**WINTER– 16 EXAMINATION**

**Model Answer** Subject Code: 17534

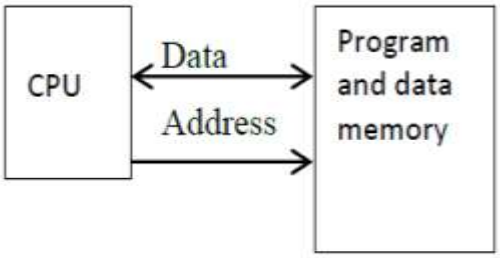
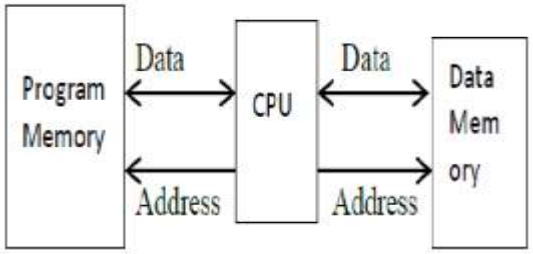
Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given more Importance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in the figure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

**Q 1 A) Attempt any three:** **Marks 12**

**a) Compare Von Neumann and Harvard architecture.(any four points)**

**Ans:(each point – 1 mark )**

| Sr.No | Von Neumann architecture | Harvard architecture |
|---|---|---|
| 1 |  |  |
| 2 | The Van Neumann architecture uses single memory for their instructions and data. | The Harvard architecture uses physically separate memories for their instructions and data. |
| 3 | Requires single bus for instructions and data | Requires separate & dedicated buses for memories for instructions and data. |
| 4 | Its design is simpler | Its design is complicated |
| 5 | Instructions and data have to be fetched in sequential order limiting the operation bandwidth. | Instructions and data can be fetched simultaneously as there is separate buses for instruction and data which increasing operation bandwidth. |
| 6 | Program segments & memory blocks for data & stacks have separate sets of addresses. | Vectors & pointers, variables program segments & memory blocks for data & stacks have different addresses in the program. |

**b) List the elements of Microcomputer. Explain any two in detail.**

**Ans(list –1 mark,explain any two elements—3 marks)**

The elements of a microcomputer are:

1) CPU

2) Program memory

3) Data memory

4) Output ports

5) Input ports

6) Clock generator:

(Any two elements can be explained)

**Central Processing Unit**: The CPU consists of ALU (Arithmetic and Logic Unit), Register unit and control unit. The CPU retrieves stored instructions and data word from memory; it also deposits processed data in memory.

**ALU (Arithmetic and Logic Unit):** This section performs computing functions on data. These functions are arithmetic operations such as additions subtraction and logical operation such as AND, OR rotate etc. Result are stored either in registers or in memory or sent to output devices.

**Register Unit:** It contains various register. The registers are used primarily to store data temporarily during the execution of a program. Some of the registers are accessible to the uses through instructions.

**Control Unit:** It provides necessary timing & control signals necessary to all the operations in the microcomputer. It controls the flow of data between the p and peripherals (input, output & memory). The control unit gets a clock which determines the speed of the p. The CPU has three basic functions

1) It fetches an instructions word stored in memory.

2) It determines what the instruction is telling it to do.(decodes the instruction)

3) It executes the instruction. Executing the instruction may include same of the following major tasks.. Transfer of data from reg. to reg. in the CPU itself.

- . Transfer of data between a CPU reg. & specified memory location.
- . Performing arithmetic and logical operations on data from a specific memory location or a designated CPU register.
- . Directing the CPU to change a sequence of fetching instruction, if processing the data created a specific condition.
- Performing housekeeping function within the CPU itself in order to establish desired condition at certain registers.

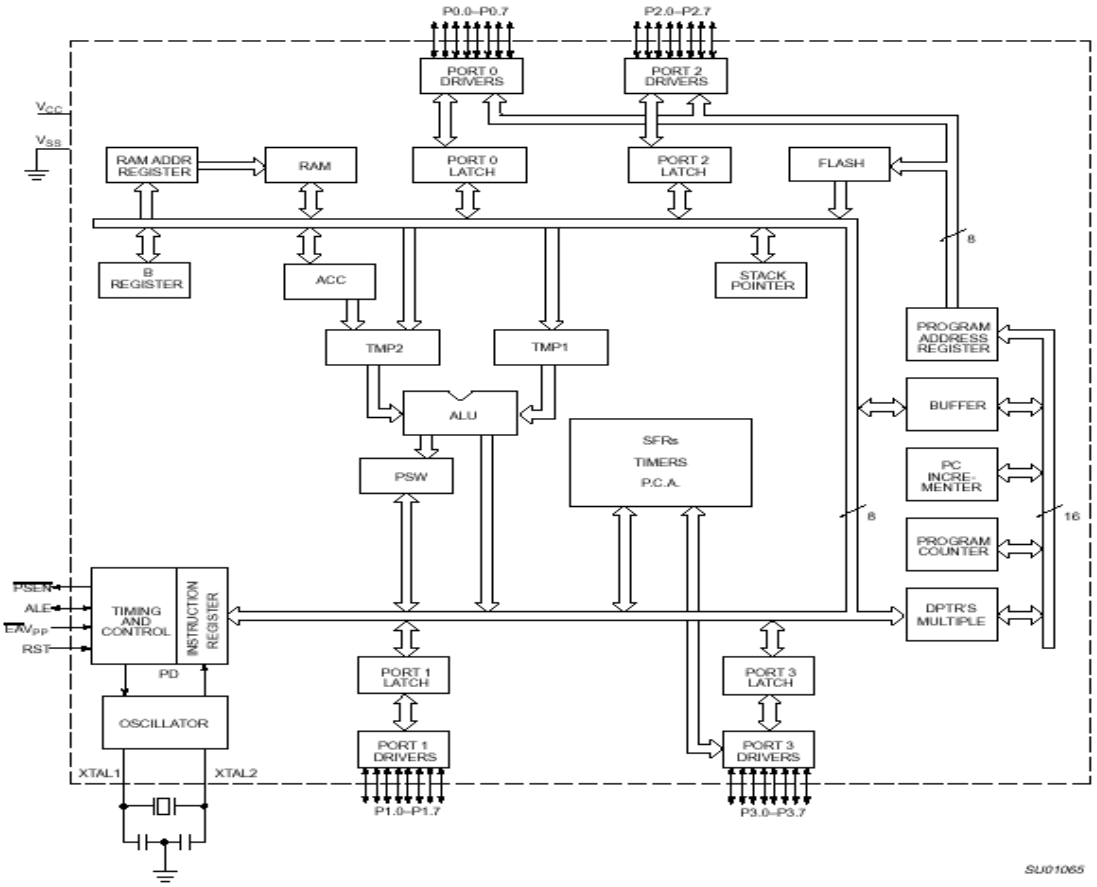
4) It looks for control signal such as interrupts and provides appropriate responses.

5) It provides states, control, and timing signals that the memory and input/output section can use.

