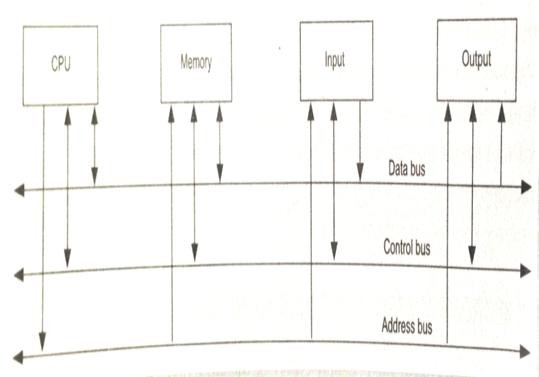
**INTRODUCTION TO MICROPROCESSOR AND MICROCOMPUTER ARCHITECTURE**:

A *microprocessor* is a programmable electronics chip that has computing and decision making capabilities similar to central processing unit of a computer. Any microprocessor- based systems having limited number of resources are called *microcomputers*. Nowadays, microprocessor can be seen in almost all types of electronics devices like mobile phones, printers, washing machines etc. Microprocessors are also used in advanced applications like radars, satellites and flights. Due to the rapid advancements in electronic industry and large scale integration of devices results in a significant cost reduction and increase application of microprocessors and their derivatives.



Microprocessor-based system

* + **Bit**: A bit is a single binary digit.
  + **Word**: A word refers to the basic data size or bit size that can be processed by the arithmetic and logic unit of the processor. A 16-bit binary number is called a word in a 16-bit processor.
  + **Bus**: A bus is a group of wires/lines that carry similar information.
  + **System Bus**: The system bus is a group of wires/lines used for communication between the microprocessor and peripherals.
  + **Memory Word**: The number of bits that can be stored in a register or memory element is called a memory word.
  + **Address Bus**: It carries the address, which is a unique binary pattern used to identify a memory location or an I/O port. For example, an eight bit address bus has eight lines and thus it can address 28 = 256 different locations. The locations in hexadecimal format can be written as 00H – FFH.
  + **Data Bus**: The data bus is used to transfer data between memory and processor or between I/O device and processor. For example, an 8-bit processor will generally have an 8-bit data bus and a 16-bit processor will have 16-bit data bus.
  + **Control Bus**: The control bus carry control signals, which consists of signals for selection of memory or I/O device from the given address, direction of data transfer and synchronization of data transfer in case of slow devices.

A typical microprocessor consists of arithmetic and logic unit (ALU) in association with control unit to process the instruction execution. Almost all the microprocessors are based on the principle of store-program concept. In *store-program concept*, programs or instructions are sequentially stored in the memory locations that are to be executed. To do any task using a microprocessor, it is to be programmed by the user. So the programmer must have idea about its internal resources, features and supported instructions. Each microprocessor has a set of instructions, a list which is provided by the microprocessor manufacturer. The instruction set of a microprocessor is provided in two forms: *binary machine code and mnemonics*.

Microprocessor communicates and operates in binary numbers 0 and 1. The set of instructions in the form of binary patterns is called a *machine language* and it is difficult for us to understand. Therefore, the binary patterns are given abbreviated names, called mnemonics, which forms the *assembly language*. The conversion of assembly-level language into binary machine-level language is done by using an application called *assembler.*

Technology Used:

The semiconductor manufacturing technologies used for chips are:

* + Transistor-Transistor Logic (TTL)
  + Emitter Coupled Logic (ECL)
  + Complementary Metal-Oxide Semiconductor (CMOS) Classification of Microprocessors:

Based on their specification, application and architecture microprocessors are classified.

*Based on size of data bus:*

* + 4-bit microprocessor
  + 8-bit microprocessor
  + 16-bit microprocessor
  + 32-bit microprocessor

*Based on application:*

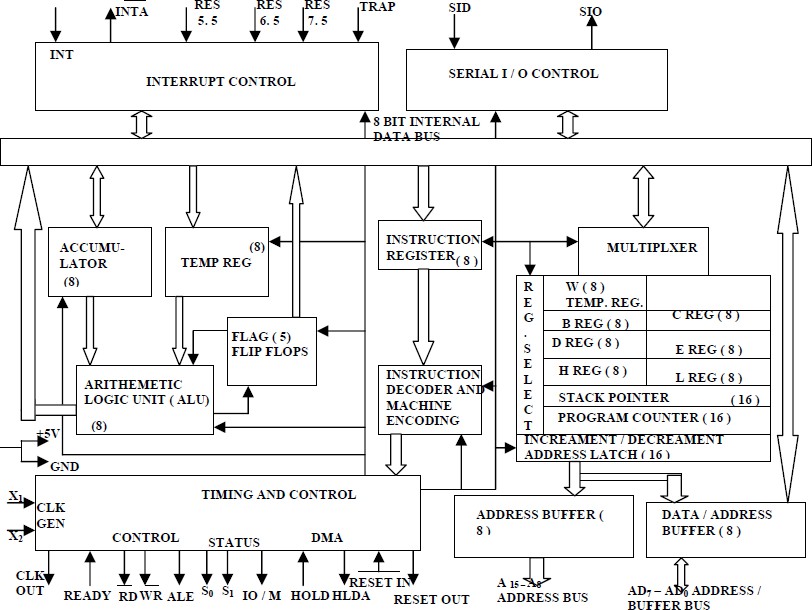
* + General-purpose microprocessor- used in general computer system and can be used by programmer for any application. Examples, 8085 to Intel Pentium.
  + Microcontroller- microprocessor with built-in memory and ports and can be programmed for any generic control application. Example, 8051.
  + Special-purpose processors- designed to handle special functions required for an application. Examples, digital signal processors and application-specific integrated circuit (ASIC) chips.

*Based on architecture:*

* + Reduced Instruction Set Computer (RISC) processors
  + Complex Instruction Set Computer (CISC) processors

## 8085 MICROPROCESSOR ARCHITECTURE

The 8085 microprocessor is an 8-bit processor available as a 40-pin IC package and uses +5 V for power. It can run at a maximum frequency of 3 MHz. Its data bus width is 8-bit and address bus width is 16-bit, thus it can address 216 = 64 KB of memory. The internal architecture of 8085 is shown is Fig. 2.



Internal Architecture of 8085

Arithmetic and Logic Unit

The ALU performs the actual numerical and logical operations such as Addition (ADD), Subtraction (SUB), AND, OR etc. It uses data from memory and from Accumulator to perform operations. The results of the arithmetic and logical operations are stored in the accumulator.

Registers

The 8085 includes six registers, one accumulator and one flag register, as shown in Fig. 3. In addition, it has two 16-bit registers: stack pointer and program counter. They are briefly described as follows.

