D 1100CST307122101 Pages: 2

Reg No.:_______________ Name:__________________________

# APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Fifth Semester B.Tech Degree Examination December 2021 (2019 scheme)

**Course Code: CST 307**

**Course Name: MICROPROCESSORS AND MICROCONTROLLERS**

Max. Marks: 100 Duration: 3 Hours

## PART A

***(Answer all questions; each question carries 3 marks)***

| | | | Marks |
|---|---|---|---|
| 1 | | List features of 8085 microprocessor. | 3 |
| 2 | | The value of Code Segment (CS) Register is 3054H and the value of different registers is as follows: BX: 4025H , IP: 1580H , DI: 5467H. Calculate the physical address of the next instruction to be fetched. | 3 |
| 3 | | State the significance of assembler directives in assembly language program and provide two examples for it. | 3 |
| 4 | | List the 8086 instructions used for transferring data between registers, memory, stack, and I/O devices. | 3 |
| 5 | | Explain how the INT n instruction finds the starting address of its interrupt service routine in IVT. | 3 |
| 6 | | Classify various categories of interrupts available in 8086. | 3 |
| 7 | | Interpret the mode and configurations of 8255 after its control word register is loaded with 86H. | 3 |
| 8 | | Explain the features of 8257 DMA controller. | 3 |
| 9 | | Differentiate between Microprocessors and Microcontrollers. | 3 |
| 10 | | List the IO ports available in 8051. | 3 |

## PART B

***(Answer one full question from each module, each question carries 14 marks)***

### Module -1

| | | | |
|---|---|---|---|
| 11 | a) | Draw and explain the internal architecture of 8086. | 10 |
| | b) | Give the architectural and signal differences between 8086 and 8088. | 4 |
| 12 | a) | Draw the Memory Read and Write timing diagrams of 8086 in Minimum mode. | 9 |
| | b) | Draw the structure of 8086 flag register and mention the purpose of each flag. | 5 |

**Module -2**

| | | | |
|---|---|---|---|
| 13 | a) | Discuss addressing modes supported by 8086 with suitable examples. | 9 |
| | b) | Discuss about the data transfer instructions with examples. | 5 |
| 14 | a) | Assume that 8086 registers having values AX=0030H, BX = 0031H, CX=0032H, DX=0033h, Flag – 0000H.<br>Predict the values of Registers and Flags [ AX, BX, CX, DX, Carry flag (CF), Zero Flag (ZF), Sign Flag (SF) ] after the execution of following instructions: (Assume each instruction are being executed independently)<br>i) ROR AX,04h ii) CMP BX, CX iii) XCHG CX, DX iv) AND AX, BX<br>v) LOOP Addr vi) XOR AX, AX vii) STC<br>Hint – Draw a table with columns *Instructions, AX, BX, CX, DX, CF, ZF, SF* and fill the answers. | 7 |
| | b) | Write an 8086-program to find the largest among 'n' numbers (each numbers and count are of one byte only).<br>Kindly assume that the size of array(count) stored in 2000h, and the numbers(array) stored from 2001h onwards up to 'n' continues locations. | 7 |

**Module -3**

| | | | |
|---|---|---|---|
| 15 | a) | Explain the stack structure of 8086. | 4 |
| | b) | Interface two 32K X 8 EPROMS and two 32K X 8 RAM chips with 8086, microprocessor and draw the suitable circuit showing their interfacing. | 10 |
| 16 | a) | Draw and explain the internal architecture of 8259. | 8 |
| | b) | State the purpose of Interrupt Vector Table of 8086 and explain its structure. | 6 |

**Module -4**

| | | | |
|---|---|---|---|
| 17 | a) | Explain the 8254 programmable timer and its operation modes with a neat block diagram. | 9 |
| | b) | Explain different modes of operation of 8255 PPI. | 5 |
| 18 | a) | With a neat diagram describe the architecture of 8255 PPI. | 8 |
| | b) | Give the registers available in 8257 DMA Controller. Explain their functions. | 6 |

**Module -5**

| | | | |
|---|---|---|---|
| 19 | a) | Explain the Internal RAM organization of 8051 with neat diagram. | 8 |
| | b) | List any four addressing modes supported by 8051 microcontrollers with one example each. | 6 |
| 20 | a) | Explain internal architecture of 8051 with neat diagram. | 9 |
| | b) | State the name and purpose of any 6 special function registers (SFRs) of 8051 microcontroller. | 5 |

*****

## PART A

1.

- 8-bit general purpose microprocessor(µp)
- Capable of addressing 64 kb of memory
- Has 40 pins
- Requires +5 v power supply
- Can operate with 3 MHz clock
- 8bit data bus
- 16 bit address bus

2. The offset of the CS Register is the IP register.

Therefore, the effective address of the memory location pointed by the CS register is calculated as follows:

Effective address= Base address of CS register X $10_H$ + Address of IP

= $3054_H$ X $10_H$ + $1580_H$
= $(30540 + 1580)_H$
= $31AC0_H$

3. Assembler directives help the assembler to correctly understand the assembly language programs to prepare the codes. Another type of hint which helps the assembler to assign a particular constant with a label or initialize particular memory locations or labels with constants is called an operator. Rather, the operators perform the arithmetic and logical tasks unlike directives that just direct the assembler to correctly interpret the program to code it appropriately. The following directives are commonly used in the assembly language programming practice using Microsoft Macro Assembler (MASM).

**DB:** Define Byte The DB directive is used to reserve byte or bytes of memorylocations in the available memory. LIST DB 0lH, 02H, 03H, 04H This statement directs the assembler to reserve four memory locations for a list named LIST andinitialize them with the above specified four values.

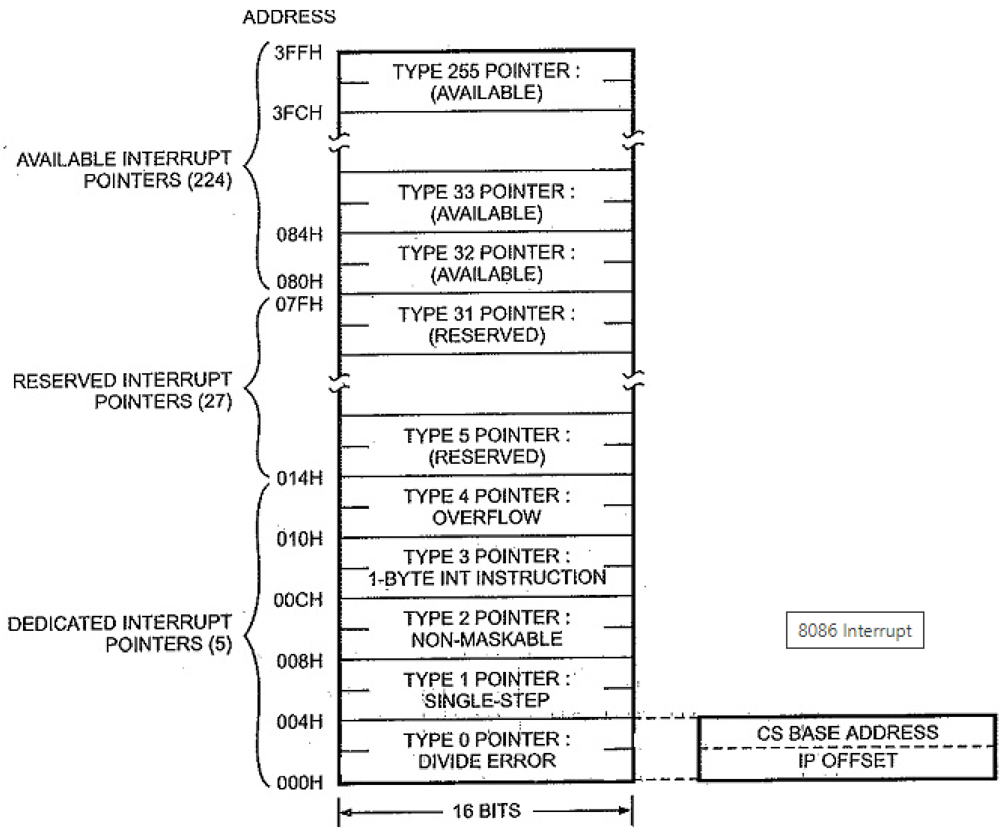
**ASSUME:** Assume Logical Segment Name The ASSUME directive is used to inform the assembler, the names of the logical segments to be assumed for different segments used in the program. In the assembly language program, each segment is given aname. For example, the code segment may be given the name CODE, data segment may be given the name DATA etc.

4.

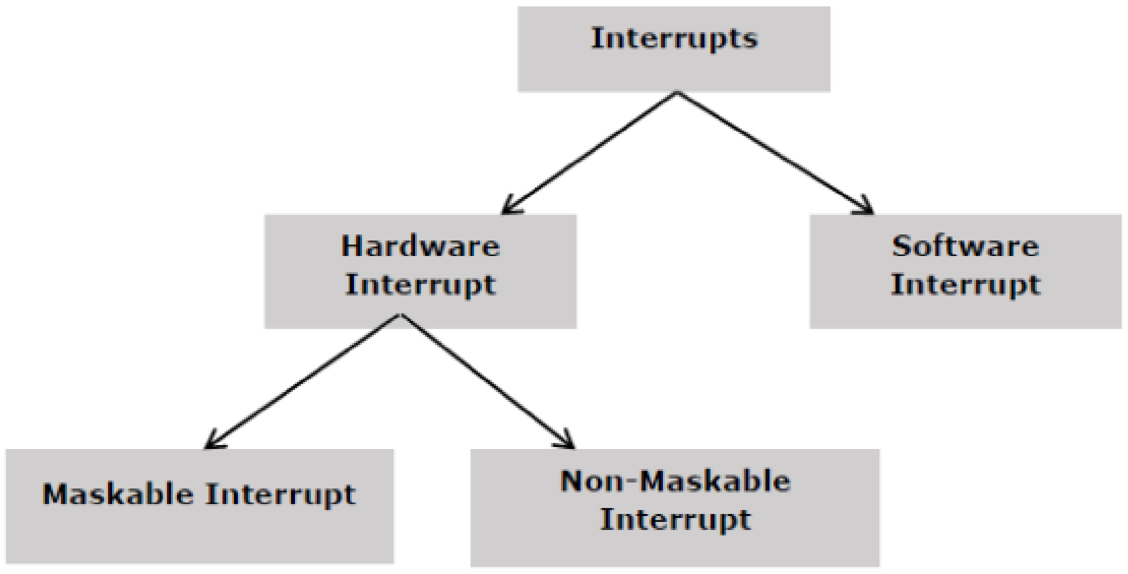
1. MOV: copy byte or word from specified source to specified destination.
2. PUSH: copy specified word to top of stack.
3. POP: copy word from top of stack to specified location.
4. PUSHA: copy all registers to stack.
5. POPA: copy words from stack to all registers.