**Program Memory**: The basic task of a microcomputer system into ensure that its CPU executes the desired instruction sequence is the program properly. The instruction sequence is stared in the program memory on initialization- usually a power up and manual reset the processor starts by executing the instruction in a predetermined location in program memory. The first instruction of the program should therefore be in this location in typical µp basic system, the program to be executed is fixed one which does not change. Therefore µp program are store on ROM, or PROM, EPROM, EEPROM. In the trainer kit, ROM contains only the monitor program. The user program is not stored in ROM because it needs not to be stored permanently

c) Draw the architecture of 8051 Microcontroller .

**Ans(correct diagram –4 marks)**




**d)Write the operation of the following instructions of 8051.**

**i) CJNE A , DIREC T,REL**

**ii) MUL AB**

**iii) XCHD A ,@$R_I$**

**iv) MOVX A, @DPTR**

**Ans: (each correct instruction explanation – 1 marks each)**

**i) CJNE A, direct, rel**

Compare the contents of the accumulator with the 8 bit data directly mentioned in the instruction and if they are not equal then jump to the relative address mentioned in the instruction.

Example: CJNE A, 04H, UP

Compare the contents of the accumulator with the 04H mentioned in the instruction and if they are not equal then jump to the line of instruction where UP label is mentioned.

**ii). MUL AB**

Multiplies the unsigned value of the Accumulator by **t**he unsigned value of the "B" register. The least significant byte (LSB) of the result is placed in the Accumulator and the most-significant-byte is placed in

the "B" register. The **Carry Flag (C)** is always cleared. The **Overflow Flag (OV)** is set if the result is greater than 255 (if the most-significant byte is not zero), otherwise it is cleared.

**Example:**
```
MOV A, #5
MOV B, # 7

MUL  AB          ; A= 35(decimal) =23H, B=00
```

iii) **XCHD A , @$R_I$**

Exchange digits

The XCHD instructions exchanges only the lower nibble of A with the lower nibble of the RAM location pointed to by Ri while leaving the upper nibbles in both places intact

**Example:** Assuming RAM location 40H has the value 97H, find its content after the following instructions

40H= (97H)

```
MOV A, #12H        ; A= 12H ( 0001   0010 binary)

MOV R1, # 40       ; R1= 40H, load pointer

XCHD A, @R1        ; exchange the lower nibble of A and RAM location 40H
```

After execution of the XCHD instruction, we have A= 17H and RAM location 40H has 92H.

**Iv). MOVX A,@DPTR**

This instruction moves the contents of the external RAM memory pointed by (or stored in) DPTR to accumulator.

**Example:**
```
MOV DPTR, # 2000H        ; DPTR = 2000H(external RAM address)

MOV A, @DPTR             ; 2000H = 0BH
                         ;A = 0BH
```

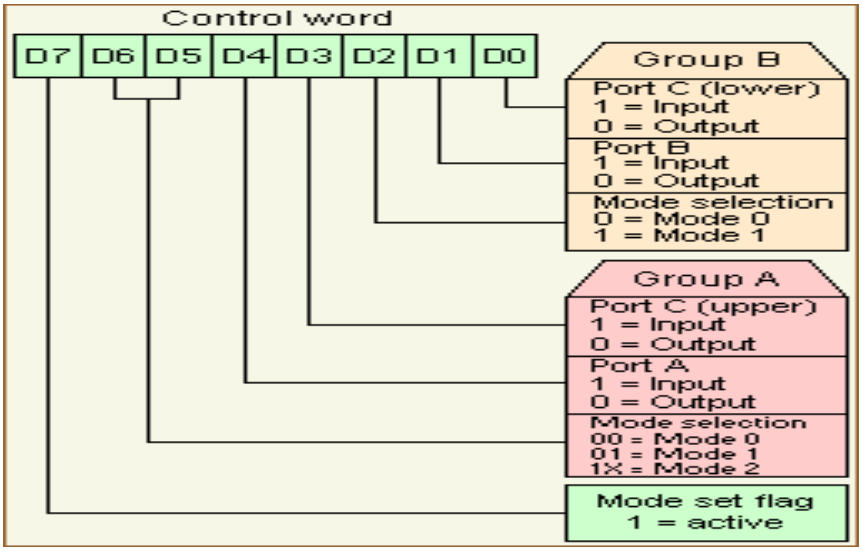
**e)** Draw the control word format of 8255 for I/O mode. write a control word to configure 8255 as below:

**i) port A as an input port in mode 0**

**ii) port B as an output port in mode 1**

**iii) port c as an input port in mode 0**

**Ans ) Control word format: 1 mark; Correct control word for i), ii) & iii): 3Marks**




**i) Port A as an input port in mode 0**

| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
|---|---|---|---|---|---|---|---|

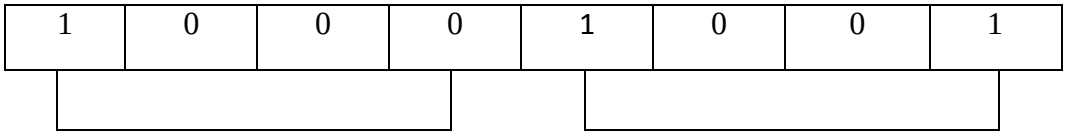
**=90H**

**ii)Port B as an output port in mode 0**

| 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
|---|---|---|---|---|---|---|---|

**=84H**

**iii)Port C as an input port in mode 0**




**=89H**

**B) Attempt any one: Marks 6**

**a) Write an assembly language program for 8051 mic to add five 8 bit numbers stored in internal RAM from 50H onwards store the result at 60H.**

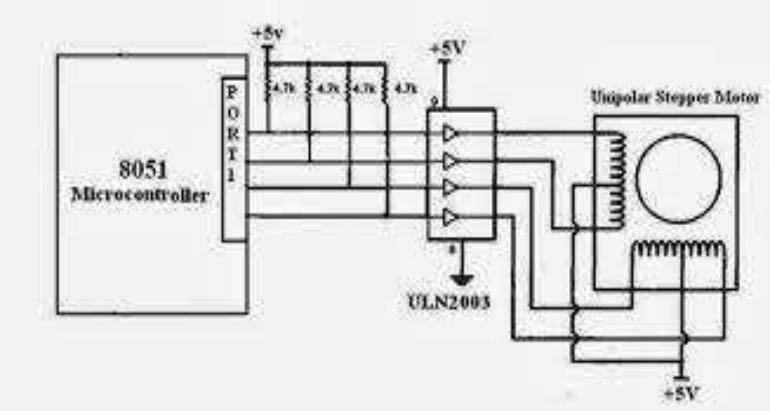
**Ans: (any correct program with comments---6 marks)**

Program for addition of five 8 bit nos.

```
        CLR PSW.3          ; Select register Bank 0
        CLR PSW.4
        MOV R0, #05H       ; Initialize byte counter
        MOV R1, #50H             ; Initialize memory pointer
        MOV A, # 00H             ; Clear Accumulator
UP:     ADD A, @R1         ; Add accumulator with number from array
        INC R1                   ; Increment memory pointer
        DJNZ R0, UP        ; Decrement byte counter,
                           ; if byte counter ≠ 0
                           ; Then go to UP
        MOV 60H, A         ; Store result in internal memory
        HERE: SJMP HERE    ; Stop
```

