## *MODEL ANSWER*

**SUMMER– 17 EXAMINATION**

**Subject Title: Microcontroller** 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.

**1. a) Attempt any THREE of the following:**

**(i) Compare between microprocessor and microcontroller wrt.**

**1) Instruction set**

**2) Applications**

**3) Memory organization**

**4) I/O compatibility**

Ans :( Each parameter 1M)

| Sr. No | **Parameter** | **Microprocessor** | **Microcontroller** |
|---|---|---|---|
| 1. | No. of instructions used | Many instructions to read/ write data to/ from external memory. | Few instruction to read/ write data to/ from external memory |

| | | | |
|---|---|---|---|
| 2. | Applications | General purpose,<br>Computers and Personal Uses | Single purpose(dedicated application), Automobile companies, embedded systems, remote control devices |
| 3. | Memory | Do not have inbuilt RAM or ROM.<br>Program and data are stored in same memory. | Inbuilt RAM or ROM<br>Separate memory to store program and data |
| 5. | I/O compatibility | I/O ports are not available requires extra device like 8155 or 8255. | I/O ports are available hence I/O compatibility |

**ii) Describe the bus structure in microcomputer.**

Ans.(correct Answer 4 M)

A Bus is a set of physical connection used for communication between CPU and peripherals.
There are three types of buses Address Bus,Data Bus and Control Bus

**(1) Address Bus**

The address bus is unidirectional over which the microcontroller sends an address code to the memory or input/output. The size of the address bus is specified by the number of bits it can handle.The more bits there are in the address bus, the more memory locations a microcontroller can access. A 16-bit address bus is capable of addressing (64k) addresses.

**(2) Data Bus**

The data bus is bi—directional on which data or instruction codes are transferred into the microcontroller or on which the result of on operation or computation is sent out from the microcontroller to the memory or input/output.Depending on the particular microcontroller, the data bus can handle 8-bit or 16-bit data.

**(3) Control Bus** :

The control bus is used by the microcontroller to send out or receive timing and control signals like read and write in order to co- ordinate and regulate its operation and to communicate with other devices i.e. memory or input/output.

**iii) List and explain the important features of 8051 microcontroller. (ANY FOUR)**

Ans: **(any correct four points – 1M each)**

Features of 8051 micro controller are as follows:-

1) 8- bit data bus and 8- bit ALU.

2) 16- bit address bus – can access maximum 64KB of RAM and ROM.

3) On- chip RAM -128 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) Works on crystal frequency of 11.0592 MHz.

9) Has power saving and idle mode in microcontroller when no operation is performed.

10) Five interrupts with six sources are available: Two interrupts Timers i.e. Timer 0 and Timer 1, two external hardware interrupts- INT0 and INT1, Serial communication interrupt ( two sources for receive and transmit).

**iv) What are assembler directives? Give any three examples with use.**

Ans.(definition 1M,examples 1M each)

Directives are directions to the assembler, but they are not instructions to 8051.They are used to define symbols ,reserve memory space ,share values in program memory,select various memory spaces ,identify the end of the source.

Assembler directives are DB,EQU , ORG ,END

**i) 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.

**ii) 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).

**iii) ORG:- Origin**

It is used to indicate the beginning of address.

Syntax:

| OR | Address |
|---|---|

The address can be given in either hex or decimal there should be a space of at least one character between ORG & address fields. Some assemblers use ORG should not begin in label field.

**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.

**v) Differentiate between linear and absolute decoding techniques (any four points).**

Ans: (1M each)

| Absolute Decoding | Linear Decoding |
|---|---|
| 1. It is also called as full decoding as all the address lines are used for decoding . | It is also called as partial decoding as all address lines are not used for decoding |

| | |
|---|---|
| 2. It is used in large memory systems. | It is used in small systems |
| 3. Hardware required for decoding logic is more | Hardware used for decoding logic is less. |
| 4. Multiple addresses are not generated | Multiple addresses are generated |

**b) Attempt any ONE of the following:**

**i) Write an assembly language program for 8051 microcontroller for finding the largest number in a given set of 05 numbers. (Assume suitable data/memory addresses)**

Ans: (6M- correct program, stepwise marks can be given even for partial correct program)

Program can be written using internal/external memory address. Any suitable address or register can be assumed

**Program:**

```
MOV DPTR, # 3000H        ;   Initialize memory pointer
MOV R0,#00h                  Assume R0=00h is largest no
MOV R1, #05                  Initialize byte counter R0=5
UP: MOVX A, @DPTR            Load number in accumulator
CJNE A, R0, NEXT             Compare the no in A with Largest no in R0
NEXT: JC SKIP                If A<=R0 skip loading new no in R0
MOV R0, A                    Store new larger number in R0
SKIP: INC DPTR               Increment memory pointer by 1
DJNZ R1, UP                  Decrement byte counter and repeat if not 0
HERE: SJMP HERE              Stop the program, largest no is in r0
```