5. There are 256 software interrupts in **the** 8086 microprocessor. **The instructions** are of **the** format **INT** type, where **the** type ranges from 00 to FF. Each interrupt type is given a number between 0 to 255 and the address of each interrupt js found by multiplying the type by 4 e.g. for type 11, interrupt address is 11 x 4 = $44_{10}$= 0002CH

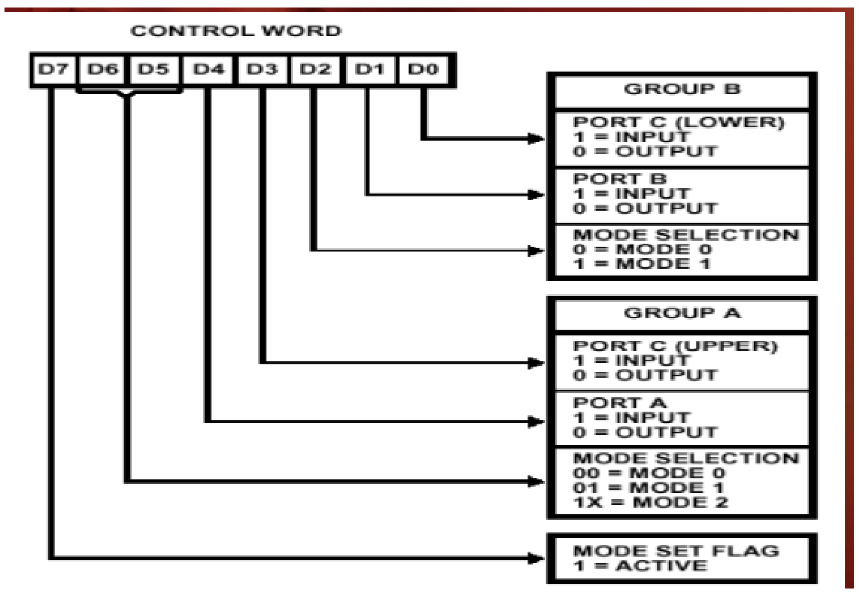
Only first five types have explicit definitions such as divide by zero and non maskable interrupt. The next 27 interrupt types, from 5 to 31, are reserved by Intel for use in future microprocessors. The upper 224 interrupt types, from 32 to 255, are available for user for hardware or software interrupts.


ADDRESS
3FFH
3FCH
084H
080H
07FH
014H
010H
00CH
008H
004H
000H
AVAILABLE INTERRUPT POINTERS (224)
RESERVED INTERRUPT POINTERS (27)
DEDICATED INTERRUPT POINTERS (5)
TYPE 255 POINTER : (AVAILABLE)
TYPE 33 POINTER : (AVAILABLE)
TYPE 32 POINTER : (AVAILABLE)
TYPE 31 POINTER : (RESERVED)
TYPE 5 POINTER : (RESERVED)
TYPE 4 POINTER : OVERFLOW
TYPE 3 POINTER : 1-BYTE INT INSTRUCTION
TYPE 2 POINTER : NON-MASKABLE
TYPE 1 POINTER : SINGLE-STEP
TYPE 0 POINTER : DIVIDE ERROR
CS BASE ADDRESS
IP OFFSET
16 BITS
8086 Interrupt

6. Categories of Interrupts



7.



**There are two basic operational modes of 8255:**

- Bit Set/Reset mode (BSR mode).

  This mode is used to set or reset the bits of port C only, and selected when the most significant bit (D7) in the control register is 0.
- Input/Output mode (I/O mode).

  This mode is selected when the most significant bit (D7) in the control register is 1. It has three operating modes: mode 0, mode 1, and mode 2.

Mode 0 is the basic input/output mode. In this mode, the 8255 can be used as an 8-bit input or output device, with each of the three ports (Port A, Port B, and Port C) being configured individually as input or output.

Mode 1 is the strobed input/output mode. This mode is similar to mode 0, but includes additional control signals to enable latching of input data and/or output data.
Mode 2 is the bidirectional mode. In this mode, Port A and Port B are configured as bidirectional I/O ports, while Port C is configured as a control port for setting various operational modes.

8. **Features of 8257**
The 8257 is a direct memory access (DMA) controller that is designed to improve the data transfer rate between external devices and the microprocessor. Some of the key features of the 8257 microprocessor are:

1. **Four independent DMA channels**: The 8257 has four independent DMA channels, which can transfer data to or from four different devices simultaneously.

2. **Address and data buses**: The 8257 has its own address and data buses, which allow it to transfer data directly to and from memory without involving the microprocessor.

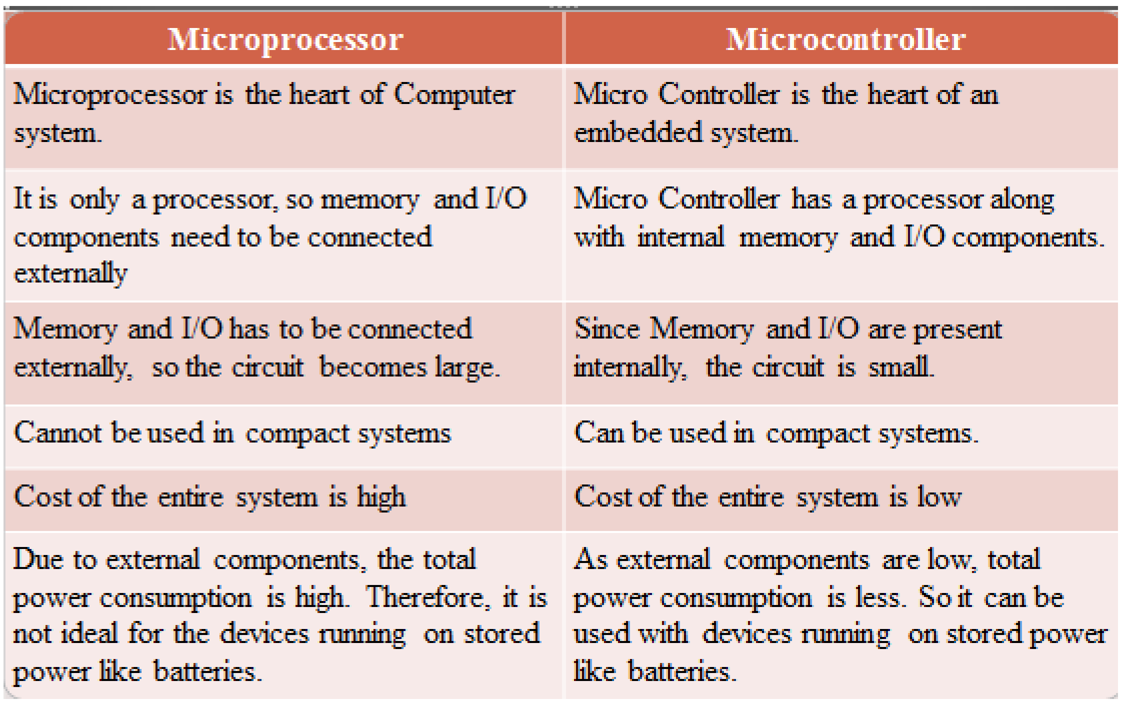
3. **Burst mode transfer**: The 8257 supports burst mode transfer, allowing it to transfer data blocks quickly and efficiently.

4. **Priority schemes**: The 8257 supports different priority schemes for each DMA channel, which allows the user to assign priorities to different devices based on their importance.

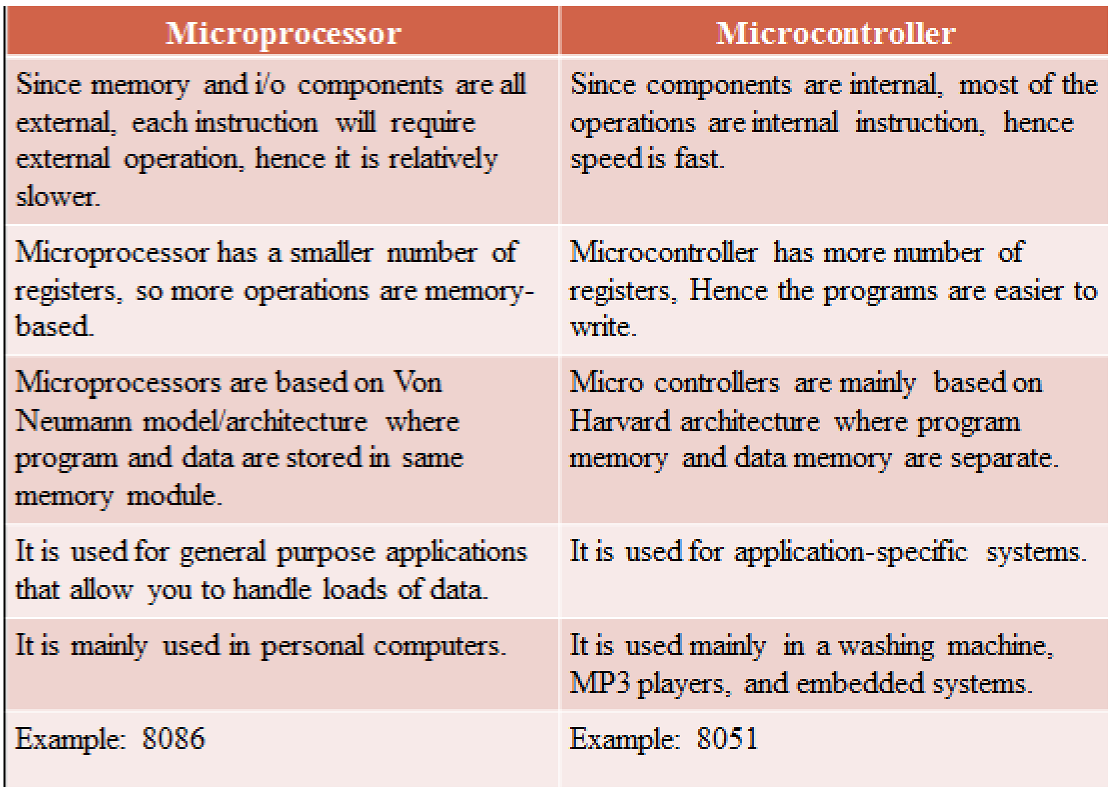
5. **Auto-initialization**: The 8257 supports auto-initialization, which allows it to automatically reload the DMA address and count registers after each data transfer.

6. **Compatible with different microprocessors**: The 8257 is compatible with a wide range of microprocessors, including the 8085, 8086, and Z80.

9.



| Microprocessor | Microcontroller |
|---|---|
| Microprocessor is the heart of Computer system. | Micro Controller is the heart of an embedded system. |
| It is only a processor, so memory and I/O components need to be connected externally | Micro Controller has a processor along with internal memory and I/O components. |
| Memory and I/O has to be connected externally, so the circuit becomes large. | Since Memory and I/O are present internally, the circuit is small. |
| Cannot be used in compact systems | Can be used in compact systems. |
| Cost of the entire system is high | Cost of the entire system is low |
| Due to external components, the total power consumption is high. Therefore, it is not ideal for the devices running on stored power like batteries. | As external components are low, total power consumption is less. So it can be used with devices running on stored power like batteries. |



| Microprocessor | Microcontroller |
|---|---|
| Since memory and i/o components are all external, each instruction will require external operation, hence it is relatively slower. | Since components are internal, most of the operations are internal instruction, hence speed is fast. |
| Microprocessor has a smaller number of registers, so more operations are memory-based. | Microcontroller has more number of registers, Hence the programs are easier to write. |
| Microprocessors are based on Von Neumann model/architecture where program and data are stored in same memory module. | Micro controllers are mainly based on Harvard architecture where program memory and data memory are separate. |
| It is used for general purpose applications that allow you to handle loads of data. | It is used for application-specific systems. |
| It is mainly used in personal computers. | It is used mainly in a washing machine, MP3 players, and embedded systems. |
| Example: 8086 | Example: 8051 |