**b) Draw the interfacing diagram of steeper motor with 8051 mic . Write an ALP to rotate the stepper motor continuously in anticlockwise direction. Assume step angle is $0.9^0$/step.**

3 marks diagram , 3 marks program




| Step no | Winding A | Winding B | Winding C | Winding D | anticlockwise |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 0 | 1 | |
| 2 | 1 | 1 | 0 | 0 | |
| 3 | 0 | 1 | 1 | 0 | |
| 4 | 0 | 0 | 1 | 1 | |

Calculation for no. of steps:

**Count = <u>Required Rotaion Angle</u>**

**1.8 degree*no. of steps**

**=180/0.9*4 steps=200/4=50**

PROGRAM:

```
        MOV A, #66H                 ; load step sequence
        MOV R0, # 50                ; count for 4 steps
BACK:   MOV P1, A                   ; issue sequence to motor
AGAIN:  RL A                        ; rotate left anticlockwise
        ACALL DELAY                 ;wait
        DJNZ R0, AGAIN
        SJMP BACK                   ;keep going
```

```
        DELAY                   ; delay subroutine.
        MOV R2, #100
H1:     MOV R3, #255
H2:     DJNZ R3, H2
        DJNZ R2, H1
        RET
```

2. Attempt any FOUR: 16 marks

a) Explain Boolean processor of 8051 microcontroller with two instructions.

**Ans: 3M- explanation 1M-example**

The 8051 instruction set is optimized for the one bit operations. So often desired in real world, real time control applications. The Boolean processor provides direct support for bit manipulation. This leads to more efficient programs that need to deal with binary input and output conditions inherent in digital control problems.

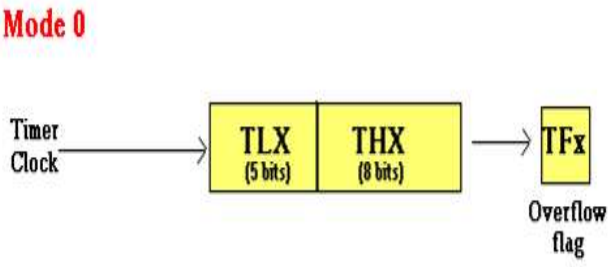
Bit addressing can be used to test pin monitoring or program control flags. For examples, instructions for Boolean function are as given below.
(a) ORL P0, # 1 ; Set P0.0
(b) XRL P0, # 1 ; Toggle P0.0

b) Explain four timer modes of 8051uC.

**Ans: 1M-each mode**

Operating modes of Timer: The timer may operate in any of the four modes that are determined byM1 and M0 bit in TMOD register.

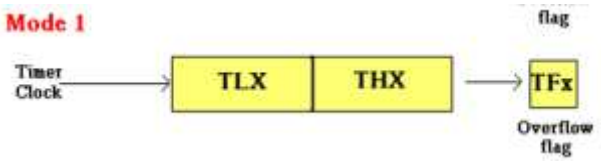
**Mode 0:**




**I**n mode0 the register THX is used as 8 bit counter and TLX is used as5 bit counter. The pulse i/p is divided by $(32)_{10}$so that TH counts. Hence original oscillator frequency is divided by $(384)_{10}$.The

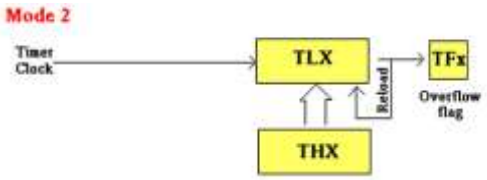
timer flag is set when THX rolls over from FF to 00H.

**Mode 1:**



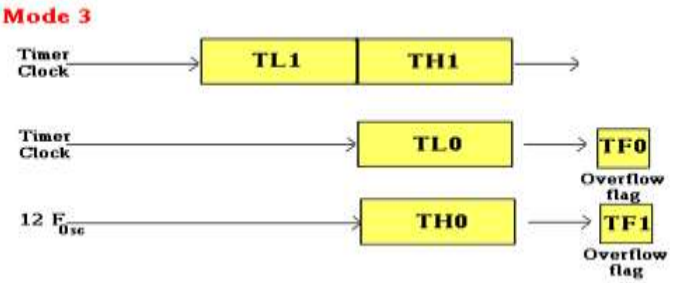

It is similar to Mode 0 except TLX is configured as a full 8-bit counter. Hence pulse input is divided by $256_{10}$ so that TH counts the timer flag is set when THX rolls over from FF to 00H.

**Mode 2**




In this mode only TLX is used as 8-bit counter. THX is used to hold the value which is loaded in TLX initially. Everytime TLX overflows from FFH to 00H the timer flag is set and the value from THX is automatically reloaded in TLX register.

**Mode 3**




In this mode, timer 0 becomes two completed separate 8-bit timers. TL0 is controlled by gate arrangement of timer 0 and sets timer 0 flag when it overflows. TH0 receives the timer clock under the control of TR1 bit and sets TF1 flag when it overflows. Timer 1 may be used in mode 0, 1 and 2 with one important exception that no interrupt will be generated by the timer when the timer 0 is using TF1 overflow flag.

c) Write an assembly language program to add two BCD numbers 66H and 95H which are stored at external memory location 3000H and 3001H respectively. Store the result at memory location 3002H.

**Ans: 3M- program,1M- comments**

**Data:**
**3000H- 66H**
**3001H- 95H**

```
MOV DPTR, #3000H        ; Initialize data pointer with external ML
MOVX A, @DPTR           ; load 1st no. in acc from the external ML
MOV R2, A               ; move the 1st no. in reg R2
```

```
        INC DPTR                ; increment data pointer
        MOVX A, @DPTR           ; load the 2nd no. in acc from external ML
        ADD A, R2               ; add 1st no. with 2nd no.
        DA A                    ; decimal adjust accumulator after addition
        INC DPTR
        MOV @DPTR, A            ; store result in internal memory
LOOP:   AJMP LOOP               ; stop
```

d) Explain the function of the following registers of 8051.

i. Stack pointer
ii. DPTR
iii. Program counter
iv. Accumulator

**Ans: 1M- each**

**i. Stack pointer:** The 8-bit stack pointer (SP) register is used to hold an internal RAM address that is called the top of stack. The address held in the SP register is the location in internal RAM where the last byte of data was stored by a stack operation.

**ii. DPTR :** The DPTR register is made up of two registers named as DPH & DPL which are used to access any memory address that may be internal & external code access & external data access. The DPTR is under the program control & can also be specified as 16-bit pointer as DPTR or by individual 8- bits as DPH & DPL. DPTR does have a single address but the DPH is assigned the address as 82 H.

**iii. Program counter:** The program counter (PC) IS A 16-bit register. It is used to hold address of a byte in memory. Program instruction bytes are fetched from locations in memory that are addressed by PC. Program ROM may be on chip at addresses OOOOH to OFFF H. external to the chip for addresses that exceeds OFFF h or totally external for all addresses from 0000H to external for all addresses from OOOOH to FFFF H. The PC is incremented automatically after every instruction byte is fetched. The PC is the only register that does not have any address.