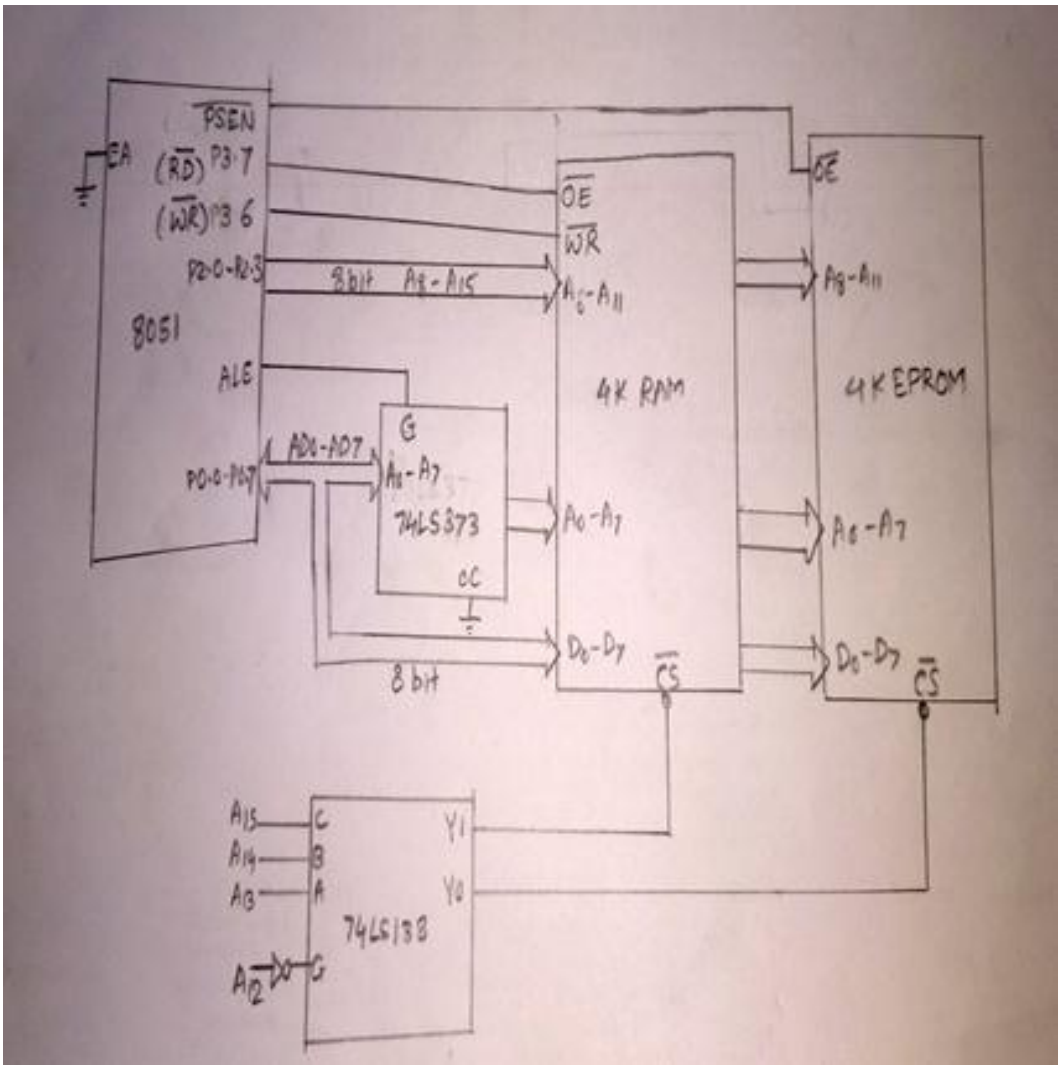
**Or**

```
MOV R0,#50h      ; Five nos are stored 59h onwards
MOV R1,#05       ;Counter for 5 nos
MOV R2,#00h      ;Assume 00 is the largest no
UP:  MOV A,@R0       ; Load the no in A
CJNE A, R2, NEXT ;Compare the no in A and R2
```

```
NEXT:   JC SKIP          ; if A<=R2 skip loading new no in R2
        MOV R2, A        ;if A>R2 load no in A into R2
SKIP:   INC R0           ; Increment pointer
        DJNZ R1, UP      ; Decrement counter and repeat the process till R1=0
HERE:   SJMP HERE        ; Stop here, the largest no is in R2
        END
```

**(ii) Draw a diagram to interface 4K byte EPROM and 4K byte RAM to 8051 microcontroller. Draw the memory map.**

Answer-(correct diagram 4M,memory map 2M)



| | A15 | A14 | A13 | A12 | A11 | A10 | A9 | A8 | A7 | A6 | A5 | A4 | A3 | A2 | A1 | A0 | HEX 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 RAM | **0** | **0** | **1** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **0** | **2000H** |
| End addr RAM | **0** | **0** | **1** | **0** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **1** | **2FFFH** |

**Starting address of RAM can be taken as 1000h and ending address 1FFFh but accordingly Decoder logic i.e. 74LS138 connections will change.**

**2. Attempt any FOUR of the following:**

**a) Defuse the program status word of 8051 microcontroller and state its functions with examples.**

A**ns: (2M-format, 2M-function)**

| CY | AC | F0 | RS1 | RS0 | OV | -- | P |
|---|---|---|---|---|---|---|---|

| | | |
|---|---|---|
| CY | PSW.7 | Carry Flag. |
| AC | PSW.6 | Auxiliary carry flag. |
| F0 | PSW.5 | Available to the user for general purpose. |
| RS1 | PSW.4 | Register bank selector bit 1. |
| RS0 | PSW.3 | Register bank selector bit 0. |
| OV | PSW.2 | Overflow flag. |
| -- | PSW.1 | User- definable bit. |
| P | PSW.0 | Parity flag. Set/cleared by hardware each instruction cycle to indicate and Odd/ even number of 1 bit in the accumulator. |

**1. CY: the carry flag.**

1. This flag is set whenever there is a carry out from the D7 bit.
2. The flag bit is affected after an 8 bit addition or subtraction.
3. It can also be set to 1 or 0 directly by an instruction such as ―SETB C and CLR C where SETB C stands for - set bit carry and CLR C for - clear carry.

**2. AC: the auxiliary carry flag**

 If there is a carry from D3 and D4 during an ADD or SUB operation, this bit is set; it is cleared. This flag is used by instructions that perform BCD (binary coded decimal) arithmetic.