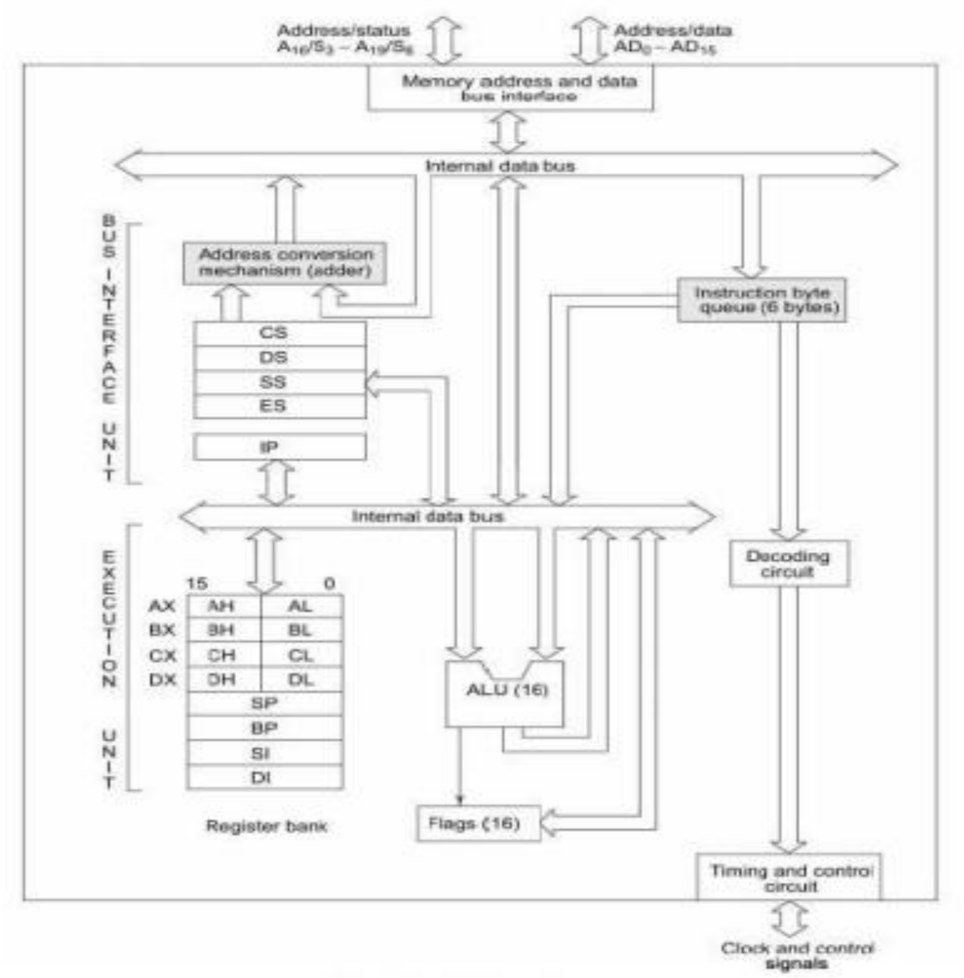
10. The 8051 has four important ports: Port 0, Port 1, Port 2 and Port 3.

- These ports allow the microcontroller to connect with the outside world.
- Each port has 8 bits. Thus the four ports jointly comprise 32 pins.
- All ports are bidirectional.

## PART B

### Module 1

11 a)

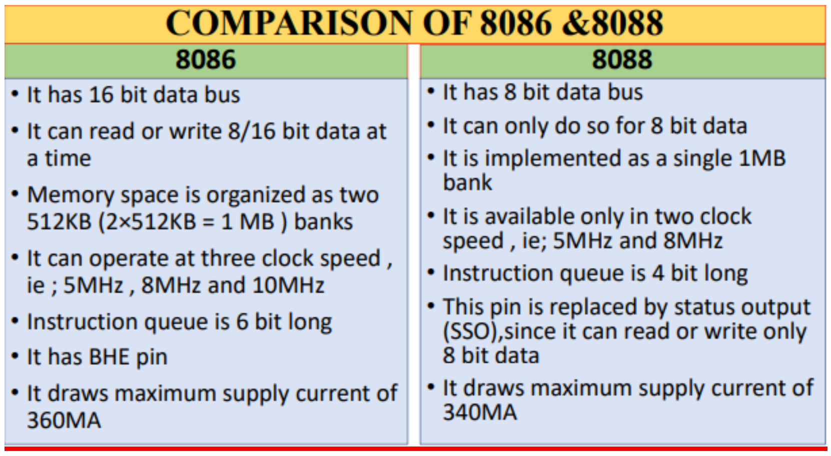


Address/status $A_{16}/S_3 - A_{19}/S_6$
Address/data $AD_0 - AD_{15}$
Memory address and data bus interface
Internal data bus
BUS INTERFACE UNIT
Address conversion mechanism (adder)
Instruction byte queue (6 bytes)
CS
DS
SS
ES
IP
Internal data bus
EXECUTION UNIT
15
0
AX
BX
CX
DX
AH
AL
BH
BL
CH
CL
DH
DL
SP
BP
SI
DI
Register bank
ALU (16)
Flags (16)
Decoding circuit
Timing and control circuit
Clock and control signals

**Fig. 1.2** *8086 Architecture*

Explain each components(General purpose registers,Segment registers,flag register,Execution unit,Bus interface unit,instruction queueaddress bus,data bus,control bus etc.Refer notes).

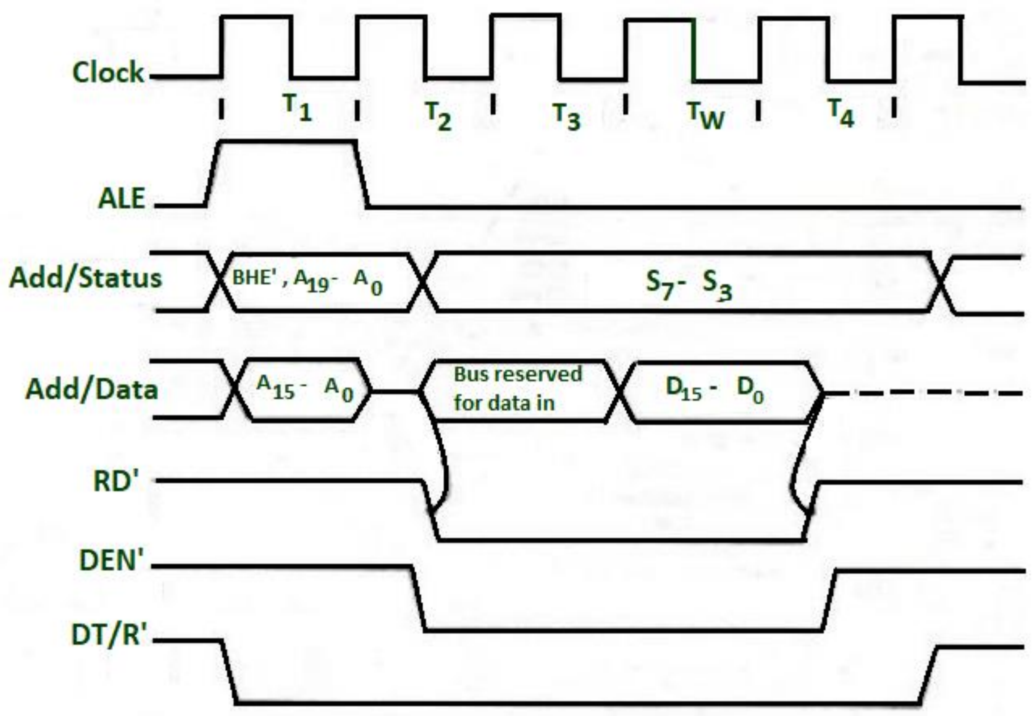
b)Architecture and signal differences between 8086 and 8088



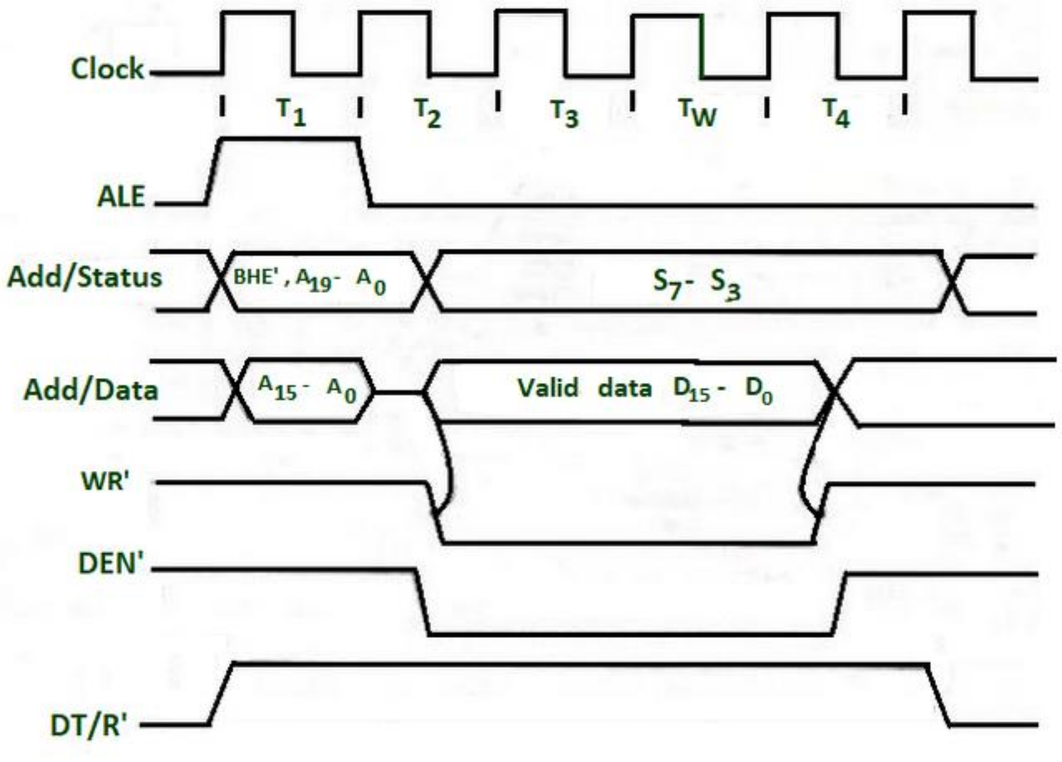
**COMPARISON OF 8086 &8088**

| 8086 | 8088 |
|---|---|
| • It has 16 bit data bus | • It has 8 bit data bus |
| • It can read or write 8/16 bit data at a time | • It can only do so for 8 bit data |
| • Memory space is organized as two 512KB (2×512KB = 1 MB ) banks | • It is implemented as a single 1MB bank |
| • It can operate at three clock speed , ie ; 5MHz , 8MHz and 10MHz | • It is available only in two clock speed , ie; 5MHz and 8MHz |
| • Instruction queue is 6 bit long | • Instruction queue is 4 bit long |
| • It has BHE pin | • This pin is replaced by status output (SSO),since it can read or write only 8 bit data |
| • It draws maximum supply current of 360MA | • It draws maximum supply current of 340MA |

12a)



*Opcode fetch or read timing diagram*



12b)

The flag register is one of the special purpose register. 8086 has 16-bit flag register, and there are 9 valid flag bits. The format of flag register is like below.

| **Bits** | **$D_{15}$** | **$D_{14}$** | **$D_{13}$** | **$D_{12}$** | **$D_{11}$** | **$D_{10}$** | **$D_9$** | **$D_8$** | **$D_7$** | **$D_6$** | **$D_5$** | **$D_4$** | **$D_3$** | **$D_2$** | **$D_1$** | **$D_0$** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Flags** | | | | | O | D | I | T | S | Z | | AC | | P | | CY |

We can divide the flag bits into two sections. The Status Flags, and the Control Flags.

**Status Flags**

In 8086 there are 6 different flags which are set or reset after 8-bit or 16-bit operations. These flags and their functions are listed below.

| **Flag Bit** | **Function** |
|---|---|
| **S** | After any operation if the MSB is 1, then it indicates that the number is negative. And this flag is set to 1 |
| **Z** | If the total register is zero, then only the Z flag is set |
| **AC** | When some arithmetic operations generates carry after the lower half and sends it to upper half, the AC will be 1 |
| **P** | This is even parity flag. When result has even number of 1, it will be set to 1, otherwise 0 for odd number of 1s |
| **CY** | This is carry bit. If some operations are generating carry after the operation this flag is set to 1 |
| **O** | The overflow flag is set to 1 when the result of a signed operation is too large to fit. |

**Control Flags**

In 8086 there are 3 different flags which are used to enable or disable some basic operations of the microprocessor. These flags and their functions are listed below.

| **Flag Bit** | **Function** |
|---|---|
| **D** | This is directional flag. This is used in string related operations. D = 1, then the string will be accessed from higher memory address to lower memory address, and if D = 0, it will do the reverse. |
| **I** | This is interrupt flag. If I = 1, then MPU will recognize the interrupts from peripherals. For I = 0, the interrupts will be ignored |
| **T** | This trap flag is used for on-chip debugging. When T = 1, it will work in a single step mode. After each instruction, one internal interrupt is generated. It helps to execute some program instruction by instruction. |

## Module 2

**13a)**

**1 Immediate:**

•In this type of addressing, immediate data is a part of instruction, and appears in the form of successive byte or bytes

•Eg: MOV AX, 0005H

**2. Direct:**

•In the direct addressing mode, a 16-bit memory address (offset) is directly specified in the instruction as a part of it.