**iv. Accumulator:** It is also called as register A. It is an 8- bit register. The CPU of 8051 is accumulator based hence it is used to hold the source operand and result of arithmetic operations like addition , subtraction, multiplication, division .However it is source as well as destination for logical operations and data movement instructions. It can be used as a look up table pointer. It is also used in RAM expansion. It is specially used for rotate,parity computation, testing for zero etc. It is bit accessible.

e) Give the address of SFRs, TCON, TMOD, IE,SCON,TL0,TL1,SBUF and IP.

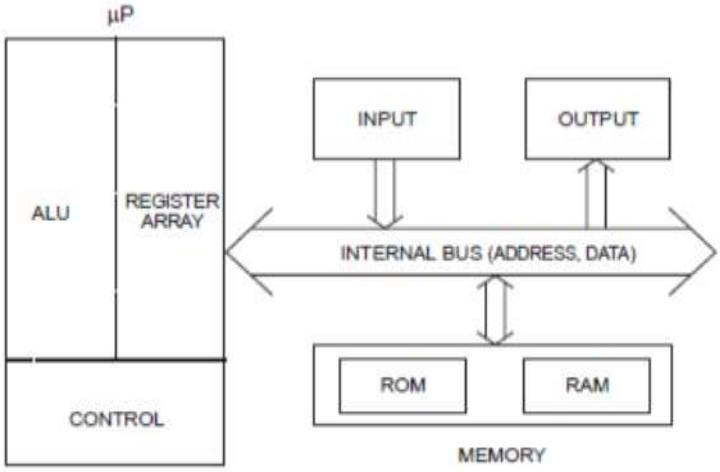
**Ans: 1/2M- each**

Addresses of SFRs are as follows:

| SFR | ADDRESS |
|---|---|
| TCON | 88H |

| | |
|---|---|
| TMOD | 89 H |
| IE | 0A0 H |
| SCON | 98H |
| TL0 | 8A H |
| TL1 | 8B H |
| SBUF | 99 H |
| IP | 0B8 H |

f) Draw the general block diagram of microprocessor. Explain the function of each block.
**Ans: 2M- diagram, 2M- explanation (marks should be given for any other correct diagram and explanation.)**




A typical microprocessor architecture is shown in Figure above. The various functional units are as follows:

1. **Buses:** A bus has a wire or line for each bit and thus allows exchange of all bits of a word in parallel. The processing of bits in the µP is also in parallel. The busses can thus be viewed as data highways. The width of a bus is the number of signal lines that constitute the bus. The figure shows for simplicity three busses for distinct functions. Over the address bus, the µP transmits the address of that I/O device or memory locations which it desires to access. This address is received by all the devices connected to the processor, but only the device which has been addressed responds. The data bus is used by the µP to send and receive data to and from different devices (I/O and memory) including instructions stored in memory. Obviously the address bus is unidirectional and the data bus is bi-directional. The control bus is used for transmitting and receiving control signals between the µP and various devices in the system.
2. **ALU:** Arithmetic-Logic Unit (ALU) The arithmetic-logic unit is a combinational network that performs arithmetic and logical operations on the data.

3. Internal registers: A number of registers are normally included in the microprocessor. These are used for temporary storage of data, instructions and addresses during execution of a program. Those in the Intel8085 microprocessor are typical as i) Accumulator ii) General Purpose Registers or Scratch Pad Memory There are six general purpose 8-bit registers that can be used by the programmer for a variety of purposes. These registers, labelled as B, C, D, E, H and L, can be used individually (e.g., when operation on 8-bit data is desired) or in pairs (e.g., when a 16-bit address is to be stored). Only B-C, D-E and H-L pairs are allowed. (iii) Instruction Register (IR) (iv) Program Counter etc.

3. **Memory unit :** Memories like RAM and ROM are not inbuilt in any microprocessor. Memory is interfaced externally to increase the storage capacity of microprocessor. The maximum memory capacity which can be interfaced is decided by the number of address lines available

4. **Control unit:** It generates the different control signals required for communication.

5. **Input /Output devices:** Microprocessors doesn't have ports, so external peripherals ICs like 8155 and 8255PPI are used to communicate with I/O devices.

3. Attempt any FOUR 16 marks

a) State the need of directives used in assembly language programming. Explain any two directives with examples.

**Ans: 1M- need, 1 ½ M-any two correct directive(any two)**

**i)** ORG:- ORG stands for Origin

Syntax: | ORG | Address |

The ORG directive is used to indicate the beginning of the address. The number that comes after ORG can be either in hex or in decimal. If the number is not followed by H, it is decimal and the assembler will convert it to hex. Some assemblers use "**.** ORG" (notice the dot) instead of "ORG" for the origin directive.

**ii)** DB:- (Data Byte)

Syntax: | Label: | DB | Byte |

Where byte is an 8-bit number represented in either binary, Hex, decimal or ASCII form. There should be at least one space between label & DB. The colon (:) must present after label. This directive can be used at the beginning of program. The label will be used in program instead of actual byte. There should be at least one space between DB & a byte. Following are some DB examples:

```
        ORG  500H
DATA1:  DB   28              ;DECIMAL(1C in hex)
DATA2:  DB   00110101B       ;BINARY (35 in hex)
DATA3:  DB   39H             ;HEX
        ORG  510H
DATA4:  DB   "2591"          ;ASCII NUMBERS
        ORG  518H
DATA6:  DB   "My name is Joe"  ;ASCII CHARACTERS
```

**iii)** EQU: Equate

It is used to define constant without occupying a memory location.

Syntax: | Name | | EQU | | Constant |

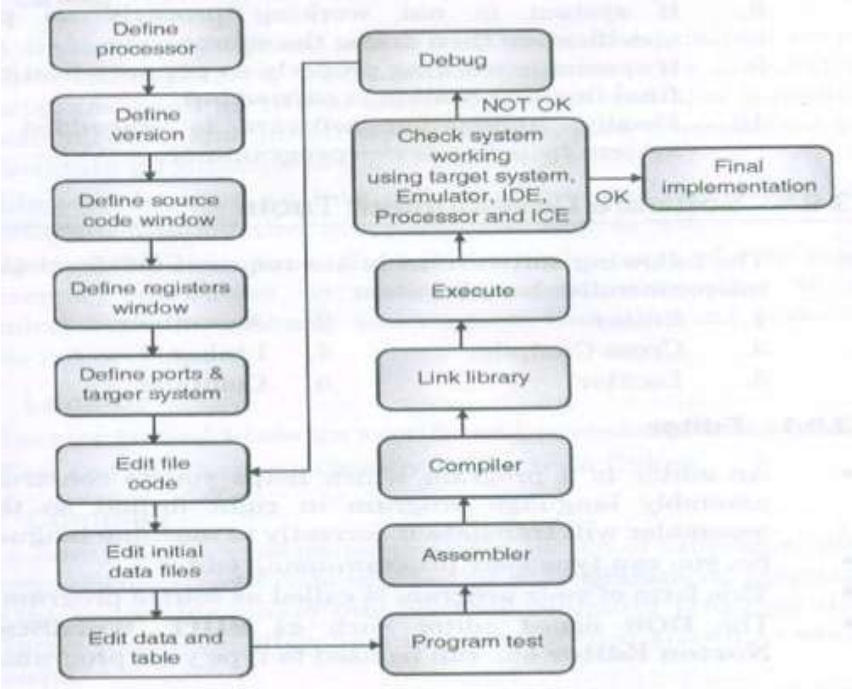
By means of this directive, a numeric value is replaced by a symbol.
For e.g. MAXIMUM EQU 99 After this directive every appearance of the label "MAXIMUM" in the program, the assembler will interpret as number 99 (MAXIMUM=99).