The 8085 has six general-purpose registers to store 8-bit data; these are identified as B, C, D, E, H and L. they can be combined as register pairs - BC, DE and HL to perform some

16-bit operations. The programmer can use these registers to store or copy data into the register by using data copy instructions.

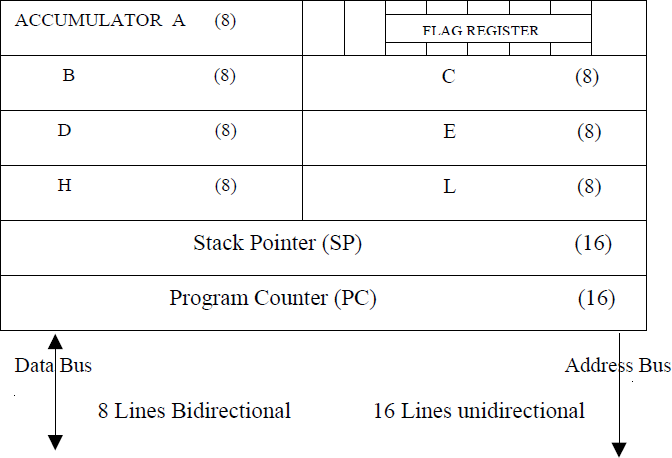


Fig. 3 Register organisation

Accumulator

The accumulator is an 8-bit register that is a part of ALU. This register is used to store 8-bit data and to perform arithmetic and logical operations. The result of an operation is stored in the accumulator. The accumulator is also identified as register A.

Flag register

The ALU includes five flip-flops, which are set or reset after an operation according to data condition of the result in the accumulator and other registers. They are called Zero (Z), Carry (CY), Sign (S), Parity (P) and Auxiliary Carry (AC) flags. Their bit positions in the flag register are shown in Fig. 4. The microprocessor uses these flags to test data conditions.

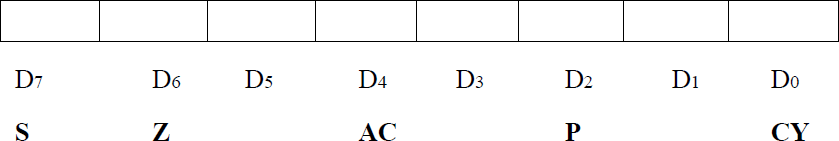


Fig. 4 Flag register

For example, after an addition of two numbers, if the result in the accumulator is larger than 8-bit, the flip-flop uses to indicate a carry by setting CY flag to 1. When an arithmetic operation results in zero, Z flag is set to 1. The S flag is just a copy of the bit D7 of the accumulator. A negative number has a 1 in bit D7 and a positive number has a 0 in 2’s complement representation. The AC flag is set to 1, when a carry result from bit D3 and passes to bit D4. The P flag is set to 1, when the result in accumulator contains even number of 1s.

Program Counter (PC)

This 16-bit register deals with sequencing the execution of instructions. This register is a memory pointer. The microprocessor uses this register to sequence the execution of the instructions. The function of the program counter is to point to the memory address from which the next byte is to be fetched. When a byte is being fetched, the program counter is automatically incremented by one to point to the next memory location.

Stack Pointer (SP)

The stack pointer is also a 16-bit register, used as a memory pointer. It points to a memory location in R/W memory, called stack. The beginning of the stack is defined by loading 16- bit address in the stack pointer.

Instruction Register/Decoder

It is an 8-bit register that temporarily stores the current instruction of a program. Latest instruction sent here from memory prior to execution. Decoder then takes instruction and decodes or interprets the instruction. Decoded instruction then passed to next stage.

Control Unit

Generates signals on data bus, address bus and control bus within microprocessor to carry out the instruction, which has been decoded. Typical buses and their timing are described as follows:

* *Data Bus*: Data bus carries data in binary form between microprocessor and other external units such as memory. It is used to transmit data i.e. information, results of arithmetic etc between memory and the microprocessor. Data bus is bidirectional in nature. The data bus width of 8085 microprocessor is 8-bit i.e. 28 combination of binary digits and are typically identified as D0 – D7. Thus size of the data bus determines what arithmetic can be done. If only 8-bit wide then largest number is 11111111 (255 in decimal). Therefore, larger numbers have to be broken down into chunks of 255. This slows microprocessor.
* *Address Bus*: The address bus carries addresses and is one way bus from microprocessor to the memory or other devices. 8085 microprocessor contain 16-bit address bus and are generally identified as A0 - A15. The higher order address lines (A8 – A15) are unidirectional and the lower order lines (A0 – A7) are multiplexed (time-shared) with the eight data bits (D0 – D7) and hence, they are bidirectional.
* *Control Bus*: Control bus are various lines which have specific functions for coordinating and controlling microprocessor operations. The control bus carries control signals partly unidirectional and partly bidirectional. The following control and status signals are used by 8085 processor:
  1. ALE (output): Address Latch Enable is a pulse that is provided when an address appears on the AD0 – AD7 lines, after which it becomes 0.
  2. RD (active low output): The Read signal indicates that data are being read from the selected I/O or memory device and that they are available on the data bus.
  3. WR (active low output): The Write signal indicates that data on the data bus are to be written into a selected memory or I/O location.
  4. IO/M (output): It is a signal that distinguished between a memory operation

and an I/O operation. When IO/M = 0 it is a memory operation and IO/M = 1 it is an I/O operation.

* 1. S1 and S0 (output): These are status signals used to specify the type of operation being performed; they are listed in Table 1.

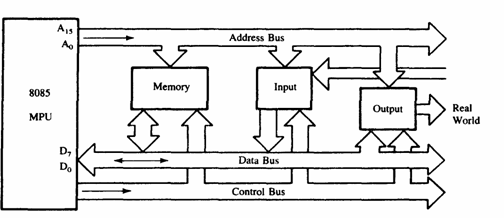
Table 1 Status signals and associated operations

|  |  |  |
| --- | --- | --- |
| S1 | S0 | States |
| 0 | 0 | Halt |
| 0 | 1 | Write |
| 1 | 0 | Read |
| 1 | 1 | Fetch |

The schematic representation of the 8085 bus structure is as shown in Fig. 5. The microprocessor performs primarily four operations:

1. Memory Read: Reads data (or instruction) from memory.
2. Memory Write: Writes data (or instruction) into memory.
3. I/O Read: Accepts data from input device.
4. I/O Write: Sends data to output device.

The 8085 processor performs these functions using address bus, data bus and control bus as shown in Fig. 5.



