FIRST TERM EXAMINATION [SEPT.-2015]
FIFTH SEMESTER [B. TECH]
MICROPROCESSOR AND MICROCONTROLLERS [ETEC-305]

Time: 1 Hrs. **M M: 30**

Note: *Q.1. is compulsory attempt any two from the remaining.*

Q.1. Attempt any five. **(5 × 2)**

Q.1. (a) State an example where the internal registers W and Z of the 8085 microprocessor are used iternally by the microprocessor.

Ans. Temporary Register: W & Z

- It is also called as operand register (8 bit).
- It provides operands to ALU. ALU can store immediate result in temporary register.
- It is not accessible by user.

Q.1. (b) Write a set of instructions to transfer logic 0 from the SOD lines of 8085 microprocessor.

Ans. SOD (Serial Output Data) Line -There is a One bit Output port inside the 8085 CPU Pin number 4 -1 bit data can be externally written in this port. To write data into this port, SIM instruction is used. The data that is to be written in this port must be stored in the A7th bit of the Accumulator. Bit A6 of the Accumulator is known as SOE (Serial output Enable). This bit Must be set to 1 to enable Serial data output.

To write a logic 1 in this SOD line, Load the accumulator with C0H

To write a logic 0 in this SOD line, Load the accumulator with 40H

Pseudo code

A<-40H

SIM

SOD<-(A7)

Q.1. (c) How much time is taken by the microrporcessor 8085 to execute conditional RET instruction of 8085 microprocessor if it is running at 2 MHZ.

Ans. F=2MHz, T = 1/F = 0.5microsec

Time is taken by the microprocessor 8085 to execute conditional RET instruction = 10*0.5microsec = 5microsec

Q.1. (d) State the function of AAA and AAS instructions of the 8086 microrprocessor.

Ans. AAA Instruction: AAA converts the result of the addition of two valid unpacked BCD digits to a valid 2-digit BCD number and takes the AL register as its implicit operand. Two operands of the addition must have its lower 4 bits contain a number in the range from 0-9.The AAA instruction then adjust AL so that it contains a correct BCD digit. If the addition produce carry (AF=1), the AH register is incremented and the carry CF and auxiliary carry AF flags are set to 1. If the addition did not produce a decimal carry, CF and AF are cleared to 0 and AH is not altered. In both cases the higher 4 bits of AL are cleared to 0.

AAA will adjust the result of the two ASCII characters that were in the range from 30h ("0") to 39h("9").This is because the lower 4 bits of those character fall in the range of 0-9.The result of addition is not a ASCII character but it is a BCD digit.

Example:

MOV AH,0; Clear AH for MSD

MOV AL,6; BCD 6 in AL

ADD AL,5; Add BCD 5 to digit in AL

AAA ;AH = 1, AL = 1 representing BCD 11.

AAS Instruction: AAS converts the result of the subtraction of two valid unpacked BCD digits to a single valid BCD number and takes the AL register as an implicit operand. The two operands of the subtraction must have its lower 4 bit contain number in the range from 0 to 9 .The AAS instruction then adjust AL so that it contain a correct BCD digit.

Q.1. (e) Describe the REP, REPZ and REPNZ prefixes used in string instruction with suitable examples.

Ans. Repeat String Operation (rep, repnz, repz)

rep;

repnz;

repz;

Operation: Repeat string-operation until tested-condition

Description: Use the rep (repeat while equal), repnz (repeat while nonzero) or repz (repeat while zero) prefixes in conjunction with string operations. Each prefix causes the associated string instruction to repeat until the count register (CX) or the zero flag (ZF) matches a tested condition.

Example: Repeat while equal: Copy the 8-bit byte from the DS:[(E)SI] to the ES:[(E)DI] register.

rep; movsb

Repeat while not zero: Compare the memory byte double-word addressed in the destination register DL, relative to the ES segment, with the contents of the AX register.

repnz; scasl

Repeat while zero: Transfer the contents of the AX register to the memory double-word addressed in the destination register DL, relative to the ES segment.

repz; stosl

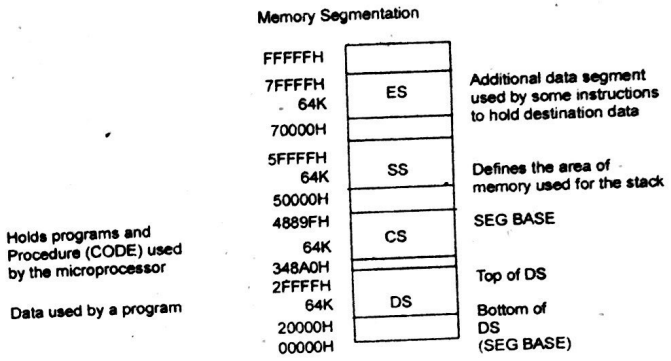
Q.1. (f) What is memory segmentation? How memory segmentation is achieved in the 8086 microprocessor? State the advantages of memory segmentation.

Ans. Memory Segmentation: The total memory size is divided into segments of various sizes. A segment is just an area in memory. The process of dividing memory this way is called Segmentation.

In memory, data is stored as bytes. Each byte has a specific address. Intel 8086 has 20 lines address bus. With 20 address lines, the memory that can be addressed is 2^{20} bytes. 2^{20} = 1,048,576 bytes (1 MB). 8086 can access memory with address ranging from 00000 H to FFFFF H.

Some of the advantages of memory segmentation in the 8086 are as follows:

- With the help of memory segmentation a user is able to work with registers having only 16-bits.
- By memory segmentation the various portions of a program can be of more than 64 Kb.
- The data and the users code can be stored separately allowing for more flexibility.
- Also due to segmentation the logical address range is from 0000H to FFFFH the code can be loaded at any location in the memory.



Q.1 (g) Why 8086 memory is mapped into banks? What logic levels are found with BHE and A0 when 8086 reads a word from the address 0A0AH?

Ans. 8086 Memory Banks: The 8086 has 20-bit address bus, so it can address 2^{20} or 1,048,576 addresses. Each address represents a stored byte. To make it possible to read or write a word with one machine cycle, the memory for an 8086 is set up in to 2 banks of up to 524,288 bytes each.

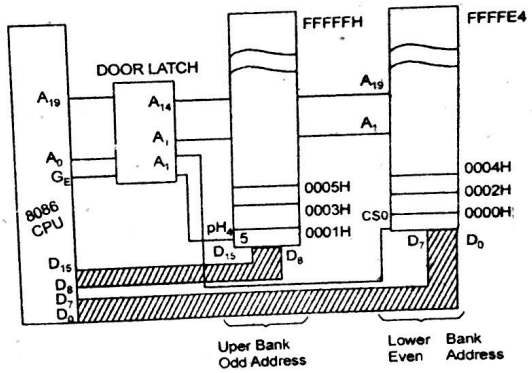


Fig. 8086 Memory Banks

One memory bank contains all the bytes which have even addresses such as 00000h, 00002h, and 00004h etc. the data lines of this bank is connected to the lower 8 bit data lines i.e. from D0 to D7 of 8086.

The other memory bank contains all bytes which have odd addresses such as 00001h, 00003h and 00005h etc. the data lines of this bank is connected to the upper 8 bit data lines i.e. from D8 to D15 of 8086.

Address line A0 is used for enabling the memory device in the lower bank. addressed memory device in this bank will be enabled when A0 is low, as it will be any even address.

Like address 00222h = 0000 0000 0010 0010 0010.

Address lines A1 to A19 are used to select the desired memory device in the ba and hence the desired byte in the device.

Address line A1 to A19 are also used to select the desired memory device in t upper bank and hence the desired byte. An additional part of enabling the upper ba memory device is handled by the BHE i.e. the bus high enable signal. This is multiplex out from the 8086 at the same time as an address is sent out. An external latc strobbed by the ALE signal, grabs the BHE signal and holds it stable for the rest of t machine cycle like it does for the address.

So now if we read a byte from or write a byte to an even address the A0 will be lo BHE will be high enabling the lower bank and disabling the upper bank.

The main reason that the A0 and BHE signal work as they do is to prevent th writing of an unwanted signal.

Q.2. (a) Write a program for 8085 to arrange numbers in ascending order. (4

Ans. A program to sort given 10 numbers from memory location 2200H in th ascending order

Source program:

MVI B, 09 : Initialize counter
START : LXI H, 2200H: Initialize memory pointer
MVI C, 09H : Initialize counter 2
BACK: MOV A, M : Get the number
INX H : Increment memory pointer
CMP M : Compare number with next number
JC SKIP : If less, don't interchange
JZ SKIP : If equal, don't interchange
MOV D, M
MOV M, A
DCX H
MOV M, D
INX H : Interchange two numbers
SKIP:DCR C : Decrement counter 2
JNZ BACK : If not zero, repeat
DCR B : Decrement counter 1
JNZ START
HLT : Terminate program execution

Q.2. (b) Given that *BX-637DH*, SI-2A9BH. Displacement 237H. Determine the effective address resulting from these register and the addressing made. (4

(i) Immediate
(ii) Direct
(iii) Register Indirect using BX

(iv) Relative Base Indexed
(v) Based Indexed
(vi) Register Relative using BX.

Ans. (i) No Effective Address
(ii) No Effective Address
(iii) Effective Address = BX = 637D
(iv) Effective Address = BX + SI + 237 = 904F
(v) Effective Address = BX + SI = 8E18
(vi) Effective Address = BX + 237 = 65B4

Q.2. (c) Describe memory mapped I/O. State its advantage over I/O mapped I/O. (2)

Ans.

Memory Mapped I/O	I/O Mapped I/O
In Memory Mapped I/O Address width is 16-bit. A0 to A15 are used to generate address of the device.	In I/O Mapped I/O Address width is 8-bit. A0 to A15 lines are used to generate address of the device.
MEMR and MEMW control signals are used to control read and write I/O operations respectively.	IOR and IOW control signals are used to control read and write I/O operations respectively.
Instructions available are STA addr, LDA addr, LDAX rp, STAX rp, ADD M, CMP M, MOV r, M, etc.	IN and OUT are the only available instructions.
Data transfer takes place between any register and I/O device.	Data transfer takes place between accumulator and I/O device.
Maximum number of I/O devices that can be addressed is 65536 (theoretically).	Maximum number of I/O devices that can be addressed is 256.
Execution speed using STA addr, LDA addr is 13 T-state and for MOV M, r, etc.,it is 7-T states.	Execution speed is 10 T-states.
It requires more hardware circuitry because it decodes 16-bit address.	It requires less hardware circuitry because it decodes 8-bit address.

Q.3. (a) What is type 2 interrupt? Explain the condition for initiating type 2 interrupt? (2)

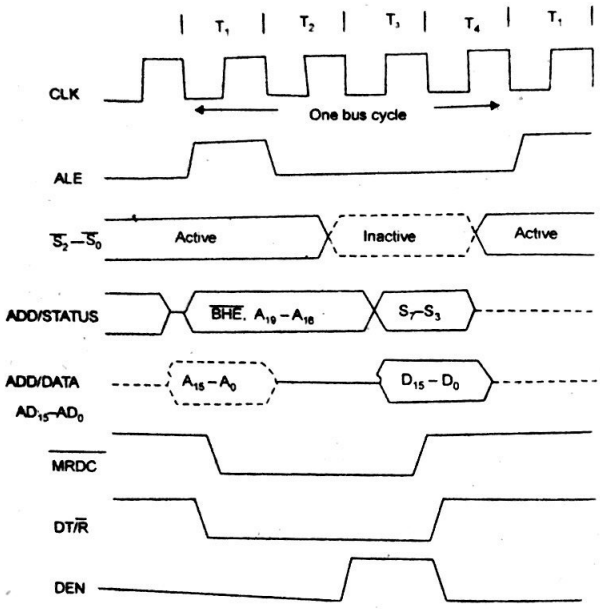
Ans. Nonmaskable Interrupt-Type 2:

The 8086 will automatically do a type 2 interrupt response when it receives a low to high transition on its NMI pin. When it does a type 2 interrupt, the 8086 will push the flags on the stack, reset TF and IF, and push the CS value and the IP value for the next instruction on the stack. It will then get the CS value for the start of the type 2 interrupt service procedure from address 0000AH and the IP value for the start of the procedure from address 00008H.

Q.3. (b) Draw the timing diagram of memory read in maximum mode configuration of 8086? (4)

Ans. The maximum mode system timing diagrams are also divided in two portions as read (input) and write (output) timing diagrams. The address/data and address/status timings are similar to the minimum mode. ALE is asserted in T1, just like minimum mode. The only difference lies in the status signals used and the available

control and advanced command signals. The fig. shows the maximum mode timings the read\operation



Q.3. (c) Design a microprocessor 8085 system having 4K × 8 EPROM, 8K × 8 RAM, one Keyboard and one display. Avoid any foldback address and use IO mapped IO technique to interface the IO devices. (4

Ans. A system requires 16kb EPROM and 16kb RAM. Also the system has 2 numbers of 8255, one number of 8279, one number of 8251 and one number of 8254. (8255 - Programmable peripheral interface; 8279-Keyboard/display controller, 8251 - USART and 8254 - Timer). Draw the Interface diagram. Allocate addresses to all the devices. The peripheral IC should be I/O mapped.

- The I/O devices in the system should be mapped by standard I/O mapping. Hence separate decoders can be used to generate chip select signals for memory IC and peripheral IC's.
- For 16kb EPROM, we can provide 2 numbers of 2764(8k × 8) EPROM.
- For 16kb RAM we can provide 2 numbers of 6264 (8k × 8) RAM.
- The 8kb memories require 13 address lines. Hence the address lines A0 - A12 are used for selecting the memory locations.
- The unused address lines A13, A14 and A15 are used as input to decoder 74LS138 (3-to-8-decoder) of memory IC. The logic low enables of this decoder are tied to IO / M(low) of 8085, so that this decoder is enabled for memory read/write operation. The

other enable pins of decoder are tied to appropriate logic levels permanently. The 4-outputs of the decoder are used to select memory IC's and the remaining 4 are kept for future expansion.

- The EPROM is mapped in the beginning of memory space from 0000H to 3FFF.
- The RAM is mapped at the end of memory space from C000 to FFFFH.
- There are five peripheral IC's to be interfaced to the system. The chip-select signals for these IC's are given through another 3-to-8 decoder 74LS138 (I/O decoder). The input to this decoder is A11, A12 and A13
- The address lines A13, A14 and A15 are logically ORed and applied to low enable of I/O decoder.
- The logic high enable of I/O decoder is tied to IO / M(low) signal of 8085, so that this decoder is enabled for I/O read/write operation.

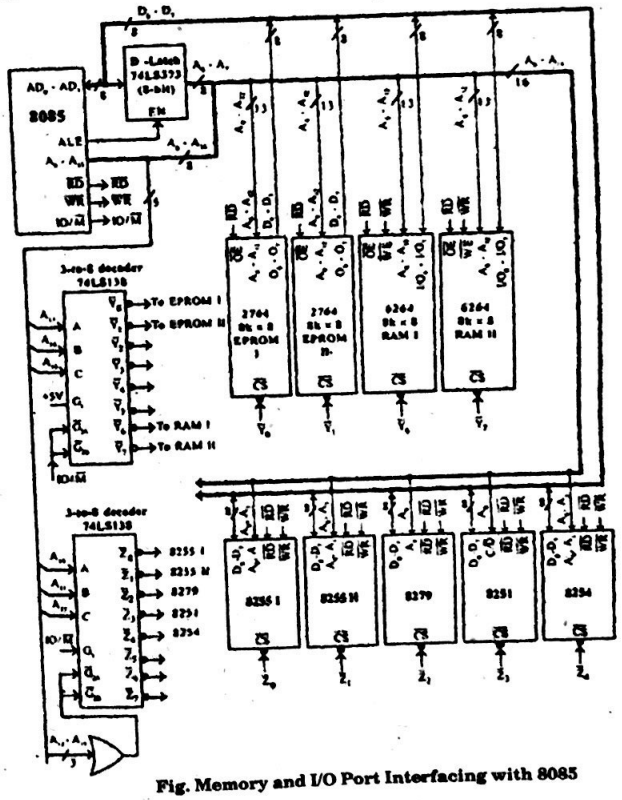


Fig. Memory and I/O Port Interfacing with 8085

Device	Bindary address															Hexa address
	Decoder input			Input to address pins of memory/8255												
	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
2764 EPROM	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0001
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0002
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1FFF
6264 RAM	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	E000
	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	E001
	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	E002
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	FFFF
8255I																	
PORT-A	0	1	0	×	×	×	×	×	×	×	×	×	×	0	0	×	4000
PORT-B	0	1	0	×	×	×	×	×	×	×	×	×	×	0	1	×	4002
PORT-C	0	1	0	×	×	×	×	×	×	×	×	×	×	1	0	×	4004
Control Register	0	1	0	×	×	×	×	×	×	×	×	×	×	1	1	×	4006
8255 II																	
PORT-A	0	1	1	×	×	×	×	×	×	×	×	×	×	0	0	×	6000
PORT-B	0	1	1	×	×	×	×	×	×	×	×	×	×	0	1	×	6002
PORT-C	0	1	1	×	×	×	×	×	×	×	×	×	×	1	0	×	6004
Control Register	0	1	1	×	×	×	×	×	×	×	×	×	×	1	1	×	6006
8255 III																	
PORT-A	1	0	0	×	×	×	×	×	×	×	×	×	×	0	0	×	8000
PORT-B	1	0	0	×	×	×	×	×	×	×	×	×	×	0	1	×	8002
PORT-C	1	0	0	×	×	×	×	×	×	×	×	×	×	1	0	×	8004
Control Register	1	0	0	×	×	×	×	×	×	×	×	×	×	1	1	×	8006

Note: The 'X' indicates that the address line is not used for the particular device and they are considered as zero.

Q.4. (a) Give the instruction sequence that compares the first 10 bytes beginning at STRG1 with the first ten bytes beginning at STRG 2 and branches to MATCH if they are equal, otherwise continues in sequence? (4)

Ans. This program to compare the given two strings for their equality using 8086 instructions with DOS interrupts.

Program:

```
NAME        BYTESEARCH
TITLE       8086 ALP FOR SEARCHING A BYTE IN AN ARRAY
DATA  SEGMENT
```



```
CR EQU 13
LF EQU 10
INMSG1    DB CR,LF,"ENTER THE STRING1: $"
INMSG2    DB CR,LF,LF,"ENTER THE STRING2: $"
STRNG1    DB 0BH,12 DUP(?)
STRNG2    DB 0BH,12 DUP(?)
SUCMSG    DB CR,LF,LF,"BOTH ARE SAME $"
FALMSG    DB CR,LF,LF,"DIFFERENT STRINGS $"
DATA      ENDS
CODE      SEGMENT
ASSUME CS:CODE, DS:DATA
START:    MOV AX,DATA
MOV DS,AX
MOV ES,AX
LEA DX,INMSG1
MOV AH,09
INT 21H
MOV DX,OFFSET STRNG1
MOV AH,0AH
INT 21H
LEA DX,INMSG2
MOV AH,09
INT 21H
MOV DX,OFFSET STRNG2
MOV AH,0AH
INT 21H
MOV SI, OFFSET STRNG1
MOV DI, OFFSET STRNG2
CLD
MOV CX,6H
REPE CMPSB
JZ SUCCESS
LEA DX,FALMSG
JMP DISPLAY
SUCCESS: LEA DX,SUCMSG
DISPLAY: MOV AH,09
INT 21H
MOV AH,4CH
INT 21H
CODE ENDS
END START
```

Q.4. (b) Explain LEA instruction of 8086. Explain how it differs from instruction with a suitable example?

Ans. LEA means Load Effective Address

MOV means Load Value

In short, LEA loads a pointer to the item you're addressing,whereas MOV loads actual value at that address.

The purpose of LEA is to allow one to perform a non-trivial address calculation store the result [for later usage]

LEA AX, [BP+SI+5] ; Compute address of value

MOV AX, [BP+SI+5] ; Load value at that address

Where there are just constants involved, MOV (through the assembler's const calculations) can sometimes appear to overlap with the simplest cases of usage of LE Where its useful is if you have a multi-part calculation with multiple base addres etc.

The instruction MOV reg,addr means read a variable stored at address addr i register reg. The instruction LEA reg,addr means read the address (not the varia stored at the address) into register reg.

Another form of the MOV instruction is MOV reg,immdata which means read immediate data (i.e. constant) immdata into register reg. Note that if the addr in LE reg,addr is just a constant (i.e. a fixed offset) then that LEA instruction is essentia exactly the same as an equivalent MOV reg,immdata instruction that loads the sa constant as immediate data.

Q.4. (c) If currently microprocessor 8086 is executing the JUMP instructi at memory location 2000H, then what will be operand of the JUMP instructi to jump at 1FF5 location?

Ans. JUMP 1FF5H.

SECOND TERM EXAMINATION [NOV.-2015]
FIFTH SEMESTER [B. TECH]
MICROPROCESSOR AND MICROCONTROLLERS [ETEC-305]

Time: 1 Hrs. **M M: 30**

Note: *Q.1. is compulsory attempt any two from the remaining.*

Q.1. Attempt any five. **(5x2)**

Q.1. (a) What is the function of data bus buffer block in the various programmable chips?