iv) **END:**

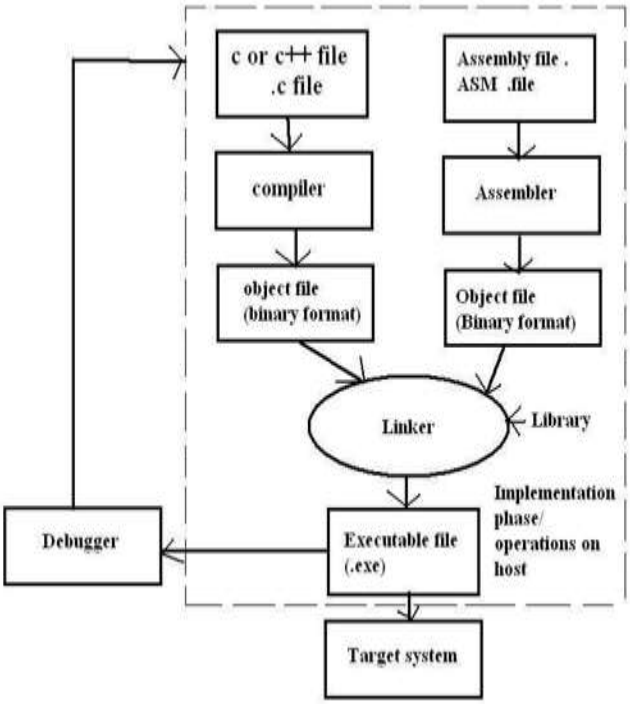
This directive must be at the end of every program. meaning that in the source code anything after the END directive is ignored by the assembler.
This indicates to the assembler the end of the source file (asm).
Once it encounters this directive, the assembler will stop interpreting program into machine code.
e.g. END ; End of the program

b) Draw the software development cycle. State the function of editor, assembler and cross compiler.

**Ans: 1M- diag, 3M- each function**




**OR**




**1) Editor:** An editor is a program which helps you to construct your assembly language program in right format so that the assembler will translate it correctly to machine language. So, you can type your program using editor. This form of your program is called as source program and extension of program must be .asm or .src depending on which assembler is used. The DOS based editor such as EDIT, WordStar, and Norton Editor etc. can be used to type your program.

**2) Assembler:** An assembler is programs that translate assembly language program to the correct binary code for each instruction i.e. machine code and generate the file called as Object file with extension .obj and list file with extension .lst extension.
Some examples of assembler are ASEM-51, Kiel's A51, AX 51 and C51, Intel PL/M-51 etc.
**3) Cross Compiler :**A cross compiler is used to create executable code for a platform other than the one on which the compiler is run. Cross compiling is compiling something for different CPU type than the one you are running on. You use a cross compiler to
produce objects for a platform other than the local host. Keil and SPJ have its own cross compiler

c) If the initial contents of ACC = 0FH. State the accumulator contents after execution of the following instructions independently.
i. CPL A ii. RRA iii. ANL A,#0FH iv. SWAP A
**Ans: 1M-each**
i. CPL A
Acc= F0H
ii. RRA
Acc= 87H
iii. ANL A,# 0FH

Acc= 0FH
iv. SWAP A
Acc = F0H

d) List any four important features of 8051 microcontroller.

**Ans: 1M-each (any four)**

Features of 8051 micro controller are as follows:-

1) 8- bit data bus and 8- bit ALU.
2) 16- bit address bus – 64KB of RAM and ROM.
3) On- chip RAM -128 (256) bytes (" Data Memory")
4) On- chip ROM – 4 KB ("Program Memory")
5) Four 8-bit bi- directional input/output ports Four 8-bit bi- directional input/ output ports.
6) Programmable serial ports i.e. One UART (serial port)
7) Two 16- bit timers- Timer 0& Timer 1
8) Six interrupts are available: Reset, Two interrupts Timers i.e. Timer 0 and Timer 1, Two external hardware interrupts- INT0 and INT1, Serial communication interrupt for both receive and transmit

e) State the alternate function of port 3 pins of 8051 microcontroller.

**Ans: ½-M each function**

| Pin | Name | Alternate Function |
|---|---|---|
| P3.0 | RXD | Serial input line(Receive) |
| P3.1 | TXD | Serial output line(Transmit) |
| P3.2 | $\overline{\text{INT0}}$ | External interrupt 0 |
| P3.3 | $\overline{\text{INT1}}$ | External interrupt 1 |
| P3.4 | T0 | Timer 0 external input |
| P3.5 | T1 | Timer 1 external input |
| P3.6 | $\overline{\text{WR}}$ | External data memory write strobe |
| P3.7 | $\overline{\text{RD}}$ | External data memory read strobe |

4. (A) Attempt any THREE 12 marks

a) Explain the following addressing modes with the help of ADD instruction in
i. Direct addressing mode
ii. Indirect addressing mode
iii. Register addressing mode
iv. Immediate addressing mode

**Ans: 1M- each**

1) Direct Addressing mode
ADD A, add (8-bit Address)

A ←A+ (add)
Eg. ADD A, 12H

The contents of memory location specified by 8 bit direct address will be logically added bit by bit with the contents of accumulator and result is stored in accumulator.

2) Indirect addressing mode

ADD A,@Ri
A ←A+ (Ri)
ADD A,@R0

The content of memory location whose address is specified by Ri (R0/R1) will be logically added bit by bit with contents of accumulator. Result is stored in accumulator. Only R0 or R1 can be used.

3) Register addressing mode

ADD A, Rn
A ←A+ Rn
ADD A, R2

The contents of specified register Rn (R0-R7) will be logically added bit by bit with the contents of accumulator and result is stored in Accumulator.

4) Immediate addressing mode

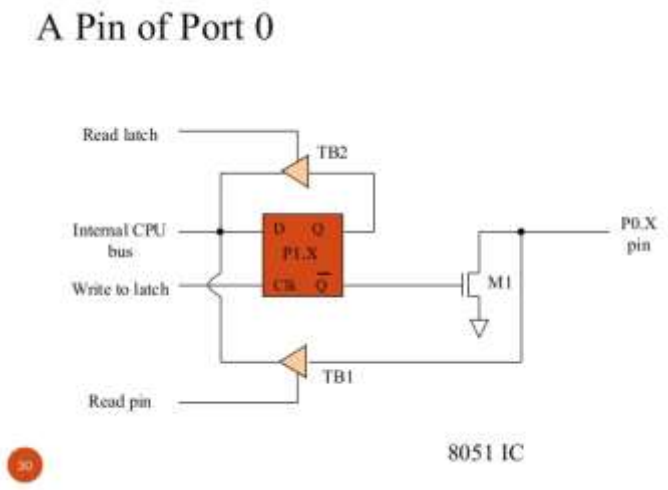
ADD A, #data(8-bit)
A ←A+ #data
ADD A, #23H

Immediate data is logically added bit by bit with contents of accumulator and result is stored in accumulator.

b) Draw the port 0 pin circuit and describe the operation.