**3. F0: Available to the user for general purposes.**

**4. RS0, RS1: register bank selects bits**

1. These two bits are used to select one of the four register banks n internal RAM in the table. By writing zeroes and ones to these bits, a group of registers R0- R7 can be used out of four registers banks in internal RAM.

| RS1 | RS0 | Space in RAM |
|---|---|---|
| 0 | 0 | Bank 0 (00H- 07H) |
| 0 | 1 | Bank 1 (08H-0FH) |
| 1 | 0 | Bank2 (10H-17H) |
| 1 | 1 | Bank3 (18H-1FH) |

**5. 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.

**6. P: Parity flag**

1. The parity flag reflects the number of 1s in the A (accumulator) register only. If the A register contains an odd number of 1's, then P=1, P=0 if A has an even number of 1's.

**b) List the pins used for accessing the external memory/devices and demultiplexing the lower address/data bus along with their description.**

Ans(list 1M,description 3M)

The pins used for accessing the external memory/devices and demultiplexing the lower address/data bus are

/PSEN

/EA

ALE

/RD(P3.7)

/WR(P3.6)

**Function of /PSEN:**

1. /PSEN stands for "program store enable." The read strobe for external Program Memory is the signal /PSEN (Program Store Enable). In an 8031-based system in which an external ROM holds the program code, this pin is connected to the /OE pin of the ROM.
2. In other words, to access external ROM containing program code, the 8031/51 uses the /PSEN signal. This read strobe is used for all external program fetches. /PSEN is not activated for internal fetches.

**Function of /EA:**

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 8K in case of 8052) 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 0FFFH 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 ROM less parts must have this pin externally strapped to VSS to enable them to execute properly.

**Function of ALE:**

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 de-multiplexing the address and data by connecting to the STB pin of the 74LS373 chip.

**Function of /RD:**
External data memory read strobe

**Function of /WR** :
External data memory write strobe.

**c) The bit addressable feature in 8051 microcontroller makes it more powerful than microprocessor. Justify your answer.!?**

**Answer (Correct justification-4M)**

The 8051 instruction set is optimized for the one bit operations so often desired in real – world, real- time control applications.

1. The Boolean processor is an internal part of 8051 architecture. It is an independent bit processor with its own accumulator and its own bit addressable RAM and I/O.

2. The Boolean processor provides direct support for bit manipulation .This leads to more efficient programs that needs to deal with binary input and output conditions inherent in digital control problems.
3. Bit addressing can be used for test pin monitoring or program control flags. For example, instructions for Boolean function are as given below.

1. SETB P1.0 ; Set P1.0
2. JB P1.0,NEXT ; Jump to label if P1.0 is set
3. ANL C, P1.4 ; AND the bit on P1.4 with carry.

Any other bit addressable instructions minimum 4 must be explained

**d) List the register banks used for in 8051 microcontroller. Give their ranges.**

Answer(List 1M, ranges 3M)

There are four register banks from 00H to 1FH. On power-up, registers R0 to R7 are located at 00H to 07H.

Bank 0 00h to 07H

Bank 1 08h to 0FH

Bank 2 10H to 17H

Bank 3 18H to 1FH

**e) State and describe the alternate functions of port 3 pins of 8051.**

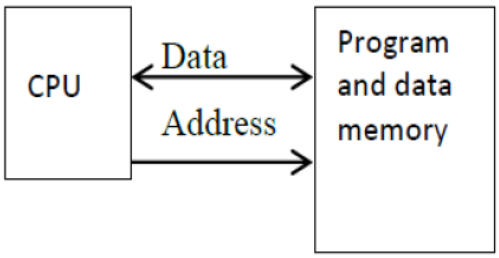
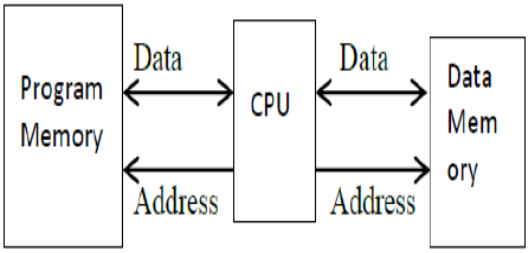
Ans. Each function ½ M)

| **Pin** | **Name** | **Alternate Function** |
|---|---|---|
| P3.0 | RXD | Serial input line |
| P3.1 | TXD | Serial output line |
| P3.2 | $\overline{INT0}$ | External interrupt 0 |
| P3.3 | $\overline{INT1}$ | External interrupt 1 |

| | | |
|---|---|---|
| P3.4 | T0 | Timer 0 external input |
| P3.5 | T1 | Timer 0 external input |
| P3.6 | $\overline{WR}$ | External data memory write strobe |
| P3.7 | $\overline{RD}$ | External data memory read strobe |

**f) Compare Harvard and Von-Neumann architecture. (4 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. |

**Q3. Attempt any FOUR of the following:**

**a) State the function of compiler, linker, assembler and editor in S/W development.**
**Ans: (1mark for each )**
**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 .c depending on which assembler is used.

**2) Assembler:** An assembler is used to find the syntax error in the programs. It 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, Keil‟s A51, AX 51 and C51, Intel PL/M-51 etc.

**3) Compiler:** It is used to find the syntax error**.** It is used to convert C language program to object or list file..

**4) Linker:** A linker is a program, which combines, if requested , more than one separately assembled object files into one executable program, such as two or more programs and also generate .abs file and initializes it with special instructions to facilitate its subsequent loading the execution. Some examples of linker are ASEM-51 BL51, Keil u Vision Debugger, LX 51 Enhanced Linker etc.

**b) Describe the function of following instructions of 8051 microcontroller:**

**i. XCH A,R1**
**ii. RRA**
**iii. MOV A,#40H**
**iv. SWAP A**

**Ans: (1mark for each with suitable example)**

**1. XCHG A,R1**

**Description**: This instruction exchanges byte of the Accumulator with byte of the register(R1) of RAM.

**Example:**
```
MOV  A, #59H        ;A= 59H
MOV R1, #30H        ;R1= 30H
XCHG  A, R1         ; A= 30H and R1= 59H
```

**ii. RRA**

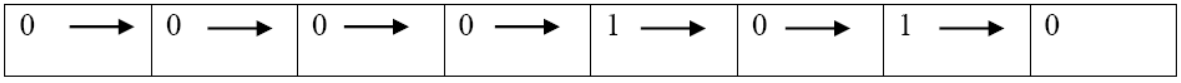
Rotate accumulator right.

