruction requires a request for data not in the queue
[2]Jump instruction occurs
[3]These bytes are discarded

INSTRUCTION PIPELINE

There are three conditions that will cause the EU to enter a "wait" mode;

1. When an instruction requires access to a memory location not in the queue. The BIU must suspend fetching instructions and output the address of this memory location. After waiting for the memory access, the EU can resume executing instruction codes from the queue (and the BIU can resume filling the queue).

2. When the instruction to be executed is a "jump" instruction. In this case control is to be transferred to a new (nonsequential) address. The queue, however, assumes that instructions will always be executed in sequence and thus will be holding the "wrong" instruction codes. The EU must wait while the instruction at the jump address is fetched. Note that any bytes presently in the queue must be discarded (they are overwritten).
3. During execution of instructions that are slow to execute. For example, the instruction AAM (ASCII Adjust for Multiplication) requires 83 clock cycles to complete. At four cycles per instruction fetch, the queue will be completely filled during the execution of this single instruction.

Programming Model

The programming model for a microprocessor shows the various internal registers in BIU and EU that are accessible to the programmer. The 14 registers of 8086 microprocessor are categorized into four groups. They are General Purpose registers (Data Registers), Pointer & Index registers, Segment registers and Flag register as shown in figure 3. In general, each register has a special function.

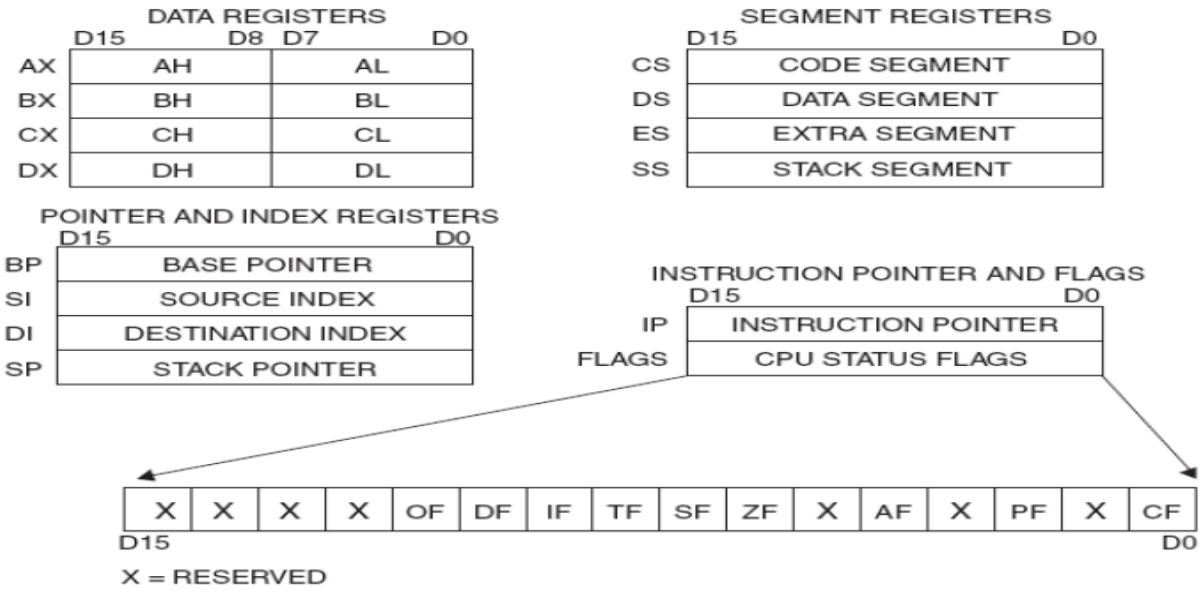
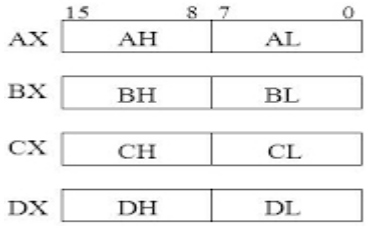


Figure-3: programming model of 8086 microprocessor.

General Purpose Registers

There are four 16-bit general purpose registers:



1. **AX Register**: AX register is also known as accumulator register. It is 16-bit registers, and it is divided into two 8-bit registers (AH and AL) to also perform 8-bit instructions. Accumulator can be used for I/O operations and string manipulation.
2. **BX Register**: This register is mainly used as a base register. It is 16-bit registers, and it is divided into two 8-bit registers (BH and BL) to also perform 8-bit instructions. It holds the starting base location (offset address) of a memory region within a data segment.
3. **CX Register**: It is defined as a counter register. It is 16-bit registers, and it is divided into two 8-bit registers (CH and CL) to also perform 8-bit instructions. .It is primarily used in loop instruction to store loop counter .
4. **DX Register**: This is the data register. It is of 16 bits, and it is divided into two 8-bit registers DH and DL to also perform 8-bit instructions. It is used in multiplication and input/output port addressing.