The 8085 bus structure

## 8085 PIN DESCRIPTION

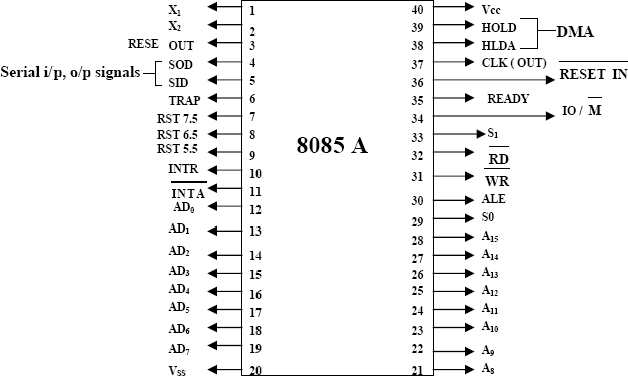
Properties:

* + It is a 8-bit microprocessor
  + Manufactured with N-MOS technology
  + 40 pin IC package
  + It has 16-bit address bus and thus has 216 = 64 KB addressing capability.
  + Operate with 3 MHz single-phase clock

 +5 V single power supply

The logic pin layout and signal groups of the 8085nmicroprocessor are shown in Fig. 6. All the signals are classified into six groups:

* + Address bus
  + Data bus
  + Control & status signals
  + Power supply and frequency signals
  + Externally initiated signals
  + Serial I/O signals



8085 microprocessor pin layout and signal groups

Address and Data Buses:

* + A8 – A15 (output, 3-state): Most significant eight bits of memory addresses and the eight bits of the I/O addresses. These lines enter into tri-state high impedance state during HOLD and HALT modes.
  + AD0 – AD7 (input/output, 3-state): Lower significant bits of memory addresses and the eight bits of the I/O addresses during first clock cycle. Behaves as data bus

during third and fourth clock cycle. These lines enter into tri-state high impedance state during HOLD and HALT modes.

Control & Status Signals:

* + ALE: Address latch enable
  + RD : Read control signal.
  + WR : Write control signal.
  + IO/M , S1 and S0 : Status signals.

Power Supply & Clock Frequency:

* + Vcc: +5 V power supply
  + Vss: Ground reference
  + X1, X2: A crystal having frequency of 6 MHz is connected at these two pins
  + CLK: Clock output

Externally Initiated and Interrupt Signals:

* + RESET IN : When the signal on this pin is low, the PC is set to 0, the buses are tri-

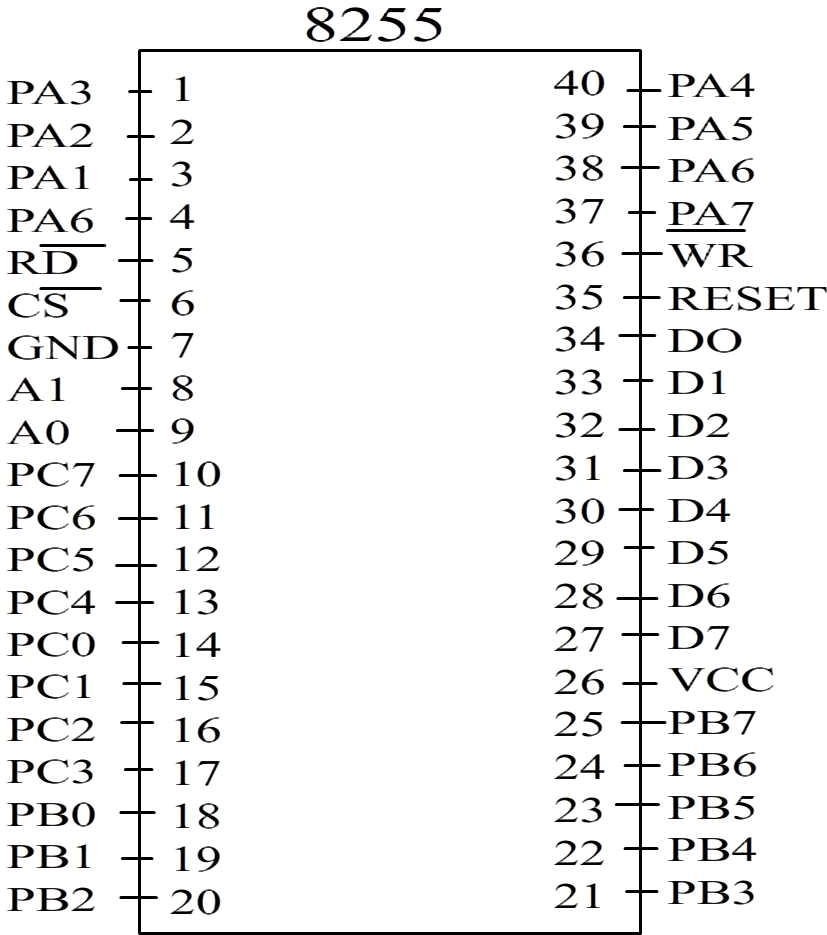
stated and the processor is reset.

* + RESET OUT: This signal indicates that the processor is being reset. The signal can be used to reset other devices.
  + READY: When this signal is low, the processor waits for an integral number of clock cycles until it goes high.
  + HOLD: This signal indicates that a peripheral like DMA (direct memory access) controller is requesting the use of address and data bus.
  + HLDA: This signal acknowledges the HOLD request.
  + INTR: Interrupt request is a general-purpose interrupt.
  + INTA : This is used to acknowledge an interrupt.
  + RST 7.5, RST 6.5, RST 5,5 – restart interrupt: These are vectored interrupts and have highest priority than INTR interrupt.
  + TRAP: This is a non-maskable interrupt and has the highest priority. Serial I/O Signals:
  + SID: Serial input signal. Bit on this line is loaded to D7 bit of register A using RIM instruction.
  + SOD: Serial output signal. Output SOD is set or reset by using SIM instruction.

# INTEL 8255: (Programmable Peripheral Interface)

The 8255A is a general purpose programmable I/O device designed for use with Intel microprocessors. It consists of three 8-bit bidirectional I/O ports (24I/O lines) that can be configured to meet different system I/O needs. The three ports are PORT A, PORT B & PORT C. Port A contains one 8-bit output latch/buffer and one 8-bit input buffer. Port B is same as PORT A or PORT B. However, PORT C can be split into two parts PORT C lower (PC0-PC3) and PORT C upper (PC7-PC4) by the control word. The three ports are divided in two groups Group A (PORT A and upper PORT C) Group B (PORT B and lower PORT C). The two groups can be programmed in three different modes. In the first mode (mode 0), each group may be programmed in either input mode or output mode (PORT A, PORT B, PORT C lower, PORT C upper). In mode 1, the second’s mode, each group may be programmed to have 8-lines of input or output (PORT A or PORT B) of the remaining 4-lines (PORT C lower or PORT C upper) 3-lines are used for hand shaking and interrupt control signals. The third mode of operation (mode 2) is a bidirectional bus mode which uses 8-line (PORT A only for a bidirectional bus and five lines (PORT C upper 4 lines and borrowing one from other group) for handshaking.

