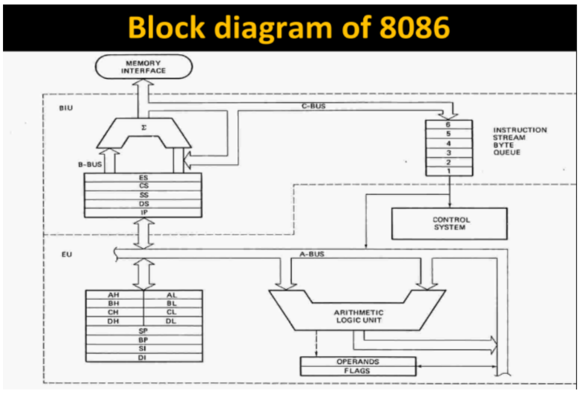
Block diagram of 8086



Q) explain Arctitecture of 8086 **microprocessor with block diagram**

8086 micro. is a 16 bit micro. Chip designed by intel in 1978 And it is 40 pin IC.

As we know that a microprocessor performs arithmetic and logic operations. And an 8086 microprocessor is able to perform these operations with 16-bit data in one cycle. Hence is a 16-bit microprocessor. Thus the size of the data bus is 16-bit as it can carry 16-bit data at a time. The architecture of 8086 microprocessor, is very much different from that of 8085 microprocessor.

The architecture of 8086 microprocessor is composed of 2 major units, the BIU i.e., Bus Interface Unit and EU i.e., Execution Unit. The figure below shows the block diagram of the architectural representation of the 8086 microprocessor:

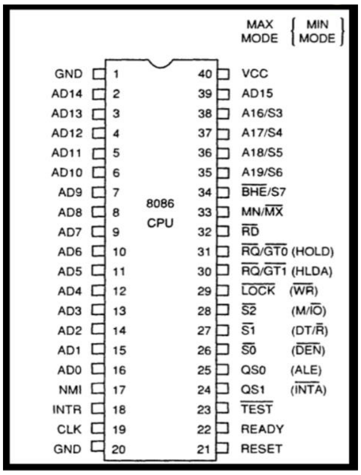
The Bus Interface Unit (BIU) manages the data, address and control buses.

The Execution Unit (EU) performs the decoding and execution of the instructions that are being fetched from the desired memory location.

It supports pipelined arctitecture

It has 256 rectored interrupts

Q***) Explain pin Diagram of 8086***



AD0-AD15 : Address/Data bus. These are low order address bus. They are multiplexed with data and A16-A19 : High order address bus. These are multiplexed with status signals.

S2, S1, S0 : Status pins. These pins are active during T4, T1 and T2 states and is returned to passive state (1,1,1 during T3 or Tw (when ready is inactive)

BHE'/S7 : Bus High Enable/Status. During T1 it is low. It is used to enable data onto the most significant half of data bus

RD': This is used for read operation. It is an output signal. It is active when low.

READY : This is the acknowledgement from the memory or slow device that they have completed the data transfer.

INTR : Interrupt Request. This is triggered input. This is sampled during the last clock cycles of each instruction for determining the availability of the request

NMI : Non maskable interrupt. This is an edge triggered input which results in a type II interrupt

INTA : Interrupt acknowledge. It is active low(0) during T2, T3 and Tw of each interrupt acknowledge cycle.

MN/MX' : Minimum/Maximum. This pin signal indicates what mode the processor will operate in

RQ'/GT1', RQ'/GT0' : Request/Grant.

LOCK' : Its an active low pin.

TEST' : This examined by a 'WAIT' instruction CLK : Clock Input Vcc : Power Supply(+5V D.C.)

RESET : This pin requires the microprocessor to terminate its present activity immediately.

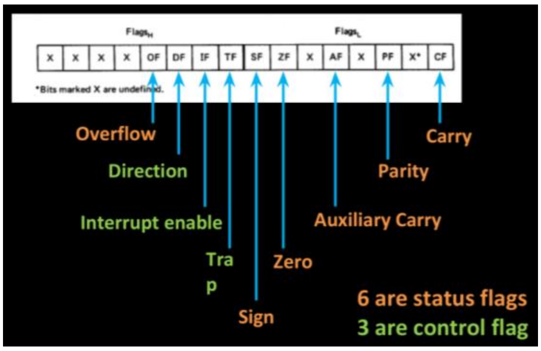
GND : Ground DEN : Data enable. QS1,QS0 : Queue Status.