It stands for rotate accumulator right. The contents of the Accumulator are rotated right bit-wise.

Example:

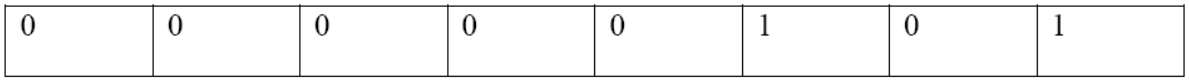
Before execution

A= 0AH




**After Execution**

**A=05H**




**1. MOV A, #40H**

This instruction moves or stores the data 40H in the Accumulator. The # symbols tells that 40 is data not address.

**Example:** MOV A, #50H ;A= 50H

**iv. SWAP A**

**Description**: This instruction exchanges the nibbles (bits 0-3 with bits 4-7) of the Accumulator. This instruction is identical to executing "RR A" or "RL A" four times.

```
Example:  MOV  A, #59H      ;A= 59H
          SWAP A            ; A= 95H
```

**c) Write an ALP for 8051 microcontroller to multiply two 8-bit numbers 23H and 15H.(Assume suitable memory address to store the result).**

**Ans: (4 marks for correct program, stepwise marks can be given for partially correct program, Any suitable address or data can be assumed)**

**Program:**

```
MOV 50H, # 23H          ; store first 8-bit no. in 50H
        MOV 51H, #15H           ; store second 8-bit no. in 51H
        MOV A, 50H              ; move first number to A
        MOV B, 51H              ; move second number to B
        MUL AB                  ; multiply the numbers
        MOV 52H, A              ; move LSB to 52H
        MOV 53H, B              ; move MSB to 53H
HERE: SJMP HERE
```

**d) List the addressing modes of 8051 microcontroller. Describe the indexed addressing mode with an example.**

**Ans: (2M-list , 2M-description )**

1. Immediate Addressing mode
2. Register addressing mode
3. Direct Addressing mode
4. Indirect addressing mode
5. Indexed Addressing mode

**Indexed Addressing Mode:**

With indexed addressing a separate register, either the program counter, PC, or the data pointer DTPR, is used as a base address and the accumulator is used as an offset address. The effective address is formed by adding the value from the base address to the value from the offset address. Indexed addressing in the 8051 is used with the JMP or MOVC instructions. Look up tables are easy to implement with the help of index addressing.

Consider the example instruction:

MOVC A, @A+DPTR

MOVC is a move instruction, which moves data from the external code memory space. The address operand in this example is formed by adding the content of the DPTR register to the accumulator value. Here the DPTR value is referred to as the base address and the accumulator value us referred to as the index address.

**e) List and describe the modes of communication used in 8051.**

**Ans: (1M each mode)**

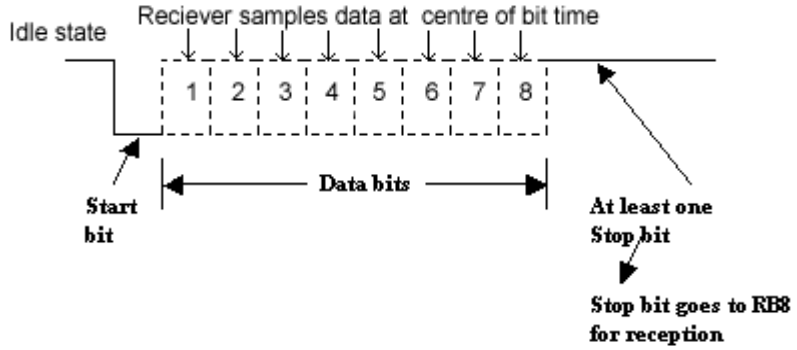
8051 micro controller communicate with another peripheral device through RXD and TXD pin of port3.controller have four mode of serial communication.

**1. Mode-0: Shift register mode (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 fosc /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 fosc /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. Mode-1: 8-bit UART (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. Mode-2: 9-bit UART (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.

**f baud = (2 SMOD /64) *fosc**

**4. Mode-3: 9-bit UART (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).

**Baud rate is same as mode 1.**

**Q.4. a) Attempt any THREE of the following:**

**i) Describe the following 8051 microcontroller instructions:**

**1. MOVX A,@ DPTR**
**2. DIV AB**
**3. SETB C**
**4. SJMP addr**

**Ans: (1M each)**

**1. MOVX A,@ DPTR**

**Description**: 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)
                                              ;Assume 2000[0Bh]
MOVX A, @DPTR                  ; 2000H [ 0BH]
                                              ;A = 0BH
```

**2. DIV AB**

**Description:** Divides the unsigned value of the Accumulator by **t**he unsigned value of the "B" register. The quotient is placed in the Accumulator and the remainder is placed in the "B" register.

```
Example:    MOV A, #0AH
            MOV B, # 05H
            DIV AB          ; A= 02(decimal) =02H, B=00
```

**3. SETB C**

**Function:** This sets high the carry bit.
**Flags:** Carry flag is affected.

**Eg.** SETB C
After execution
Set carry flag CY=1

**4. SJMP addr**

SJMP addr: Short address

$PC_{new} \Leftarrow PC_{old}$ +addr

The program control jumps max 128 bytes backwards or max 127 bytes forward with respect to the current PC. Next instruction is executed from new PC.