•Eg: MOV AX,[5000H],

–Effective address= 10H*DS +5000H

**3. Register:**

•In the register addressing mode, the data is stored in a register and it is referred using the particular register

•All the registers, except IP, may be used in this mode.

•Eg: MOV AX, BX

**4 .Register Indirect:**

•Sometimes, the address of the memory location which contains data or operand is determined in an indirect way, using the offset registers.

•This mode of addressing is known as register indirect mode

•In this addressing mode, the offset address of data is in either BX or SI or DI register.

•The default segment is either DS or ES. The data is supposed to be available at the address pointed to by the content of any of the above registers in the default data segment.

•Eg: MOV AX,[BX]

–Effective address is 10H*DS+[BX]

**5. Indexed:**

•In this addressing mode, offset of the operand is stored in one of the Index registers.

•DS is the default segment for index registers SI and DI

•In the case of string instructions DS and ES are default segments for SI and DI respectively.

•This mode is a special case of the above discussed register indirect addressing mode

•Eg: MOV AX,[SI]

–effective address is 10H*DS+[SI]

**6. Register Relative:**

•In this addressing mode, the data is available at an effective address formed by adding an 8-bit or 16-bit displacement with the content of any one of the registers BX, BP, SI and DI in the default (either DS or ES) segment.

•Eg: MOV AX,50H[BX]

–Effective address is 10H*DS+50H+[BX]

**7. Based Indexed:**

•The effective address of the data is formed, in this addressing mode, by adding the content of a base register (any one of BX or BP) to the content of an index register (any one of SI or DI)
•The default segment register may be DS or ES
–Eg: MOV AX,[BX][SI]
•effective address is 10H*DS +[BX]+[SI]

**8. Relative Based Indexed:**

•The effective address is formed by adding an 8-bit or 16-bit displacement with the sum of contents of anyone of the base registers (BX or BP) and any one of the index registers (SI or DI), in a default segment.
•Eg: MOV AX,50H[BX][SI]

For the control transfer instructions, the addressing modes depend upon whether the destination location is within the same segment or in a different one.

•It also depends upon the method of passing the destination address to the processor.
•Basically there are two addressing modes for the control transfer instructions, viz, intersegment and intrasegment addressing modes.
•If the location to which the control is to be transferred lies in a different segment other
than the current one, the mode is called intersegment mode.

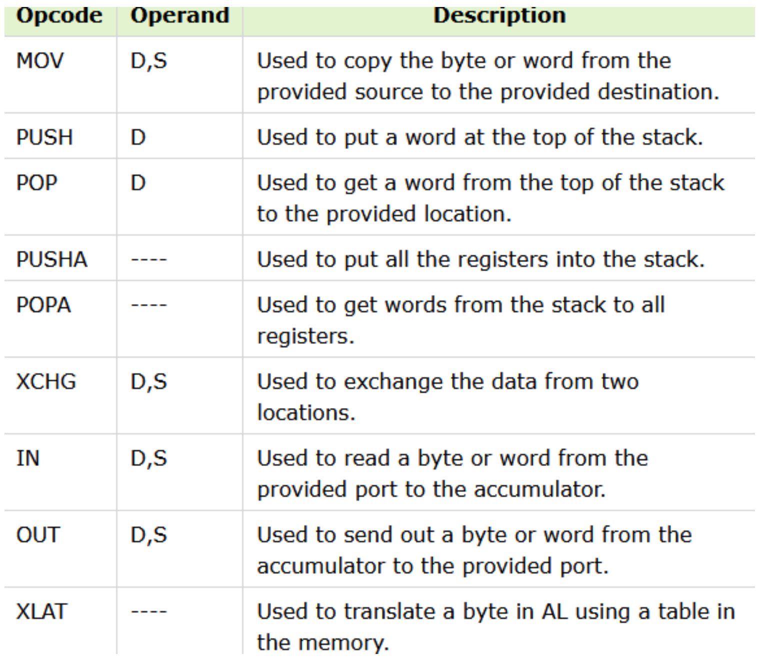
**9.Inter segment direct**

**10. Inter segment indirect**

**11. Intra segment direct**

**12. Intra segment indirect**

**13b)**



| Opcode | Operand | Description |
|---|---|---|
| MOV | D,S | Used to copy the byte or word from the provided source to the provided destination. |
| PUSH | D | Used to put a word at the top of the stack. |
| POP | D | Used to get a word from the top of the stack to the provided location. |
| PUSHA | ---- | Used to put all the registers into the stack. |
| POPA | ---- | Used to get words from the stack to all registers. |
| XCHG | D,S | Used to exchange the data from two locations. |
| IN | D,S | Used to read a byte or word from the provided port to the accumulator. |
| OUT | D,S | Used to send out a byte or word from the accumulator to the provided port. |
| XLAT | ---- | Used to translate a byte in AL using a table in the memory. |

| | | |
|---|---|---|
| LAHF | ---- | Used to load AH with the low byte of the flag register. |
| SAHF | ---- | Used to store AH register to low byte of the flag register. |
| PUSHF | ---- | Used to copy the flag register at the top of the stack. |
| POPF | ---- | Used to copy a word at the top of the stack to |

**14a)**

| Instruction | AX | BX | CX | DX | CF | ZF | SF |
|---|---|---|---|---|---|---|---|
| ROR AX,04H | 0003H | 0031H | 0032H | 0033H | 0000H(If rotate with carry,CF is affected) | 0000H | 0000H |
| CMP BX,AX | 0030H | 0031H | 0032H | 0033H | If BX<AX CF is set | If BX=AX ZF is set | |
| XCHG CX,DX | 0030H | 0031H | 0032H | 0033H | 0000H | 0000H | 0000H |
| AND AX,BX | 0030H | 0031H | 0032H | 0033H | 0000H | 0000H | 0000H |
| LOOP Addr | 0030H | 0031H | 0032H | 0033H | 0000H | 0001H | 0000H |
| XOR AX,BX | 0001H | 0031H | 0032H | 0033H | 0000H | 0001H | 0000H |
| STC | 0030H | 0031H | 0032H | 0033H | 0001H | 0000H | 0000H |

**14 b)**

| | |
|---|---|
| 4000 | MOV SI, 2000 |
| 4003 | MOV CL, [SI] |
| 4005 | MOV CH, 00 |
| 4007 | INC SI |
| 4008 | MOV AL, [SI] |
| 400A | DEC CL |
| 400C | INC SI |
| 400D | CMP AL, [SI] |
| 400F | JNC 4013 |
| 4011 | MOV AL, [SI] |
| 4013 | INC SI |

|  |  |
|---|---|
| 4014 | LOOP 400D |
| 4016 | MOV [3000], AL |
| 401A | HLT |

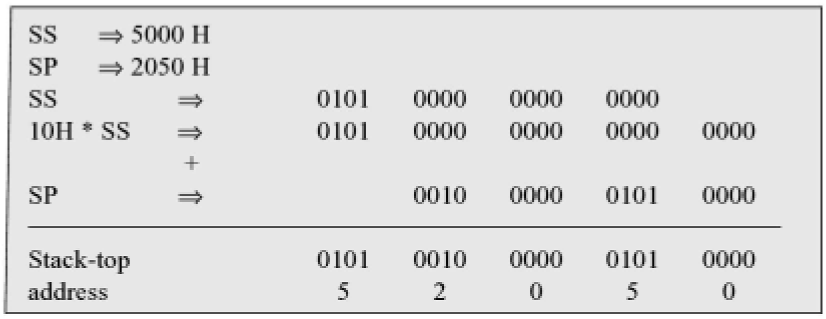
**Module 3**

**15 a)**

THE STACK STRUCTURE OF 8086:

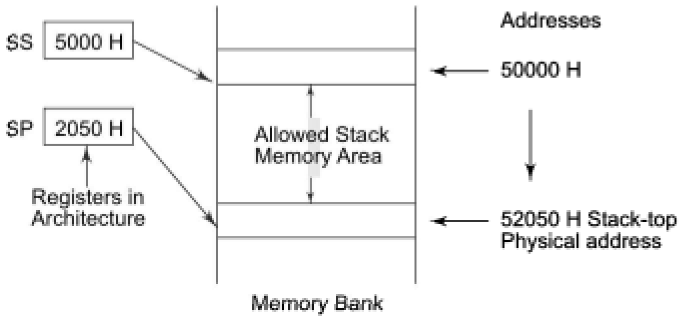
●The stack is a block of memory that may be used for temporarily storing the contents of the registers inside the CPU. It is a top-down data structure whose elements are accessed using the stack pointer (SP) which gets decremented by two as we store a data word in the stack and gets incremented by two as we retrieve a data word from the stack back to the CPU register.

●The stack segment, like any other segment, may have a memory block of a maximum of 64 Kbytes locations and thus may overlap with any other segments. The stack Segment register (SS) contains the base address of the stack segment in the memory. In 8086 microprocessor-based system, the stack is created by loading a 16-bit base address in the Stack Segment (SS) register and a 16-bit offset address in Stack Pointer (SP). The 20-bit physical address of the stack is computed by multiplying the contents of the SS register by 10H and then adding the contents of SP to this product. Here the content of SP is the offset address of the stack.



```
SS        ⇒ 5000 H
SP        ⇒ 2050 H
SS             ⇒        0101   0000   0000   0000
10H * SS       ⇒        0101   0000   0000   0000   0000
               +
SP             ⇒               0010   0000   0101   0000
-----------------------------------------------------------
Stack-top               0101   0010   0000   0101   0000
address                   5      2      0      5      0
```

Thus the stack top address is 52050 H. Figure 4.1 makes the concept more clear.