The 8255 is contained in a 40-pin package, whose pin out is shown below:



The block diagram is shown below:

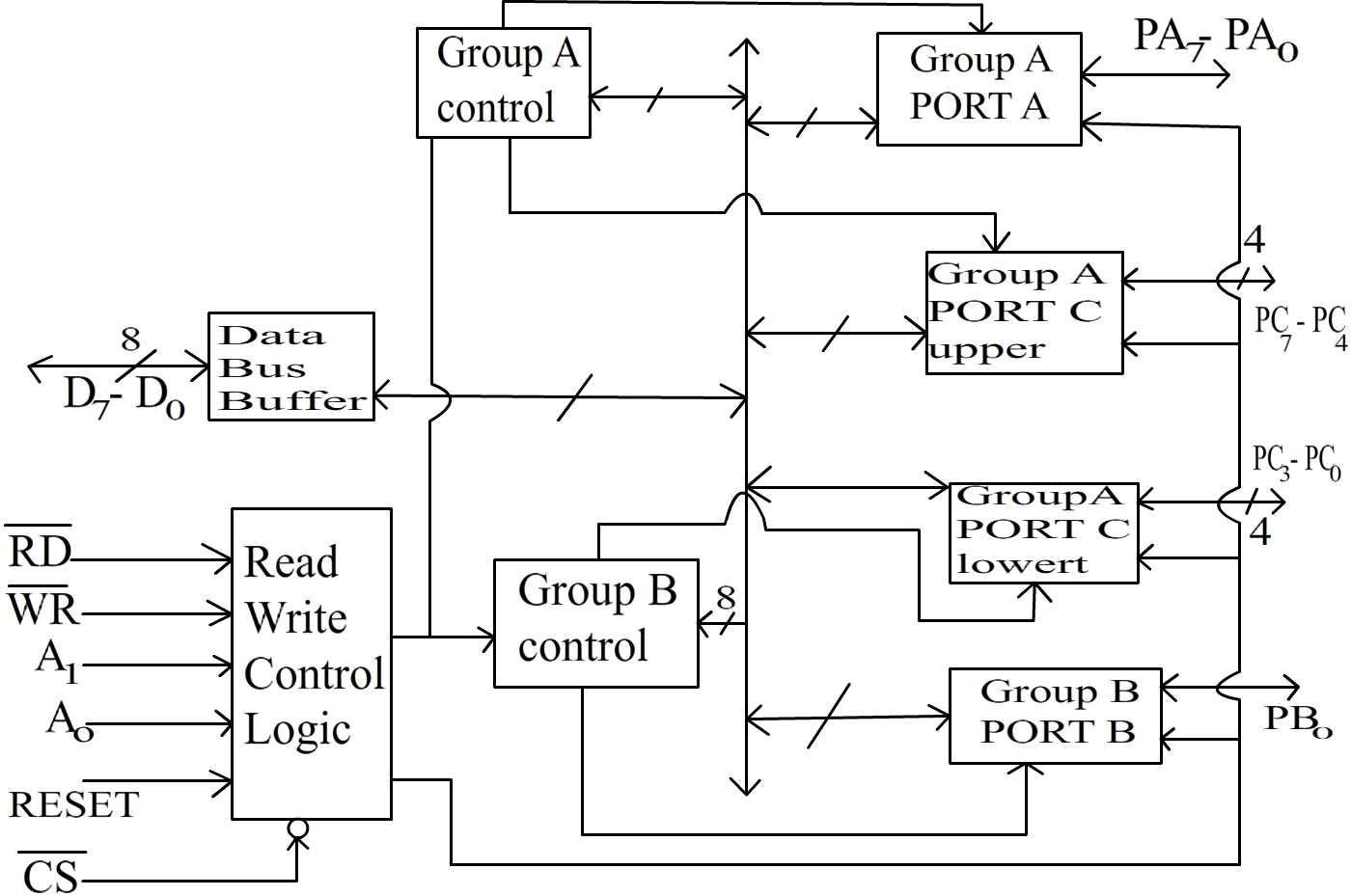
PIN Names RESET – Reset input

- Chip selected

* Read input
* Write input

A0 A1 – Port Address PA7 – PA0 – PORT A PB7 – PB0 – PORT B PC7 – PC0 – PORT C VCC - +5v

GND - Ground



Functional Description:

This support chip is a general purpose I/O component to interface peripheral equipment to the microcomputer system bus. It is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

Data Bus Buffer:

It is a tri-state 8-bit buffer used to interface the chip to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer. The data lines are connected to BDB of p.

Read/Write and logic control:

The function of this block is to control the internal operation of the device and to control the transfer of data and control or status words. It accepts inputs from the CPU address and control buses and in turn issues command to both the control groups.

 Chip Select:

A low on this input selects the chip and enables the communication between the 8255 A & the CPU. It is connected to the output of address decode circuitry to select the device when it  (Read). A low on this input enables the 8255 to send the data or status information to the CPU on the data bus.

 (Write):

A low on this input pin enables the CPU to write data or control words into the 8255 A.

A1, A0 port select:

These input signals, in conjunction with the  and  inputs, control the selection of one of the three ports or the control word registers. They are normally connected to the least significant bits of the address bus (A0 and A1).

Following Table gives the basic operation,

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A1 | A0 |  |  |  | Input operation | |
| 0 | 0 | 0 | 1 | 0 | PORT A | Data bus |
| 0 | 1 | 0 | 1 | 0 | PORT B | Data bus |
| 1 | 0 | 0 | 1 | 0 | PORT C | Data bus |
| 0 | 0 | 1 | 0 | 0 | Output operation  Data bus PORT A | |
| 0 | 1 | 1 | 0 | 0 | Data bus | PORT B |
| 1 | 0 | 1 | 0 | 0 | Data bus | PORT C |
| 1 | 1 | 1 | 0 | 0 | Data bus | control |

All other states put data bus into tri-state/illegal condition.

# RESET:

A high on this input pin clears the control register and all ports (A, B & C) are initialized to input mode. This is connected to RESET OUT of 8255. This is done to prevent destruction of circuitry connected to port lines. If port lines are initialized as output after a power up or

reset, the port might try to output into the output of a device connected to same inputs might destroy one or both of them.

# PORTs A, B and C:

The 8255A contains three 8-bit ports (A, B and C). All can be configured in a variety of functional characteristic by the system software.