Ans. Data Bus Buffer: This three-state bi-directional 8-bit buffer is used to interface the programmable chips 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.

Q.1. (b) What do you mean by STB⁻ and IBF⁻ signals of 8255 in Input mode of Mode I.

Ans. Input Handshaking signals

1. $\overline{IBF}$ (Input Buffer Full) - It is an output indicating that the input latch contains information.

2. $\overline{STB}$ (Strobed Input) - The strobe input loads data into the port latch, which holds the information until it is input to the microprocessor via the IN instruction.

Q.1. (c) Explain the Read Back Command of 8254.

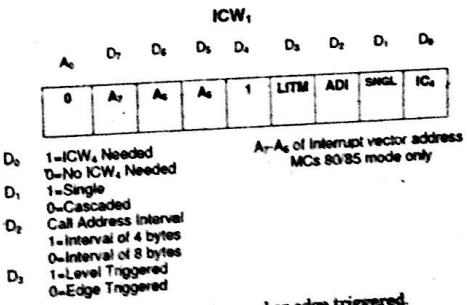
Ans. Read-Back Command:

(a) This command allows the user to check the count value, programmed Mode, and current states of the OUT pin and Null Count flag of the selected counter(s).

(b) This command is similar to several Counter Latch Commands, one for each counter latched.

Q.1. (d) Explain the SNGL and LTIM bits of ICW1 of 8259.

Ans. The SNGL bit in ICW1 indicates whether the 8259A is in cascade mode or not.



LITM bit in ICW1 indicates level triggered or edge triggered.

Q.1(e) Explain the DSR and DTR signals of 8251.

Ans. DSR (Input terminal)

This is an input port for MODEM interface. The input status of the terminal can be recognised by the CPU reading status words.

DTR (Output terminal)

This is an output port for MODEM interface. It is possible to set the status of DT by a command.

Q.1(f) Give the priority level of the various interrupts of 8051.

Ans. Interrupt priority upon reset when the 8051 is powered up, the priorities a assigned according to Table. From Table we see, that if external hardware interrupts and 1 are activated at the same time, external interrupt 0 (INTO) is responded to firs Only after INTO has been serviced is INT1 serviced, since INT1 has the lower priorit In reality, the priority scheme in the table is nothing but an internal polling sequence which the 8051 polls the interrupts in the sequence listed in Table and respond accordingly.

Table. 8051/52 Interrupt Priority Upon Rest

Highest to Lowest Priority	
External Interrupt 0	(INT0)
Timer Interrupt 0	(TF0)
External Interrupt 1	(INT 1)
Timer Interrupt 1	(TF 1)
Serial Communication	(R1 + T1)
Timer 2 (8052 only)	TF2

Q.1. (g) What do you mean by Bit and Byte addressable memory.

Ans. Bit Addresses for I/O and RAM: Many microprocessors such as the 386 or Pentium allow programs to access registers and I/O ports in byte size only. In other words, if you need to check a single bit of an I/O port, you must read the entire byte first and then manipulate the whole byte with some logic instructions to get hold of the desired single bit. This is not the case with the 8051. Indeed, one of the most important features of the 8051 is the ability to access the registers, RAM, and I/O ports in bits instead of bytes. This is a very unique and powerful feature for a microprocessor made in the early 1980s. In this section we show address assignment of bits of I/O, register, and memory, in addition to ways of programming them.

Bit-addressable RAM: Of the 128-byte internal RAM of the 8051, only 16 bytes are bit-addressable. The rest must be accessed in byte format. The bit-addressable RAM locations are 20H to 2FH. These 16 bytes provide 128 bits of RAM bit-addressability since 16×8= 128. They are addressed as 0 to 127 (in decimal) or 00 to 7FH. Therefore, the bit addresses 0 to 7 are for the first byte of internal RAM location 20H, and 8 to OFH are the bit addresses of the second byte of RAM location 21H, and so on. The last byte of 2FH has bit addresses of 78H to 7FH. Internal RAM locations 20 – 2FH are both byte-addressable and bit-addressable.

Q.2. (a) Interface an 8-bit DAC 0808 to 8086 through 8255 in IO mapped IO technique. Write an Assembly Language Program to generate square wave. (5)

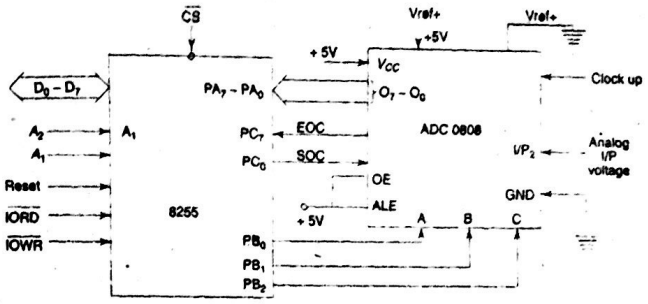
Ans. Figure shows the interfacing connections of ADC0808 with 8086 using 8255.

The analog input I/P 2 is used and therefore address pins A, B, C should be 0,1,0 respectively to select I/P2. The OE and ALE pins are already kept at +5V to select the ADC and enable the outputs. Port C upper acts as the input port to receive the EOC signal while port C lower acts as the output port to send SOC to the ADC. 12Port A acts as a 8-bit input data port to receive the digital data output from the ADC. The 8255 control word is written as follows:

Control word = 98 H

The required ALP is given as follows:

```
MOV AL, 98 H ; Initialisation of 8255
OUT CWR, AL
MOV AL, 02 H ; Select I/P 2 as analog
```


```
OUT Port B, AL ; input.
MOV AL,00H ; Give start of conversion
OUT Port C,AL ; pulse to the ADC
MOV AL,01H
OUT Port C,AL
Wait: IN AL, PortC ; Check for EOC by
RCR ; reading port C upper and rotate through carry
JNC Wait
IN AL, PortA ; If EOC, read digital equivalent in AL
HLT ; Stop
```

Q.2. (b) Interface stepper motor to 8086 using 8255 and write Assembly Language Program to rotate stepper motor in a clockwise direction and in full stepping. Assume that 8255 interfaced with 8086 in memory mapped IO. (5)

Ans. To Interface Stepper Motor to 8086 using 8255 and write Assembly Language Program to rotate Stepper Motor in Clockwise & Anticlockwise direction.

Stepper motor is a device used to obtain an accurate position control of rotating shafts. A stepper motor employs rotation of its shaft in terms of steps, rather than continuous rotation as in case of AC or DC motor. To rotate the shaft of the stepper motor, a sequence of pulses is needed to be applied to the windings of the stepper motor, in proper sequence. The numbers of pulses required for complete rotation of the shaft of the stepper motor are equal to the number of internal teeth on its rotor. The stator teeth and the rotor teeth lock with each other to fix a position of the shaft. With a pulse applied to the winding input, the rotor rotates by one teeth position or an angle x. the angle x may be calculated as.

$x = 3600 / \text{no. of rotor teeth}$

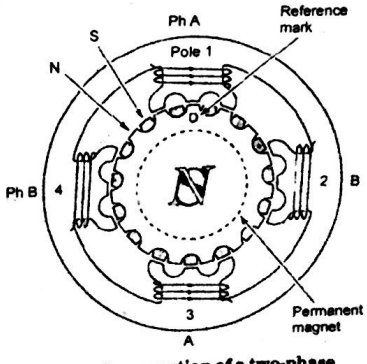


Fig. Cross-section of a two-phase hybrid motor

After the rotation of the shaft through angle x, the rotor locks it self with the ne tooth in the sequence on the internal surface of the stator. The typical schematic o typical stepper motor with four windings is as shown below.

The stepper motors have been designed to work with digital circuits. Binary lev pulses of 0-5V are required at its winding inputs to obtain the rotation of the shaft The sequence of the pulses can be decided, depending upon the required motion of th shaft. By suitable sequence of the pulses the motor can be used in three modes operation.

One phase ON (medium torque)

Two phase ON (high torque)

Half stepping (low torque)

Motion	Steps	A	B	C	D	Hex value
Clockwise	1	0	0	1	1	03 H
	2	0	1	1	0	06 H
	3	1	1	0	0	0C H
	4	1	0	0	1	09 H
	5	0	0	1	1	03 H
Anticlockwise	1	0	0	1	1	03 H
	2	1	0	0	1	09 H
	3	1	1	0	0	0C H
	4	0	1	1	0	06 H
	5	0	0	0	0	00 H

WORKING:

8255 is interfaced with 8086 in I/O mapped I/O. port C (PC0, PC1, PC2, PC3) is used to give pulse sequence to stepper motor. The 8255 provides very less current which will not be able to drive stepper motor coils so each of the winding of a stepper motor needs to be interfaced using high speed switching Darlington transistors with max 1A, 80V rating with heat sink, with the output port of 8255. Output the sequence in correct order to have the desired direction to rotate the motor.

Assembly Language Program to rotate Stepper Motor in Clockwise direction

```
MODEL SMALL
.STACK 100
.DATA
PORTA EQU FFC0H ; PORTA ADDRESS
PORTB EQU FFC2H ; PORTB ADDRESS
PORTC EQU FFC4H ; PORTC ADDRESS
CWR EQU FFC6H ; CONTROL PORT ADDRESS
PHASEC EQU 03H
PHASEB EQU 06H ; SEQUENCE IN SERIES TO ROTATE MOTOR
PHASED EQU 0CH ; IN CLOCKWISE DIRECTION
PHASEA EQU 09H
.CODE
START:
MOV AL,@DATA
MOV DX,CTL
OUT DX,AL
AGAIN
```

```
MOV DX,PORTC
OUT DX,AL
MOV CX,0FFFFH
UP:
LOOP UP
MOV AL,PHASEA
MOV DX,PORTC
OUT DX,AL
MOV CX,0FFFFH
UP1:
LOOP UP1
MOV AL,PHASED
MOV DX,PORTC
OUT DX,AL
MOV CX,0FFFFH
UP2:
LOOP UP2
MOV AL,PHASEB
MOV DX,PORTC
OUT DX,AL
MOV CX,0FFFFH
UP3:
LOOP UP3
JMP AGAIN ; REPEATE OUTPUT SEQUENCE
INT 03H
END START
```

Q.3 (a) Write a program to initialize 8253/54 in mode 1 to read and load lower-8-bits only assuming that 8253/54 is interfaced in IO mapped IO. Assume counter and CWR addresses. (4)

Ans. 8254 used in mode 1

Now CWR data is 33_H

Let Counter 0 address is 40
Counter 1 address is 42
Counter 2 address is 44
CWR address is 46
Program:

```
MOV AL, 33H
OUT 46, AL
MOV AL, 98
OUT 40, AL
MOV AL, 3AH
OUT 40, AL
HLT
```

Q.3. (b) Draw and explain the block diagram of 8259 A. (6)

Ans. The Intel 8259 is a Programmable Interrupt Controller (PIC) designed for the Intel 8085 and Intel 8086 microprocessors. The initial part was 8259, a later A suffix version was upward compatible and usable with the 8086 or 8088 processor. The 8259 combines multiple interrupt input sources into a single interrupt output to the host microprocessor, extending the interrupt levels available in a system beyond the one or

two levels found on the processor chip. The 8259A was the interrupt controller for the ISA bus in the original IBM PC and IBM PC AT.

Features of 8259:

8 levels of interrupts.
Can be cascaded in master-slave configuration to handle 64 levels of interrupts.
Internal priority resolver.
Fixed priority mode and rotating priority mode.
Individually maskable interrupts.
Modes and masks can be changed dynamically.
Accepts IRQ, determines priority, checks whether incoming priority > current level being serviced, issues interrupt signal.
In 8085 mode, provides 3 byte CALL instruction. In 8086 mode, provides 8 bit vector number.
Polled and vectored mode.
Starting address of ISR or vector number is programmable.
No clock required.

Refer Q.No.9(c) of End Term 2016

Q.4 (a) Explain clearly the operation of receiver section of 8251. (4)

Ans. The receiver section consists of three blocks — receiver buffer register, input register and the receiver control logic block. Serial data from outside world is delivered to the input register via RXD line, which is subsequently put into parallel form and placed in the receiver buffer register. When this register is full, the RXRDY (receiver ready) line becomes high. This line is then used either to interrupt the MPU or to indicate its own status. MPU then accepts the data from the register.

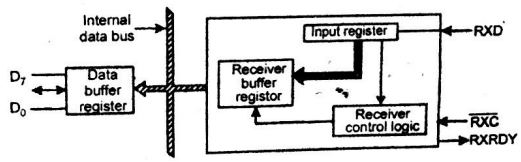


Fig. Receiver section

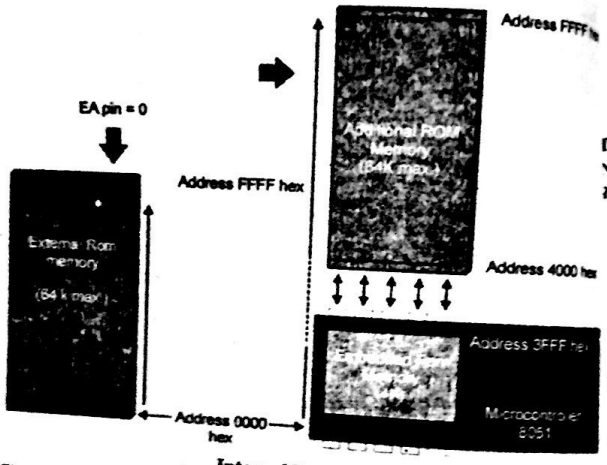
RXC line stands for receiver clock. This clock signal controls the rate at which bits are received by the input register. The clock can be set to 1, 16 or 64 times the baud in the asynchronous mode.

Q.4(b) Explain the internal RAM organization of 8051? Discuss how switching between register banks is possible? Give a sequence of instructions to switch from bank-0 to bank-2? (6)

Ans. 8051 Memory Organization

The 8051 microcontroller's memory is divided into Program Memory and Data Memory. Program Memory (ROM) is used for permanent saving program being executed, while Data Memory (RAM) is used for temporarily storing and keeping intermediate results and variables.

Up to 256 bytes of internal data memory are available depending on the 8051 derivative. Locations available to the user occupy addressing space from 0 to 7Fh, i.e. first 128 registers and this part of RAM is divided in several blocks. The first 128 bytes of internal data memory are both directly and indirectly addressable. The upper 128 bytes of data memory (from 0x80 to 0xFF) can be addressed only indirectly.

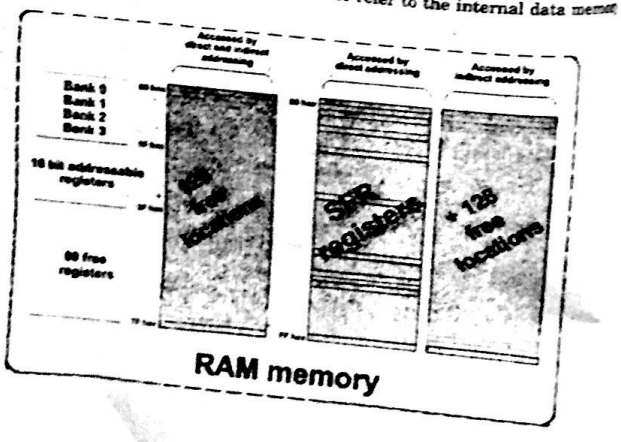


Internal Data Memory

Since internal data memory is used for CALL stack also and there is only 256 by splited over few different memory areas fine utilizing of this memory is crucial for h and compact code. See types efficiency also.

Memory block in the range of 20h to 2Fh is bit-addressable, which means that ea bit being there has its own address from 0 to 7Fh. Since there are 16 such registers, th block contains in total of 128 bits with separate addresses (Bit 0 of byte 20h has the b address 0, and bit 7 of byte 2Fh has the bit address 7Fh).

Three memory type specifiers can be used to refer to the internal data memor data, idata, and bdata.



END TERM EXAMINATION [DEC.-2015] FIFTH SEMESTER [B. TECH] MICROPROCESSOR AND MICROCONTROLLERS [ETEC-305]

M M: 75

Time: 3 Hrs.

Note: *Attempts any Five questions including Q.No. 1 which is compulsory. Select one question from each unit.*

Q.1 (a) Which mode of the 8254 is used to generate symmetric square waves? **(2.5 × 10 = 25)**

Ans. Mode 3

Q.1 (b) Explain the following signals of 8086:-

(*i*) DT/$\overline{R}$, (*ii*) $\overline{BHE}$, (*iii*) $\overline{DEN}$, (*iv*) Q_{S_0}, Q_{S_1} (*v*) MN/$\overline{MX}$, (*vi*) RESET.

Ans. (*i*) DT/$\overline{R}$: (Data Transmit or receive)is an output signal required in system that uses the data bus transceiver

(*ii*) $\overline{BHE}$ /S7 is used as bus high enable during the 1st clock cycle of an instruction execution. The $\overline{BHE}$ can be used in conjunction with AD0 to select the memory

(iii) DEN (O): Data Enable: It is provided as an output enable for the 8286/8287 in a minimum system which uses the transceiver $\overline{DEN}$ is active LOW during each memory and IO access. It will be low beginning with T2 until the middle of T4, while for a write cycle, it is active from the beginning of T2 until the middle of T4. It floats to tri-state off during local bus "hold acknowledge".

(iv) QS0, QS1 (O): Queue – Status: Queue Status is valid during the clock cycle after which the queue operation is performed. QS0, QS1 provide status to allow external tracking of the internal 8086 instruction queue. The condition of queue status is shown in table:

QS1	QS1	Characteristics
0	0	No operation
0	1	First byte of opcode from queue
1	0	Empty the queue
1	1	Subsequent byte from queue

(*v*) MN/$\overline{MX}$: is an input pin used to select one of this mode. When **MN/$\overline{MX}$ is high the 8086 operates in minimum mode** .In this mode the 8086 is configured to support small single processor system using a few devices that the system bus .when **MN/MX is low 8086 is** configured to support multiprocessor system.

(*vi*) RESET is the system set reset input signal it terminates all the activities it clear PSW, IP, DS, SS, ES and the instruction Queue.

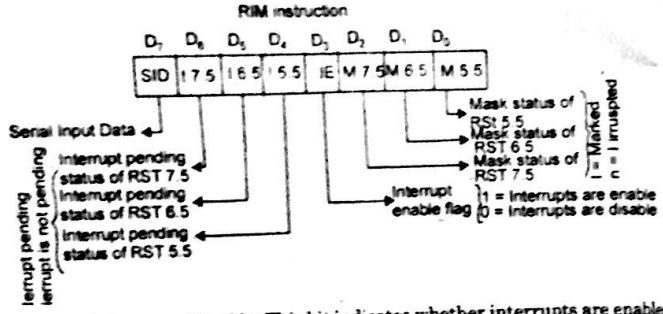
Q.1(*c*) Explain the function of RIM instruction?

Ans. The RIM instruction reads the following bits into the accumulator:

Bit D7 (SID-Serial Input Data) This is the input pin of the serial data interface **which is connected** to pin 5 of the 8085, and indicates the high/low status of that pin.

Bits D6-D4 (I 7.5, I 6.5, I 5.5) These bits indicates that an interrupt is pending for **these three** 8085 interrupts 7.5, 6.5, and 5.5. If interrupts 5.5 or 6.5 have been masked **off by bits** D0 or D1, bits D4 and D5 will not be set. Bit D6, which corresponds to the 7.5

interrupt, will be set on to indicate that an interrupt 7.5 was requested, even if it masked off.



Bit D3 (IE-Interrupt Enable) This bit indicates whether interrupts are enabled using the EI (Enable Interrupts) instruction, or disabled (0) using the DI (Disa is made up of 32 bits =2 Word's or 4 BYTE Interrupts) instruction.

Bits 2-D0 (M 7.5, M6.5, M5.5) **Mask status of interrupts 7.5, 6.5, and 5** Corresponds to bits D2-D0 of the SIM instruction. 1 if masked, 0 if enabled.

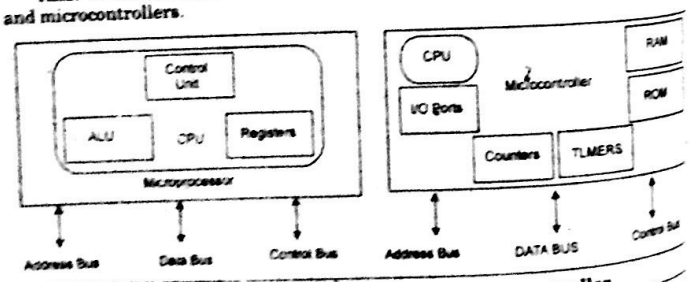
So the SIM and RIM instructions are typically used to either output to or in from 8085 serial interface, or enable/disable/read the interrupt masks for interru 7.5, 6.5, 5.5, *but usually not at the same time.*

Q.1(d) For comparing two blocks of data using string instructions, wh two segments are mandatory?

Ans. Extra Segment & Data Segment

Q.1(e) Discuss four major differences between a microprocessor and mirocontroller.