```
SJMP NEXT
-
-
-
NEXT: MOV A,#30h; Label should be within the range of +127 to -128
```

**ii) State the use of SCON and SBUF register of 8051.**

**Ans: (2M each mode)**

**SCON :** : It is an 8 bit control register used in serial communication of 8051.Its format is as shown below with functions of each bit.

| SM0 | SM1 | SM2 | REN | TB8 | RB8 | TI | RI |
|---|---|---|---|---|---|---|---|

SM0 SCON.7 Serial port mode specifier
SM1 SCON.6 Serial port mode specifier.
SM2 SCON.5 Used for multiprocessor communication (Make it 0.)
REN SCON.4 Set/ cleared by software to enable/ disable reception.
TB8 SCON.3 $9^{th}$ bit to be transmitted in mode 2 and 3.
RB8 SCON.2 9th received bit in mode 2 and 3
TI SCON.1 Transmit interrupt flag. Set by hardware at the beginning of the stop Bit in mode 1.Must be cleared by software.
RI SCON.0 Receive interrupt flag. Set by hardware halfway through the stop bit time in mode 1.Must be cleared by software.

**SBUF**: It is an 8 bit register used in serial communication of 8051.Serial data sending is done by writing to the register SBUF while data receiving is done by reading the same register. The SBUF is as shown below:

| Bit 7 | | | | | | | Bit 0 |
|---|---|---|---|---|---|---|---|

SBUF has physically two registers, one write only and other is read only. Both registers use one address 99H

**iii) Write an assembly language program to send message "HELLO" serially at 4800 baud rate continuously(crystal frequency = 11.0592MHz)**
**Ans:(3M-prog,1M-comments)**

```
MOV TMOD, #20H                  ; timer 1, mode2
    MOV TH1,#-6 or MOV TH1,#0FAh        ; 4800 baud rate
    MOV SCON, #50H              ; 8-bit data,1 stop bit, REN enabled
    SETB TR1                    ; Start timer 1
  AGAIN: MOV A, #"H"                    ; transfer "H"
    ACALL MESSAGE               ; Some delay
        MOV A, #"E"                     ; transfer "E"
    ACALL MESSAGE
MOV A, #"L"                 ; transfer "L"
    ACALL MESSAGE
        MOV A, #"L"                     ; transfer "L"
    ACALL MESSAGE
        MOV A, #"O"                     ; transfer "O"
    ACALL MESSAGE
        SJMP AGAIN

 MESSAGE: MOV SBUF, A;
```

JNB TI, HERE;
CLR TI;
RET

**iv)Write an assembly language program to toggle all the bits of port2 of 8051 continuously.**
**Ans:(4M-for correct prog, stepwise marks can be given for partially correct program )**

**Program**

```
        MOV A, #0FFH            ; Store FFH in A
BACK: MOV P2, A             ; move FF to P2 to set all the bits
        ACALL DELAY             : wait for some time
        CPL A                   ; make the bits 0
        SJMP BACK
```

**Delay Program:**

```
DELAY: MOV R3, #255
        DJNZ R3, DELAY
        RET
```

**Q4. b) Attempt any ONE of the following:**
**i) Wrte an ALP to arrange the given data in ascending order in 8051 microcontroller:**

**Data: 40H =09H**
**41H = 07H**
**42H = 12H**
**43H = 25H**
**44H = 01H**

**Store the result in 50H to 54H.**

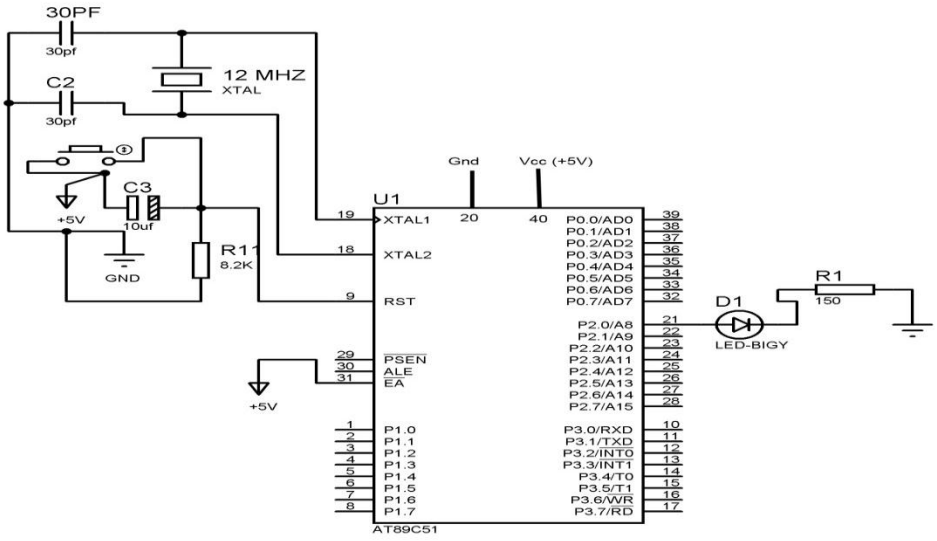
**Ans: (6M- for correct prog, stepwise marks can be given for partially correct program)**
**Any registers can be considered for counter but only R0 and R1 for pointers.**

```
      MOV R4,#04H       ; initialize pass counter i.e maximum no (5)-1
UP1:  MOV R3,#04H       ; initialize byte counter i.e. maximum no (5)-1 as one no is
                        ;compared with four other nos
      MOV R0,#40H       ; initialize memory pointer
UP:   MOV A,@R0         ; read first number
      MOV R1,A          ; copy to R1 register first no in R1
      INC R0            ;point to next no
      MOV A,@R0         ;next no in A
      CJNE A,R1,NEXT    ; compare the first number with the second 2nd no – 1st no
NEXT: JNC SKIP          ; if number 2nd no > 1st no next number go to skip else exchange it
      MOV @R0,R1        ; 1st no at next position
      DEC R0            ; point to previous position
      MOV @R0,A         ; 2nd no at previous position
```