SS
5000 H
SP
2050 H
Registers in Architecture
Allowed Stack Memory Area
Memory Bank
Addresses
50000 H
52050 H Stack-top Physical address

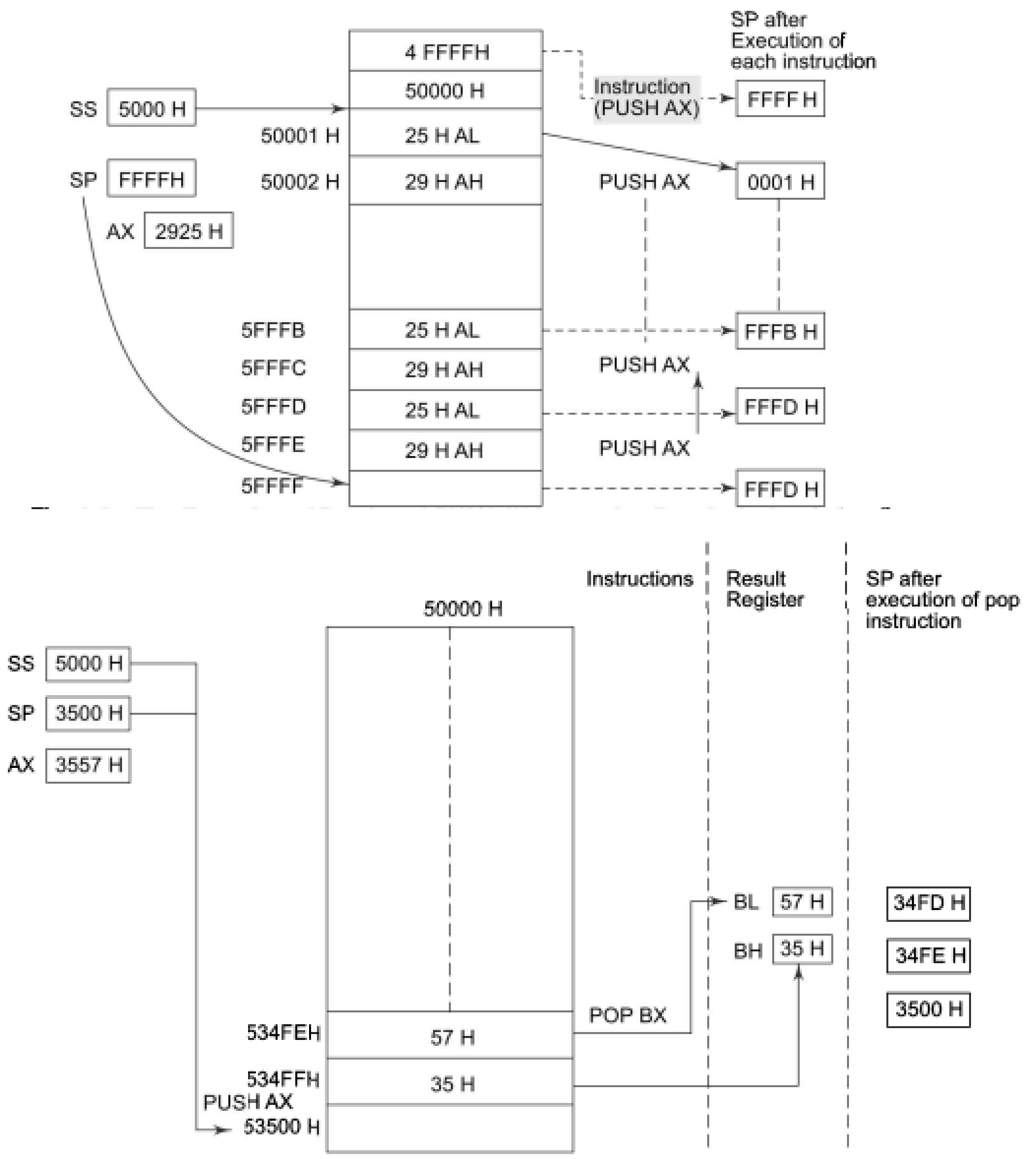


**Fig. 4.3** *Effect of PUSH and POP on SP*

**15b)**

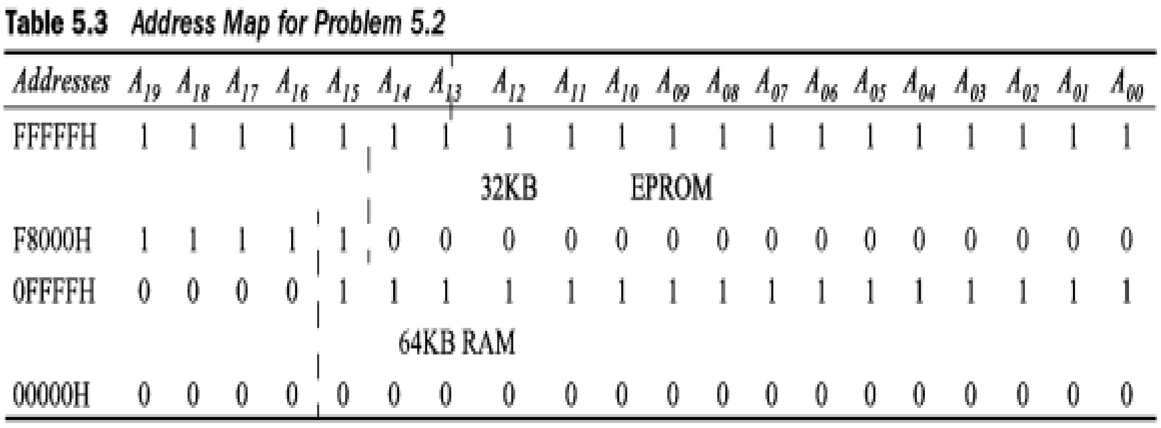


**Table 5.3** *Address Map for Problem 5.2*

| Addresses | $A_{19}$ | $A_{18}$ | $A_{17}$ | $A_{16}$ | $A_{15}$ | $A_{14}$ | $A_{13}$ | $A_{12}$ | $A_{11}$ | $A_{10}$ | $A_{09}$ | $A_{08}$ | $A_{07}$ | $A_{06}$ | $A_{05}$ | $A_{04}$ | $A_{03}$ | $A_{02}$ | $A_{01}$ | $A_{00}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FFFFFH | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | | | | | | | | 32KB | | | EPROM | | | | | | | | | |
| F8000H | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0FFFFH | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | | | | | | | 64KB RAM | | | | | | | | | | | | | |
| 00000H | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

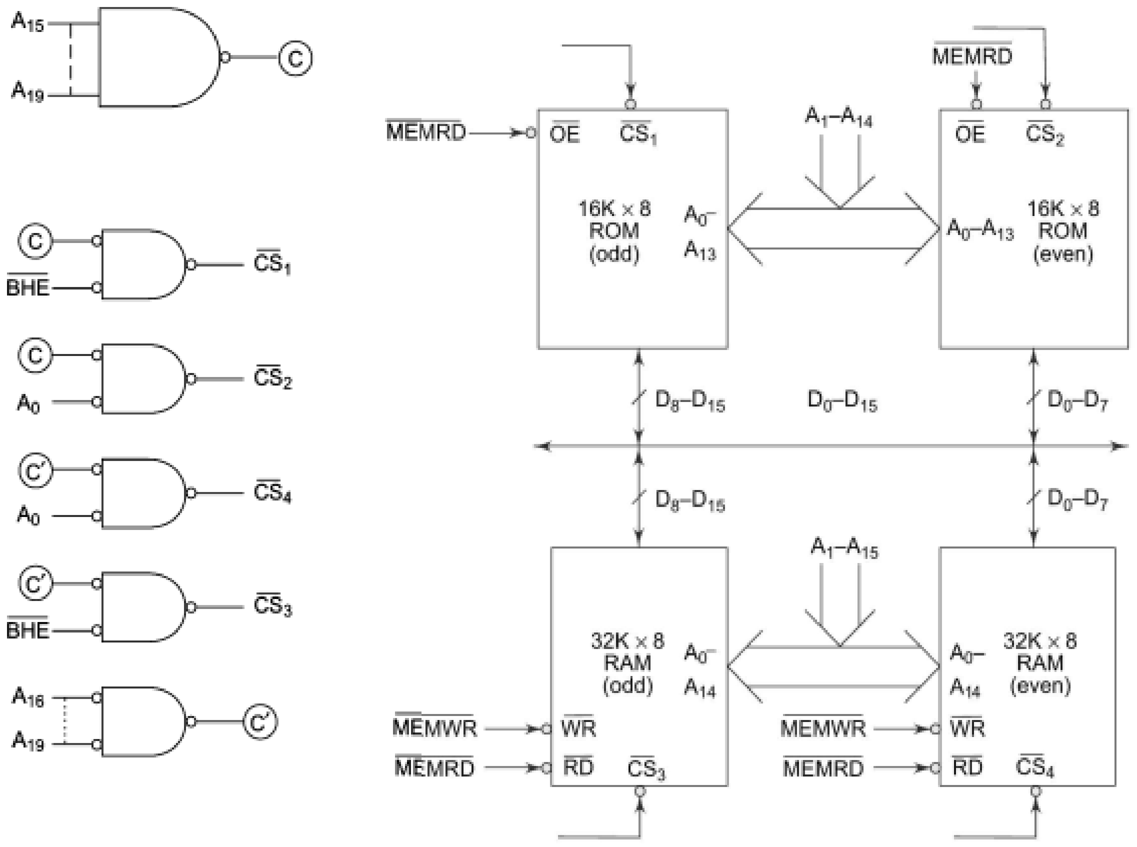
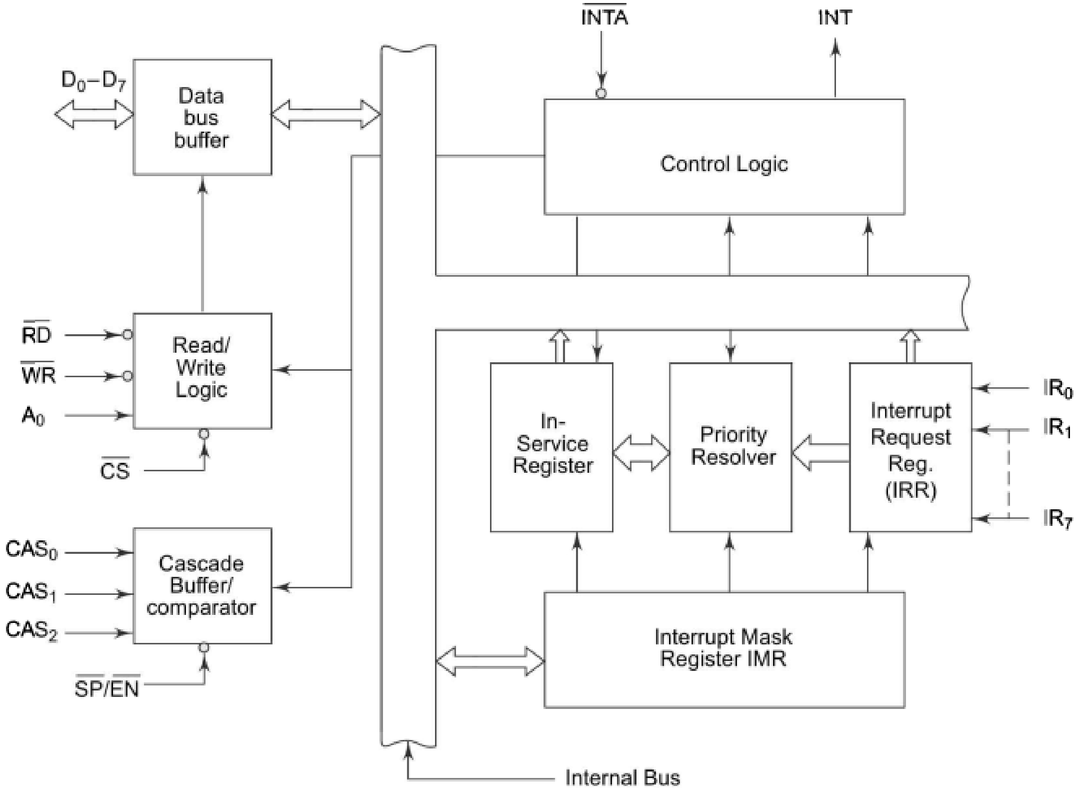