**Ans:2M- diag, 2M- explanation**

Port 0: It may be used as input/output or bidirectional low order address and data bus for external memory. It does not have internal pull up resistors.




c) Explain the four operating modes of serial communication of 8051 microcontroller.

**Ans: 1M- each mode**

8051 micro controller communicate with another peripheral device through RXD and TXD pin of port3.Controller have four mode of serial communication. This four mode of serial communication are below.

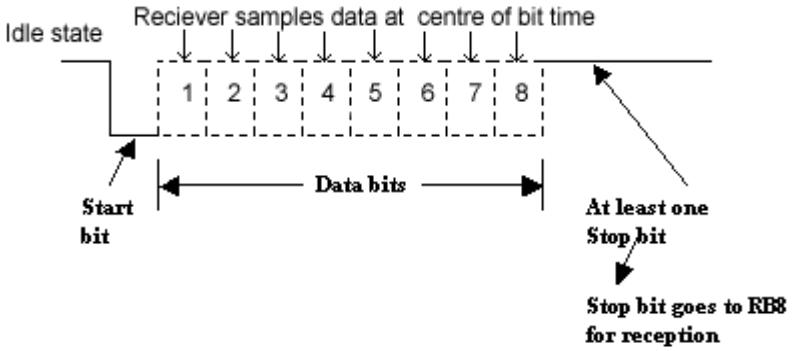
1. Serial data mode 0-fixed baud rate.
2. Serial data mode 1-variable baud rate.
3. Serial data mode 2 -fixed baud rate.
4. Serial Data mode 3 -variable baud rate.

**1. Serial Data Mode-0 (Baud Rate Fixed)**

In this mode, the serial port works like a shift register and the data transmission works synchronously with a clock frequency of $f_{osc}$ /12. Serial data is received and transmitted through RXD. 8 bits are transmitted/ received aty a time. Pin TXD outputs the shift clock pulses of frequency $f_{osc}$ /12, which is connected to the external circuitry for synchronization. The shift frequency or baud rate is always 1/12 of the oscillator frequency.

**2. Serial Data Mode-1 (standard UART mode)(baud rate is variable)**

In mode-1, the serial port functions as a standard Universal Asynchronous Receiver Transmitter (UART) mode. 10 bits are transmitted through TXD or received through RXD. The 10 bits consist of one start bit (which is usually '0'), 8 data bits (LSB is sent first/received first), and a stop bit (which is usually '1'). Once received, the stop bit goes into RB8 in the special function register SCON. The **baud rate is variable**.




$$f_{baud} = \frac{2^{SMOD}}{32} X \frac{fosc}{12 X [256-(TH1)]}$$

**3. Serial Data Mode-2 Multiprocessor (baud rate is fixed)**

In this mode 11 bits are transmitted through TXD or received through RXD.The various bits are as follows: a start bit (usually '0'), 8 data bits (LSB first), a programmable 9 $^{th}$ (TB8 or RB8)bit and a stop bit (usually '1'). While transmitting, the 9 $^{th}$ data bit (TB8 in SCON) can be assigned the value '0' or '1'. For example, if the information of parity is to be transmitted, the parity bit (P) in PSW could be moved into TB8.On reception of the data, the 9 $^{th}$ bit goes into RB8 in 'SCON', while the stop bit is ignored. The baud rate is programmable to either 1/32 or 1/64 of the oscillator frequency.

$$\mathbf{f_{baud} = (2^{SMOD}/64) f_{osc}}$$

**4. Serial Data Mode-3 - Multi processor mode(Variable baud rate)**

In this mode 11 bits are transmitted through TXD or received through RXD.The various bits are: a start bit (usually '0'), 8 data bits (LSB first), a programmable 9 th bit and a stop bit (usually '1'). Mode-3 is same as mode-2, except the fact that the baud rate in mode-3 is variable (i.e., just as in mode-1).

$$f_{baud} = (2^{SMOD} /32) * ( f_{osc} / 12 (256-TH1))$$

d) Write an assembly language program for 8051 microcontroller to transfer letter 'A' serially at 4800 baud rate continuously.

**Ans: 1M- calculation,**

**Calculation :**

The machine cycle frequency of 8051 = 11.0592 / 12 = 921.6 kHz, and 921.6 kHz / 32 = 28.800 Hz is frequency by UART to timer 1 to set baud rate.

28800/6=4800 where -6=FAH is loaded into TH1.

```
MOV TMOD, #20H          ; Timer 1 , Mode 2
MOV TH1, #-6            ; 4800 Baud
MOV SCON, #50H          ; 8Bit ,1 Stop Bit
 SETB TR1
 AGAIN: MOV SBUF, # ”A”  ; Letter A to be transferred
HERE: JNB TI, HERE       ; Wait for the last bit
 CLR TI
 SJMP AGAIN
 END
```