```
        INC R0          ; point to next no
SKIP:   DJNZ R3,UP      ; decrement byte counter if not zero go to UP
        DJNZ R4,UP1     ; Repeat this process total nos-1 times
        MOV R0,#40H     ; Load source memory pointer
        MOV R1,#50h     ; Load destination memory pointer
        MOV R5,#05H     ; initialize the counter register for 5 nos
AGAIN:  MOV A,@R0       ; copy the number into accumulator
        MOV @R1,A       ; copy the number into memory pointer 50H
        INC R0          ; increment memory pointer
        INC R1          ; increment memory pointer
        DJNZ R5,AGAIN   ; decrement counter register if not zero go to AGAIN
        SJMP $          ; stop
```

**ii) Draw the interfacing diagram of 8 LED's to port 2 of 8051 microcontroller. Write an ALP to make LED ON and OFF after 100ms delay.**

**Ans: (2M-diagram, 4M- for correct program) Assume crystal freq=12MHz, as required delay is given we must use timer for delay.**




**(Note: LEDs should be connected to all the 8 lines of Port 2)**

**Program**

```
        MOV A, #0FFH            ; Store FFH in A
START:MOV P2, A                 ; move FF to P2 to glow all the LEDs
        ACALL DELAY             : wait for some time
        CPL A                   ; turn off the LEDs
 ACALL DELAY                    : wait for some time
        SJMP START
```

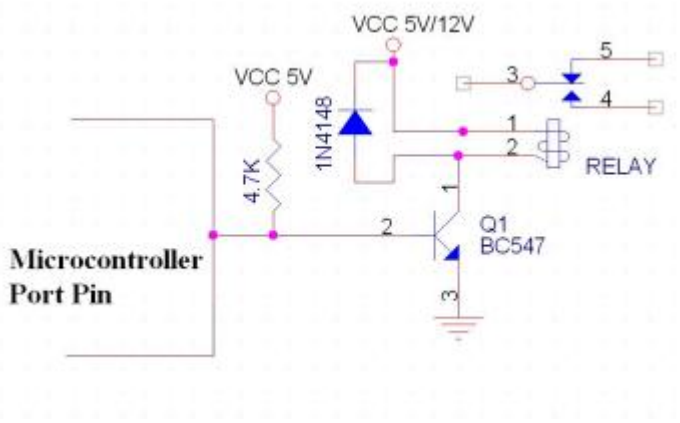
SJMP $

**Delay Program:**

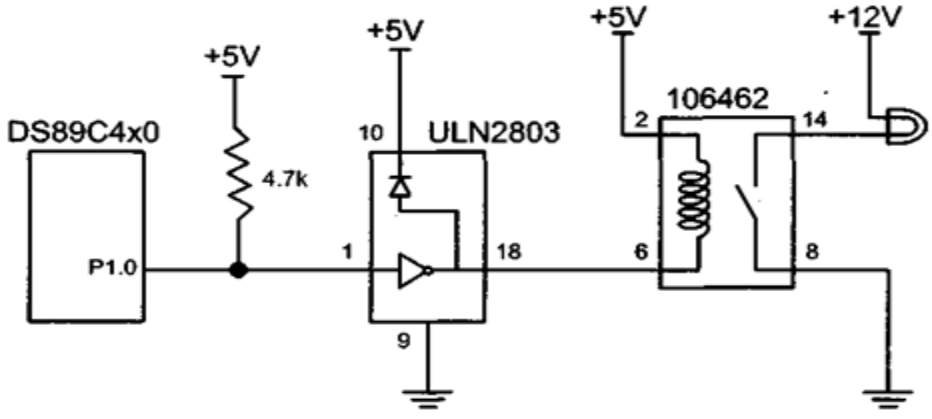
```
DELAY:   MOV TMOD, #10H        ;Timer1, mode 1
HERE:    MOV R0,#02H           ;Counter for 100ms (50*2)delay
BACK:    MOV TL1, # B0H        ; load count value in TL1
         MOV TH1, #3CH         ; load count value in TH1
         SETB TR1              ; start Timer 1
AGAIN:   JNB TF1, AGAIN        ; stay until timer rolls over
         CLR TR1               ; stop timer
CLR TF1                        ; clear timer flag
DJNZ R0, BACK                  ; if R0 is not equal to 0, reload timer
RET
```

**iii) Draw interfacing diagram of relay with 8051 microcontroller. Write an ALP to turn ON and OFF relay.**

**Ans: (3M-diagram, 3M- for correct program)**




**OR**



**Program:**

```
        ORG 0H
MAIN:
        SETB P1.0
        MOV R5, #55
        ACALL DELAY
        CLR P1.0
        MOV R5, #55
        ACALL DELAY
        SJMP MAIN
DELAY:
H1:     MOV R4,#100
H2:     MOV R3,#253
H3:     DJNZ R3, H3
        DJNZ R4, H2
        DJNZ R5, H1
        RET
        END
```

Or

Program

```
AGAIN:  SETB P1.0
        ACALL DELAY         ; call delay
        CLR P1.0
        ACALL DELAY         ; call delay
        SJMP AGAIN
        SJMP $