# PORTA:

One 8-bit data output latch/buffer and one 8-bit data input latch.

# PORT B:

One 8-bit data output latch/buffer and one 8-bit data input buffer.

# PORT C:

One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signals inputs in conjunction with ports A and B.

# Group A & Group B control:

The functional configuration of each port is programmed by the system software. The control words outputted by the CPU configure the associated ports of the each of the two groups. Each control block accepts command from Read/Write content logic receives control words from the internal data bus and issues proper commands to its associated ports.

Control Group A – Port A & Port C upper Control Group B – Port B & Port C lower

The control word register can only be written into No read operation if the control word register is allowed.

# Operation Description:

**Mode selection:**

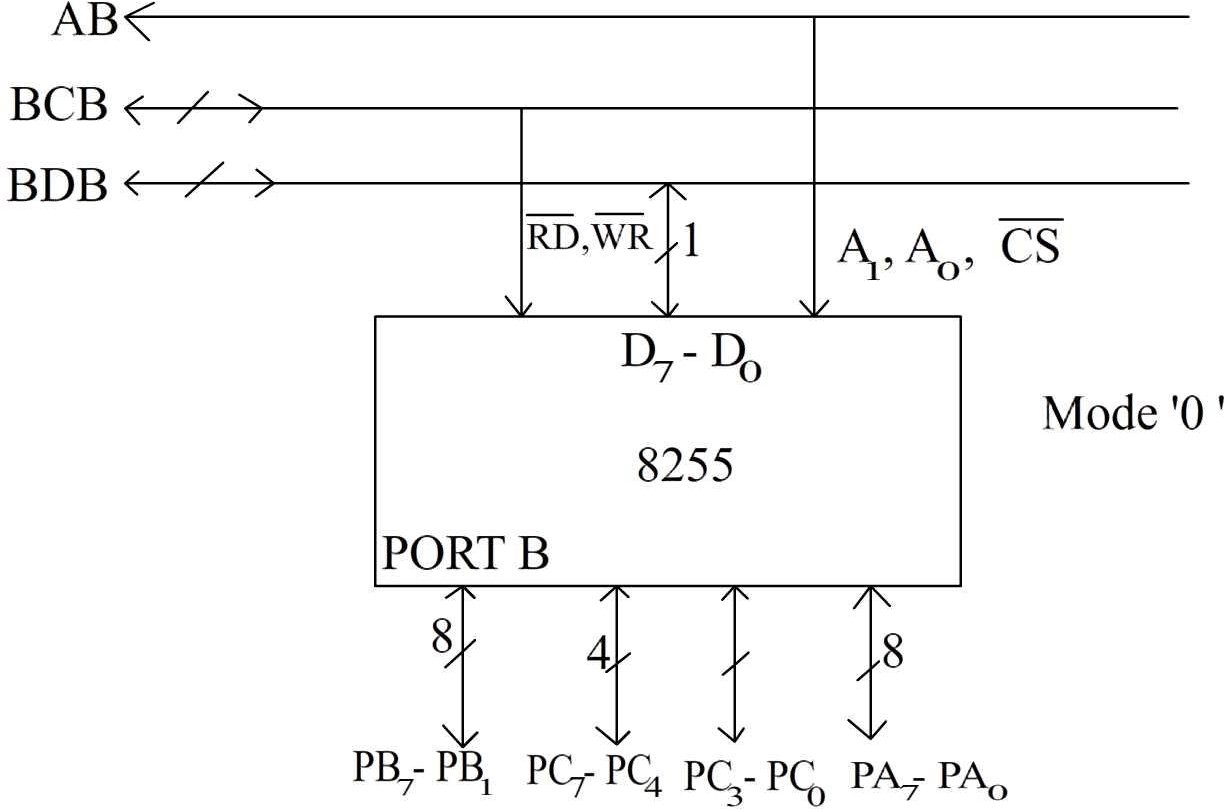
There are three basic modes of operation that can be selected by the system software.

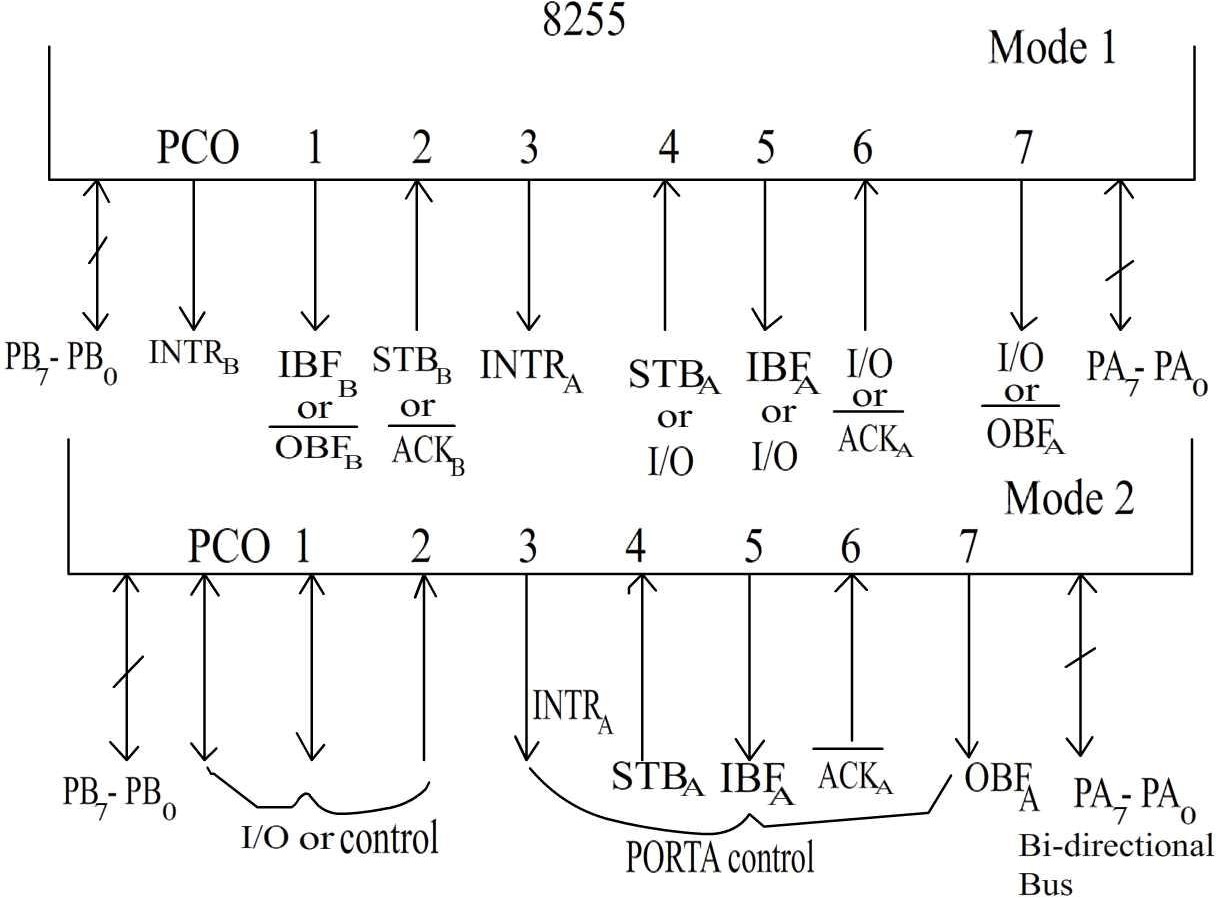
Mode 0: Basic Input/output Mode 1: Strobes Input/output Mode 2: Bi-direction bus.