Ans. The following table shows some of the differences between microprocess and microcontrollers.



Microprocessor	Microcontroller
Microprocessor assimilates the function of a central processing unit (CPU) on to a single integrated circuit (IC).	Microcontroller can be considered as a computer which has a processor and other components in order to make computer
Microprocessors are mainly used in designing general purpose systems from small to large and complex	Microcontrollers are used in automatic controlled devices.

Microprocessors are basic components of personal computers	Microcontrollers are generally used in embedded systems
Computational capacity of microprocessor is very high. Hence can perform complex tasks	Less computational capacity when compared to microprocessors. Usually used for simpler tasks

Q.1(f) What is the difference in the functioning of the CY and OV flags?

Ans. CY, the carry flag: This flag is set whenever there is a carry out from the D7 bit. This flag bit is affected after an 8-bit addition or subtraction.

OV, the overflow flag: This flag is set whenever the result of a signed number operation is too large, causing the high-order bit to overflow into the sign bit. In general, the carry flag is used to detect errors in unsigned arithmetic operations. The overflow flag is only used to detect errors in signed arithmetic operations

Q.1(g) Explain the following assembler directive of 8086:

(*i*) DD (*ii*) ASSUME (*iii*) ORG (*iv*) ENDP (*v*) PROC

Ans. (*i*) **DD:** The DD directive is used to declare a DWORD - A DWORD double word

Declaration examples

Dword1 DW 12345678h

Dword2 DW 4294967295; 0FFFFFFFFh.

(*ii*) ASSUME Directive: The ASSUME directive is used to tell the assembler that the name of the logical segment should be used for a specified segment. The 8086 works directly with only 4 physical segments. a Code segment, a data segment, a stack segment, and an extra segment

Example: ASUME CS:CODE; This tells the assembler that the logical segment named CODE contains the instruction statements for the program and should be treated as a code segment.

(*iii*) ORG: Sets the current origin to a new value. This is used to set the program or register address during assembly. For example, ORG 0100h tells the assembler to assemble all subsequent code starting at address 0100h.

(*iv*) ENDP: ENDP directive is used along with the name of the procedure to indicate the end of a procedure to the assembler

Example: SQUARE_NUM PROCE; It start the procedure: Some steps to find the square root of a number SQUARE_NUM ENDP. Hear it is the End for the procedure

(*v*) PROC: The PROC directive is used to identify the start of a procedure. The term near or far is used to specify the type of the procedure

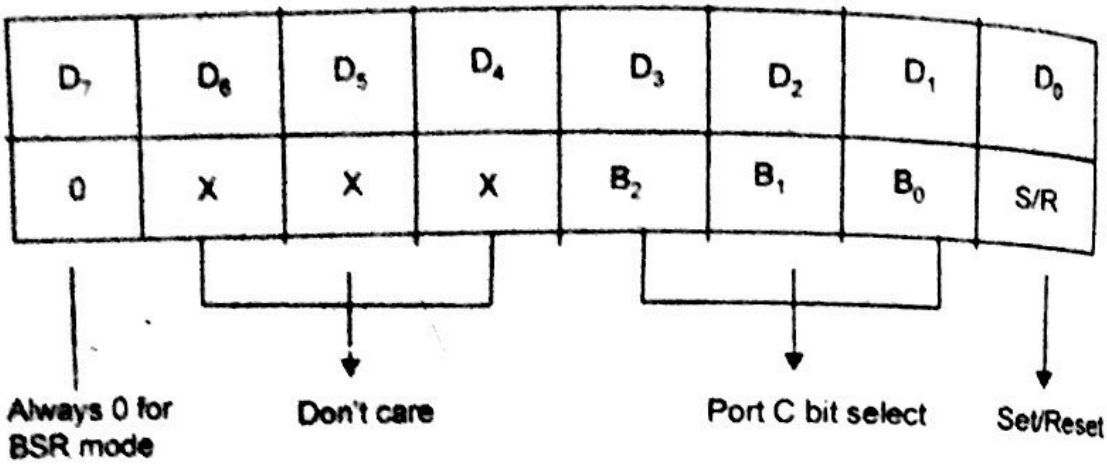
Example: SMART PROC FAR; This identifies that the start of a procedure named as SMART and instructs the assembler that the procedure is far . SMART ENDP This PROC is used with ENDP to indicate the break of the procedure

Q.1. (*h*) Explain the 8255 control word format for BSR mode.

Ans. The Bit Set/Reset (BSR) mode is applicable to port C only. Each line of port C (PC_0 - PC_7) can be set/reset by suitably loading the control word register. BSR mode and I/O mode are independent and selection of BSR mode does not affect the operation of other ports in I/O mode

8255 BSR mode

- D_7 bit is always 0 for BSR mode.
- Bits D_6, D_5 and D_4 are don't care bits
- Bits D_3, D_2 and D_1 are used to select the pin of Port C
- Bit D_0 is used to set/reset the selected pin of Port C



8255 COntrol Register formet for BSR Mode

Selection of port C pin is determined as follows:

B3	B2	B1	Bit/pin of port C selecte
0	0	0	PC_0
0	0	1	PC_1
0	1	0	PC_2
0	1	1	PC_3
1	0	0	PC_4
1	0	1	PC_5
1	1	0	PC_6
1	1	1	PC_7

Q.1(*i*) Name any four bit manipulation instruction in 8051.

Ans. CLR bit. Zero the specified bit.

SETB bit. Putting a specified bit.

CPL bit. Complement the bit indicated.

Bit destino MOV bit procedencia. Transfer or move a bit.

ALU: ALU performs the airthmetic operations and logical operation.

Flag Registers: It consists of 5 flip flop which changes its status according its the result stored in an accumulator. It is also known as **status registers.** It i connected to the ALU.

There are five flip-flops in the flag register are as follows:

1.Sign(S) 2.zero(z)
3.Auxiliary carry(AC) 4.Parity(P)
5.Carry(C)

The bit position of the flip flop in flag register is:

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
S	Z	X	AC	X	P	X	CY

All of the three flip flop set and reset according to the stored result in th accumulator.

1.Sign: If D7 of the result is 1 then sign flag is set otherwise reset. As we know that a number on the D7 always desides the sign of the number.

if D7 is 1: the number is negative.

if D7 is 0: the number is positive.

2. Zeros(Z): If the result stored in an accumulator is zero then this flip flop is set otherwise it is reset.

3. Auxiliary carry(AC): If any carry goes from D3 to D4 in the output then it is set otherwise it is reset.

4. Parity(P): If the no of 1's is even in the output stored in the accumulator then it is set otherwise it is reset for the odd.

5. Carry(C): If the result stored in an accumulator generates a carry in its final output then it is set otherwise it is reset.

Instruction registers(IR): It is a 8-bit register. When an instruction is fetched from memory then it is stored in this register.

Instruction Decoder: Instruction decoder identifies the instructions. It takes the informations from instruction register and decodes the instruction to be performed.

Program Counter: It is a 16 bit register used as memory pointer. It stores the memory address of the next instruction to be executed. So we can say that this register is used to sequencing the program. Generally the memory have 16 bit addresses so that it has 16 bit memory.

The program counter is set to 0000H.

Stack Pointer: It is also a 16 bit register used as memory pointer. It points to the memory location called stack. Generally stack is a reserved portion of memory where information can be stores or taken back together.

Timing and Control Unit: It provides timing and control signal to the microprocessor to perform the various operation.It has three control signal. It controls all external and internal circuits. It operates with reference to clock signal. It synchronizes all the data transfers.

There are three control signal:

1. ALE: Airthmetic Latch Enable, It provides control signal to synchronize the components of microprocessor.

2. RD: This is active low used for reading operation.

3.WR-This is active low used for writing operation.

There are three status signal used in microprocessor S0, S1 and IO/M. It changes its status according the provided input to these pins.

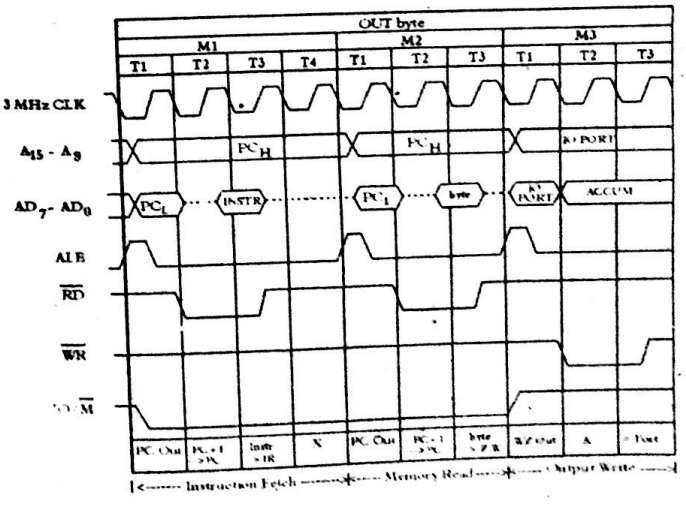
IO/M (Active Low)	S1	S2	Data Bus Status (output)
0	0	0	Halt
0	0	1	Memory Write
0	1	0	Memory Read
1	0	1	IO Write
1	1	0	IO Write
0	1	1	Opcode fetch
1	1	1	Interrupt acknowledge

Serial Input Output Control-There are two pins in this unit. This unit is used for serial data communication.

Interrupt Unit-There are 6 interrupt pins in this unit. Generally an external hardware is connected to these pins. These pins provide interrupt signal sent by external hardware to microprocessor and microprocessor sends acknowledgement for receiving the interrupt signal. Generally INTA is used for acknowledgement.

Q.2 (b) Draw and explain the timing diagram of instruction OUT port address. **(6.5)**

Ans. The above figure gives an example of 8085 timing, showing the value of external control signals. Of course, at the same time internal control signals are being generated by the control unit to control internal data transfers. Three machine cycles (M1, M2, M3) are needed. During the first, the OUT instruction is fetched. The second machine



cycle fetches the second half of the instruction, which contains the number of the I/O device selected for output. During the third cycle, the contents of the AC are written out to the selected device over the data bus.

The start of each machine cycle is signaled by the Address Latch Enabled (ALE) pulse from the control unit. The ALE pulse alerts external circuits. During timing state T1, of machine cycle M1, the control unit sets the Io/M signal to indicate that this is a memory operation. Also, the control unit causes the contents of the PC to be placed on the address bus (A15 - A8) and address/data bus (AD7 - AD0). With the falling edge of the ALE pulse, the other modules on the bus store the address.

During timing state T2, the addressed memory module places the contents of the addressed memory location on the address / data bus. The control unit sets the Read Control (RD) signal to indicate a read, but it waits until T3 to copy the data from the bus. This gives the memory module time to put the data on the bus and for the signal levels to stabilize. The final state, T4, is a *bus idle* state during which the CPU decodes the instruction. The remaining machine cycle proceed in a similar fashion.

OR

Q.3 (a) Differentiate between the following: (6)

Q.3. (a) (i) Maskable and Non Maskable interrupt.

Ans. (*i*) The interrupt which can be ignored by the processor ,while performing its operations are called maskable interrupts. generally maskable interrupts are the interrupts that comes from the pheripheral devices. where as the non maskable interrupts are the interrupts which cannot be ignored. generally these type of interrupts are specified to be software interrupts. The examples of maskable are: mouseclick,memoryread etc the examples of non_maskable are:powerfailure,software corruptedetc

Q.3. (a) (ii) Vectored and Non Vectored Interrupt.

Ans. (*ii*) Interrupts stop the routine procedure of incrementing Program counter by 1 and instead take the Program Counter to some other specific value. In laymen terms, it basically stops the current flow of the program and instead starts executing something else. Now this something else can be something which you define which is stored at some particular location decided by you or it can be something which is preloaded in the microprocessor at some already default location. The first one is called non-vectored interrupt whereas the second one is called vectored interrupt.

Basically in vectored interrupt processor automatically generates the new address Program Counter (PC) is taken to eg. when 8085 in interrupted with RST 5.5 pin it automatically takes PC to the address 002CH. On other hand, if user has to provide this subroutine, then it is unvectored interrupt eg. in the case of CALL instruction in 8085.

Q.3. (a) (iii) PCHL and SPHL.

Ans. (iii) SPHL: This instruction copies H and L register to the stack pointer. The instruction loads the contents of the H and L registers into the stack pointer register, the contents of the H register provide the highorder address and the contents of the L register provide the low-order address. The contents of the H and L registers are not altered.

Eg: SPHL

PCHL: Load program counter with HL contents. The contents of registers H and L are copied into the program counter. The contents of H are placed as the high-order byte and contents of L as the low order byte.

Q.3. (b) An array of data bytes are stored in memory location 2000H to 2009H. Draw a flowchart and write an assembly language program to transfer the entire block of data to new memory locations starting at 3070 H. (6.5)

Ans. A block of data consisting of 10 bytes is stored in memory starting at 2000H. This block is to be shifted (relocated) in memory from 3070H onwards. Do not shift the block or part of the block anywhere else in the memory.

- MVI C, 0AH :" Initialize counter"
- LX I H, 2000H :" Initialize source memory pointer 314FH"
- LXI D, 3070H :" Initialize destination memory pointer"
- BACK: MOV A, M :" Get byte from source memory block"
- STAX D :" Store byte in the destination memory block"
- DCX H :" Decrement source memory pointer"
- DCX :" Decrement destination memory pointer"
- DCR C :" Decrement counter"
- JNZ BACK :" If counter 0 repeat"
- HLT :" Stop execution"

UNIT II

Q.4 (a) Explain the following instruction: (6)
(i) CWD (ii) SAHF (iii) DAS (iv) IRET (v) IDIV (vi) NEG.

Ans. (i) CWD: Convert Word to Double word; No operands
Algorithm: if high bit of AX = 1
then DX = 65535 (0FFFFh)
else DX = 0
Example: MOV DX, 0; DX = 0

Ans. (ii) SAHF: Store AH register into low 8 bits of Flags register.

Ans. (iii) DAS: Decimal adjust After Subtraction. Corrects the result of subtraction of two packed BCD values.

Ans. (iv) IRET: Interrupt Return.

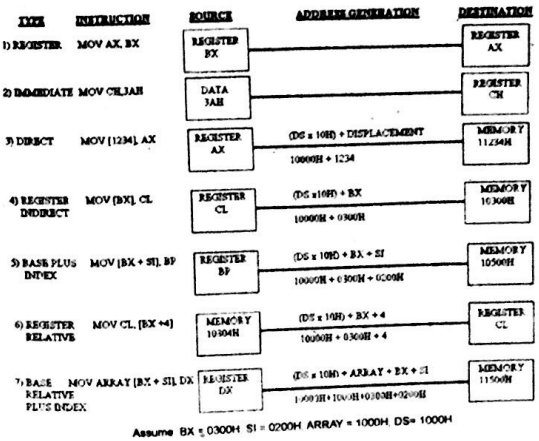
Ans. (v) IDIV: Signed divide.

Ans. (vi) NEG: Negate. Makes operand negative (two's complement).

Q.4 (b) What do you understand by addressing modes? Discuss all 12 addressing modes of 8086 microprocessor with supporting one example of each. (6.5)

Ans.

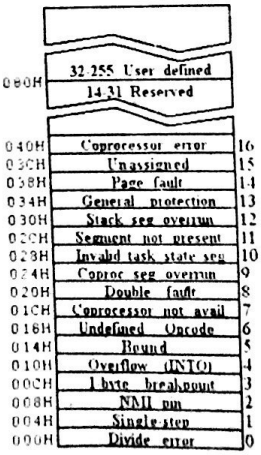
8086 ADDRESS MODES



OR

Q.5 (a) Draw and explain the interrupt vectord table of 8086 microprocessor. Also explain the interrupt response sequence of 8086.

Ans. Interrupts of 8086 The 8086 microprocessor has 256 types of interrupts. INTEL has assigned a type number to each interrupt. The type numbers are in the range of 0 to 255. The 8086 processor has dual facility of initiating these 256 interrupts. The interrupts can be initiated either by executing "INT n" instruction where n is the type number or the interrupt can be initiated by sending an appropriate signal to INTR input pin of the processor. For the interrupts initiated by software instruction "INT n" the type number is specified by the instruction itself. When the interrupt is initiated through INTR pin, then the processor runs an interrupt acknowledge cycle to get the type number (i.e., the interrupting device should supply the type number through D0–D7 lines when the processor requests for the same through interrupt acknowledge cycle). The kinds of interrupts and their designated types are summarized in fig. 4.5 by illustrating the layout of their pointers within the memory. Only the first five types have explicit definitions; the other types may be used by interrupt instructions or external interrupts. From the figure it is seen that the type associated with a division error interrupt is 0. Therefore, if a division by 0 is attempted, the processor will push the current contents of the PSW, CS and IP into the stack, fill the IP and CS registers from the addresses 00000 to 00003, and continue executing at the address indicated by the new contents of IP and CS. A division error interrupt occurs any time a DIV or IDIV instruction is executed with the quotient exceeding the range, regardless of the IF (Interrupt flag) and TF (Trap flag) status. The type 1 interrupt is the single-step interrupt (Trap interrupt) and is the only interrupt controlled by the TF flag. If the TF flag is enabled, then an interrupt will occur at the end of the next instruction that will cause a branch to the location indicated by the contents of 00004H to 00007H. The single step interrupt is used primarily for debugging which gives the programmer a snapshot of his program after each instruction is executed.



The interrupt vector table is located in the first 1024 bytes of memory at addresses 000000H through 0003FFH.

There are 256 4-byte entries (segment and offset in real mode).

Seg high	Seg low	Offset high	Offset low
Byte 3	Byte 2	Byte 1	Byte 0

The type 2 interrupt is the non maskable external interrupt. It is the only external interrupt that can occur regardless of the IF flag setting. It is caused by a signal sent to the CPU through the non maskable interrupt pin. The remaining interrupt types correspond to interrupts instructions imbedded in the interrupt program or to external interrupts. The interrupt instructions are summarized below and their interrupts are not controlled by the IF flag.

Interrupt Response

As said earlier 8086 checks for interrupts after executing each instructions. These checking can be organized in following steps:

(1) Decrements the stack pointer by 2 and pushes the flag register.

(2) Disables INTR interrupt input by clearing IF flag (interrupt flag).

(3) Resets the trap flag (TF).

(4) Decrements the stack by 2 and pushes current code segment register onto the stack.

(5) Decrements the stack by 2 and pushes instruction pointer IP onto the stack.

(6) Does an indirect far jump to the start of the procedure to respond to the interrupt.

Q.5 (b) (i) Write a program to generate a delay of five minutes using an 8086 system that runs on 5 MHz frequence. (3)

(ii) Explain how many times the following loop will be executed. (3.5)

```
LXI B, 007 H
LOOP; DCX B
MOV A, B
ORAC
JNZ LOOP
```

Ans. (i) Time delay= 5 minutes, and frequency= 5MHz

	Program:	T STATES REQUIRED
	MOV AX, COUNT1	4
L1:	MOV BX, COUNT2	4
L2:	NOP	3
	DEC BX	2
	JNZ L2	16/4
	DEC AX	2
	JNZ L1	16/4
	RET	8

Here we have two nested loops for decrementing the two counter register
Let the count2 = FFFFH or 65536 in decimal
Let the execution time for inner loop be t1
t1 = [4*0.2 + (3 + 2 + 16)*65535*0.2] microsec
Let the execution time for outer loop once be t2
t2 = [t1 + (2*16)*0.2]micro sec
required delay td = 5*60 = 300sec
count 1= td/t2
exact delay = t1*count1

Ans. (ii) 7 times.

UNIT-III

Q.6.(a) Explain the various modes of operation of 8253. (8)

Ans. Operation Modes: The D3, D2, and D1 bits of the Control Word set the operating mode of the timer. There are 6 modes in total; for modes 2 and 3, the D3 bit is

ignored, so the missing modes 6 and 7 are aliases for modes 2 and 3. Notice that, fo modes 0, 2, 3 and 4, GATE must be set to HIGH to enable counting. For mode 5, th rising edge of GATE starts the count. For details on each mode, see the reference link

Mode 0 (000): Interrupt on Terminal Count: Mode 0 is used for the generation o accurate time delay under software control. In this mode, the counter will start countin from the initial COUNT value loaded into it, down to 0. Counting rate is equal to th input clock frequency.

The OUT pin is set low after the Control Word is written, and counting starts on clock cycle after the COUNT programmed. OUT remains low until the counter reache 0, at which point OUT will be set high until the counter is reloaded or the Control Wor is written. The Gate signal should remain active high for normal counting. If Gate goe low counting gets terminated and current count is latched till Gate pulse goes high again. the first byte of the new count when loaded in the count register,stop the previou count.

Mode 1 (001): Programmable One Shot

In this mode 8253 can be used as Monostable Multivibrator. GATE input is used as trigger input.

OUT will be initially high. OUT will go low on the Clock pulse following a trigger to begin the one-shot pulse, and will remain low until the Counter reaches zero. OUT will then go high and remain high until the CLK pulse after the next trigger.