DELAY:  MOV R2,#250
XX:     DJNZ R2, XX
        RET
        END
```

**5. Attempt any FOUR of the following** **(16)**

**a) Draw the format of IP register of 8051 microcontroller. Describe the function of each bit in it.**

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 |

**b) Draw the format of TMOD register of uc 8051 and describe function of each bit in it.**

**Ans:- ( Format- 2 mks, function of each bit-2 mks)**

**TMOD: TIMER/COUNTER MODE CONTROL REGISTER. NOT BIT ADDRESSABLE.**

| GATE | C/$\overline{T}$ | M1 | M0 | GATE | C/$\overline{T}$ | M1 | M0 |
|---|---|---|---|---|---|---|---|
| TIMER 1 | | | | TIMER 0 | | | |

| | |
|---|---|
| GATE | When TRx (in TCON) is set and GATE = 1, TIMER/COUNTERx will run only while INTx pin is high (hardware control). When GATE = 0, TIMER/COUNTERx will run only while TRx = 1 (software control). |
| C/$\overline{T}$ | Timer or Counter selector. Cleared for Timer operation (input from internal system clock). Set for Counter operation (input from Tx input pin). |
| M1 | Mode selector bit. (NOTE 1) |
| M0 | Mode selector bit. (NOTE 1) |

**NOTE 1:**

| M1 | M0 | Operating Mode | |
|---|---|---|---|
| 0 | 0 | 0 | 13-bit Timer (MCS-48 compatible) |
| 0 | 1 | 1 | 16-bit Timer/Counter |
| 1 | 0 | 2 | 8-bit Auto-Reload Timer/Counter |
| 1 | 1 | 3 | (Timer 0) TL0 is an 8-bit Timer/Counter controlled by the standard Timer 0 control bits, TH0 is an 8-bit Timer and is controlled by Timer 1 control bits. |
| 1 | 1 | 3 | (Timer 1) Timer/Counter 1 stopped. |

**c) Describe the interrupts in 8051 microcontroller with their priorities.**

Ans:- (3 mks for Diagram and description and 1 mark for priority)

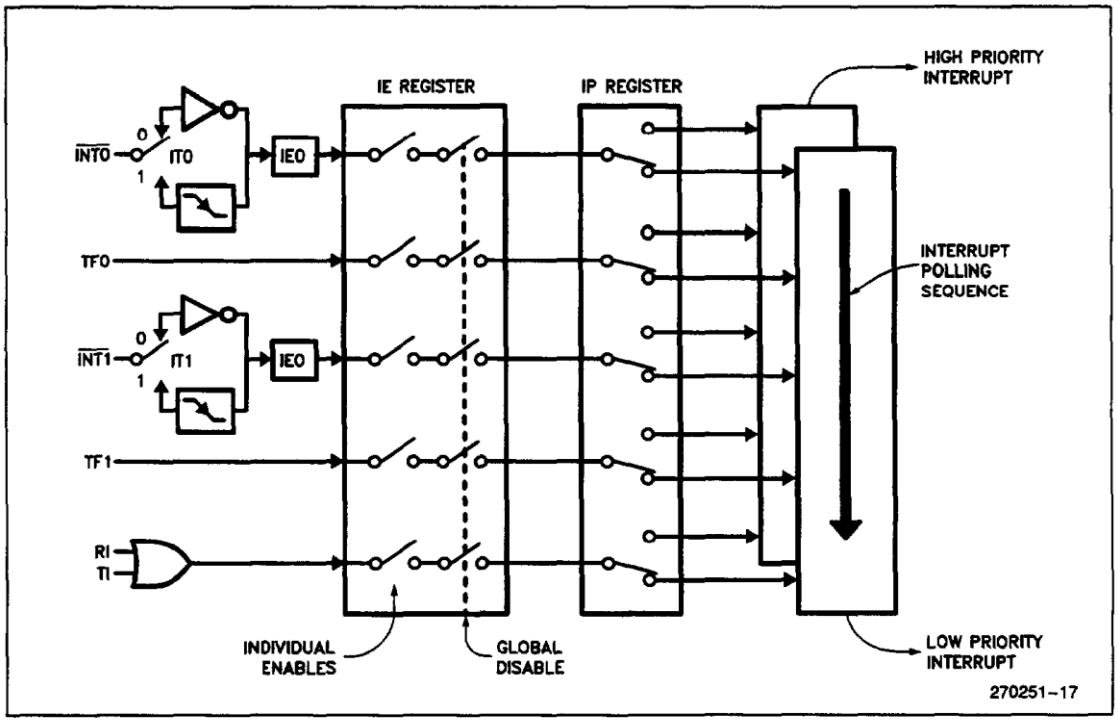



**Figure 19. 8051 Interrupt Control System**

| 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 |

All the 5 interrupts of 8051 has got different priorities. Interrupts are serviced according to it's priority order. From the table above, you can see that INT0 has the highest priority of 1 and Timer 0 comes next with priority value 2. The order of priority works like this – consider a case where two interrupts are raised at the same time – one from INT0 and another from Timer 1 interrupt. In such a case, processor would serve the interrupt according to it's priority. In our case INT0 is of high priority (priority order 1)and Timer 1 interrupt is of low priority (priority order 4). So processor will execute ISR of INTO first and then later, after finishing ISR of INT0, processor will begin executing ISR of Timer 1 interrupt.

**d) Write an ALP for microcontroller to generate square wave on port pin p2.1 using delay subroutine.**

Ans:- (Proper relevant program – 4 mks any delay can be considered as frequency of square wave is not given)

```
ORG         0000
MOV         TMOD,# 01H          ; Mode 1
HERE :      MOV     TL0,# 0F2H      ; Lower byte of timer 0
            MOVE    TH0, # 0FF      ;Higher byte of timer 0
            CPL     P2.1            ; toggle P 2.1
            ACALL   DELAY
            SJMP    HERE
; delay using timer 0
DELAY :     SETB    TR0             ; Start time 0
AGAIN :     JNB     TF0, AGAIN
            CLR     TR0             ; Stop timer 0
            CLR     TF0
            RET
```

Or

```
            ORG 0000h
            SJMP START
            ORG 0030h
START:      CPL P2.1
            ACALL DELAY
            SJMP START
            SJMP $