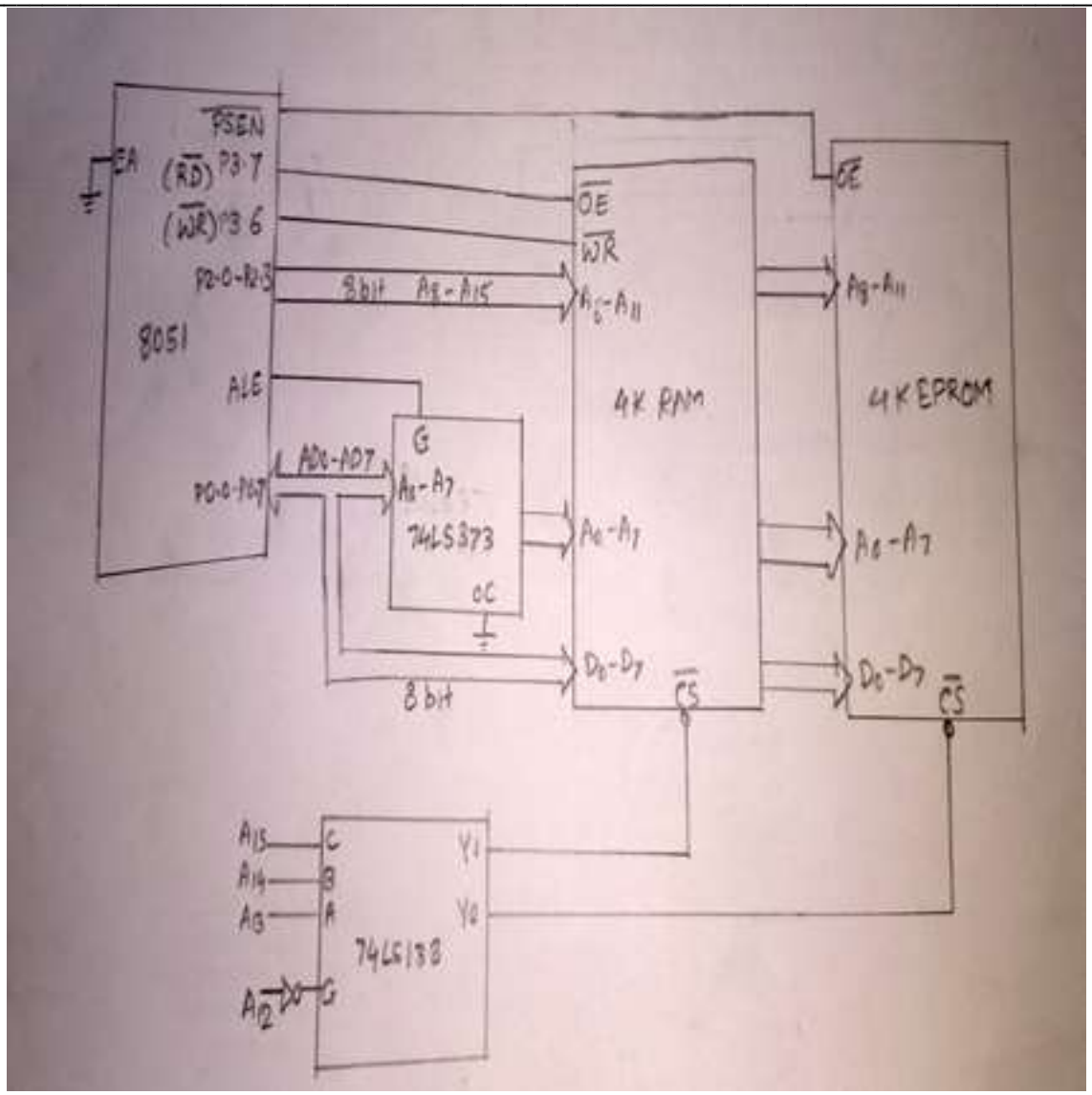
B) Attempt any ONE 4 marks

a) Write an ASL program in 8051uc to find largest number from the array of ten numbers stored in external RAM memory. Starting at 3000H.
Ans: 3M- correct prog, 1M- comments

**Program:**

```
     CLR PSW.3              ; Select Bank 0  PSW.3
    MOV R1, 0AH             ; Initialize byte counter
    MOV DPTR, # 3000H       ; Initialize memory pointer
    DEC R1                  ; Decrement byte counter by 1
    MOV X A, @DPTR          ; Load number in accumulator
    MOV 40 H, A             ; Store number in memory location
UP: INC DPTR                ; Increment memory pointer by 1
    MOVXA, @DTPR            ; Read next number
    CJNE A, 40 H, DN        ; if number≠ next number, and then go to NEXT
DN: JC NEXT                 ; If next number > number then go to NEXT
    MOV 40H, A              ; Else replace NEXT number with number
NEXT: DJNZ R1, UP           ; Decrement byte counter by 1, if byte counter≠  0 then go to UP
    INC DPTR                ; Increment memory pointer by 1
    MOV A, 40H
    MOVX@ DPTR, A           ; Store result t external memory location
LOOP: AJMP LOOP             ; Stop
```

b) Draw interfacing diagram of 4Kbyte EPROM and 4Kbyte RAM to 8051 microcontroller. Draw memory map.( 2 marks  diagram , 2 marks memory map )


PSEN
EA
(RD) P3.7
(WR) P3.6
P2.0-P2.3
8bit A8-A15
8051
ALE
P0.0-P0.7
AD0-AD7
G
A0-A7
74LS373
OC
8 bit
OE
WR
A0-A11
4K RAM
D0-D7
CS
OE
A0-A11
4K EPROM
A0-A7
D0-D7
CS
A15
A14
A13
C
B
A
74LS138
Y1
Y0
G
A12

| | A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | ADDR |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Start addr of EPROM | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0000H** |
| End addr of EPROM | **0** | **0** | **0** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **0FFFH** |
| Start addr of RAM | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **2000H** |
| End addr of RAM | **0** | **0** | **1** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **2FFFH** |

**5 Attempt any four** **Marks 16**

**a).List the interrupts used in 8051.Give the priorities and addresses.**

**Ans(list -1 mark, priorities and address—3 marks)**

| **Interrupt Source** | **Vector address** | **Interrupt priority** |
|---|---|---|
| External Interrupt 0 –INT0 | 0003H | 1 |
| Timer 0 Interrupt | 000BH | 2 |
| External Interrupt 1 –INT1 | 0013H | 3 |
| Timer 1 Interrupt | 001BH | 4 |
| Serial Interrupt | 0023H | 5 |

**b), Describe the function of following pins of 8051 microcontroller ( 2 marks each function )**

**i) EA/VPP**

**ii) ALE/PROG.**

**Function of EA/VPP**

1. EA which stands for external access is pin number 31 in the DIP packages. It is an input pin and must be connected to either Vcc or GND. In other words, it cannot be left unconnected.

2. The lowest 4K (or SK or 16K) bytes of Program Memory can be either in the on-chip ROM or in an external ROM. This selection is made by strapping the EA (External Access) pin to either VCC or Vss.

3. In the 4K byte ROM devices, if the pin is strapped to Vcc, then program fetches to addresses 0000H through OFFFH are directed to the internal ROM. Program fetches to addresses 1000H through FFFFH are directed to external ROM.

4. If the pin is strapped to Vss, then all program fetches are directed to external ROM. The ROMless parts must have this pin externally strapped to VSS to enable them to execute properly.

**Function of ALE/ PROG:**

1. ALE stands for address latch enable. It is an output pin and is active high for latching the low byte of address during accesses to external memory.

2. The ALE pin is used for demultiplexing the address and data by connecting to the G pin of the 74LS373 chip.

**c)Draw the format of IE register and describe it.**

Ans: **(2M- format, 2M –explanation)**

**IE: INTERRUPT ENABLE REGISTER. BIT ADDRESSABLE.**

If the bit is 0, the corresponding interrupt is disabled. If the bit is 1, the corresponding interrupt is enabled.

| EA | — | ET2 | ES | ET1 | EX1 | ET0 | EX0 |
|---|---|---|---|---|---|---|---|

| | | |
|---|---|---|
| EA | IE.7 | Disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit. |
| — | IE.6 | Not implemented, reserved for future use.* |
| ET2 | IE.5 | Enable or disable the Timer 2 overflow or capture interrupt (8052 only). |
| ES | IE.4 | Enable or disable the serial port interrupt. |
| ET1 | IE.3 | Enable or disable the Timer 1 overflow interrupt. |
| EX1 | IE.2 | Enable or disable External Interrupt 1. |
| ET0 | IE.1 | Enable or disable the Timer 0 overflow interrupt. |
| EX0 | IE.0 | Enable or disable External Interrupt 0. |

*User software should not write 1s to reserved bits. These bits may be used in future MCS-51 products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1.