**Fig. 5.2** *Interfacing Problem 5.2*

**16a)**



**Explain each components**

**16b)**

**IVT of 8086**

## Interrupt cycle of 8086/88

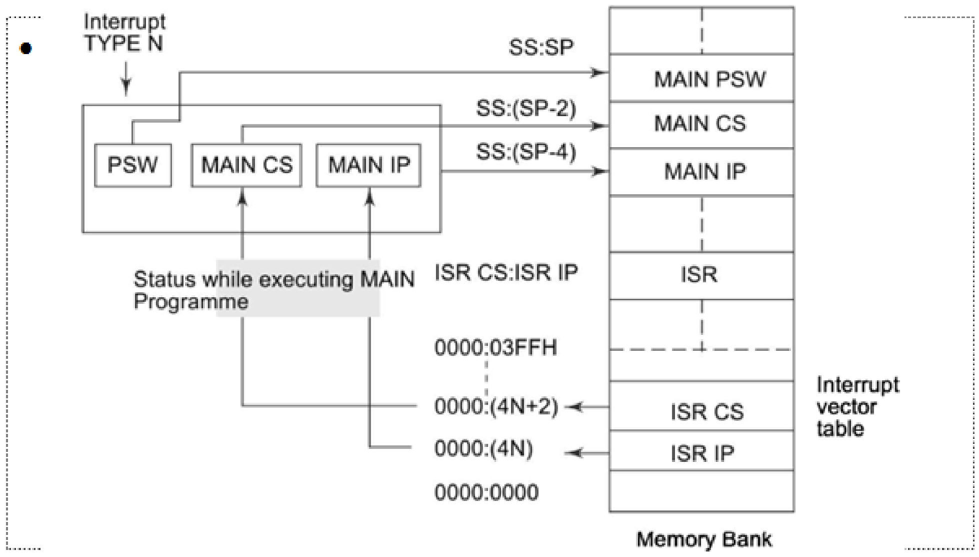
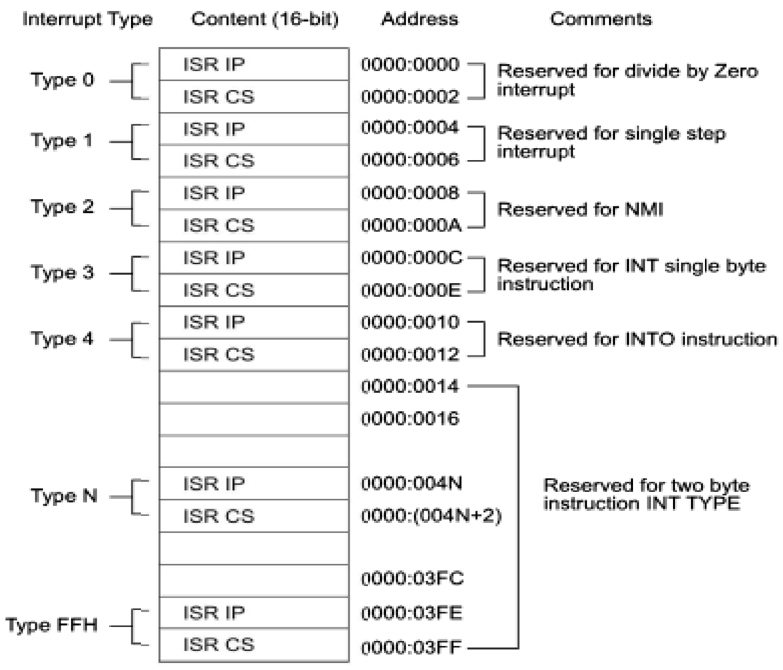


**Fig. 4.4** ***Interrupt Response Sequence***



| Interrupt Type | Content (16-bit) | Address | Comments |
|---|---|---|---|
| Type 0 | ISR IP | 0000:0000 | Reserved for divide by Zero interrupt |
| | ISR CS | 0000:0002 | |
| Type 1 | ISR IP | 0000:0004 | Reserved for single step interrupt |
| | ISR CS | 0000:0006 | |
| Type 2 | ISR IP | 0000:0008 | Reserved for NMI |
| | ISR CS | 0000:000A | |
| Type 3 | ISR IP | 0000:000C | Reserved for INT single byte instruction |
| | ISR CS | 0000:000E | |
| Type 4 | ISR IP | 0000:0010 | Reserved for INTO instruction |
| | ISR CS | 0000:0012 | |
| | | 0000:0014 | Reserved for two byte instruction INT TYPE |
| | | 0000:0016 | |
| | | | |
| Type N | ISR IP | 0000:004N | |
| | ISR CS | 0000:(004N+2) | |
| | | | |
| | | 0000:03FC | |
| Type FFH | ISR IP | 0000:03FE | |
| | ISR CS | 0000:03FF | |

ISR : Interrupt Service Routine

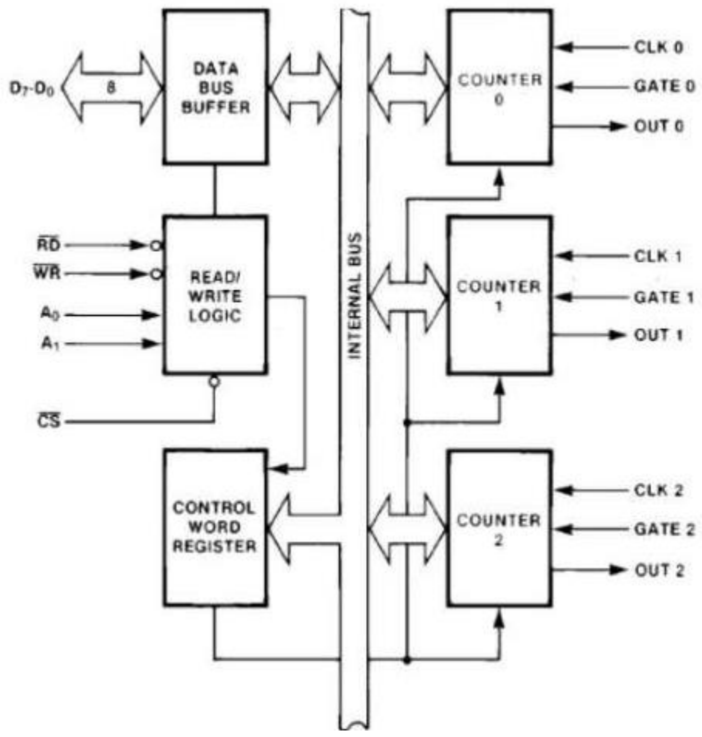
**Fig. 4.5** ***Structure of Interrupt Vector Table of 8086/88***

**Explain the procedure**

## Module 4

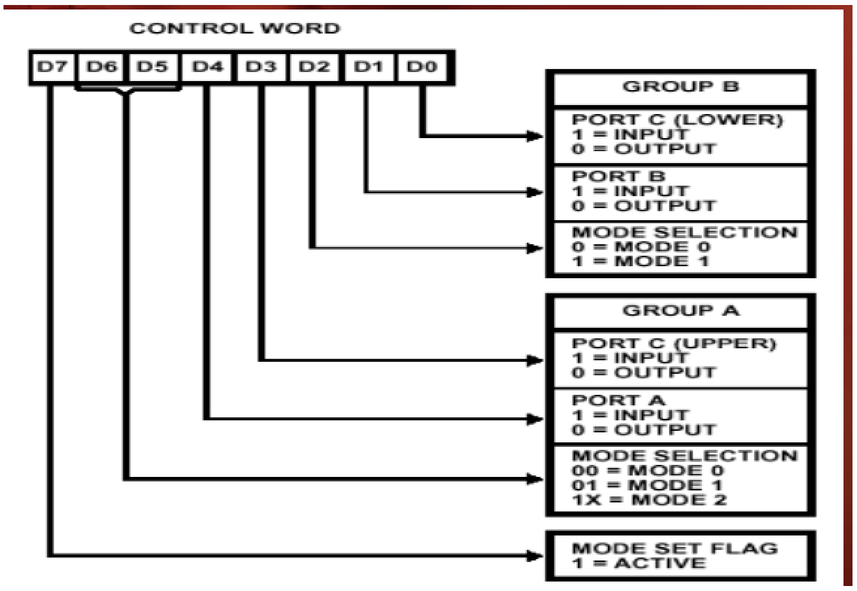
**17a)**

The architecture of 8254 looks as follows –



**Explain each components(refer notes)**

**17b) There are two basic operational modes of 8255:**



CONTROL WORD
D7
D6
D5
D4
D3
D2
D1
D0
GROUP B
PORT C (LOWER) 1 = INPUT 0 = OUTPUT
PORT B 1 = INPUT 0 = OUTPUT
MODE SELECTION 0 = MODE 0 1 = MODE 1
GROUP A
PORT C (UPPER) 1 = INPUT 0 = OUTPUT
PORT A 1 = INPUT 0 = OUTPUT
MODE SELECTION 00 = MODE 0 01 = MODE 1 1X = MODE 2
MODE SET FLAG 1 = ACTIVE

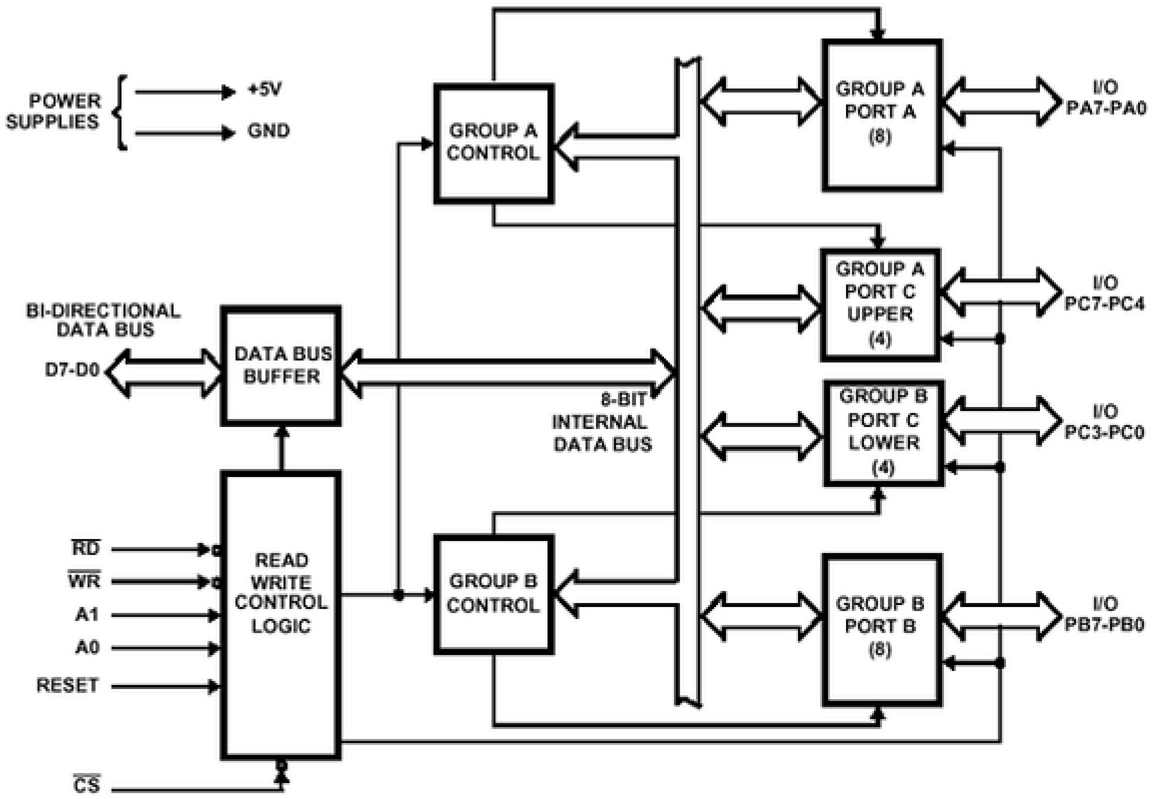
- Bit Set/Reset mode (BSR mode).

  This mode is used to set or reset the bits of port C only, and selected when the most significant bit (D7) in the control register is 0.
- Input/Output mode (I/O mode).

This mode is selected when the most significant bit (D7) in the control register is 1. It has three operating modes: mode 0, mode 1, and mode 2.