DELAY:      MOV R2,#250
XX:         DJNZ R2,XX
            RET
END
```

---

**e) Describe the alternative functions of port'0' and port2 in 8051.**

Ans:- ( alternate 2 functions - 2 mks each)

Port 0: It can be used as

a) Simple input/output

b) Bidirectional low order address / data bus for external memory.

Port 2: It can be used as

a) Simple input/output port

b) the alternative use is to supply a higher order address byte in conjunction with the port 0 lower order byte to address external memory.

**6. Attempt any FOUR of the following (16)**

**a) Draw the format of IE register of µc 8051 and describe function of each bit in 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 |
|---|---|---|---|---|---|---|---|

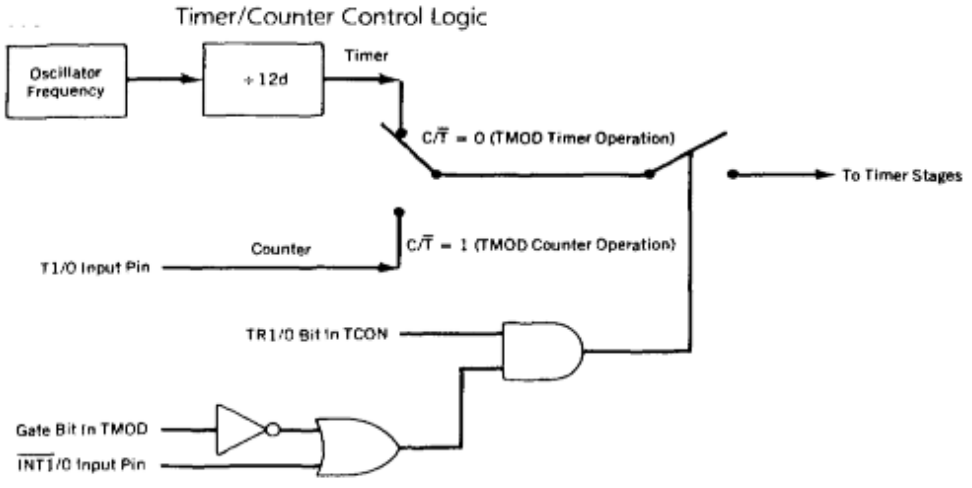
| | | |
|---|---|---|
| EA | IE.7 | Disables all interrupts. If EA = 0, no interrupt will be acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit. |
| — | IE.6 | Not implemented, reserved for future use.* |
| ET2 | IE.5 | Enable or disable 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.

**b) Differentiate between the uses of 8051 as timer and counter.**

Ans:- ( Relevant any 4 points – 1 mks each)

| SR.NO. | 8051 USE AS TIMER | 8051 USE AS COUNTER |
|---|---|---|
| 1. | It is used for delay implementation | It is used to measure external event such as pulse width measurement |
| 2. | External pin T0/T1 is not used | External pin T0/T1 is monitored |
| 3. | Counter is incremented for $1/12^{th}$ of Fosc | Counter is incremented for every pulse on T0 or T1 pin, irrespective of it's frequency. |
| 4. | THx and TLx is loaded as per required Delay | THx and TLx must be initially cleared for counter |
| 5. | C/T=0 | C/T=1 |




( diagram is optional)

**c) Write an ALP for 8051 microcontroller to generate a delay of 500 ms by using timer 1.**

**Assume crystal frequency = 12 MHz.**

ANS: (1 mark for calculation and 3 marks for program)

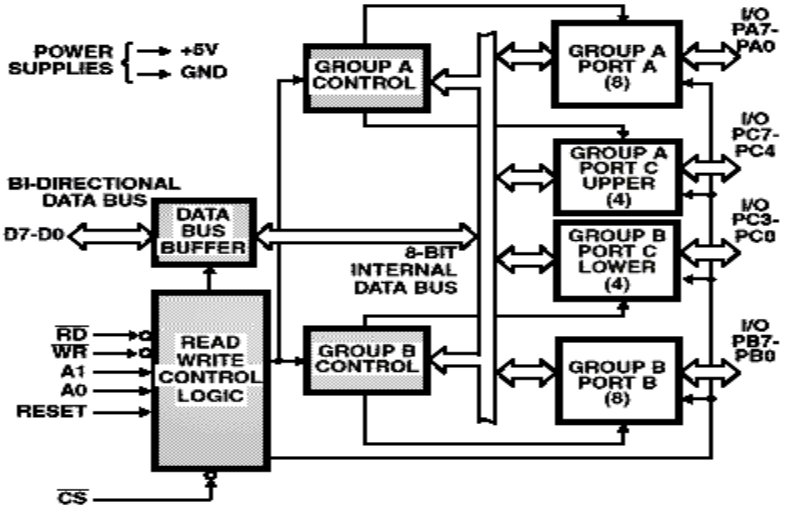
Crystal freq=12MHz
Timer frequency=12MHz/12=1MHz
Time=1/1MHz=1µs
For delay of 50 ms,
50ms/1µs=50000
Therefore count to be loaded in TH1 and TL1 can be calculated as 65536-50000=15535D=3CB0H

PROGRAM

```
DELAY:   MOV TMOD, #10H          ;Timer1, mode 1
HERE:    MOV R0,#0AH             ;Counter for 500ms (50*10)delay
BACK:    MOV TL1, # B0H              ; load count value in TL1
         MOV TH1, #3CH               ; load count value in TH1
         SETB TR1                    ; start Timer 1
AGAIN:   JNB TF1, AGAIN              ; stay until timer rolls over
         CLR TR1                     ; stop timer
CLR TF1                      ; clear timer flag
DJNZ R0, BACK                ; if R0 is not equal to 0, reload timer
RET                          ; repeat
```

**d) Draw the block diagram of IC 8255 and describe its operating mode.**

Ans:- ( Block diagram- 3 mks, operating mode – 1 mks)




Operating mode-

1) **BSR MODE**: It is used to set or reset a particular bit of port C. It affects on only one bit of port C. To set or reset a selected bit of port C, the selected port C line should be initialized in output mode prior to this command.

2) **I/O mode**: I/O command word is used to set operating modes of port A, B and C. It is also used to set direction of port A,B and C.

**e) Give any four addressing modes of µc 8051 with examples of each.**

Ans:- (Any four modes with correct example -4mks, i.e. 1 mark each along with one example)

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.

5. Index Addressing mode: In this mode the base address is given through DPTR or PC and offset is given through A.

MOVC A,@A*DPTR

( any four addressing modes can be explained)