DT/R : Data Transmit/Receive

HOLD/HOLDA : HOLD indicates that another master has been requesting a local bus

ALE : Address Latch Enable. ALE is provided by the microprocessor to latch the address into the 8282 or 8283 address latch

Q) EXPLAIN FLAGS REGISTER.



The Flag register is a Special Purpose Register. Depending upon the value of result after any arithmetic and logical operation the flag bits become set (1) or reset (0).

A) STATUS FLAGS – There are 6 flag registers in 8086 microprocessor which become set(1) or reset(0) depending upon condition after either 8-bit or 16-bit operation. These flags are conditional/status flags. 5 of these flags are same as in case of 8085 microprocessor and their working is also same as in 8085 microprocessor. The sixth one is the overflow flag.
The 6 status flags are:

Sign Flag(S),Zero Flag(Z),Auxiliary Cary Flag(AC),Parity Flag (P), Carry Flag (CY)
These first five flags are defined here

Overflow Flag (O) – This flag will be set (1) if the result of a signed operation is too large to fit in the number of bits available to represent it, otherwise reset (0). After any operation, if D[6] generates any carry and passes to D[7] OR if D[6] does not generates carry but D[7] generates, overflow flag becomes set, i.e., 1. If D[6] and D[7] both generate carry or both do not generate any carry, then overflow flag becomes reset, i.e., 0.
Example: On adding bytes 100 + 50 (result is not in range -128…127), so overflow flag will set.

B) CONTROL FLAGS – The control flags enable or disable certain operations of the microprocessor. There are 3 control flags in 8086 microprocessor and these are:
Directional Flag (D) – This flag is specifically used in string instructions.
If directional flag is set (1), then access the string data from higher memory location towards lower memory location.
If directional flag is reset (0), then access the string data from lower memory location towards higher memory location.
Interrupt Flag (I) – This flag is for interrupts.
If interrupt flag is set (1), the microprocessor will recognize interrupt requests from the peripherals.
If interrupt flag is reset (0), the microprocessor will not recognize any interrupt requests and will ignore them.
Trap Flag (T) – This flag is used for on-chip debugging. Setting trap flag puts the microprocessor into single step mode for debugging. In single stepping, the microprocessor executes a instruction and enters into single step ISR.
If trap flag is set (1), the CPU automatically generates an internal interrupt after each instruction, allowing a program to be inspected as it executes instruction by instruction.
If trap flag is reset (0), no function is performed.

Q) EXPLAIN MAXIMUM MODE

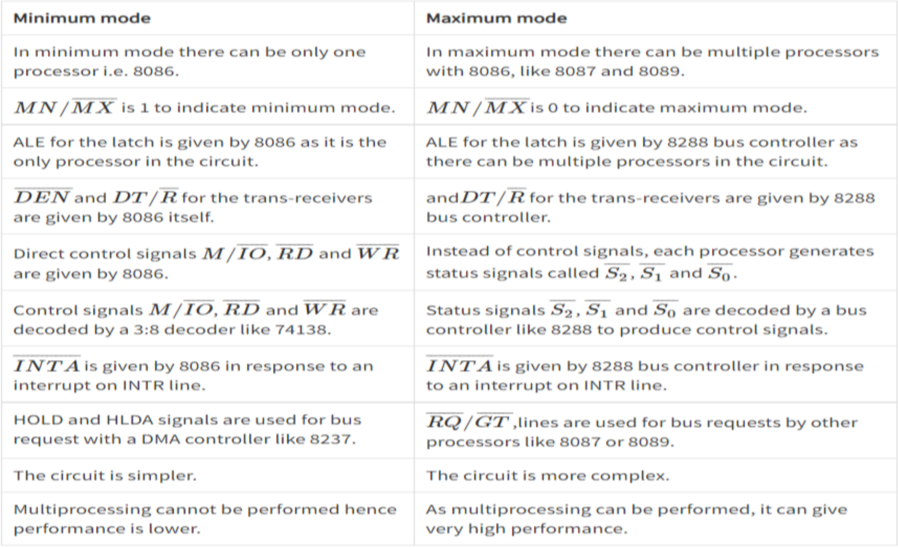
-In this we can connect more processors to 8086 (8087/8089).

-8086 max mode is basically for implementation of allocation of global resources and passing bus control to other coprocessor(i.e. second processor in the system), because two processors can not access system bus at same instant.