**c) Write an ALP to get a byte of data from port 0.If it is greater than 99H ,send it to port 2.**

**Ans(Any correct program with comments—4 marks)**

| | |
|---|---|
| MOV A,P0 | Send a byte of data from port 0 to Accumulator |
| CJNE A,#99H,NEXT | Compare number stored in accumulator with 99H |
| NEXT: JC SKIP | If number >99H then go to skip |
| MOV P2,A | Else send to port 2 |
| SKIP : MOV P1,A | Send the number to P1 |
| HERE: SJMP HERE | |

**e)Draw the format of PCON SFR. How is it used to double the baud rate in serial communication.**

**Ans(format and explain –4 mark,)**

| SMOD | -- | -- | -- | GF1 | GF0 | PD | IDL |
|---|---|---|---|---|---|---|---|

**Fig: Format of PCON**

SMOD - Double Baud rate bit. If timer 1 is used to generate baud rate and SMOD = 1, the baud rate is doubles when the serial port is used in modes 1, 2, or 3.

--- - Not Implemented, Reserved for future use.

--- - Not Implemented, Reserved for future use.

--- - Not Implemented, Reserved for future use. GF1 - General Purpose Flag Bit

GF0 - General Purpose Flag Bit

PD - Power down Bit. Setting this bit activates Power Down operation in 8051. IDL - Idle Mode bit. Setting this bit activates Idle Mode operation in 8051.

**Q6) Attempt any four: Marks: 16**

**a) Write an ALP to generate continuous square wave of 2KHz frequency on pin P1.3,using mode 1 of Timer 0.Assume crystal freq= 11.0592MHz.**

**Ans: ( 1M- calculation, 2M- Program,1m- comments)**

Crystal frequency= 11.0592 MHz

I/P clock = $(11.059 \times 10^6)/12 = 1000000 = 921.58KHz$

$T_{in} = 1.085\mu$ sec

For 2 kHz square wave

$F_{out} = 2$ KHz

$T_{out} = 1/ 2X\ 10^3$

$T_{out} = 0.5msec = 500us$

Consider half of it = $T_{out} = 250\mu$ sec

$N = T_{out} / T_{in} = 250/1.085 = 230.41$

$65536-231 = (65305)_{10} = (FF19)_{16}$

Program:-

```
MOV TMOD, # 01H          ; Set timer 0 in Mode 1, i.e., 16 bit timer
L2:  MOV TLO, # 19H      ; Load TL register with LSB of count MOV  THO, # FFH
; load TH register with MSB of count
     SETB TR0            ; start timer 0
L1:  JNB TFO, L1         ; poll till timer roll over
     CLR TR0             ; stop timer 0
     CPL P1.3            ; complement port 1.4 line to get high or low
     CLR TF0             ; clear timer flag 0
     SJMP L2             ; re-load timer with count as mode 1 is not auto reload
```

**b)Draw and explain the format of IP register of 8051 mic.**

**Ans: (2M- format, 2M –explanation)**

| D7 | | | | | | | D0 |
|---|---|---|---|---|---|---|---|
| -- | -- | PT2 | PS | PT1 | PX1 | PT0 | PX0 |

Priority bit = 1 assigns high priority. Priority bit = 0 assigns low priority.

| | | |
|---|---|---|
| -- | IP.7 | Reserved |
| -- | IP.6 | Reserved |
| **PT2** | IP.5 | Timer 2 interrupt priority bit (8052 only) |
| **PS** | IP.4 | Serial port interrupt priority bit |
| **PT1** | IP.3 | Timer 1 interrupt priority bit |
| **PX1** | IP.2 | External interrupt 1 priority bit |
| **PT0** | IP.1 | Timer 0 interrupt priority bit |
| **PX0** | IP.0 | External interrupt 0 priority bit |

**c)Describe the function of following handshaking signals of 8255**

**i) IBF**

**II)STB**

**iii) OBF**

**iv) ACK**

**Ans . (Function of each signal: 1 Marks)**

**IBF:**

This is active high output signal generated by 8255 to peripheral. A high on this output indicates that data has been loaded into input latch.

$\overline{\text{STB}}$

This is active low input signal to 8255.A low on this input loads data into the input latch. This signal is generated by the peripheral to indicate that data is available on input ports lines.

**$\overline{\text{OBF}}$:**

The OBF output will go low to indicate that the CPU has written data to the specified port. This signal is generated by 8255 for output peripheral to indicate data is available and latched on the port lines.

$\overline{ACK}$

A low on this input informs the 8255 that the data from port A or port B is accepted by the output peripheral In essence peripheral device generates ACK signal indicating that data is accepted from the output port.

**d)Explain TCON register of 8051 with its format.**

**Ans(Format –1 mark,explain each bit—3 marks)**

| | | |
|---|---|---|
| TF1 | TCON. 7 | Timer 1 overflow flag. Set by hardware when the Timer/Counter 1 overflows Cleared by hardware as processor vectors to the interrupt service routine. |
| TR1 | TCON. 6 | Timer 1 run control bit. Set/cleared by software to turn Timer/Counter 1ON/OFF. |
| TF0 | TCON. 5 | Timer 0 overflow flag. Set by hardware when the Timer/Counter . 0 overflows Cleared by hardware as processor vectors to the service routine. |
| TR0 | TCON. 4 | Timer 0 run control bit. Set/cleared by software to turn Timer/Counter 0 ON/OFF. |
| IE1 | TCON. 3 | External Interrupt 1 edge flag. Set by hardware when External Interrupt edge is detected. Cleared by hardware when interrupt is processed |
| IT1 | TCON. 2 | Interrupt 1 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt. |
| IE0 | TCON. 1 | External Interrupt 0 edge flag. Set by hardware when External Interrupt edge detected. Cleared by hardware when interrupt is processed. |
| IT0 | TCON. 0 | Interrupt 0 type control bit. Set/cleared by software to specify falling edge/low level triggered External Interrupt. |

**e)Describe any four selection factors of microcontroller.**

**Ans(Any four factors ,each –1 mark)**

The selection of microcontroller depends upon the type of application. The following factors must be considered while selecting the microcontroller.

1. Word length: The word length of microcontroller is either 8, 16 or 32 bit. As the word length increases, the cost, power dissipation and speed of the microcontroller increases.
2. Power dissipation: It depends upon various factors like clock frequency, speed, supply voltage, VLSI technology etc. For battery operated embedded systems, we must use low power microcontrollers.
3. Clock frequency: The speed of an embedded system depends upon the clock frequency. The clock frequency depends upon the application.

4. Instruction Set: On the basis of instructions microcontrollers are classified into two categories 1. CISC 2. RISC.

   CISC system improves software flexibility. Hence it is used in general purpose systems.

   RISC improves speed of the system for the particular applications.
5. Internal resources: The internal resources are ROM, RAM, EEPROM, FLASH ROM, UART, TIMER, watch dog timer, PWM, ADC, DAC, network interface, wireless interface etc. It depends upon the application for which microcontroller is going to be used.
6. I/O capabilities: The number of I/O ports, size and characteristics of each I/O port, speed of operation of the I/O port, serial port or parallel ports. These are the considerations needed to ascertain