After writing the Control Word and initial count, the Counter is armed. A trigger results in loading the Counter and setting OUT low on the next CLK pulse, thus starting the one-shot pulse. An initial count of N will result in a one-shot pulse N CLK cycles in duration.

The one-shot is retriggerable, hence OUT will remain low for N CLK pulses after any trigger. The one-shot pulse can be repeated without re-writing the same count into the counter. GATE has no effect on OUT. If a new count is written to the Counter during a oneshot pulse, the current one-shot is not affected unless the counter is retriggered. In that case, the Counter is loaded with the new count and the oneshot pulse continues until the new count expires.

Mode 2 (X10): Rate Generator

In this mode, the device acts as a divide-by-n counter, which is commonly used to generate a real-time clock interrupt.

Like other modes, counting process will start the next clock cycle after COUNT is sent. OUT will then remain high until the counter reaches 1, and will go low for one clock pulse. OUT will then go high again, and the whole process repeats itself.

The time between the high pulses depends on the preset count in the counter's register, and is calculated using the following formula:

$$\text{Value to be loaded into counter} = \frac{f_{input}}{f_{output}}$$

Note that the values in the COUNT register range from n to 1; the register never reaches zero.

Mode 3 (X11): Square Wave Generator

This mode is similar to mode 2. However, the duration of the high and low clock pulses of the output will be different from mode 2.

Suppose is the number loaded into the counter (the COUNT message), the output will be

- high for $\frac{n}{2}$ counts, and low for $\frac{n}{2}$ counts, if n is even.

- high for $\frac{n+1}{2}$ counts, and low for $\frac{n-1}{2}$ counts, if n is odd.

Mode 4 (100): Software Triggered Strobe

After Control Word and COUNT is loaded, the output will remain high until the counter reaches zero. The counter will then generate a low pulse for 1 clock cycle (a strobe) – after that the output will become high again.

Mode 5 (101): Hardware Triggered Strobe

This mode is similar to mode 4. However, the counting process is triggered by the GATE input.

After receiving the Control Word and COUNT, the output will be set high. Once the device detects a rising edge on the GATE input, it will start counting. When the counter reaches 0, the output will go low for one clock cycle – after that it will become high again, to repeat the cycle on the next rising edge of GATE.

Q.6. (b) Explain the block diagram of 8279. **(6.5)**

Ans. 8279 Programmable Keyboard/display interface

This is a hardware approach to interface a matrix keyboard and a multiplexed display. The display can be set as a right entry or a left entry.

8279 Block Diagram

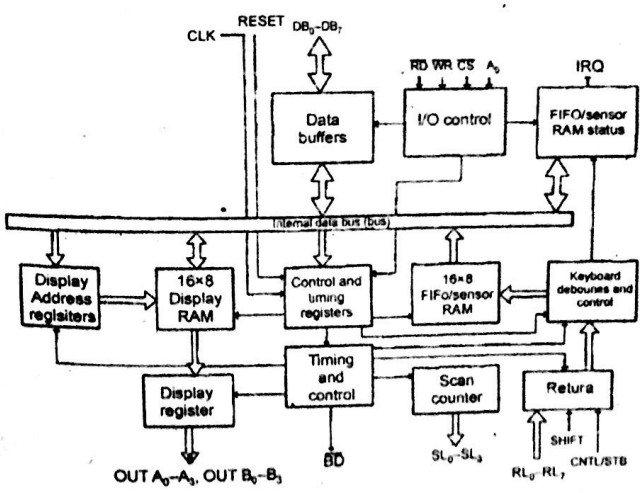


Fig. 8279 Block Diagram

Keyboard Section: This sectio has 8 lines. RL0 - RL7. Plus 2 additional lines. Shift and CNTL/STB. The keys are automatically debounced and keyboard can operate in two modes:

>>two key lockout mode or

>>N-key rollover.

In two key lockout mode if 2 keys are pressed simultaneously only firstke recongnized.

In N key rollover mode, simultaneous keys are recongnized and stored in inter buffer: it can also be set up so that no key is recongnized until only one key is remai pressed.

This has a FIFO RAM.

The status logic keeps track of number of entries and provides IRQ(interr request) signal when

FIFO is empty.

DISPLAY SECTION

This section has 8 output lines divided into 2 groups of 4. A0 - A3 and B0 - These lines can be used in both ways 8 lines or 2 sets of 4 lin The display can be blanked using BD line. The section has 16x8 display RAM.

SCAN SECTION

This section has scan counter and 4 scan lines. SL0 - SL3. These 4 scan lines can decoded using a 4 - 16 decoder to generate 16 lines for scannin These 16 lines can be connected to rows of a matrix keyboard and digit driveers multiplexed display.

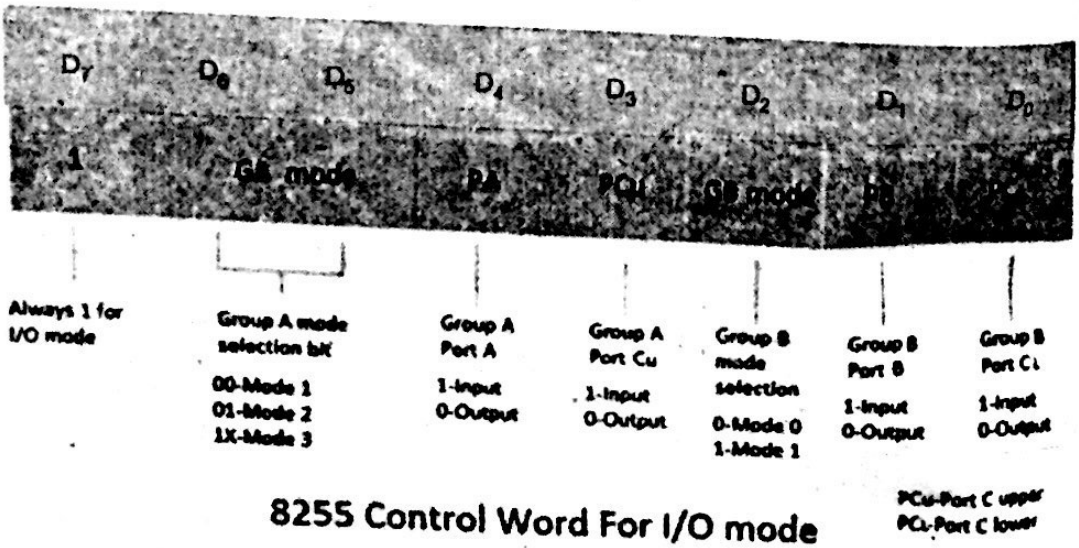
MPU INBTERFACE SECTION

This section has 8 bidirectional lines. DB0 - DB7. 1 interrupt request line(IRQ 6 lines for interfacing including buffer address lines A0. When A0 is high - signals a interpreted as control word or status. When A0 is low - signal is interpreted as da IRQ goes high whenever data is ready to be loaded into MPU.

OR

Q.7 (a) Write instruction in 8086 to define port A as I/P port in mode 0, port as O/P port in mode 1, port C upper as I/P in mode 0 and port C lower as O/P por in mode 1. 8255 port A address in 0840 H.

Ans. CWR of 8255



8255 Control Word For I/O mode

now CWR data is $9C_H$

Port A address is 0840

Port B address is 0842

Port C address is 0844

Table 1 Operation between a CPU and 8251

Control Words

There are two types of control word.

1. Mode instruction (setting of function)
2. Command (setting of operation)

1. Mode Instruction: Mode instruction is used for setting the function of the 8[illegible]. Mode instruction will be in "wait for write" at either internal reset or external re[illegible]. That is, the writing of a control word after resetting will be recognized as a "m[illegible] instruction."

Items set by mode instruction are as follows:

- Synchronous/asynchronous mode
- Stop bit length (asynchronous mode)
- Character length
- Parity bit
- Baud rate factor (asynchronous mode)
- Internal/external synchronization (synchronous mode)
- Number of synchronous characters (Synchronous mode)

The bit configuration of mode instruction is shown in Figures 2 and 3. In the case synchronous mode, it is necessary to write one-or two byte sync characters. If sy characters were written, a function will be set because the writing of sync charact constitutes part of mode instruction.

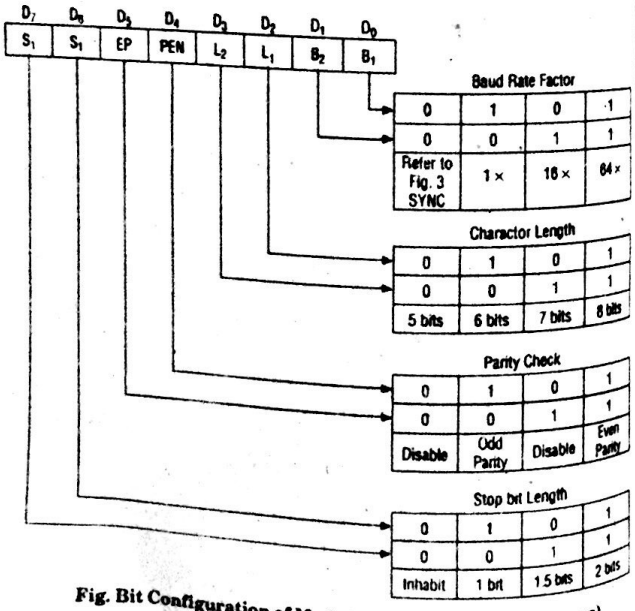


Fig. Bit Configuration of Mode Instruction (Asynchronous)

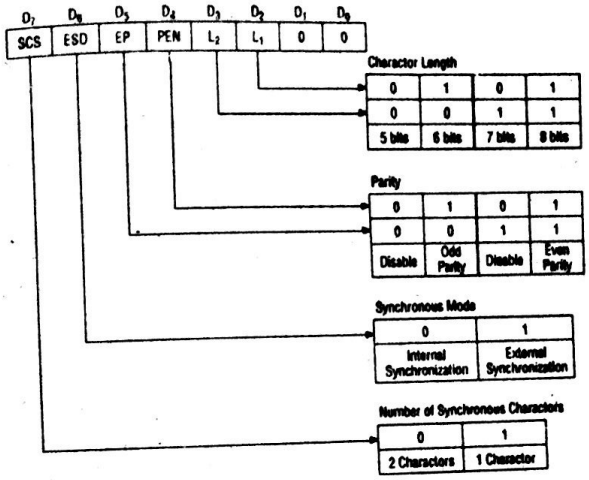


Fig. Bit Configuration of Mode Instruction (Synchronous)

2. Command: Command is used for setting the operation of the 8251. It is possible to write a command whenever necessary after writing a mode instruction and sync characters.

Items to be set by command are as follows:

- Transmit Enable/Disable
- Receive Enable/Disable
- DTR, RTS Output of data.
- Resetting of error flag.
- Sending to break characters
- Internal resetting
- Hunt mode (synchronous mode)

UNIT-IV

Q.8 (a) Draw and explain the internal architecture of 8051 microcontroller. (6)

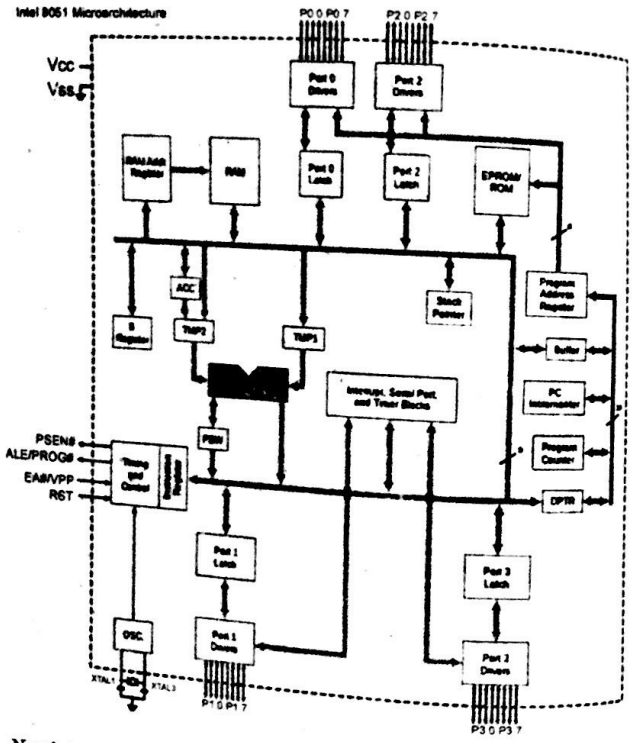
Ans. Major components of Intel 8051 microcontroller

The 8051 microcontroller is an 8-bit microcontroller. The major components of 8051 microcontroller and their functions.

An 8051 microcontroller has the following 12 major components:

1. ALU (Arithmetic and Logic Unit)
2. PC (Program Counter)
3. Registers
4. Timers and counters
5. Internal RAM and ROM
6. Four general purpose parallel input/output ports

7. Interrupt control logic with five sources of interrupt
8. Serial date communication
9. PSW (Program Status Word)
10. Data Pointer (DPTR)
11. Stack Pointer (SP)
12. Data and Address bus.



Now let us see the functions of each of these components

1. ALU: All arithmetic and logical functions are carried out by the ALU.

Addition, subtraction with carry, and multiplication come under arithmetic operations.

Logical AND, OR and exclusive OR (XOR) come under logical operations.

2. Program Counter (PC): A program counter is a 16-bit register and it has no internal address. The basic function of program counter is to fetch from memory the address of the next instruction to be executed. The PC holds the address of the next instruction residing in memory and when a command is encountered, it produces that instruction. This way the PC increments automatically, holding the address of the next instruction.

3. Registers: Registers are usually known as data storage devices. 8051 microcontroller has 2 registers, namely Register A and Register B. Register A serves as an accumulator while Register B functions as a general purpose register. These registers are used to store the output of mathematical and logical instructions.

The operations of addition, subtraction, multiplication and division are carried out by Register A. Register B is usually unused and comes into picture only when multiplication and division functions are carried out by Register A. Register A also involved in data transfers between the microcontroller and external memory.

8051 microcontroller also has 7 Special Function Registers (SFRs). They are

1. Serial Port Data Buffer (SBUF)
2. Timer/Counter Control (TCON)
3. Timer/Counter Mode Control (TMOD)
4. Serial Port Control (SCON)
5. Power Control (PCON)
6. Interrupt Priority (IP)
7. Interrupt Enable Control (IE)

4. Timers and Counters: Synchronization among internal operations can be achieved with the help of clock circuits which are responsible for generating clock pulses. During each clock pulse a particular operation will be carried out, thereby, assuring synchronization among operations. For the formation of an oscillator, we are provided with two pins XTAL1 and XTAL2 which are used for connecting a resonant network in 8051 microcontroller device. In addition to this, circuit also consists of four more pins. They are,

Internal operations can be synchronized using clock circuits which produce clock pulses. With each clock pulse, a particular function will be accomplished and hence synchronization is achieved. There are two pins XTAL1 and XTAL2 which form an oscillator circuit which connect to a resonant network in the microcontroller. The circuit also has 4 additional pins .

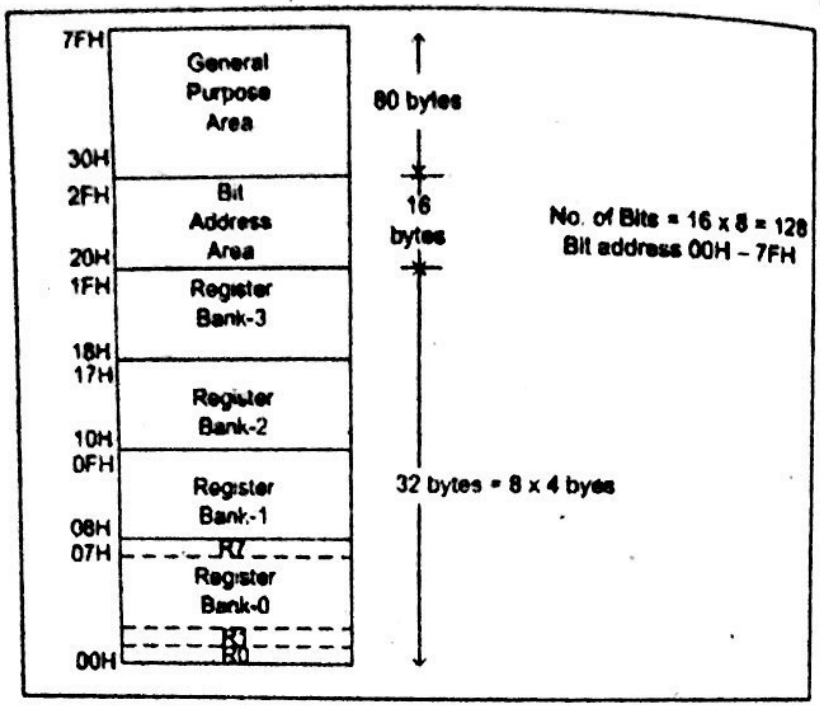
1. EA: External enable
2. ALE: Address latch enable
3. PSEN: Program store enable and
4. RST: Reset.

Quartz crystal is used to generate periodic clock pulses.

5. Internal RAM and ROM

ROM: A code of 4K memory is incorporated as on-chip ROM in 8051. The 8051 ROM is a non-volatile memory meaning that its contents cannot be altered and hence has a similar range of data and program memory, i.e, they can address program memory as well as a 64K separate block of data memory.

RAM: The 8051 microcontroller is composed of 128 bytes of internal RAM. This is a volatile memory since its contents will be lost if power is switched off. These 128 bytes of internal RAM are divided into 32 working registers which in turn constitute 4 register banks (Bank 0-Bank 3) with each bank consisting of 8 registers (R0 - R7). There are 128 addressable bits in the internal RAM.



6. Four General Purpose Parallel Input/Output Ports: The 8051 microcontroller has four 8-bit input/output ports. These are: PORT P0: When there is no external memory present, this port acts as a general purpose input/output port. In the presence of external memory, it functions as a multiplexed address and data bus. It performs a dual role.

PORT P1: This port is used for various interfacing activities. This 8-bit port is a normal I/O port i.e. it does not perform dual functions.

PORT P2: Similar to PORT P0, this port can be used as a general purpose port when there is no external memory but when external memory is present it works in conjunction with PORT PO as an address bus. This is an 8-bit port and performs dual functions.

PORT P3: PORT P3 behaves as a dedicated I/O port

Q.8 (b) Write a Program to generate a square wave of 10 kHz using Timer 1 in mode 1 using 8051. (6.5)

Ans. 10 KHz square wave using 8051 timer.

```
MOV P1,#0000000B
MOV TMOD,#0000001B
MAIN: SETB P1.0
        ACALL DELAY
        CLR P1.0
        ACALL DELAY
        SJMP MAIN
DELAY: MOV TH0,#0FFH
        MOV TL0,#0CEH
        SETB TR0
HERE:JNB TF0,HERE
        CLR TR0
        CLR TF0
        SETB P1.0
RET
END
```

- Timer 0 overflow interrupt- TF0
- Timer 1 overflow interrupt- TF1
- External hardware interrupt- INT0
- External hardware interrupt- INT1
- Serial communication interrupt- RI/TI

The Timer and Serial interrupts are internally generated by the microcont whereas the external interrupts are generated by additional interfacing dev switches that are externally connected to the microcontroller. These external inte can be edge triggered or level triggered. When an interrupt occurs, the microcont executes the interrupt service routine so that memory location corresponds interrupt that enables it. The Interrupt corresponding to the memory location is in the interrupt vector table below.

Interrupt vector Table

Interrupt Number	Interrupt Description	Address
0	External INT 0	0003h
1	Timer/Counter 0	0006h
2	External INT 1	00013h
3	Timer Counter 1	001Bh
4	Serial Port	0023h

Interrupt Structure of 8051 Microcontroller

Upon 'RESET' all the interrupts get disabled, and therefore, all these interr must be enabled by a software. In all these five interrupts, if anyone or all are activa this sets the corresponding interrupt flags as shown in the figure. All these interru can be set or cleared by bit in some special function register that is Interrupt Enab (IE), and this in turn depends on the priority, which is executed by IP interrupt prior register.