-All processors execute their own program.

-The resources which are common to all processors are known as global resources.
-The resources which are allocated to a particular processor are known as local or private resources

Q) DIFFERENCE BETWEEN MINIMUM AND MAXIMUM MODE



Minimum mode	Maximum mode
In minimum mode there can be only one processor i.e. 8086.	In maximum mode there can be multiple processors with 8086, like 8087 and 8089.
$MN/\overline{MX}$ is 1 to indicate minimum mode.	$MN/\overline{MX}$ is 0 to indicate maximum mode.
ALE for the latch is given by 8086 as it is the only processor in the circuit.	ALE for the latch is given by 8288 bus controller as there can be multiple processors in the circuit.
$\overline{DEN}$ and $DT/\overline{R}$ for the trans-receivers are given by 8086 itself.	and$DT/\overline{R}$ for the trans-receivers are given by 8288 bus controller.
Direct control signals $M/\overline{IO}$, $\overline{RD}$ and $\overline{WR}$ are given by 8086.	Instead of control signals, each processor generates status signals called $\overline{S_2}$, $\overline{S_1}$ and $\overline{S_0}$.
Control signals $M/\overline{IO}$, $\overline{RD}$ and $\overline{WR}$ are decoded by a 3:8 decoder like 74138.	Status signals $\overline{S_2}$, $\overline{S_1}$ and $\overline{S_0}$ are decoded by a bus controller like 8288 to produce control signals.
$\overline{INTA}$ is given by 8086 in response to an interrupt on INTR line.	$\overline{INTA}$ is given by 8288 bus controller in response to an interrupt on INTR line.
HOLD and HLDA signals are used for bus request with a DMA controller like 8237.	$\overline{RQ}/\overline{GT}$,lines are used for bus requests by other processors like 8087 or 8089.
The circuit is simpler.	The circuit is more complex.
Multiprocessing cannot be performed hence performance is lower.	As multiprocessing can be performed, it can give very high performance.

Q) EXPLAIN MINIMUM MODE

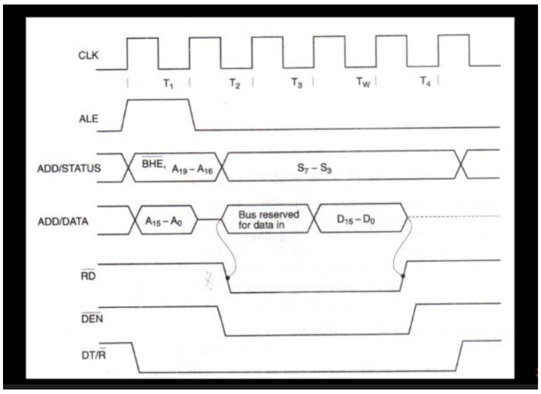
-The 8086 microprocessor operates in minimum mode when MN/MX’ = 1.
-In minimum mode,8086 is the only processor in the system which provides all the control signals which are needed for memory operations and I/O interfacing.
-Here the circuit is simple but it does not support multiprocessing.
The other components which are transceivers, latches, 8284 clock generator, 74138 decoder, memory and i/o devices are also present in the system.
-The address bus of 8086 is 20 bits long. By this we can access 2^{20} byte memory i.e. 1MB . Out of 20 bits, 16 bits A_0 to A_{15}(or 16 lines) are multiplexed with a data bus. By multiplexing, it means they will act as address lines during the first T state of the machine cycle and in the rest, they act as data lines. A_{16} to A_{19} are multiplexed S_3 to S_6 and BHE’ is multiplexed with S_7.

-**8282(8 BITS) LATCH**: The latches are buffered D FF. They are used to separate the valid address from the multiplexed Address/data bus by using the control signal ALE, which is connected to strobe(STB) of 8282. The ALE is active high signal. Here three such latches are required because the address is 20 bits.