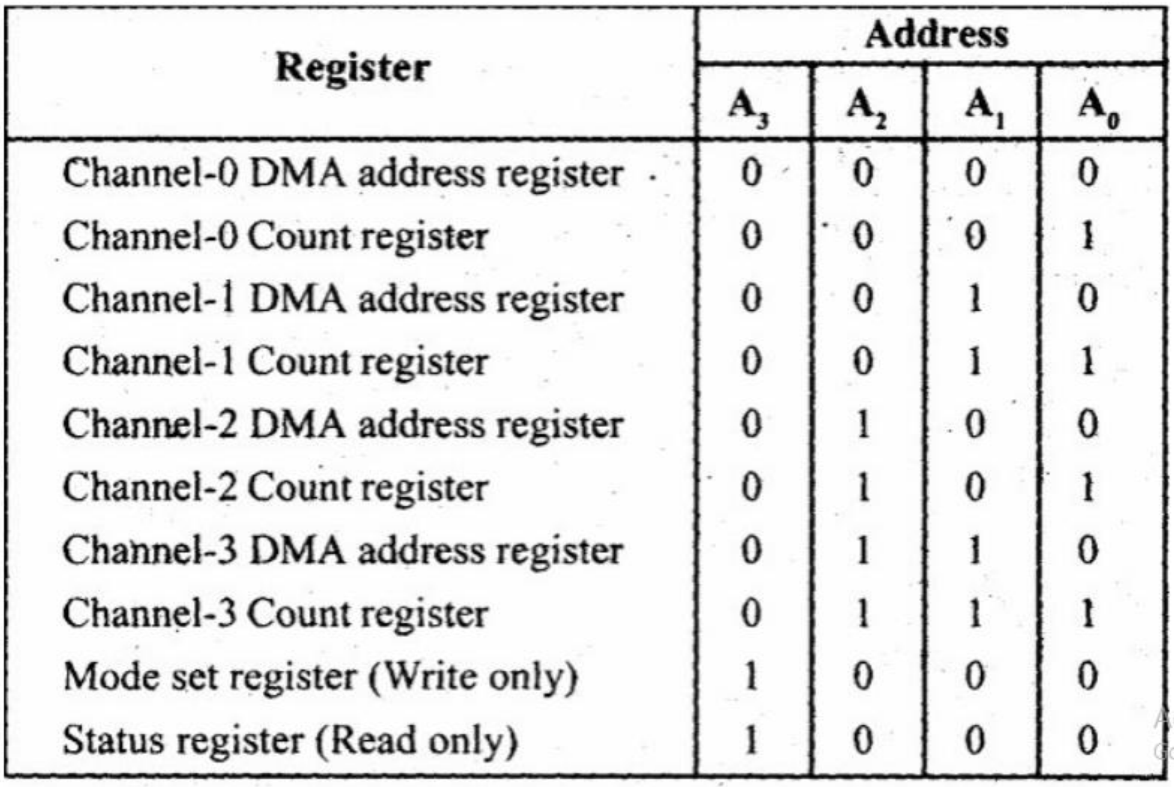
Mode 0 is the basic input/output mode. In this mode, the 8255 can be used as an 8-bit input or output device, with each of the three ports (Port A, Port B, and Port C) being configured individually as input or output.
Mode 1 is the strobed input/output mode. This mode is similar to mode 0, but includes additional control signals to enable latching of input data and/or output data.
Mode 2 is the bidirectional mode. In this mode, Port A and Port B are configured as bidirectional I/O ports, while Port C is configured as a control port for setting various operational modes.

**18a)**



**Explain each components**

**18b)**



| Register | Address | | | |
|---|---|---|---|---|
| | $A_3$ | $A_2$ | $A_1$ | $A_0$ |
| Channel-0 DMA address register | 0 | 0 | 0 | 0 |
| Channel-0 Count register | 0 | 0 | 0 | 1 |
| Channel-1 DMA address register | 0 | 0 | 1 | 0 |
| Channel-1 Count register | 0 | 0 | 1 | 1 |
| Channel-2 DMA address register | 0 | 1 | 0 | 0 |
| Channel-2 Count register | 0 | 1 | 0 | 1 |
| Channel-3 DMA address register | 0 | 1 | 1 | 0 |
| Channel-3 Count register | 0 | 1 | 1 | 1 |
| Mode set register (Write only) | 1 | 0 | 0 | 0 |
| Status register (Read only) | 1 | 0 | 0 | 0 |

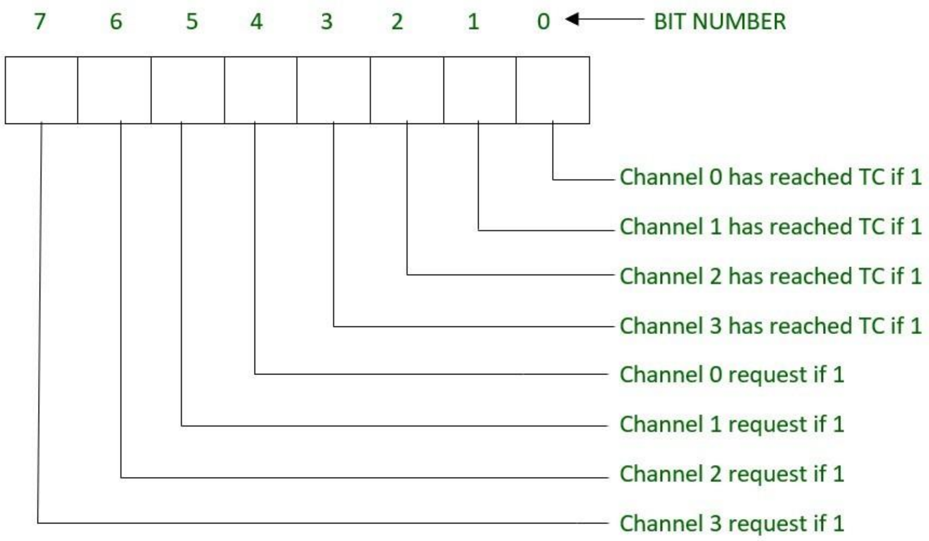
**1.** Base Address Register (16 bit)
**2.** Base Word Count Register (16 bit)
**3.** Current Address Register (16 bit)
**4.** Current Word Count Register (16 bit)
**5.** Temporary Address Register (16 bit)
**6.** Temporary Word Count Register (16 bit)
**7.** Status Register (8 bit)
**8.** Command Register (8 bit)
**9.** Temporary Register (8 bit)
**10.** Mode Register (8 bit)
**11.** Mask Register (4 bit)
**12.** Request Register (4 bit)

These are explained as following below.

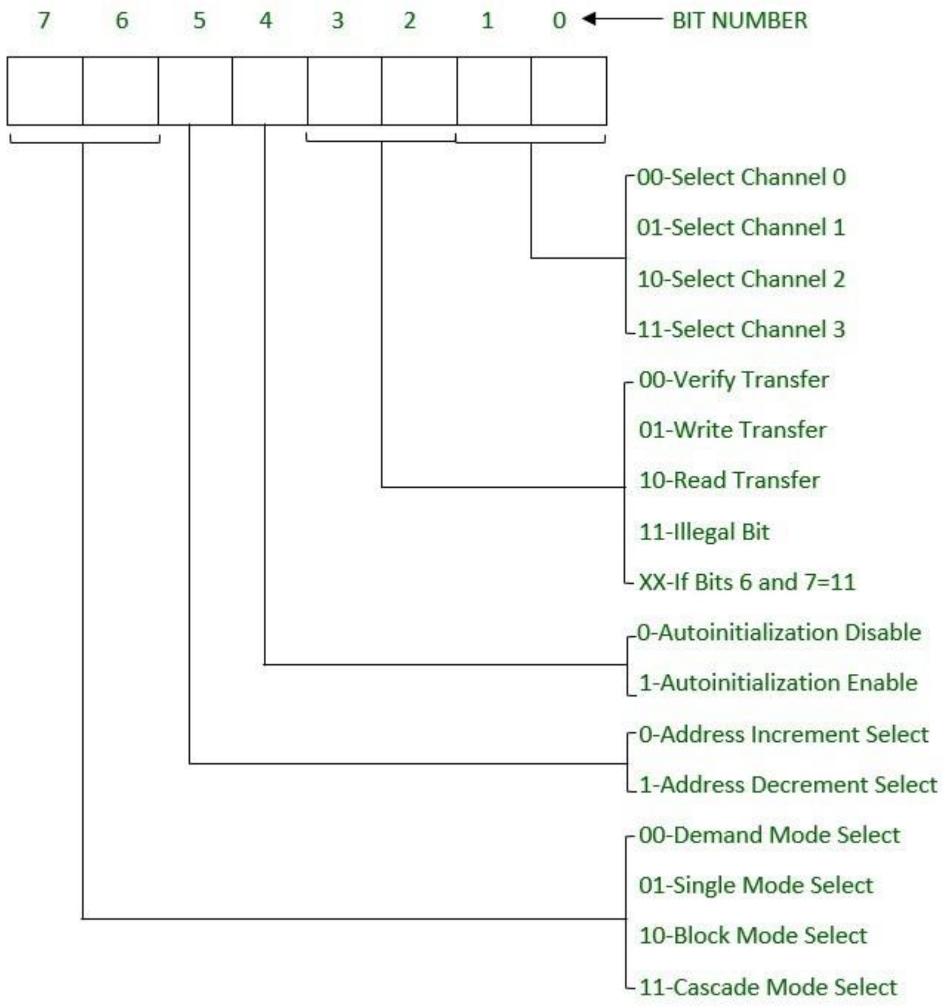
1. **Base Address Register:**
   It is a 16 bit register that stores the initial address from where the data transfer will take place in a DMA Controller. It is used to reload the Current Address Register after every operation.
2. **Base Word Count Register:**
   It is a 16 bit register that stores the number of transfers to be performed during an operation. It is used to reload the Current Word Count Register after every operation.
3. **Current Address Register:**
   It is a 16 bit register that stores the memory address for DMA data transfer. The value

automatically increases or decreases after every operation based on how it is programmed. Each channel has its own Current Address Register.

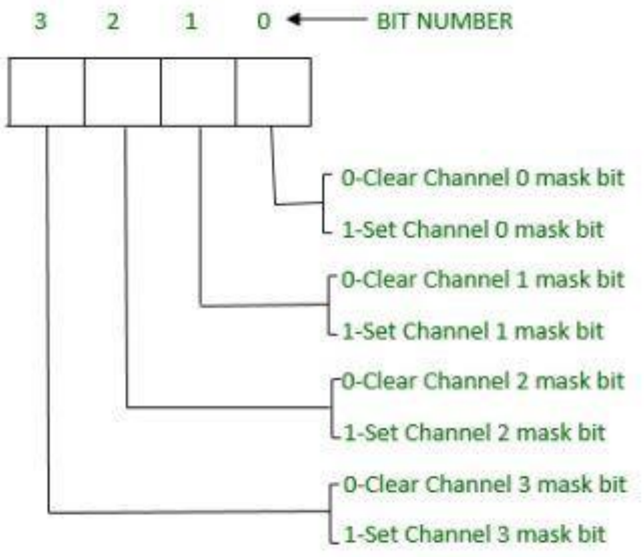
4. **Current Word Count Register:**
   It is a 16 bit register that stores the number of transfers remaining to be performed during an operation. The value automatically decreases after every operation.
5. **Temporary Address Register:**
   It is a 16 bit register that stores the address of data during memory to memory transfer in a DMA Controller.
6. **Temporary Word Count Register:**
   It is a 16 bit register that stores the number of transfers to be performed during a memory to memory transfer in a DMA Controller.
7. **Status Register:**
   It is a 8 bit register that indicates which channel is currently under DMA services or which channels has reached its terminal count. It basically gives the status of the channels. The terminal counts(TC) bits indicates if the channel has reached its terminal count. If terminal count is reached, the transfers are terminated.