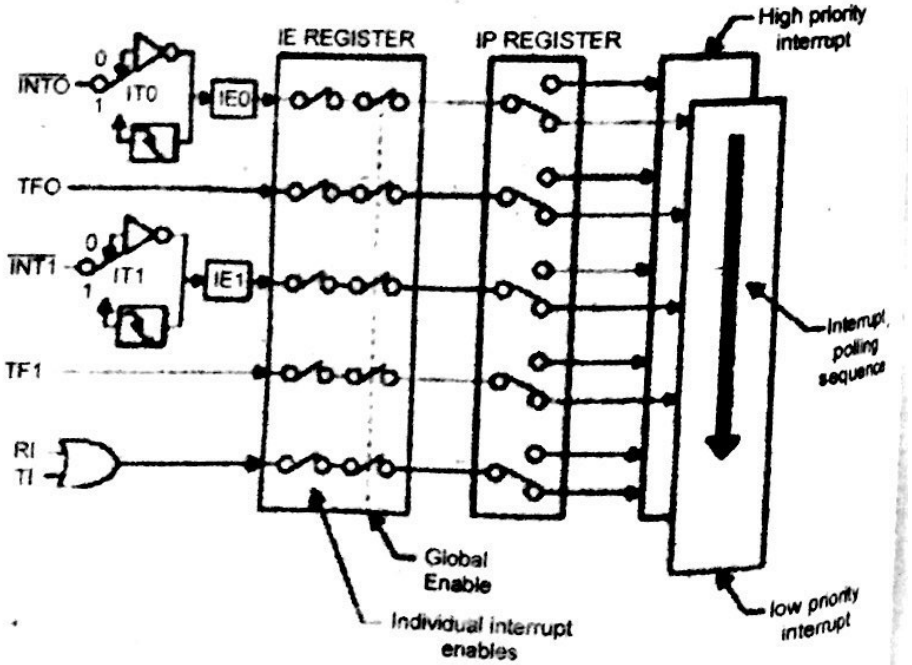
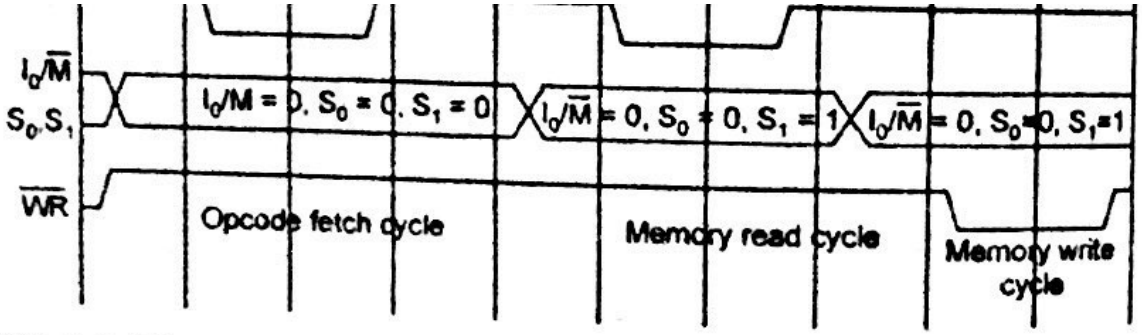


Fig. Interrupt structure of 8051 microcontroller



Q.3. (a) Identify addressing modes used in each of the following 8086 instructions. (2)

(i) MOV BX, 0354H

(ii) ADD AL, 04 [BX]

(iii) MOV AX, [BX] [SI]

(iv) MOV AX, [BX+SI+04)

Ans. (i) MOV BX, 0354H; Immediate Addressing Mode

(ii) ADD AL,04 [BX]; Register Relative Addressing Mode

(iii) MOV AX, [BX] [SI]; Base Plus Addressing Mode

(iv) MOV AX, [BX+SI+04]: Base Relative Plus Index Addressing Mode

Q.3. (b) Write a program to generate Fibonacci series for microprocessor 8086. (4)

Ans. Fibonacci series

Steps to be performed

→ Clear accumulator, take Ist number = 0

→ Take 2nd number = 1

→ Add Ist number with 2nd number

→ Store the sum

→ Add sum with next and so on

→ Continue till counter = 0

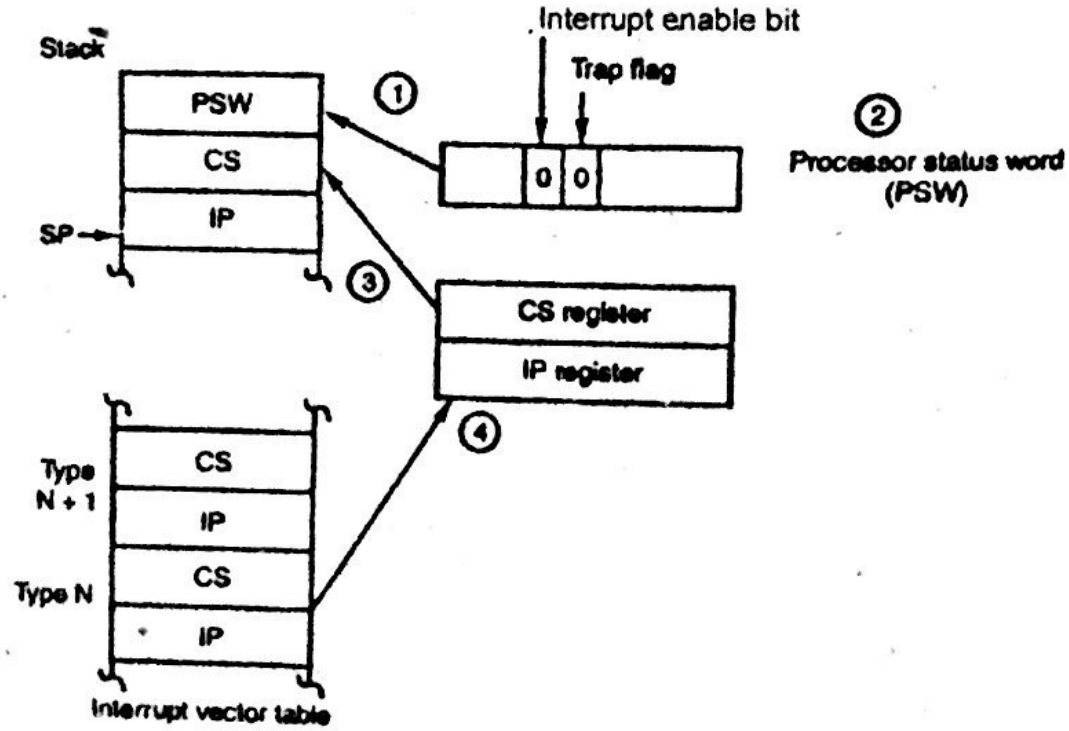
```
        MOV     SI,     OFFSET LIST
        MOV     CX,     OA_H
```

8086 goes when it receive an interrupt. (2

Ans. When an interrupt is acknowledged, the processor goes through the following sequences events before executing the interrupt service routine.

1. The flag register is pushed onto the stack.
2. The INTR inputs are disabled and TF is also cleared.
3. The CS register is pushed onto the stack.
4. The IP register is pushed onto the stack.
5. The IP register is loaded with the interrupt vector address (offset).
6. The CS register is loaded with the interrupt vector address (segment).
7. The ISR is executed from the address formed by the new CS and IP registers.

These sequences are summarized in Fig.



Q.4. (c) Interface four chips of 2K RAM and two chips of 2K ROM with microprocessor 8086. Give the complete memory map of the system. (6)

Ans. Four chips of 2K RAM, two chips of 2K ROM

Half memory capacity i.e. 2K RAM (1) + 2 RAM (2) & 2K ROM (1) + 2K ROM (2) will be connected in even memory bank and another will be in odd memory bank.

Address decoding table

Total Memory	Total address	Even address	Binary Addr $A_{19}A_{18}A_{17}A_{16}$ - - - - A_{11} -- A_3 A_2 A_1 A_0
4 KB ROM	$F0000_H$	$F0000_H$	1 1 1 1 - - - - 0 0 0 0
	$FOFFF_H$	$FOFFE_H$	1 1 1 1 - - - - 1 1 1 0
8KB RAM	F1000	F1000	1 1 1 1 - - - - 0 0 0 0
	F1FFF	F1FFE	1 1 1 1 - - - 1 1 1 0

END TERM EXAMINATION [DEC. 2016]
FIFTH SEMESTER [B.TECH]
MICROPROCESSOR AND MICRO CONTROLLER [ETEC–305]

Time : 3 hrs. M.M. :

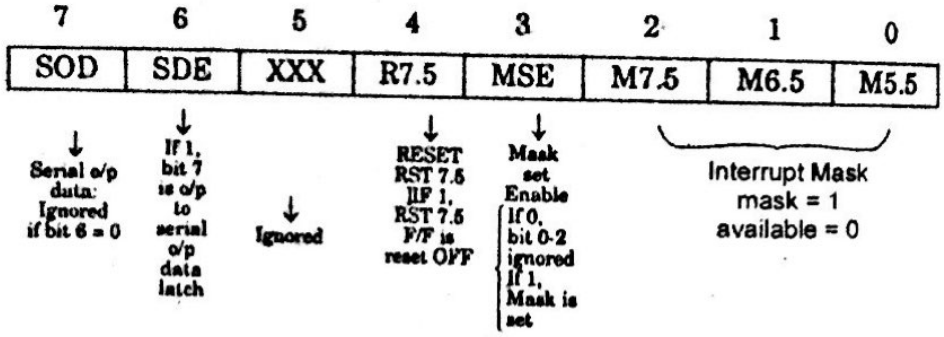
Note: *Attempt any five questions in cluding Q. No 1 which is compulsory. Attempt on question from each unit. Assume suitable missing data if any.*

Q. 1. Attempt all:

Q. 1. (a) Explain the function of SIM instruction? (2.5 x 10 = 25

Ans. SIM (Set interrupt mask)

Used to mask any one of the interrupt RST 7.5, 6.5, 5.5. and for Serial data out



Q.1. (b) Distinguish between ACALL and LCALL instructions in 8051?

Ans. ACALL and LCALL

ACALL stands for absolute call while LCALL stands for long call. These tw instructions allow the programmer to call a subroutine. There is a slight differenc between the two instructions, the same as the difference between AJMP and LJMP.

ACALL allows you to jump to a subroutine within the same 2K pag while LCALL allows you to jump to a subroutine anywhere in the 64K code space.

The advantage of ACALL over LCALL is that it is a 2-byte instructio while LCALL is a 3-byte instruction.

Q.1. (c) What do you mean by maximum and minimum mode of 8086?

Ans. Minimum mode configuration of 8086: If pin number 33 of 8086 is connecte to logic 1 then 8086 operates in the minimum mode. Pins 24 to 31 of 8086 will have th function as shown in the parenthesis, next to these pins.

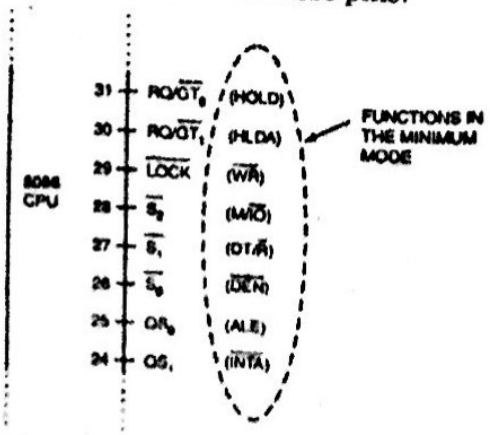


Fig.

2. $\overline{RQ}/\overline{GT1}$ and $\overline{RQ}/\overline{GT0}$: The request grant pins request ... mode operation. Both these lines are bidirectional and are needed to request and gra... a DMA operation.

3. $\overline{LOCK}$: The lock output is used to lock peripherals off the system. This pin ... activated by using the clock prefix on any instruction.

4. QS_1 and QS_0: The queue status bits show the states of the internal instructio... queue. These pins are provided to access numeric math co-processor (8087). The operation of the queue status bits is in truth-table below:

QS_1	QS_0	Function
0	0	Queue is idle (NOP)
0	1	First byte is opcodes
1	0	Queue is empty
1	1	Subsequent byte of opcode

Q.1. (d) Explain the function of the following signals.

(i) $\overline{EA}$ (ii) NMI (iii) $\overline{LOCK}$ (iv) $\overline{PSEN}$ (v) $\overline{TEST}$

Ans. (i) ($\overline{EA}$)/VPP: It is an active low I/P to 8051 microcontroller. when ($\overline{EA}$) = 0, then 8051 microcontroller access from external program memory (ROM) only. When ($\overline{EA}$) = 1 then it access internal and external program memories (ROM).

(ii) NMI: An edge triggered input, causes a type-2 interrupt. A subroutine is vectored to via the interrupt vector look up table located in system memory. NMI is not maskable internally by software. A transition from a LOW to HIGH on this pin initiates the interrupt at the end of the current instruction. This input is internally synchronized.

(iii) $\overline{LOCK}$: It indicates to another system bus master, not to gain control of the system bus while $\overline{LOCK}$ is active low. The LOCK signal is activated by the "LOCK" prefix instruction and remains active until the completion of the instruction. This signal is active low and floats to tri-state OFF during 'hold acknowledge'.

(iv) ($\overline{PSEN}$): It is active low O/P signal. It is used to enable external program memory (ROM). When ($\overline{PSEN}$) = 0, then external program memory becomes enabled and microcontroller read content of external memory location. Therefore it is connected to OE, of external ROM. It is activated twice every external ROM memory cycle.

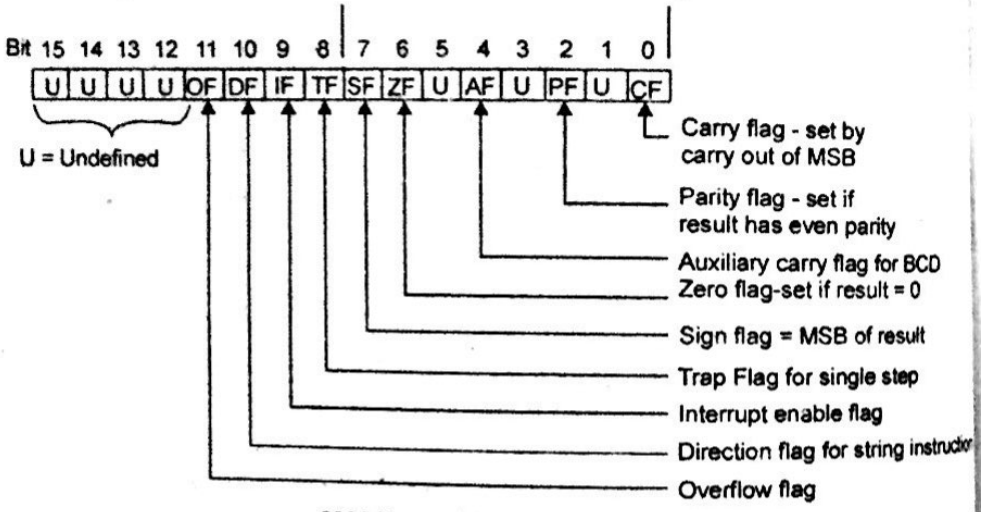
(v) $\overline{TEST}$: Pin is examined by the "WAIT" instruction. If the TEST pin is low, execution continues. Otherwise the processor waits in an "idle" state. This input is synchronized internally during each clock cycle on the leading edge of CLK.

EQU (equate): This is used to define a constant without occupying a memo location. The EQU directive does not set aside storage for a data item but associate constant value with a data label so that when the label appears in the program.

DB (define byte): The DB directive is the most widely used data directive in t assembler. It is used to define the 8-bit data. When DB is used to define data, t numbers can be in decimal, binary, hex, or ASCII formats.

Q.1. (g) Explain the flag register of 8086?

Ans. FLAGS register: The **FLAGS register** is the status **register** in Intel x microprocessors that contains the current status of the processor. This **register** is bits wide.



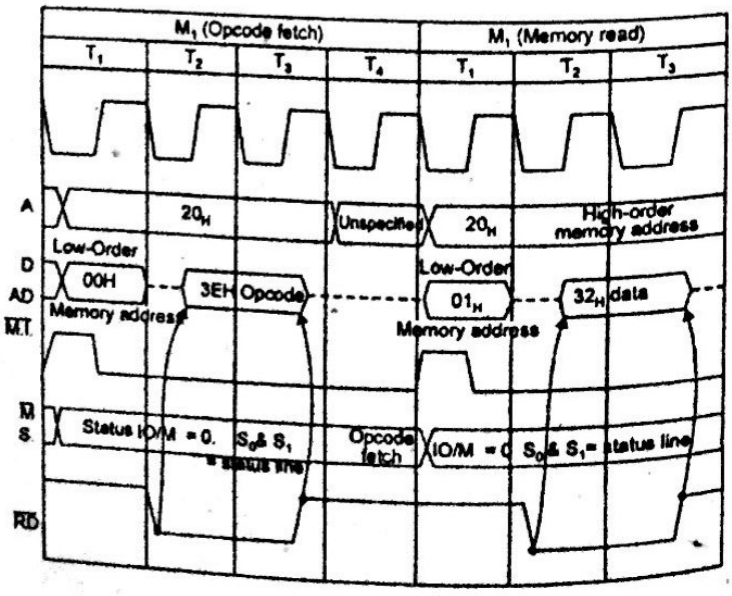
8086 flag register format

Q.1. (*h*) What is the use of BSR mode in 8255?

Ans. Refer to Q.1. (h) End Term Examination 2015.

Q.1. (i) Draw the timing diagram of instruction MVI B, 05H?

Ans. Timing diagram for MVIA, 32 H.



The vector ...

Interrupt	Vector address
RST 0	0000_H
RST 1	0008_H
RST 2	0010_H
RST 3	0018_H
RST 4	0020_H
RST 5	0028_H
RST 6	0030_H
RST 7	0038_H

Interrupt	Vector address
RST 7.5	$003C_H$
RST 6.5	0034_H
RST 5.5	$002C_H$
TRAP	0024_H

When the processor encounters the software instruction, it pushes the content ... PC (Program Counter) to stack. Then loads the Vector address in PC and starts executing the Interrupt Service Routine (ISR) stored in this vector address. At the end of ISR ... return instruction - RET will be placed. When the RET instruction is executed, th... processor POP the content of stack to PC. Hence the processor control returns to th... main program after servicing the interrupt. Execution of ISR is referred to as servicing of interrupt.

All software interrupts of 8085 are vectored interrupts. The software interrupts cannot be masked and they cannot be disabled. The software interrupts are RST 0, RST1, ...RST 7 (8 Nos).

Hardware Interrupts of 8085

An external device, initiates the hardware interrupts of 8O85 by placing an appropriate signal at the interrupt pin of the processor. The processor keeps on checking the interrupt pins at the second T -state of last machine cycle of every instruction. If the processor finds a valid interrupt signal and if the interrupt is unmasked and enabled then the processor accepts the interrupt. The acceptance of the interrupt is acknowledged by sending an $\overline{INTA}$ signal to the interrupted device.

The processor saves the content of PC (program Counter) in stack and then loads the vector address of the interrupt in PC. (If the interrupt is non-vectored, then the interrupting device has to supply the address of ISR when it receives INTA signal). It starts executing ISR in this address. At the end of ISR, a return instruction, RET will be placed When the processor executes the RET instruction, it POP the content of top of stack to PC. Thus the processor control returns to main program after servicing interrupt. The hardware interrupts of 8085 are TRAP, RST 7.5, RST 6.5, RST 5.5 and INTR.

Q.2. (b) Write an assembly language program to generate Fibonacci series along with the flow chart? (6.5)

Ans. To write an assembly language program to generate the Fibonacci series with given first two terms.

C100 LXI H C300 21 ; Load the HL register pair immediately
C103 MOV B M 46 ; Move the memory content in to B register

Q.3. (a) Explain the function of following instructions with suitable examples: (6)

(i) DAD (ii) DAA (iii) CMC (iv) JPO, JPE (v) EI, DI (vi) RET

Ans. *(i)* **DAD:** The 16-bit contents of the specified register pair are added to the contents of the HL register and the sum is stored in the HL register. The contents of the source register pair are not altered. If the result is larger than 16 bits, the CY flag is set. No other flags are affected.

Example: DAD H

(ii) **DAA:** The contents of the accumulator are changed from a binary value to two 4-bit binary coded decimal (BCD) digits. This is the only instruction that uses the auxiliary flag to perform the binary to BCD conversion, and the conversion procedure is described below. S, Z, AC, P, CY flags are altered to reflect the results of the operation.

If the value of the low-order 4-bits in the accumulator is greater than 9 or if AC flag is set, the instruction adds 6 to the low-order four bits.

If the value of the high-order 4-bits in the accumulator is greater than 9 or if the Carry flag is set, the instruction adds 6 to the high-order four bits.

Example: DAA

(iii) CMC: The Carry flag is complemented. No other flags are affected.

Example: CMC

(iv) JPO, JPE: The program sequence is transferred to the memory location specified by the 16-bit address given in the operand based on the specified flag of the PSW

JPO; Jump if Odd parity

JPE; Jump if even parity

(v) EI, DI: DI; The interrupt enable flip-flop is reset and all the interrupts except the TRAP are disabled. No flags are affected.

Example: DI

EI; The interrupt enable flip-flop is set and all interrupts are enabled. No flags are affected. After a system reset or the acknowledgement of an interrupt, the interrupt enable flipflop is reset, thus disabling the interrupts. This instruction is necessary to reenable the interrupts (except TRAP).

Example: EI

(vi) **RET:** The program sequence is transferred from the subroutine to the calling program. The two bytes from the top of the stack are copied into the program counter, and program execution begins at the new address.