When the reset input goes HIGH all poets are set to mode’0’ as input which means all 24 lines are in high impedance state and can be used as normal input. After the reset is removed the 8255A remains in the input mode with no additional initialization. During the execution of the program any of the other modes may be selected using a single output instruction.

The modes for PORT A & PORT B can be separately defined, while PORT C is divided into two portions as required by the PORT A and PORT B definitions. The ports are thus divided into two groups Group A & Group B. All the output register, including the status flip-flop will be reset whenever the mode is changed. Modes of the two group may be combined for any desired I/O operation e.g. Group A in mode ‘1’ and group B in mode ‘0’.

The basic mode definitions with bus interface and the mode definition format are given in fig (a) & (b),





### 8086 Microprocessor Architecture and Operation:

It is a 16 bit µp. 8086 has a 20 bit address bus can access upto 220 memory locations ( 1 MB) . It can support upto 64K I/O ports. It provides 14, 16-bit registers. It has multiplexed address and data bus AD0- AD15 and A16 – A19. It requires single phase clock with 33% duty cycle to provide internal timing. 8086 is designed to operate in two modes, Minimum and Maximum. It can prefetches upto 6 instruction bytes from memory and queues them in order to speed up instruction execution. It requires +5V power supply. A 40 pin dual in line package.

### Minimum and Maximum Modes:

The minimum mode is selected by applying logic 1 to the MN / MX# input pin. This is a single microprocessor configuration. The maximum mode is selected by applying logic 0 to the MN / MX# input pin. This is a multi micro processors configuration.

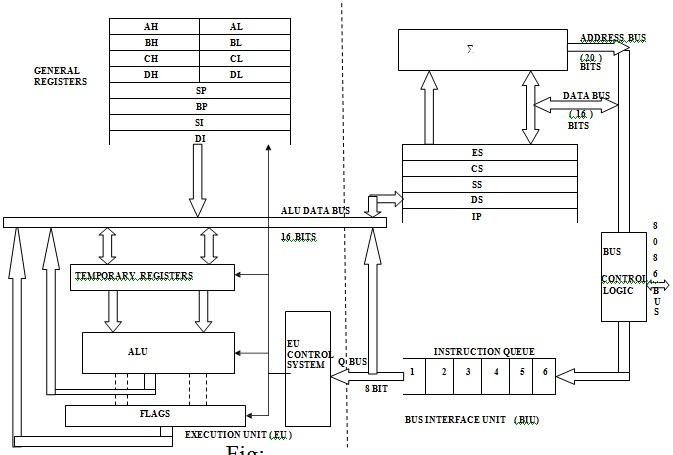
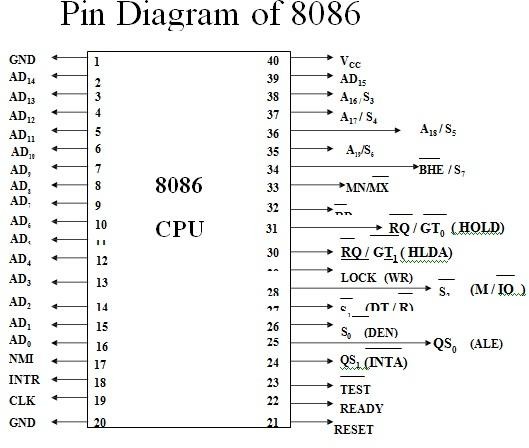


Fig. Architecture of 8086



Internal Architecture of 8086

8086 has two blocks BIU and EU. The BIU performs all bus operations such as instruction fetching, reading and writing operands for memory and calculating the addresses of the memory operands. The instruction bytes are transferred to the instruction queue. EU executes instructions from the instruction system byte queue. Both units operate asynchronously to give the 8086 an overlapping instruction fetch and execution mechanism which is called as Pipelining. This results in efficient use of the system bus and system performance. BIU contains Instruction queue, Segment registers, Instruction pointer, Address adder. EU contains Control circuitry, Instruction decoder, ALU, Pointer and Index register, Flag register.

Bus Interfacr Unit**:**

It provides a full 16 bit bidirectional data bus and 20 bit address bus. The bus interface unit is responsible for performing all external bus operations.

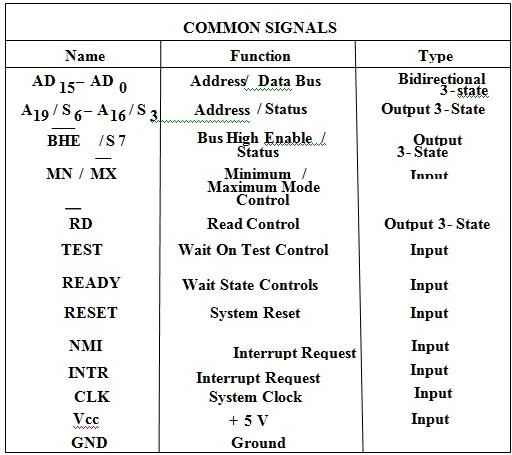
***Specifically it has the following functions***:

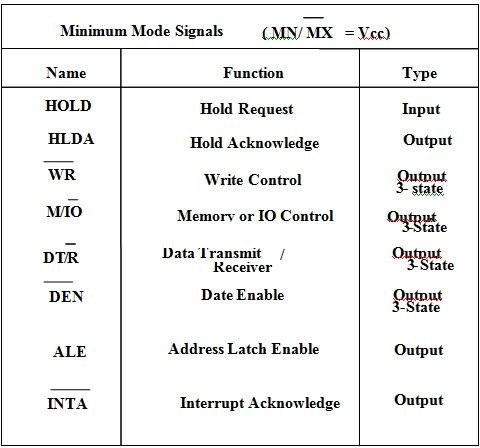
Instruction fetch, Instruction queuing, Operand fetch and storage, Address relocation and Bus control. The BIU uses a mechanism known as an instruction stream queue to implement a ***pipeline architecture.***