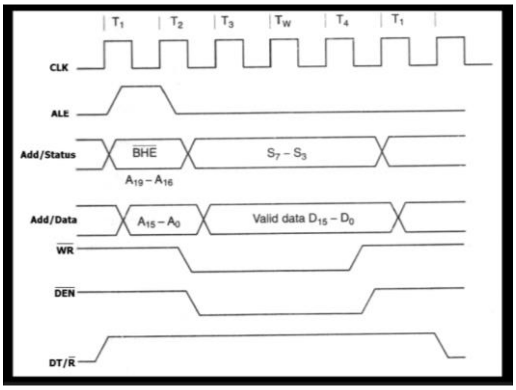
-8286 (8 BITS) TRANSCEIVERS : They are bidirectional buffers and also known as data amplifiers. They are used to separate the valid data from multiplexed add/data bus. Two such transceivers are needed because the data bus is 16 bits long. 8286 is connected to DT/R’ and DEN’ signals. They are enabled through the DEN signal .The direction of data on the data bus is controlled by the DT/R’ signal. DT/R’ is connected to T and DEN’ is connected to OE’.

TIMING DIADRAM:

‘Read’ Cycle timing Diagram for Minimum Mode



‘Write’ Cycle timing Diagram for Minimum Mode



Q) INSTRUCTION SETS

Arithmetic Instructions

ADD- Adds data to the accumulator i.e. AL or AX register or memory locations.

SUB- Subtract immediate data from accumulator, memory or register.

MUL- Unsigned 8-bit or 16-bit multiplication

DIV- Unsigned 8-bit or 16-bit division.

Data Transfer instruction

MOV- Moves data from register to register, register to memory, memory to register, memory to accumulator, accumulator to memory, etc

PUSH- Pushes (sends, writes or moves) the content of a specified register or memory location(s) onto the top of the stack.

POP- Pops (reads) two bytes from the top of the stack and keeps them in a specified register, or memory location(s).

Logical Instructions

AND- Performs bit by bit logical AND operation of two operands and places the result in the specified destination.

OR-Performs bit by bit logical OR operation of two operands and places the result in the specified destination.

XOR- Performs bit by bit logical XOR operation of two operands and places the result in the specified destination

NOT- Takes one's complement of the content of a specified register or memory location(s).

Q) MEMORY SEGMENTATION

Segmentation is the process in which the main memory of the computer is logically divided into different segments and each segment has its own base address. It is basically used to enhance the speed of execution of the computer system, so that the processor is able to fetch and execute the data from the memory easily and fast.

Types Of Segmentation –

1. **Overlapping Segment –** A segment starts at a particular address and its maximum size can go up to 64kilobytes. But if another segment starts along with this 64kilobytes location of the first segment, then the two are said to be *Overlapping Segment*.
2. **Non-Overlapped Segment –** A segment starts at a particular address and its maximum size can go up to 64kilobytes. But if another segment starts before this 64kilobytes location of the first segment, then the two segments are said to be *Non-Overlapped Segment*.

NEED FOR SEGMENTATION-

The no. of address lines in 8086 is 20. 8086 biu will send 20 bit address

But the bus interface unit(BIU) contains fourc16bit special purpose registers(mentioned below) called as segment registers.

-CODE SEGMENT REGISTERS(cs)= is used for addressing code segment,where the program is stored

-DATA SEGMENT(DS)= is used for addressing data segment of memory where the data is stored

-EXTRA SEGMENT(ES)= is used to address extra segment memory which stores data.

-STACK SEGMENT(SS)= is used for addressing stack segment. The stack segment is used to store stack data

Advantages of the Segmentation-

-It provides a powerful memory management mechanism.
-Data related or stack related operations can be performed in different segments.
-Code related operation can be done in separate code segments.
-It allows to processes to easily share data.
-It allows to extend the address ability of the processor, i.e. segmentation allows the use of 16 bit registers to give an addressing capability of 1 Megabytes. Without segmentation, it would require 20 bit registers.
-It is possible to enhance the memory size of code data or stack segments beyond 64 KB by allotting more than one segment for each area.

Q) EXPLAIN ADDRESSING MODES

Register mode – In this type of addressing mode both the operands are registers.

Immediate mode – In this type of addressing mode the source operand is a 8 bit or 16 bit data. Destination operand can never be immediate data.

Displacement or direct mode – In this type of addressing mode the effective address is directly given in the instruction as displacement.

Register indirect mode – In this addressing mode the effective address is in SI, DI or BX.

Based indexed mode – In this the effective address is sum of base register and index register.

Indexed mode – In this type of addressing mode the effective address is sum of index register and displacement.

Based mode – In this the effective address is the sum of base register and displacement.

Based indexed displacement mode – In this type of addressing mode the effective address is the sum of index register, base register and displacement.

Relative mode –
In this the effective address is calculated with reference to instruction pointer.