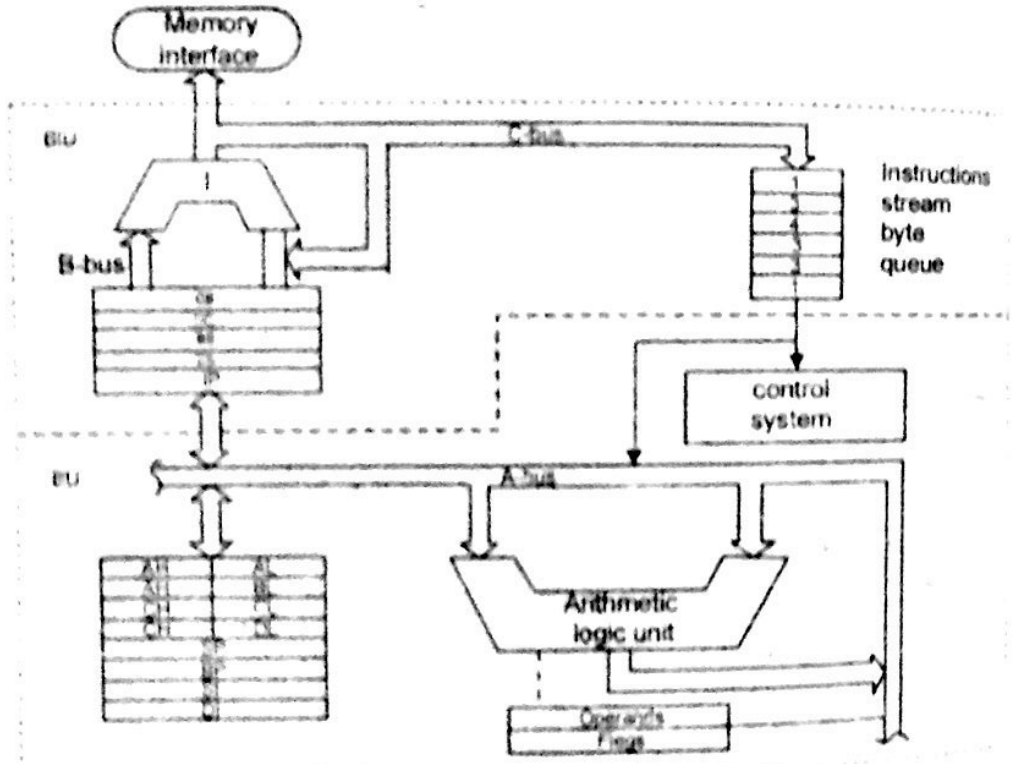
Example: RET

Q.3. (b) Explain the difference between partial and full address decoding scheme? Connect 16k bytes with the systems lines of microprocessor 8085 performing partial address decoding i.e. use A_{15} line for generating chip select logic. The memory IC available is EPROM with starting memory address is 9000H? (6.5)

Ans. In practice, all memory locations are not implemented, Hence all the address lines of the microprocessor are not used by the memory module to select particular memory locations. Unused address lines are decoded to generate chip select signals. There are mainly two types of address decoding.

The Bus Interface Unit: The BIU handles all data and addresses on the b
the execution unit such as it sends out addresses, fetches instructions from me
reads data from ports and memory as well as writes data to ports and memory In
there are so many functional groups or parts these are as follows

Instruction Queue: To increase the execution speed, BIU fetches as many as
instruction bytes ahead to time from memory The pre fetched instruction bytes
held for the EU in a first in first out group of registers called a instruction queue Wh
the EU is ready for its next instruction, it simply reads the instruction from t
instruction queue This is much faster than sending out an address to the syst
memory and to send back the next instruction byte Fetching the next instruction wh
the current instruction executes is called pipelining



8086 Internal Block Digram

Segment Registers: The BIU contains four 16-bit segment registers. They are
the extra segment (ES) register, the code segment (CS) registers, the data segment (DS
registers, and the stack segment (SS) registers These segment registers are used t
hold the upper 16 bits of the starting address for each of the segments The part of
segment starting address stored in a segment register is often called the segment base

1. Code Segment (CS): The CS register is used for addressing a memory location
in the Code Segment of the memory, where the executable program is stored

2. Data Segment (DS): The DS contains most data used by program Data are
accessed in the

Data Segment by an offset address or the content of other register that holds the
offset address

3. Stack Segment (SS): SS defined a section of memory to store addresses and
data while a subprogram executes

• **Auxiliary Flag (AF):** If an operation performed in ALU generates a carry/barrow from lower nibble (i.e. $D_0 - D_3$) to upper nibble (i.e. $D_4 - D_7$), the AF flag is set i.e. carry given by D_3 bit to D_4 is AF flag. This is not a general-purpose flag; it is used internally by the processor to perform Binary to BCD conversion.

• **Parity Flag (PF):** This flag is used to indicate the parity of result. If lower order 8-bits of the result contains even number of 1's, the Parity Flag is set to one and for odd number of 1's, the Parity Flag is reset i.e. zero.

• **Zero Flag (ZF):** It is set to one; if the result of arithmetic or logical operation is zero else it is reset.

• **Sign Flag (SF):** In sign magnitude format the sign of number is indicated by MSB bit. If the result of operation is negative, sign flag is set to one.

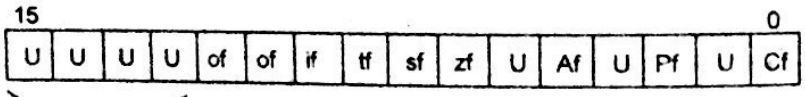
• **Overflow Flag (OF):** It occurs when signed numbers are added or subtracted. An OF indicates that the result has exceeded the capacity of machine.

2. Control Flags: Control flags are intentionally set or reset to control certain operations of the processor with specific instructions put in the program from the user. Control flags are as follows:

(a) Trap Flag (TP): It is used for single step control. It allows user to execute one instruction of a program at a time for debugging. When trap flag is set, program can be run in single step mode.

(b) Interrupt Flag (IF): It is an interrupt enable/disable flag, i.e. used to allow/prohibit the interruption of a program. If it is set, the maskable interrupt is enabled and if it is reset, the interrupt is disabled.

(c) Direction Flag (DF): It is used in string operation. If it is set, string bytes are accessed from higher memory address to lower memory address. When it is reset, the string bytes are accessed from lower memory address to higher memory address.



CF: Contains carry and of MSH of result
FF: Indicates result has even parity
AF: Certains carry cull of his 3 in AL
ZF: Indicates if result equals area
SP: Indicates if result is negative
OF: Indicates that in on covered in result
IF: Enebled Disables inegrimity
DF: Ceitains pointer updates string occuration
TF: Provides single-step crgebility for detugging

8086 Flag Register Format

General Purpose Registers: The EU has eight general purpose registers labeled AH, AL, BH, BL, CH, CL, DH, and DL. These registers can be used individually for temporary storage of 8-bit data. The AL register is also called the accumulator. Certain pairs of these general purpose registers can be used together to store 16-bit data. The valid register pairs are AH and AL, BH and BL, CH and CL and DH and DL. These register pairs is referred to the AX, BX, CX, and DX resp.

1. AX Register: For 16-bit operations, AX is called the accumulator register that stores operands for arithmetic operations.

Q.5. (a) Explain the functions of instructions with suitable examples: (6)

(i) LEA (ii) AAA (iii) CWD (iv) MOVSB (v) REP (vi) LOOP

Ans. (i) LEA – Used to load the address of operand into the provided register.

(ii)AAA– Used to adjust ASCII after addition.

(iii)CWD " Used to fill the upper word of the double word with the sign bit of the lower word.

(iv) MOVS/MOVSB/MOVSW– Used to move the byte/word from one string to another.

(v) REP– Used to repeat the given instruction till CX ≠ 0.

(vi) LOOP– Used to loop a group of instructions until the condition satisfies, i.e., CX = 0

Q.5. (b) What do you mean by segment over ride prefix? Write the instruction format for MOV DS: 2345 [BP], DX (6.5)

Ans. Segment override prefix

Consider an example

MOV[BX],DL

In this instruction, offset present in BX is added to the data segment base in DS to produce the physical address

But if we want that offset should be added to other segment base not to DS, then we should use segment override prefix:

MOVCS:[BX],DL

Here the offset is added to CS. The code byte for segment override prefix has the format 001XX110, where X's represent the 2-bit code corresponding to the required segment.

X	X	Segment base
0	0	*ES*
0	1	*CS*
1	0	*SS*
1	1	*DS*

MOV DS: 2345 [BP], DX

Here we have to specify DX using REG field. The D bit must be 0, indicating that DX is the source register. The REG field must be 010 to indicate DX register. The w bit must be 1 to indicate it is a word operation. 2345 [BP] is specified with MOD = 10 and R/M = 110 and displacement = 2345H.

Whenever BP is used to generate the effective address (EA), the default segment would be SS. In this example we want the segment register to be DS, so we have to provide the segment override prefix byte (3EH byte) to start with. The SOP byte is 001[illegible] where XX value is provided as per table shown below.

Table 1: Operation between a CPU and 8251

$\overline{CS}$	C/$\overline{D}$	$\overline{RD}$	$\overline{WR}$	
1	×	×	×	Data Bus 3-State
0	×	1	1	Data Bus 3-State
0	1	0	1	Status → CPU
0	1	1	0	Control Word← CPU
0	0	0	1	Data→CPU
0	0	1	0	Data ← CPU

Control Words

There are two types of control word.

1. Mode instruction (setting of function)
2. Command (setting of operation)

1. Mode Instruction: Mode instruction is used for setting the function of the 8251 Mode instruction will be in "wait for write" at either internal reset or external reset. That is, the writing of a control word after resetting will be recognized as a "mode instruction."

Items set by mode instruction are as follows:

- Synchronous/asynchronous mode
- Stop bit length (asynchronous mode)
- Character length
- Parity bit

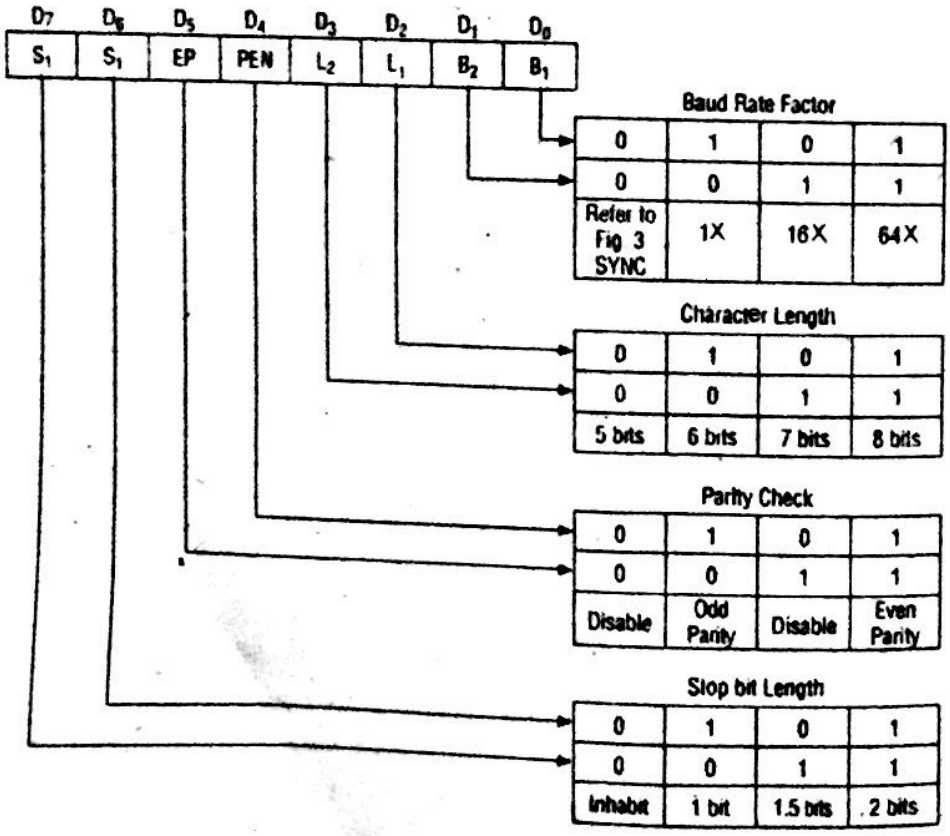


Fig. Bit configuration of mode Instruction (Asynchronous)

Mode 0 (000): Interrupt on Terminal Count: Mode 0 is used for the generation of accurate time delay under software control. In this mode, the counter will start counting from the initial COUNT value loaded into it, down to 0. Counting rate is equal to the input clock frequency.

The OUT pin is set low after the Control Word is written, and counting starts one clock cycle after the COUNT programmed. OUT remains low until the counter reaches 0, at which point OUT will be set high until the counter is reloaded or the Control Word is written. The Gate signal should remain active high for normal counting. If Gate goes low counting gets terminated and current count is latched till Gate pulse goes high again, the first byte of the new count when loaded in the count register stop the previous count.

Mode 1 (001): Programmable One Shot

In this mode 8253 can be used as Monostable Multivibrator. GATE input is used as trigger input.

OUT will be initially high. OUT will go low on the Clock pulse following a trigger to begin the one-shot pulse, and will remain low until the Counter reaches zero. OUT will then go high and remain high until the CLK pulse after the next trigger.

After writing the Control Word and initial count, the Counter is armed. A trigger results in loading the Counter and setting OUT low on the next CLK pulse, thus starting the one-shot pulse. An initial count of N will result in a one-shot pulse N CLK cycles in duration.

The one-shot is retriggerable, hence OUT will remain low for N CLK pulses after any trigger. The one-shot pulse can be repeated without rewriting the same count into the counter. GATE has no effect on OUT. If a new count is written to the Counter during a oneshot pulse, the current one-shot is not affected unless the counter is retriggered. In that case, the Counter is loaded with the new count and the oneshot pulse continues until the new count expires.

Mode 2 (X10): Rate Generator

In this mode, the device acts as a divide-by-n counter, which is commonly used to generate a real-time clock interrupt.

Like other modes, counting process will start the next clock cycle after COUNT is sent. OUT will then remain high until the counter reaches 1, and will go low for one clock pulse. OUT will then go high again, and the whole process repeats itself.

The time between the high pulses depends on the preset count in the counter's register, and is calculated using the following formula:

$$\text{Value to be loaded into counter} = \frac{f_{input}}{f_{output}}$$

Note that the values in the COUNT register range fromfj, to 1; the register never reaches zero.

Mode 3 (X11): Square Wave Generator

This mode is similar to mode 2. However, the duration of the high and low clock pulses of the output will be different from mode 2.

Suppose is the number loaded into the counter (the COUNT message), the output will be

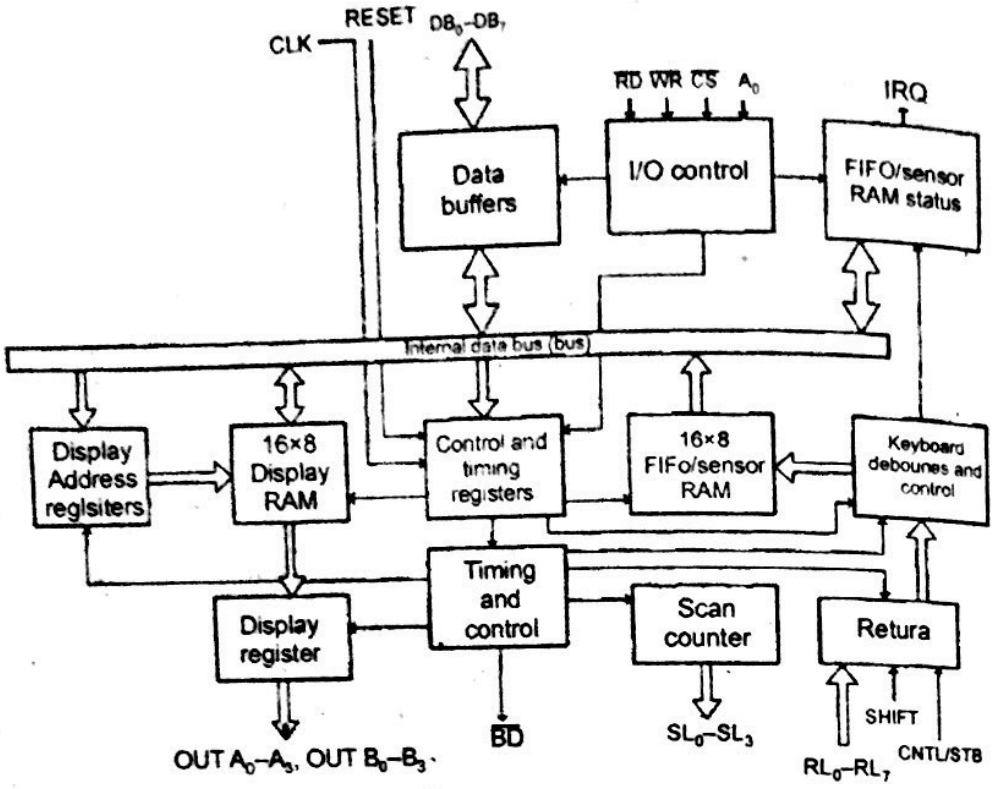


Fig. 8279 Block Diagram

Keyboard Section: This sectio has 8 lines. RL0 - RL7. Plus 2 additional lines. Shift and CNTL/STB. The keys are automatically debounced and keyboard can operate in two modes:

>>two key lockout mode or

>>N-key rollover.

In two key lockout mode if 2 keys are pressed simultaneously only firstkey is recongnized.

In N key rollover mode, simultaneous keys are recongnized and stored in internal buffer: it can also be set up so that no key is recongnized until only one key is remained pressed.

This has a FIFO RAM.

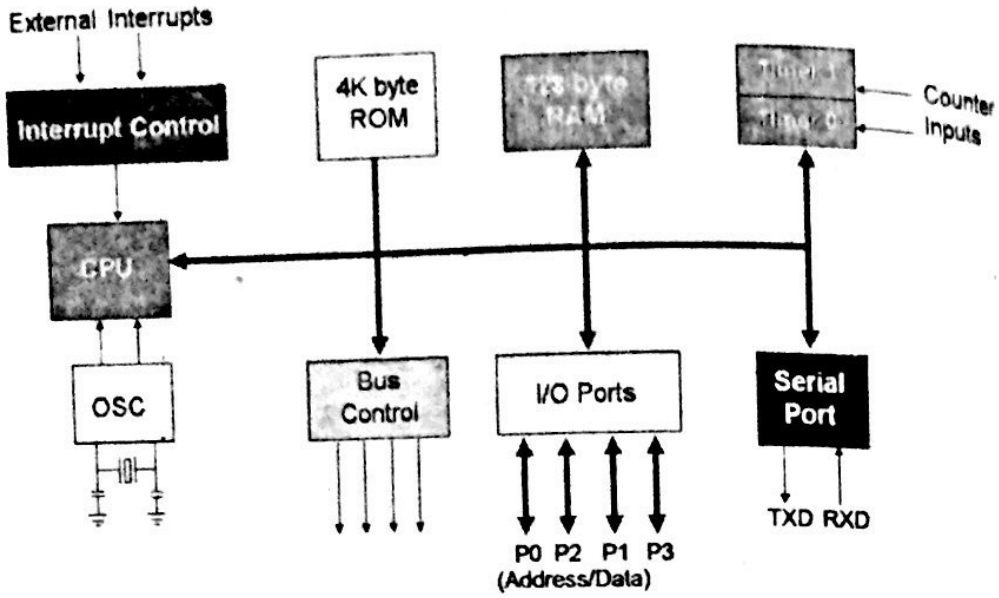
The status logic keeps track of number of entries and provides IRQ(interrupt request) signal when

FIFO is empty.

DISPLAY SECTION: This section has 8 output lines divided into 2 groups of 4. A0 - A3 and B0 - B3. These lines can be used in both ways 8 lines or 2 sets of 4 lines. The display can be blanked using BD line. The section has 16x8 display RAM.

SCAN SECTION: This section has scan counter and 4 scan lines. SL0 - SL3. These 4 scan lines can be decoded using a 4 - 16 decoder to generate 16 lines for scanning. These 16 lines can be connected to rows of a matrix keyboard and digit drivers of multiplexed display.

The 8051 Block Diagram



Now let us see the functions of each of these components.

1. ALU: All arithmetic and logical functions are carried out by the ALU.

Addition, subtraction'with carry, and multiplication come under arithmetic operations.

Logical AND, OR and exclusive OR (XOR) come under logical operations.

2. Program counter (PC): A program counter is a 16-bit register and it has no internal address. The basic function of program counter is to fetch from memory the adress of the next instruction to be executed. The PC holds the address of the next instruction residing in memory and when a command is encountered, it produces that instruction. This way the PC increments automatically, holding the address of the next instruction.

3. Registers: Registers are usually known as data storage devices. 8051 microcontroller has 2 registers, namely Register A and Register B. Register A serves as an accumulator while Register B functions as a general purpose register. These registers are used to store the output of mathematical and logical instructions.

The operations of addition, subtraction, multiplication, and division are carried out by Register A. Register B is usually unused and comes into picture only when multiplication and division functions are carried out by Register A. Register A also involved in data transfers between the microcontroller and external memory.