7 6 5 4 3 2 1 0
BIT NUMBER
Channel 0 has reached TC if 1
Channel 1 has reached TC if 1
Channel 2 has reached TC if 1
Channel 3 has reached TC if 1
Channel 0 request if 1
Channel 1 request if 1
Channel 2 request if 1
Channel 3 request if 1

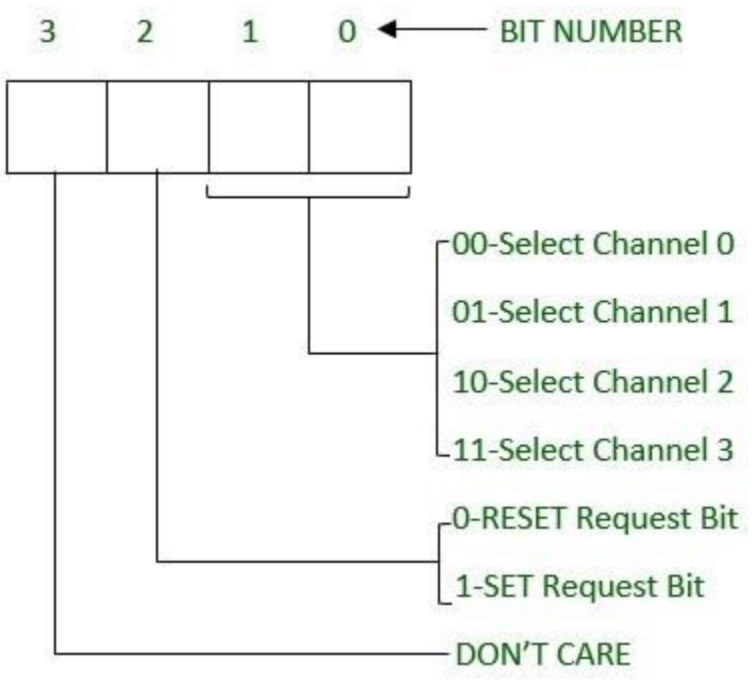
8. **Command Register:**
   It is a 8 bit register that programs the DMA operation and initializes the channel to be used for data transfer.
9. **Temporary Register:**
   It is a 8 bit register that holds data during memory to memory data transfer. It always contain the last byte transferred in previous memory to memory transfer operation.
10. **Mode Register:**
    It is a 8 bit register that determines the operating mode, i.e., the transfer mode and other transfer parameters, for a channel. Each channel has its own mode register which is selected by bit positions 0 and 1.



11. **Mask Register:**
It is a 4 bit register that is used to mask a channel from requesting the DMA Services. When the mask on a channel is SET, the channel is disabled. It sets or clears all the mask on all the channels with just one command.



12. **Request Register:**
It is a 4 bit register that is used to request DMA data transfer by the software. It determines which channel is requesting for the data transfer.



**Module 5**

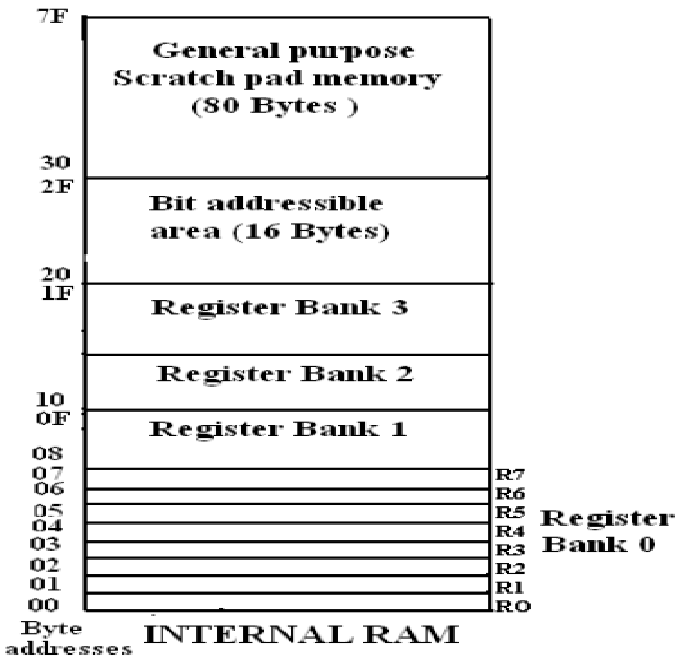
**19a)**

- The Internal RAM is found on-chip in 8051. So it is the fastest RAM available.
- Internal RAM is volatile, so when the 8051 is reset this memory is cleared.
- The 128 bytes of internal RAM is organized as below.

(i) Four register banks (Bank0,Bank1, Bank2 and Bank3) each of 8-bytes. The default bank register is Bank 0. The Banks are selected with the help of RS0 and RS1 bits of PSW Register.

(ii) 16 bytes of bit addressable area and

(iii) 80 bytes of general purpose area (Scratch pad memory) as shown in the diagram below.



**19b)**

**Immediate addressing mode**

In this Immediate Addressing Mode, the data is provided in the instruction itself. The data is provided immediately after the opcode. These are some examples of Immediate Addressing Mode.

MOVA, #0AFH;
MOVR3, #45H;
MOVDPTR, #FE00H;

**Register addressing mode**

In the register addressing mode the source or destination data should be present in a register (R0 to R7). These are some examples of RegisterAddressing Mode.

MOVA, R5;
MOVR2, #45H;
MOVR0, A;

**Direct Addressing Mode**

In the Direct Addressing Mode, the source or destination address is specified by using 8-bit data in the instruction. Only the internal data memory can be used in this mode. Here some of the examples of direct Addressing Mode.

MOV80H, R6;
MOVR2, 45H;
MOVR0, 05H;

**Register indirect addressing Mode**

In this mode, the source or destination address is given in the register. By using register indirect addressing mode, the internal or external addresses can be accessed. The R0 and R1 are used for 8-bit addresses, and DPTR is used for 16-bit addresses, no other registers can be used for addressing purposes. Let us see some examples of this mode.

MOV0E5H, @R0;
MOV@R1, 80H

**20a)**

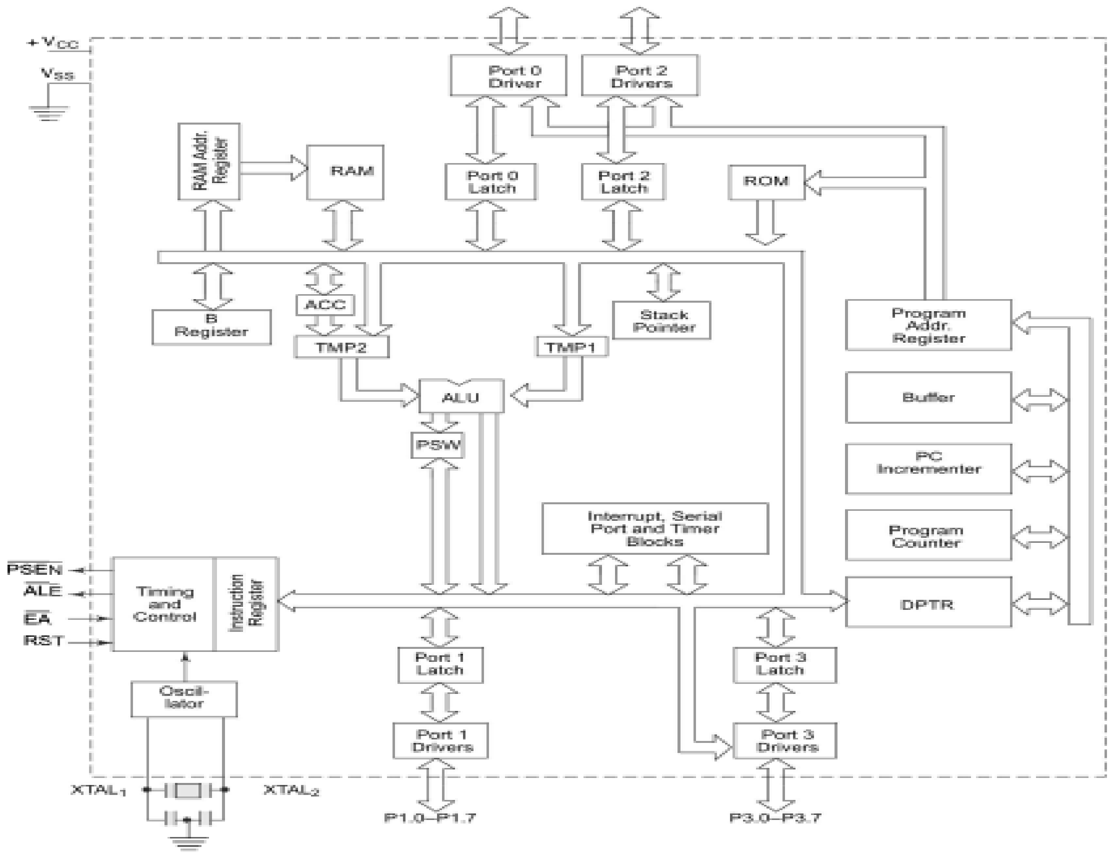
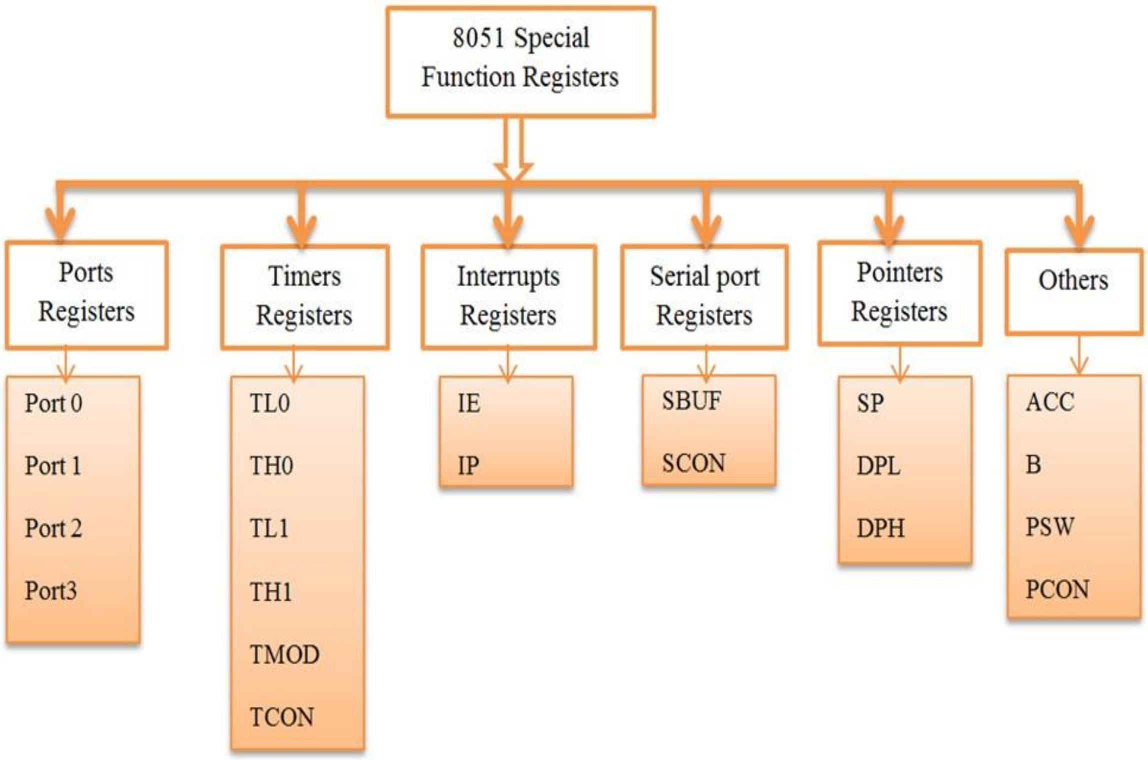


Fig. 17.2 8051 Block Diagram (Intel Corp.)

**Explain each components**

**20b)**


8051 Special Function Registers
Ports Registers
Port 0
Port 1
Port 2
Port3
Timers Registers
TL0
TH0
TL1
TH1
TMOD
TCON
Interrupts Registers
IE
IP
Serial port Registers
SBUF
SCON
Pointers Registers
SP
DPL
DPH
Others
ACC
B
PSW
PCON