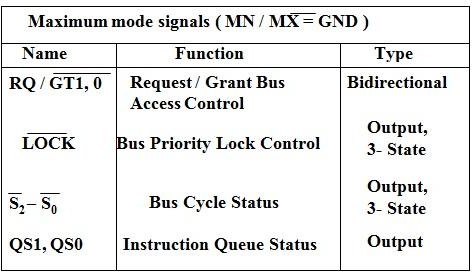
This queue permits prefetch of up to six bytes of instruction code. When ever the queue of the BIU is not full, it has room for at least two more bytes and at the same time the EU is not requesting it to read or write operands from memory, the BIU is free to look ahead in the program by prefetching the next sequential instruction. These prefetching instructions are held in its FIFO queue. With its 16 bit data bus, the BIU fetches two instruction bytes in a single memory cycle. After a byte is loaded at the input end of the queue, it automatically shifts up through the FIFO to the empty location nearest the output.

The EU accesses the queue from the output end. It reads one instruction byte after the other from the output of the queue. If the queue is full and the EU is not requesting access to operand in memory. These intervals of no bus activity, which may occur between bus cycles are known as ***Idle state****.* If the BIU is already in the process of fetching an instruction when the EU request it to read or write operands from memory or I/O, the BIU first completes the instruction fetch bus cycle before initiating the operand read / write cycle. The BIU also contains a dedicated adder which is used to generate the 20 bit physical address that is output on the address bus. This address is formed by adding an appended 16 bit segment address and a 16 bit offset address. For example, the physical address of the next instruction to be fetched is formed by combining the current contents of the code segment CS register and the current contents of the instruction pointer IP register. The BIU is also responsible for generating bus control signals such as those for memory read or write and I/O read or write.

**EXECUTION UNIT** : The Execution unit is responsible for decoding and executing all instructions. The EU extracts instructions from the top of the queue in the BIU, decodes them, generates operands if necessary, passes them to the BIU and requests it to perform the read or write bys cycles to memory or I/O and perform the operation specified by the instruction on the operands. During the execution of the instruction, the EU tests the status and control flags and updates them based on the results of executing the instruction. If the queue is empty, the EU waits for the next instruction byte to be fetched and shifted to top of the queue. When the EU executes a branch or jump instruction, it transfers control to a location corresponding to another set of sequential instructions. Whenever this happens, the BIU automatically resets the queue and then begins to fetch instructions from this new location to refill the queue.







Internal Registers of 8086

The 8086 has four groups of the user accessible internal registers. They are the instruction pointer, four data registers, four pointer and index register, four segment registers.

The 8086 has a total of fourteen 16-bit registers including a 16 bit register called the ***status register***, with 9 of bits implemented for status and control flags. Most of the registers contain data/instruction offsets within 64 KB memory segment. There are four different 64 KB segments for instructions, stack, data and extra data. To specify where in 1 MB of processor memory these 4 segments are located the processor uses four segment registers:

**Code segment** (CS) is a 16-bit register containing address of 64 KB segment with processor instructions. The processor uses CS segment for all accesses to instructions referenced by instruction pointer (IP) register. CS register cannot be changed directly. The CS register is automatically updated during far jump, far call and far return instructions.

**Stack segment** (SS) is a 16-bit register containing address of 64KB segment with program stack. By default, the processor assumes that all data referenced by the stack pointer (SP) and base pointer (BP) registers is located in the stack segment. SS register can be changed directly using POP instruction.

**Data segment** (DS) is a 16-bit register containing address of 64KB segment with program data. By default, the processor assumes that all data referenced by general registers (AX, BX, CX, DX) and index register (SI, DI) is located in the data segment. DS register can be changed directly using POP and LDS instructions.

**Extra segment** (ES) is a 16-bit register containing address of 64KB segment, usually with program data. By default, the processor assumes that the DI register references the ES segment in string manipulation instructions. ES register can be changed directly using POP and LES instructions. It is possible to change default segments used by general and index registers by prefixing instructions with a CS, SS, DS or ES prefix.

All general registers of the 8086 microprocessor can be used for arithmetic and logic operations. The general registers are:

**Accumulator** register consists of two 8-bit registers AL and AH, which can be combined together and used as a 16-bit register AX. AL in this case contains the low-order byte of the word, and AH contains the high-order byte. Accumulator can be used for I/O operations and string manipulation.

**Base** register consists of two 8-bit registers BL and BH, which can be combined together and used as a 16-bit register BX. BL in this case contains the low-order byte of the word, and BH contains the high-order byte. BX register usually contains a data pointer used for based, based indexed or register indirect addressing.

**Count** register consists of two 8-bit registers CL and CH, which can be combined together and used as a 16-bit register CX. When combined, CL register contains the low-order byte of the word, and CH contains the high-order byte. Count register can be used in Loop, shift/rotate instructions and as a counter in string manipulation,.

**Data** register consists of two 8-bit registers DL and DH, which can be combined together and used as a 16-bit register DX. When combined, DL register contains the low-order byte of the word, and DH contains the high- order byte. Data register can be used as a port number in I/O operations. In integer 32-bit multiply and divide instruction the DX register contains high-order word of the initial or resulting number.

The following registers are both general and index registers:

**Stack Pointer** (SP) is a 16-bit register pointing to program stack.

**Base Pointer** (BP) is a 16-bit register pointing to data in stack segment. BP register is usually used for based, based indexed or register indirect addressing.

**Source Index** (SI) is a 16-bit register. SI is used for indexed, based indexed and register indirect addressing, as well as a source data address in string manipulation instructions.

**Destination Index** (DI) is a 16-bit register. DI is used for indexed, based indexed and register indirect addressing, as well as a destination data address in string manipulation instructions.

Other registers:

**Instruction Pointer** (IP) is a 16-bit register.

**Flags** is a 16-bit register containing 9 one bit flags.

**Overflow Flag** (OF) - set if the result is too large positive number, or is too small negative number to fit into destination operand.

**Direction Flag** (DF) - if set then string manipulation instructions will auto-decrement index registers. If cleared then the index registers will be auto-incremented.

**Interrupt-enable Flag** (IF) - setting this bit enables maskable interrupts.

**Single-step Flag** (TF) - if set then single-step interrupt will occur after the next instruction.

**Sign Flag** (SF) - set if the most significant bit of the result is set.

**Zero Flag** (ZF) - set if the result is zero

**Auxiliary carry Flag** (AF) - set if there was a carry from or borrow to bits 0-3 in the AL register.

**Parity Flag** (PF) - set if parity (the number of "1" bits) in the low-order byte of the result is even.

**Carry Flag** (CF) - set if there was a carry from or borrow to the most significant bit during last result calculation.