8051 microcontroller also has 7 Special Function Registers (SFRs). They are:

1. Serial Port Data Buffer (SBUF)
2. Timer/Counter Control (TCON)
3. Timer/Counter Mode Control (TMOD)
4. Serial Port Control (SCON)
5. Power Control (PCON)
6. Interrupt Priority (IP)

6. Four General Purpose Parallel Input/Output Ports: The 8051 microcontroller has four 8-bit input/output ports. These are: PORT PO: When there is no external memory present, this port acts as a general purpose input/output port. In the presence of external memory, it functions as a multiplexed address and data bus. It performs a dual role.

PORT P1: This port is used for various interfacing activities. This 8-bit port is a normal I/O port i.e. it does not perform dual functions.

PORT P2: Similar to PORT PO, this port can be used as a general purpose port when there is no external memory but when external memory is present it works in conjunction with PORT PO as an address bus. This is an 8-bit port and performs dual functions.

PORT P3: PORT P3 behaves as a dedicated I/O port

PSW (Program status word) is an 8-bit register. It is used to indicate some conditions that result after an instruction is executed.

CY	AC	F0	RSI	RSO	OV	-	P

Cy →Carry Flag → shows carry or borrow after addition or subtraction.

AC →Auxilliary carry flag→ it is set if there is a carry from D_3 to D_4 during an ADD or SUB operation.

P → Parity Flag → it reflects the number if 1's in Accumulator. P = 1, for odd no. of 1's & P = 0 for even no. of 1's

OV → overflow flag → this flag is set whenever the result of a signed no. operation is too large, causing the high-order bit to overflow into the sign bit

RS1, RS0 → Register bank select bits

RS1	RS0	Reg. bank
0	0	0
0	1	1
1	0	2
1	1	3

F0 → Available to the user for general purpose.

Q.8. (c) Explain the TMOD register? (3.5)

Ans.

TMOD: Timer/Counter Mode Control Register (Not Bit Addressable)

GATE	c/T	MI	M0	Gate	c/T	MI	M0
Timer 1				Timer 0			

Gate When TRx (in TCON) is set and Gate = 1, Timer Counters will run only while INTx pin is high (hardware control) When Gate = 0 Timer/Counterx will run only while TRx = 1 (software) control)

C/T Timer or Counter selector Cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin).

MI Mode selector bit (NOTE I)

M0 Mode selector bit (NOTE 1)

Note 1

MI	M0	Operating Mode
0	0	0 13-bit Timer
0	1	1 16- bit Timer/Counter
1	0	2 8-bit Auto Released Timer/Counter
1	0	3 (Timer 0) TL 0 in an 8 bit Timer Counter controlled by the standard Timer 0 control bits. THO is an 8-bit Timer and controlled by Timer 1 control bits
1	1	3 (Timer 1) Timer Counter 1 stopped

```
SJMP Again ; reload timer since
           mode 1 is not all to reload
```

Square wave = 25 KHz

$T = 1/f = 1/25 \times 10^3 = 40\ \mu s$

Half of it for the high and low portion of the pulse is 20 μs.

40 μ/1.085 ms = 36 and 65536–36 = 65500

$= (FFDC)_H$

TL1 = DC & TH1 = FF

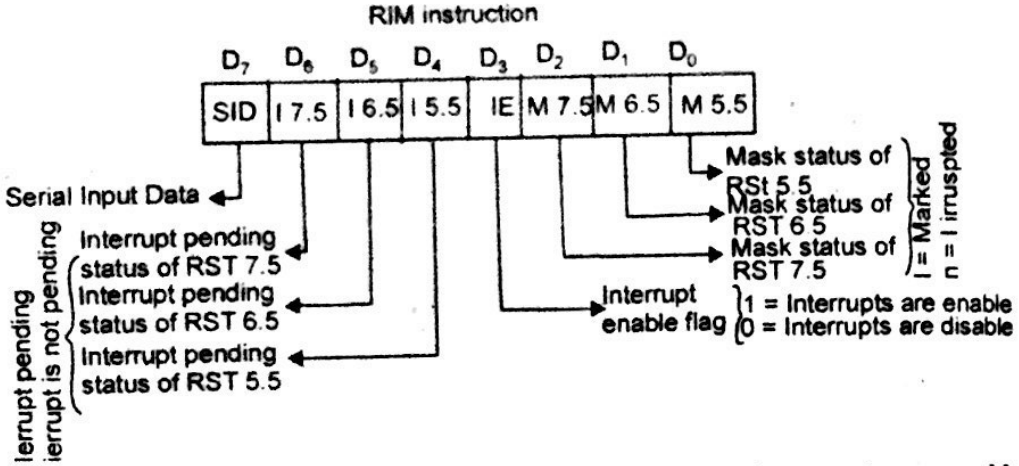
```
MOV TMOD, # 10H
Again  MOV TL1, # DCH
MOV TH1, # OFFH
SETB TR1
BACk  JNB TF1 Back
CLR TR1
CLR P2.3
CLR TF1
SJMP Again
HLT
```

Q.2.(a) What is the use of RIM instruction? State the bit-wise significance of accumulator data following the RIM instruction. (5)

Ans. The RIM instruction reads the following bits into the accumulator:

Bit D7 (SID-Serial Input Data) This is the input pin of the serial data interface which is connected to pin 5 of the 8085, and indicates the high/low status of that pin.

Bits D6-D4 (I 7.5, I 6.5, I 5.5) These bits indicates that an interrupt is pending for these three 8085 interrupts 7.5, 6.5, and 5.5. If interrupts 5.5 or 6.5 have been masked off by bits D0 or D1, bits D4 and D5 will not be set. Bit D6, which corresponds to the 7.5 interrupt, will be set on to indicate that an interrupt 7.5 was requested, even if it was masked off.



Bit D3 (IE-Interrupt Enable) This bit indicates whether interrupts are enabled (1) using the EI (Enable Interrupts) instruction, or disabled (0) using the DI (Disable Interrupts) instruction.

Bits 2-D0 (M 7.5, M6.5, M5.5) Mask status of interrupts 7.5, 6.5, and 5.5. Corresponds to bits D2-D0 of the SIM instruction. 1 if masked, 0 if enabled.

So the SIM and RIM instructions are typically used to either output to or input from 8085 serial interface, or enable/disable/read the interrupt masks for interrupts 7.5, 6.5, 5.5, *but usually not at the same time.*

Q.2.(b) How the memory of 8086 is divided into even bank and odd memory banks? How these memory banks are selected for byte and word access? (5)

Ans. In 8086 system, the 1M bytes memory is physically organised as an odd bank and an even bank, each of 512 Kbytes, addressed in parallel by the processor. Byte data with an even address is transferred on D_7–D_0, while the byte data with an odd address is transferred on D_{15}–D_8 bus lines. The processor provides two enable signals, $\overline{BHE}$ and A_0 for selection of either even or odd or both the banks. To read or write a complete word from/to memory, if it is located at an even address, only our read or write cycle is required It the word is located at an odd address, the first read or write cycle is required for accessing the lower byte while the second our is required for accessing the upper byte.

BHE	A_0	Indication
0	0	Whole word
0	1	Upper byte from to odd address
1	0	Lower byte from or to even address
1	1	None

Dedicated Interrupts Supported by 8086: There are five dedicated interrupts type 0 to type 4.

→ Type 0 to type 4.

→ **Type 0 or divide by zero interrupt:** Whenever the operation of DIV operation or IDIV operation is too large to fit in the destination register, the 8086 will automatically do a type 0 interrupt.

→ **Type 1 or single-step interrupt:** If the trap flag is set then type 1 interrupt will occur. In it, the processor will execute one instruction and stop. We can then examine the contents of registers and memory locations.

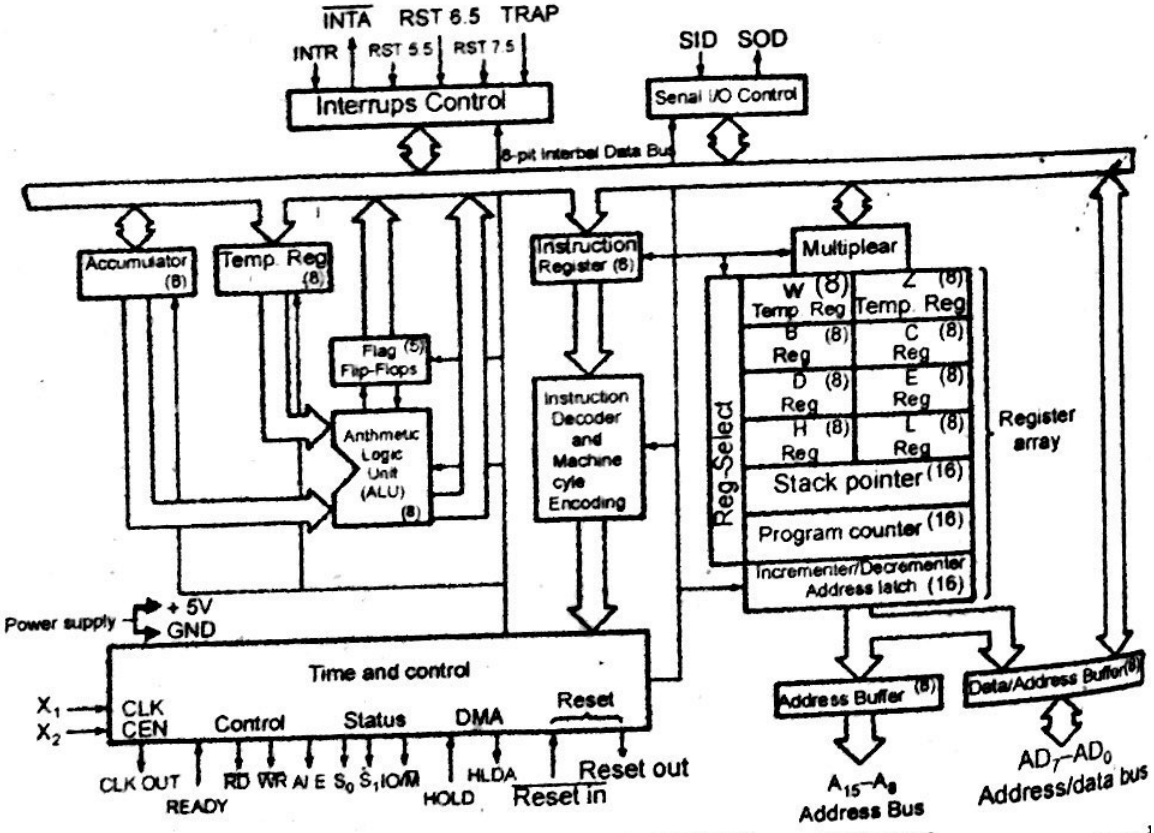
→ **Type 2 or Nonmaskable Interrupt:** 8086 will do a type 2 interrupt when it receives a low-to-high transition on its NMI input Pin.

→ **Type 3 Breakpoint Interrupt:** It is produced by execution of the INT 3 instruction. The main use of type 3 interrupt is to implement a breakpoint function in a system.

→ **Type 4 or Overflow interrupt:** The 8086 overflow flag (OF) will be set if the signed result of an arithmatic operation on two signed no. is too large to be represented in the destination register or memory location.

Q.4. (a) Write an assembly language program to arrange the data in ascending order in 8085. (5)

Ans. Architechture of 8085



This is the functional Block Diagram of 8085 Microprocessor.

Acumulator: It is a 8-bit register which is used to perform airthmetical and logical operation. It stores the output of any operation. It also works as registers for I/O accesses.

Instruction Decoder: Instruction decoder identifies the instructions. It takes the informations from instruction register and decodes the instruction to be performed.

Program Counter: It is a 16 bit register used as memory pointer. It stores the memory address of the next instruction to be executed. So we can say that this register is used to sequencing the program. Generally the memory have 16 bit addresses so that it has 16 bit memory.

The program counter is set to 0000H.

Stack Pointer: It is also a 16 bit register used as memory pointer. It points to the memory location called stack. Generally stack is a reserved portion of memory where information can be stores or taken back together.

Timing and Control Unit: It provides timing and control signal to the microprocessor to perform the various operation.It has three control signal. It controls all external and internal circuits. It operates with reference to clock signal.It synchronizes all the data transfers.

There are three control signal:

1. ALE: Airthmetic Latch Enable, It provides control signal to synchronize the components of microprocessor.

2. RD: This is active low used for reading operation.

3.WR-This is active low used for writing operation.

There are three status signal used in microprocessor S0, S1 and IO/M. It changes its status according the provided input to these pins.

IO/M (Active Low)	S1	S2	Data Bus Status (output)
0	0	0	Halt
0	0	1	Memory Write
0	1	0	Memory Read
1	0	1	IO Write
1	1	0	IO Write
0	1	1	Opcode fetch
1	1	1	Interrupt acknowledge

Serial Input Output Control-There are two pins in this unit. This unit is used for serial data communication.

Interrupt Unit-There are 6 interrupt pins in this unit. Generally an external hardware is connected to these pins. These pins provide interrupt signal sent by external hardware to microprocessor and microprocessor sends acknowledgement for receiving the interrupt signal. Generally INTA is used for acknowledgement.

Q.4.(b) Explain the addressing modes of 8086 with suitable examples. (5)

***Ans.* Addressing modes:** The method by which address of source of data is given alongwith instruction is called as addressing mode of source.

1. Immediate addressing mode (IAM): If 8/16 bit data required for executing the instruction is given alongwith the instruction, then it is called immediate addressing mode.

Example:

1. *MOV AL, 75H ; 78H →* [75] *AL*

END TERM EXAMINATION [DEC-2017]
FIFTH SEMESTER [B.TECH.]
MICROPROCESSOR AND MICROCONTROLLER [ETEC–305]

Time : 3 hrs. **M.M. : 75**

Note: *Attempt any five questions including Q. no. 1 which is compulsory. Select one question from each unit. Assume suitable missing data if any.*

Q. 1. Attempt all:

Q.1. (a) If the 8085 adds 87H and 79H specify the contents of the accumulator and the status of the S, Z and CY flags. **(2)**

Ans. 87H+ 79H= 100000000

S = 0, Z = 1, CY = 1

Q.1.(b) How does the microprocessor differentiate among a positive number, a negative number and a bit pattern. **(2)**

Ans. A *digital comparator* or magnitude comparator is a hardware electronic device that takes two numbers as input in binary form and determines whether one number is greater than or less than or equal to the other number.

Comparators are used in central processing units (CPU) and microcontrollers.

In comparator hardware some gates are used (AND, OR, NAND, NOR, XOR, etc). These gates take binary inputs and give result in binary. The output can be seen from a truth table.

Inputs		Outputs		
A	**B**	**A>B**	**A=B**	**A<B**
0	0	0	1	0
0	1	0	0	1
1	0	1	0	0
1	1	0	1	0

Here 0 and 1 are electronic voltages for the gate.

1 - Represents some threshold voltage which indicates some positive voltage.

0 - Represents the voltage below than the threshold.

E.g. Suppose a comparator works on 5 volt (it is consideration for explanation) then:

Voltage more than 3 volt can be considered as binary-1.

Voltage below than 3 volt be considered as binary-0

Q.1.(c) What is the function of A_1 and A_0 lines in 8255? **(2)**

Ans. The 8255 gives a CPU or digital system access to programmable parallel I/O. The 8255 has 24 input/output pins. These are divided into three 8-bit ports (A, B, C). Port A and port B can be used as 8-bit input/output ports. Port C can be used as an 8-bit input/output port or as two 4-bit input/output ports or to produce handshake signals for ports A and B.

The three ports are further grouped as follows:

1. Group A consisting of port A and upper part of port C.
2. Group B consisting of port B and lower part of port C.

Eight data lines (D_0–D_7) are available (with an 8-bit data buffer) to read/write data into the ports or control register under the status of the

RD (pin 5) and WR (pin 36), which are active low signals for read and write operations respectively. Address lines A_1 and A_0 allow to access a data register for each port or a control register, as listed below:

A_1	A_0	Port selected
0	0	port A
0	1	port B
1	0	port C
1	1	control register

Q.1.(d) Explain the display RAM section of 8279? **(2)**

Ans. DISPLAY SECTION: This section has 8 output lines divided into 2 groups of 4. $A_0 - A_3$ and $B_0 - B_3$. These lines can be used in both ways 8 lines or 2 sets of 4 lines. The display can be blanked using BD line. The section has 16×8 display RAM.

SCAN SECTION: This section has scan counter and 4 scan lines. $SL_0 - SL_3$. These 4 scan lines can be decoded using a 4 – 16 decoder to generate 16 lines for scanning. These 16 lines can be connected to rows of a matrix keyboard and digit driveers of multiplexed display.

MPU INBTERFACE SECTION: This section has 8 bi directional lines. $DB_0 - DB_7$. 1 interrupt request line(IRQ). 6 lines for interfacing including buffer address lines A_0.

When A_0 is high – signals are interpreted as control word or status.

When A_0 is low – signal is interpreted as data.

IRQ goes high whenever data is ready to be loaded into MPU.

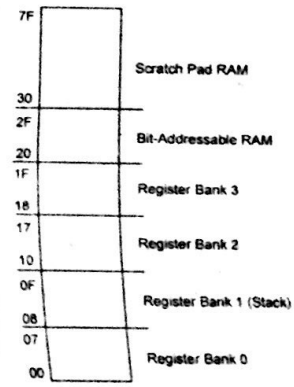
Q.1.(e) Explain the register banks of 8051? **(2)**

Ans. 8051 REGISTER BANKS AND STACK

The 80 51 microcontroller has a total of 128 bytes of RAM. The allocation of these 128 bytes of RAM and examine their usage as registers and stack.

RAM memory space allocation in the 8051

There are 128 bytes of RAM in the 8051 (some members, notably the 8052, have 256 bytes of RAM). The 128 bytes of RAM inside the 8051 are assigned addresses 00 to 7FH. As we will see in Chapter 5, they can be accessed directly as memory locations. These 128 bytes are divided into three different groups as follows.



RAM Allocation in the 8051:

1. A total of 32 bytes from locations 00 to IF hex are set aside for register banks and the stack.

2. A total of 16 bytes from locations 20H to 2FH are set aside for bit-addressable read/write memory.

3. A total of 80 bytes from locations 30H to 7FH are used for read and write storage, or what is normally called a *scratch pad*. These 80 locations of RAM are widely used for the purpose of storing data and parameters by 8051 programmers.

Q.1.(f) How Type 1 dedicated interrupt is generated. What is the vector location? (2)

Ans. Type 1: Single Step Interrupt (INT1)

- The microprocessor executes this interrupt after every instruction if the TF is set.
- It puts microprocessor in single stepping mode i.e. the microprocessor pauses after executing every instruction. This is very useful during debugging.
- Its ISR generally displays contents of all registers. Its ISR address is stored at location 1 x 4 = 00004H in the IVT.

Q.1.(g) How much time is taken by the 8085 microprocessor to execute CALL instruction if it is running at 2MHz? (2)

Ans. T states required by CALL instruction = 18 T states

F = 2MHz

T = 1/F= 0.5μs

Total time = 18 × 0.5μs= 9μs

Q.1. (h) Explain the flag register of 8086 microprocessor. (4)

Ans. Refer to Q. 1. (g) of End Term Examination 2016.

Q.1.(i) Explain the difference between MOV, MOVC and MOVX instruction in 8051. (4)

Ans. The **MOVX** instruction transfers data between the accumulator and external data memory. External memory may be addressed via 16-bits in the DPTR register or via 8-bits in the R0 or R1 registers. When using 8-bit addressing, Port 2 must contain the high-order byte of the address.

MOVX @Ri, A

MOVC: The **MOVC** instruction moves a byte from the code or program memory to the accumulator

MOVC A, @A+DPTR MOV The **MOV** instruction moves data bytes between the two specified operands. The byte specified by the second operand is copied to the location specified by the first operand. The source data byte is not affected.

MOV @Rn, #immediate

Q.1.(j) Check whether the following statements are true and false: (3)

(i) Instruction EI is necessary to implement the TRAP interrupt, but external hardware and the SIM instruction are unnecesary.

(ii) The execution of instruction MVI A, 10H and SIM will enable all three interrupts (RST 7.5, RST 6.5, and RST 5.5).

(iii) Instruction RIM is used to disable the interrupts RST 7.5, RST 6.5 and RST 5.5.

Ans. (i) False (ii) False (iii) True

UNIT-I

Q.2.(a) Explain the function of following instructions with suitable examples: (6)

(i) DCX (ii) CPI (iii) XRA (iv) LDAX (v) EI (vi) CMC

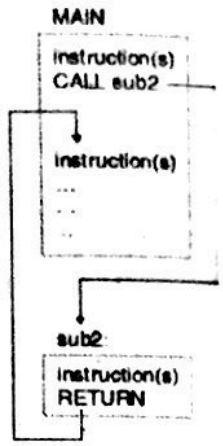
Ans. (i) **DCX:** Decrement register pair by 1

The contents of the designated register pair are decremented by 1 and the result is stored in the same place.

Example: DCX H

(ii) CPI: Compare immediate with accumulator

to a label specified after the CALL keyword. When the subroutine ends with the RETURN instruction, the instructions following CALL are processed.



When calling an external subroutine, CALL passes control to the program name that is specified after the CALL keyword. When the external subroutine completes, you can use the RETURN instruction to return to where you left off in the calling program.

Q.3. (ii) PUSH vs POP

Ans. A stack is a data structure that is used in programming. There are two basic operations that can be performed on a stack to modify its contents, which are called PUSH and POP. The main difference between PUSH and POP is what they do with the stack. PUSH is used when you want to add more entries to a stack while POP is used to remove entries from it.

A stack is so named because it places the individual data entries just like a stack of books. The first one goes to the bottom and you can only add or remove items at the top of the stack. If you want something from the middle or bottom of the stack, you need to first remove everything on top of it in order to get the item you want. This is often referred to as a Last In, First Out structure or LIFO.

Aside from how they modify the stack, there are also differences on the commands or the arguments they take to be specific. PUSH takes two arguments, the name of the stack to add the data to and the value of the entry to be added. In comparison, POP only needs the name of the stack and the value is no longer relevant. POP automatically removes the entry at the stop of the stack or the one that was last added to it.

When adding, there is always a point where you can't add anymore. When the stack is filled and another PUSH command is issued, you get a stack overflow error. It basically tells you that the stack can no longer accommodate the last PUSH. And with POP, a stack underflow error occurs when you try to POP an already empty stack. These errors basically tell you the limits of your stack and can be captured to provide an alternative or to provide a cleaner and more informative error to the user or programmer.

Stacks are quite important tools, despite being quite simple, in programming. Programs that utilize stacks intensively have other operations built on top of PUSH and POP that either provides better functionality or simplifies commonly done tasks.

Q.3.(b) Write an assembly language program to checks a set of six signed numbers and adds the positive number. The number are stored in memory locations starting from XX60H and the final result is expected to be less than FFH and stored in location XX70H. **(6.5)**

Ans. MVI B, 00_H ; clear (B) to save result
MVI C, 06_H; Set up register C to count six numbers
LXI H, $XX60_H$; Set up HL as a memory pointer
NEXT: MOV A, M ; Get a byte
RAL : Place D_7 into C4 flag
JC REJECT; If $D_7 = 1$, reject the byte
RAR; Restore the byte
ADD B; Add the previous sum
MOV B, A ; Save the sum

...ments the stack pointer by 2 and pushes the flag register. ...following steps:

(2) Disables INTR interrupt input by clearing IF flag.(interrupt flag)

(3) Resets the trap flag (TF).

(4) Decrements the stack by 2 and pushes current code segment register onto the stack.

(5) Decrements the stack by 2 and pushes instruction pointer IP onto the stack.

(6) Does an indirect far jump to the start of the procedure to respond to the interrupt.

Q.5. (a) Explain the function of following assembler directives with suitable examples:

(i) DB (ii) ASSUME (iii) EQU (iv) ORG (v) ENDP (vi) PTR **(6)**

Ans. (i) DB - The DB directive is used to declare a BYTE -2-BYTE variable - A BYTE is made up of 8 bits.

Declaration examples:

```
Byte1 DB 10h
Byte2 DB 255 ; 0FFh, the max. possible for a BYTE
CRLF DB 0Dh, 0Ah, 24h ;Carriage Return, terminator BYTE
```

(ii) ASSUME Directive - The ASSUME directive is used to tell the assembler that the name of the logical segment should be used for a specified segment. The 8086 works directly with only 4 physical segments: a Code segment, a data segment, a stack segment, and an extra segment. Example: ASUME CS:CODE ; This tells the assembler that the logical segment named CODE contains the instruction statements for the program and should be treated as a code segment.

(iii) EQU - This EQU directive is used to give a name to some value or to a symbol. Each time the assembler finds the name in the program, it will replace the name with the value or symbol you given to that name. Example: FACTOR EQU 03H

(iv) ORG - The *origin directive* tells the assembler where to load instructions and data into memory. It changes the program counter to the value specified by the expression in the operand field. Subsequent statements are assembled into memory locations starting with the new program/location counter value. If no ORG directive is encountered in a source program, the program counter is initialized to zero.

(v) ENDP - ENDP directive is used along with the name of the procedure to indicate the end of a procedure to the assembler Example: SQUARE_NUM PROCE; It start the procedure.

Some steps to find the square root of a number SQUARE_NUM ENDP ; Hear it is the End for the procedure

(vi) PTR - This PTR operator is used to assign a specific type of a variable or to a label. Example: INC [BX] ; This instruction will not know whether to increment the byte pointed to by BX or a word pointed to by BX. INC BYTE PTR [BX] ; increment the byte ; pointed to by BX This PTR operator can also be used to override the declared type of variable . If we want to access the a byte in an array WORDS DW 437Ah, 0B97h, MOV AL, BYTE PTR WORDS

1. Control Logic: The control logic block accepts control bus signals as well as inputs from the address bus, and issues commands to the individual group control blocks (Group A control and Group B control). It issues appropriate enabling signals to access the required data/control words or status word. The input pins for the control logic section are described here.

2. Group A and Group B Controls: Each of the Group A and Group B control blocks receives control words from the CPU and issues appropriate commands to the ports associated with it. The Group A control block controls Port A and PC_7–PC_4 while the Group B control block controls Port B and PC_3–PC_0.

3. Port A: This has an 8-bit latched and buffered output and an 8-bit input latch. It can be programmed in three modes: mode 0, mode 1 and mode 2.

4. Port B: This has an 8-bit data I/O latch/ buffer and an 8-bit data input buffer. It can be programmed in mode 0 and mode 1.

5. Port C: This has one 8-bit unlatched input buffer and an 8-bit output latch/buffer. Port C can be spitted into two parts and each can be used as control signals for ports A and B in the handshake mode. It can be programmed for bit set/reset operation.

Q.6.(b) Design a programmable timer using 8254 and 8086. Interface at an address 0080H for counter 0 and write the following assmebly language program. The 8086 and 8254 run at 6 MHz and 1.5 MHz respectively. **(6.5)**

(i) To generate a square wave of period 10 ms.

(ii) To interrupt the processro after 10 ms.

Ans. Neglecting the higher order address lines (A_{16}-A_8), the interfacing circuit diagram is shown in fig.. The 8254 is interfaced with lower order data bus (D_0-D_7), hence A_0 is used for selecting the even bank. The A_0 and A_1 of the 8254 are connected with with A_1 and A_2 of the processor. The counter address can be decoded as given below. If A_0 is the 8254 will not be selected at all.

A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
0	1	0	0	0	0	0	0	= 80H Counter 0
					0	0	0	= 82H Counter 1
					1	0	0	= 84H Counter 2
					1	1	0	= 86H Control word Reg.

(i) For generating a square wave, 8254 should be used in mode 3.

Let us select counter 0 for this purpose. That will be operated in BCD mode (may even be operated in HEX mode). Now suitable count is to be calculated for generating 10ms time period.

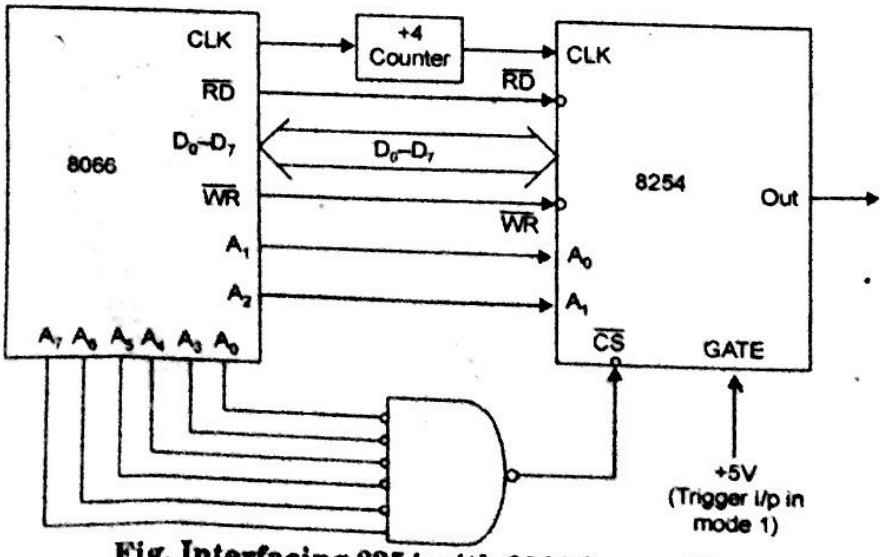


Fig. Interfacing 8254 with 8086 for problem.

```
        MOV AL,98H    : Load 98H as LSB of ocunt
        OUT 82H,AL    : in count reg of counter 1
        MOV AL,3AH    : then load 3AH is MSB
        OUT 82 H, AL  : of counter 1
        MOV AH 4CH    : RETURN TO DOS
        INT 21H
CODE    ENDS
        END START
```

Q.7.(a) Explain the difference between 8253/8254. Briefly explain its different modes of operation? (6)

Ans. Operation Modes: The D3, D2, and D1 bits of the Control Word set the operating mode of the timer. There are 6 modes in total; for modes 2 and 3, the D3 bit is ignored, so the missing modes 6 and 7 are aliases for modes 2 and 3. Notice that, for modes 0, 2, 3 and 4, GATE must be set to HIGH to enable counting. For mode 5, the rising edge of GATE starts the count. For details on each mode, see the reference links.

Mode 0 (000): Interrupt on Terminal Count: Mode 0 is used for the generation of accurate time delay under software control. In this mode, the counter will start counting from the initial COUNT value loaded into it, down to 0. Counting rate is equal to the input clock frequency.

The OUT pin is set low after the Control Word is written, and counting starts one clock cycle after the COUNT programmed. OUT remains low until the counter reaches 0, at which point OUT will be set high until the counter is reloaded or the Control Word is written. The Gate signal should remain active high for normal counting. If Gate goes low counting gets terminated and current count is latched till Gate pulse goes high again. the first byte of the new count when loaded in the count register,stop the previous count.

Mode 1 (001): Programmable One Shot: In this mode 8253 can be used as Monostable Multivibrator. GATE input is used as trigger input.

OUT will be initially high. OUT will go low on the Clock pulse following a trigger to begin the one-shot pulse, and will remain low until the Counter reaches zero. OUT will then go high and remain high until the CLK pulse after the next trigger.

After writing the Control Word and initial count, the Counter is armed. A trigger results in loading the Counter and setting OUT low on the next CLK pulse, thus starting the one-shot pulse. An initial count of N will result in a one-shot pulse N CLK cycles in duration.

The one-shot is retriggerable, hence OUT will remain low for N CLK pulses after any trigger. The one-shot pulse can be repeated without rewriting the same count into the counter. GATE has no effect on OUT. If a new count is written to the Counter during a oneshot pulse, the current one-shot is not affected unless the counter is retriggered. In that case, the Counter is loaded with the new count and the oneshot pulse continues until the new count expires.

Mode 2 (X10): Rate Generator: In this mode, the device acts as a divide-by-n counter, which is commonly used to generate a real-time clock interrupt.

Like other modes, counting process will start the next clock cycle after COUNT is sent. OUT will then remain high until the counter reaches 1, and will go low for one clock pulse. OUT will then go high again, and the whole process repeats itself.

The time between the high pulses depends on the preset count in the counter's register, and is calculated using the following formula:

$$\text{Value to be loaded into counter} = \frac{f_{input}}{f_{output}}$$

Keyboard Section: This sectio has 8 lines. RL0 - RL7. Plus 2 additional lines, Shift and CNTL/STB. The keys are automatically debounced and keyboard can operate in two modes:

>>two key lockout mode or

>>N-key rollover.

In two key lockout mode if 2 keys are pressed simultaneously only firstkey is recongnized.

In N key rollover mode, simultaneous keys are recongnized and stored in internal buffer: it can also be set up so that no key is recongnized until only one key is remained pressed.

This has a FIFO RAM.

The status logic keeps track of number of entries and provides IRQ(interrupt request) signal when

FIFO is empty.

DISPLAY SECTION: This section has 8 output lines divided into 2 groups of 4. A0 - A3 and B0 - B3. These lines can be used in both ways 8 lines or 2 sets of 4 lines. The display can be blanked using BD line. The section has 16x8 display RAM.

SCAN SECTION: This section has scan counter and 4 scan lines. SL0 - SL3. These 4 scan lines can be decoded using a 4 - 16 decoder to generate 16 lines for scanning. These 16 lines can be connected to rows of a matrix keyboard and digit driveers of multiplexed display

MPU INBTERFACE SECTION: This section has 8 bidirectional lines. DB0 - DB7. 1 interrupt request line(IRQ). 6 lines for interfacing including buffer address lines A0. When A0 is high - signals are interpreted as control word or status. When A0 is low - signal is interpreted as data. IRQ goes high whenever data is ready to be loaded into MPU.

UNIT - IV

Q.8.(a) Explain the block diagram and flag register of 8051 microcontroller? (6)

Ans. Major components of Intel 8051 microcontroller

The 8051 microcontroller is an 8-bit microcontroller. The major **components of 8051 microcontroller** and their functions..

An 8051 microcontroller has the following 12 major components:

1. ALU (Arithmetic and Logic Unit)
2. PC (Program Counter)
3. Registers
4. Timers and counters
5. Internal RAM and ROM
6. Four general purpose **parallel** input/output ports
7. Interrupt control logic with **five sources** of interrupt
8. Serial date communication
9. PSW (Program Status Word)
10. Data Pointer (DPTR)
11. Stack Pointer (SP)
12. Data and Address bus.

4. Timers and Counters: Synchronization among internal operations can be achieved with the help of clock circuits which are responsible for generating clock pulses. During each clock pulse a particular operation will be carried out, thereby, assuring synchronization among operations. For the formation of an oscillator, we are provided with two pins XTAL1 and XTAL2 which are used for connecting a resonant network in 8051 microcontroller device. In addition to this, circuit also consists of four more pins. They are,

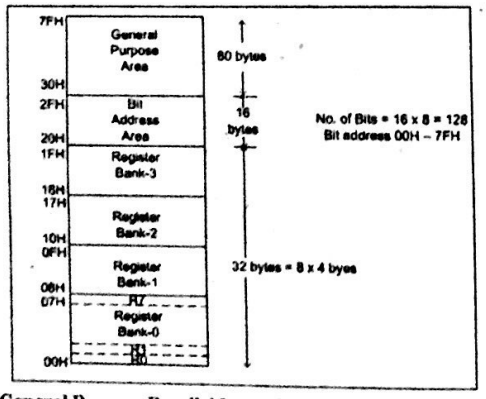
Internal operations can be synchronized using clock circuits which produce clock pulses. With each clock pulse, a particular function will be accomplished and hence synchronization is achieved. There are two pins XTAL1 and XTAL2 which form an oscillator circuit which connect to a resonant network in the microcontroller. The circuit also has 4 additional pins.

1. EA: External enable
2. ALE: Address latch enable
3. PSEN: Program store enable and
4. RST: Reset.

Quartz crystal is used to generate periodic clock pulses.

5. Internal RAM and ROM: ROM: A code of 4K memory is incorporated as on-chip ROM in 8051. The 8051 ROM is a non-volatile memory meaning that its contents cannot be altered and hence has a similar range of data and program memory, i.e, they can address program memory as well as a 64K separate block of data memory.

RAM: The 8051 microcontroller is composed of 128 bytes of internal RAM. This is a volatile memory since its contents will be lost if power is switched off. These 128 bytes of internal RAM are divided into 32 working registers which in turn constitute 4 register banks (Bank 0-Bank 3) with each bank consisting of 8 registers (R0 - R7). There are 128 addressable bits in the internal RAM.



6. Four General Purpose Parallel Input/Output Ports: The 8051 microcontroller has four 8-bit input/output ports. These are: PORT P0: When there is no external memory present, this port acts as a general purpose input/output port. In the presence of external memory, it functions as a multiplexed address and data bus. It performs a dual role.

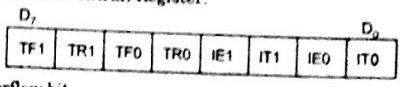
PORT P1: This port is used for various interfacing activities. This 8-bit port is a normal I/O port i.e. it does not perform dual functions.

PORT P2: Similar to PORT P0, this port can be used as a general purpose port when there is no external memory but when external memory is present it works in conjunction with PORT P0 as an address bus. This is an 8-bit port and performs dual functions.

PORT P3: PORT P3 behaves as a dedicated I/O port.

Q.8.(b) Explain the TMOD and TCON register? **(6.5)**

Ans. TCON: (Timer Contral) Register:



D_7 : Timer 1 overflow bit — [Set by hardware when Timer 1/0 overflows]

D_5 : Timer 0 overflow bit

D_6 : Timer 1 run bit — [Set cleared by software to turn time 1/0 ON/OFF.]

D_4 : Timer 0 run bit

D_3 : External interrupt 1 edge flag. — [Set by hardware when a H-L edge is detected on INT1/INT0]

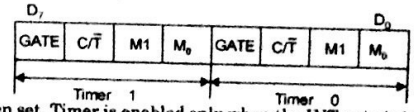
D_1 : External interrupt 0 edge flag

D_2 : Interrupt 1 triggering — [Set/cleared by software to make INT1/INT0 trigger on falling edge/low-level]

D_0 : Interrupt 0 triggering

[Set/cleared by software to make INT1/INT0 trigger on falling edge/low-level]

TMOD (Timer Mode) Register:



GATE: When set, Timer is enabled only when the INTx pin is high and TRx bit is set (H/W control) when cleared, Timer is enabled only when TR*x* = 1 (S/w control).

C/T: When set, counter operation is selected (i/p from Tx i/p pin). When cleared, Timer operation is selected (i/p from internal system clock).

M_1, M_0 :Mode bits

M1	M0	Model
0	0	13-bit Timer Mode → Mode 0
0	1	16-bit Timer Mode → Mode 1
1	0	8-bit Auto-reload →Mode 2
1	1	Split Timer Mode→Mode 3

Q.9.(a) Describe interrupts used in 8051. Give their priority and address. **(6)**

Ans. Interrupts are of different types like software and hardware, maskable and non-maskable, fixed and vector interrupts, and so on. Interrupt Service Routine (ISR) comes into the picture when interrupt occurs, and then tells the processor to take appropriate action for the interrupt, and after ISR execution, the controller jumps into the main program.

Types of Interrupts in 8051 Microcontroller

The 8051 microcontroller can recognize five different events that cause the main program to interrupt from the normal execution. These five sources of interrupts in 8051 are:

- Timer 0 overflow interrupt- TF0
- Timer 1 overflow interrupt- TF1
- External hardware interrupt- INT0
- External hardware interrupt- INT1
- Serial communication interrupt- RI/TI

The Timer and Serial interrupts are internally generated by the microcontroller, whereas the external interrupts are generated by additional interfacing devices or switches that are externally connected to the microcontroller. These external interrupts can be edge triggered or level triggered. When an interrupt occurs, the microcontroller

executes the interrupt service routine so that memory location corresponds to the interrupt that enables it. The Interrupt corresponding to the memory location is given in the interrupt vector table below

Interrupt vector Table

Interrupt Number	Interrupt Description	Address
0	External INT 0	0003h
1	Timer/Counter 0	0006h
2	External INT 1	00013h
3	Timer Counter 1	001Bh
4	Serial Port	0023h

Interrupt Structure of 8051 Microcontroller

Upon 'RESET' all the interrupts get disabled, and therefore, all these interrupts must be enabled by a software. In all these five interrupts, if anyone or all are activated, this sets the corresponding interrupt flags as shown in the figure. All these interrupts can be set or cleared by bit in some special function register that is Interrupt Enabled (IE), and this in turn depends on the priority, which is executed by IP interrupt priority register.

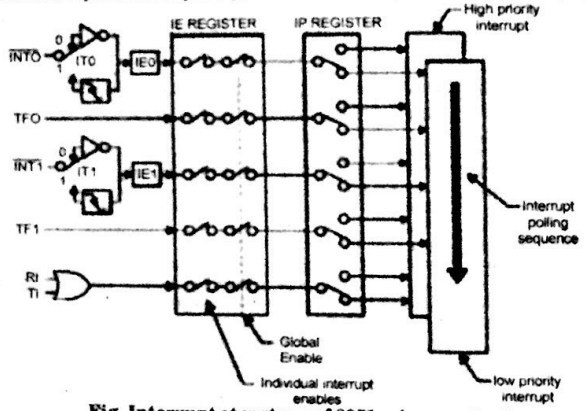


Fig. Interrupt structure of 8051 microcontroller

Q.9.(b) Write a program to generate a square wave of 5 KHz using timer 1 in mode 0 using 8051. **(6.5)**

Ans. We must toggle the bit to generate the square wave.

1.T = 1 / f = 1 / 5 kHz = 200 μs the period of the square wave.

2.1/2 of it for the high and low portions of the pulse is 100 μs.

3.100 μs / 1.085 μs = 92 and 65536 – 92 = 65444. which in hex is $FFA4_H$.

4.TL = $A4_H$ and TH = FF_H. all in hex.

The program is as follows.

```
MOV TMOD, #10H
Again: MOV TL1, #A4H
MOV TH1, #0FFH
SETB TR1
Back: JNB TF1, Back
CPL P1.5
CLR TF1
SJMP Again
HLT
```