### Addressing Modes

**Implied** - the data value/data address is implicitly associated with the instruction.

**Register** - references the data in a register or in a register pair.

**Immediate** - the data is provided in the instruction.

**Direct** - the instruction operand specifies the memory address where data is located.

**Register indirect** - instruction specifies a register containing an address, where data is located. This addressing mode works with SI, DI, BX and BP registers.

**Based** :- 8-bit or 16-bit instruction operand is added to the contents of a base register (BX or BP), the resulting value is a pointer to location where data resides.

**Indexed** :- 8-bit or 16-bit instruction operand is added to the contents of an index register (SI or DI), the resulting value is a pointer to location where data resides.

**Based Indexed** :- the contents of a base register (BX or BP) is added to the contents of an index register (SI or DI), the resulting value is a pointer to location where data resides.

**Based Indexed with displacement** :- 8-bit or 16-bit instruction operand is added to the contents of a base register (BX or BP) and index register (SI or DI), the resulting value is a pointer to location where data resides.

### Interrupts

The processor has the following interrupts:

**INTR** is a maskable hardware interrupt. The interrupt can be enabled/disabled using STI/CLI instructions or using more complicated method of updating the FLAGS register with the help of the POPF instruction.

When an interrupt occurs, the processor stores FLAGS register into stack, disables further interrupts, fetches from the bus one byte representing interrupt type, and jumps to interrupt processing routine address of which is stored in location 4 \* <interrupt type>. Interrupt processing routine should return with the IRET instruction.

**NMI** is a non-maskable interrupt. Interrupt is processed in the same way as the INTR interrupt. Interrupt type of the NMI is 2, i.e. the address of the NMI processing routine is stored in location 0008h. This interrupt has higher priority then the maskable interrupt.

**Software interrupts** can be caused by:

INT instruction - breakpoint interrupt. This is a type 3 interrupt.

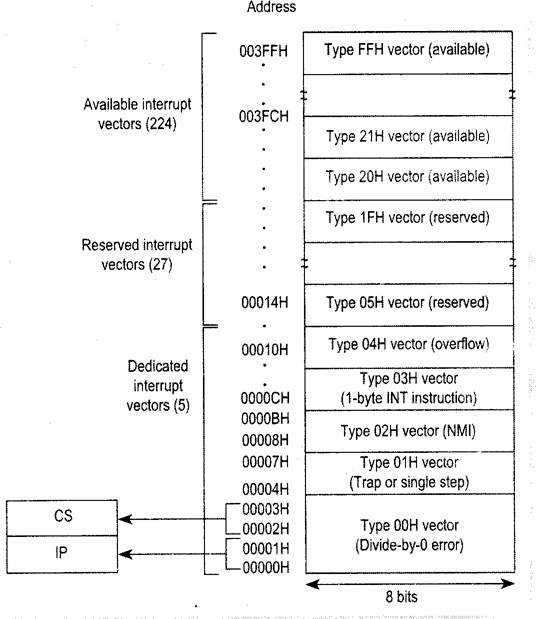
INT <interrupt number> instruction - any one interrupt from available 256 interrupts. INTO instruction - interrupt on overflow

Single-step interrupt - generated if the TF flag is set. This is a type 1 interrupt. When the CPU processes this interrupt it clears TF flag before calling the interrupt processing routine.

**Processor exceptions**: Divide Error (Type 0), Unused Opcode (type 6) and Escape opcode (type 7).

Software interrupt processing is the same as for the hardware interrupts.

The figure below shows the 256 interrupt vectors arranged in the interrupt vector table in the memory.



Interrupt Vector Table in the 8086

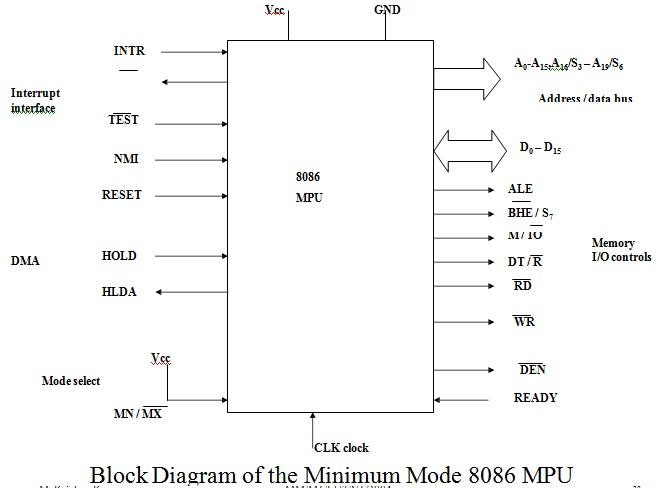
### Minimum Mode Interface

When the Minimum mode operation is selected, the 8086 provides all control signals needed to implement the memory and I/O interface. The minimum mode signal can be divided into the following basic groups : address/data bus, status, control, interrupt and DMA.

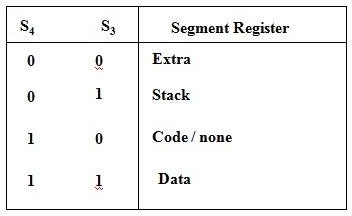
**Address/Data Bus** : these lines serve two functions. As an address bus is 20 bits long and consists of signal lines A0 through A19. A19 represents the MSB and A0 LSB. A 20bit address gives the 8086 a 1Mbyte memory address space. More over it has an independent I/O address space which is 64K bytes in length.

The 16 data bus lines D0 through D15 are actually multiplexed with address lines A0 through A15 respectively. By multiplexed we mean that the bus work as an address bus

during first machine cycle and as a data bus during next machine cycles. D15 is the MSB and D0 LSB. When acting as a data bus, they carry read/write data for memory, input/output data for I/O devices, and interrupt type codes from an interrupt controller.



**Status signal** : The four most significant address lines A19 through A16 are also multiplexed but in this case with status signals S6 through S3. These status bits are output on the bus at the same time that data are transferred over the other bus lines. Bit S4 and S3 together from a 2 bit binary code that identifies which of the 8086 internal segment registers are used to generate the physical address that was output on the address bus during the current bus cycle. Code S4S3 = 00 identifies a register known as ***extra segment register*** as the source of the segment address.



Memory segment status code

Status line S5 reflects the status of another internal characteristic of the 8086. It is the logic level of the internal enable flag. The last status bit S6 is always at the logic 0 level.