MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER-15 EXAMINATION
Model Answer

Subject Code: 17626 **Subject Name: Embedded System**

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in themodel 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 judgment 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: **Marks 12**

(i) Draw the format of PSW. Explain the function of each bit.

(PSW format- 2 Marks,Explanation - 2 Marks)

Ans:

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CY	AC	FO	RS1	RS0	OV	UD	P

Explanation :

F0: Flag 0 (Available to the user for General Purpose)

UD : User definable flag

RS0 and RS1 : resister bank selection

RS1	RS0	Working Register Bank and Address
0	0	Bank0 (D:0x00 - D:0x07)
0	1	Bank1 (D:0x08 - D:0x0F)
1	0	Bank2 (D:0x10 - D:0x17)
1	1	Bank3 (D:0x18H - D:0x1F)

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. 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".

AC, the auxiliary carry flag

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

P, the parity flag

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

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.

(ii) List various branching instructions in microcontroller 8051.

(List - 4 Marks)

Ans:

Mnemonic	Description
ACALL addr11	Absolute subroutine call
LCALL addr16	Long subroutine call
RET	Return from subroutine
RETI	Return from interrupt
AJMP addr11	Absolute jump
LJMP addr16	Long jump
SJMP rel	Short jump
JMP @A+DPTR	Jump indirect
JZ rel	Jump if A=0
JNZ rel	Jump if A NOT=0
CJNE A,direct,rel	Compare and Jump if Not Equal
CJNE A,#data,rel	
CJNE Rn,#data,rel	
CJNE @Ri,#data,rel	
DJNZ Rn,rel	Decrement and Jump if Not Zero
DJNZ direct,rel	
NOP	No Operation

(iii)State various software tools available in IDE. Explain any one in brief.

(State software tools- 2 Marks , Any one Explanation - 2 Marks)

Ans:

Software tools

- Compiler
- Cross assembler
- Cross compiler
- Locators

- Loaders
- Simulators
- Debugger
- Integrated development environment (IDE)

Explanation (any one of the following)

Compiler:

It is a computer program that transforms the source code written in a programming or source language into another computer language i.e. target language i.e. binary code known as object code.

Cross assembler:

It is useful to convert object codes for microcontrollers or processor to other codes for another microcontrollers or processor and vice versa.

Cross compiler:

It is used to create executable code other than one on which the compiler is run. They are used to generate executables for embedded systems or multiple platforms.

Linker/Locator:

It is used for relocation process .

It is done during compilation also it can be done at run time by a relocating loader.

It is a program that takes one or more objects generated by compiler and combines them into a single executable program.

Simulators:

A simulator is the s/w that simulates a h/w unit like emulator, peripheral, network and I/O devices on a PC

- It defines a processor or processing device as well as various versions for the target system
- Monitors the detailed information of as source code part with labels and symbols during the execution for each single step.
- Provides the detailed information of the status of memory RAM and simulated ports, simulated peripheral devices of the defined target system

Integrated Development Environment (IDE) :-

- It supports for defining a processor family and its version
- Support a user definable assembler to support a new version or a type of processor.
- Provides multiuser environment
- Supports conditional and unconditional break points
- Provide debugger.

(iv) What is deadlock? How can it be prevented?

(Dead lock - 2 Marks, Prevention - 2 Marks)

Ans:

Deadlock:

A deadlock, also called as deadly embrace, is a situation in which two threads are each unknowingly waiting for resource held by other.

- Assume thread T1 has exclusive access to resource R1.
- Thread T2 has exclusive access to resource R2.
- If T1 needs exclusive access to R2 and T2 needs exclusive access to R1,
- Neither thread can continue.

o They are deadlocked.

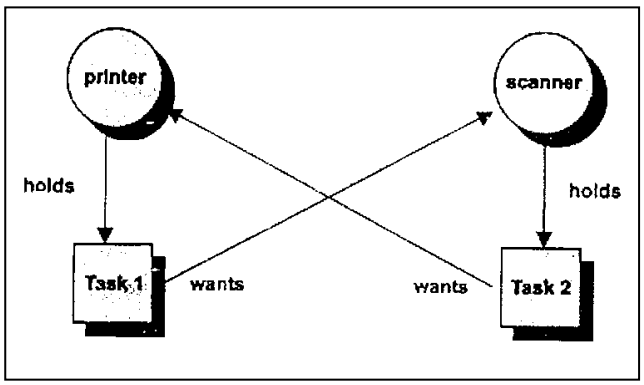
o The simplest way to avoid a deadlock is for threads to:

- Acquire all resources before proceeding
- Acquire the resources in the same order
- Release the resource in the revere order

How to Prevent deadlock:

- Deadlock is the situation in which multiple concurrent threads of execution in a system are blocked permanently because of resources requirement that can never be satisfied.
- A typical real-time system has multiple types of resources and multiple concurrent threads of execution contending for these resources. Each thread of execution can acquire multiple resources of various types throughout its lifetime.
- Potential for deadlock exist in a system in which the underlying RTOS permits resources sharing among multiple threads of execution.

Following is a deadlock situation between two tasks.



- In this example, task #1 wants the scanner while holding the printer. Task #1 cannot proceed until both the printer and the scanner are in its possession.
- Task #2 wants the printer while holding the scanner. Task #2 cannot continue until it has the printer and the scanner.

- Because neither task #1 nor task#2 is willing to give up what it already has, the two tasks are now deadlocked because neither can continue.

b) Attempt any ONE of the following: Marks 06

(i) Write 8051 program in assembly or C to send message 'INDIA' serially. Assume baud rate as 9600 and crystal frequency as 11.0952 MHz.

(Counter calculations- 2 Marks, Correct program- 4 Marks)

Ans:

Calculation for baud rate 9600.

- Timer clock Frequency is = XTAL / 12
  = 11.0592MHz / 12
  = 921.6 KHz
- UART Frequency is = Timer clock Frequency / 32
  = 921.6KHz / 32
  = 28.8 KHz
- Baud rate = UART Frequency / COUNTER Value
- COUNTER Value = UART Frequency / Baud Rate
  = 28.8 KHz / 9600
  = 3

As timer in microcontroller is up counter / timer, so counter value is = -3 or FDh

Program :

```
                ORG    0000h
                MOV    TMOD,#20h        ;timer 1, mode 2
                MOV    TH1,#-3          ;9600 baud
                MOV    SCON,#50h        ;8-bit, 1 stop bit, REN enabled
                SETB   TR1              ;start timer 1
AGAIN:          MOV    A,#'I'           ;transfer 'I'
                ACALL  TRANS
                MOV    A,#'N'           ;transfer 'N'
                ACALL  TRANS
                MOV    A, #'D'          ;transfer 'D'
                ACALL  TRANS
                MOV    A,#'I'           ;transfer 'I'
                ACALL  TRANS
                MOV    A,#'A'           ;transfer 'A'
                ACALL  TRANS
                SJMP   AGAIN            ;keep doing it
;--Serial data transfer subroutine
TRANS:          MOV    SBUF,A           ;load SBUF
HERE:           JNB    TI,HERE          ;wait for last bit to transfer
                CLR    TI
                RET
                END
```

OR

```
#include <reg51.h>
void SerTx(unsignedchar);
void main(void)
{
        TMOD = 0x20;
        TH1 = 0xFD;
        SCON = 0x50;
        TR1 = 1;
        SerTx('I');
        SerTx('N');
        SerTx('D');
        SerTx('I');
        SerTx('A');
}
void SerTx(unsignedchar x)
{
        SBUF = x;
        while(TI==0);
        TI = 0;
}
```

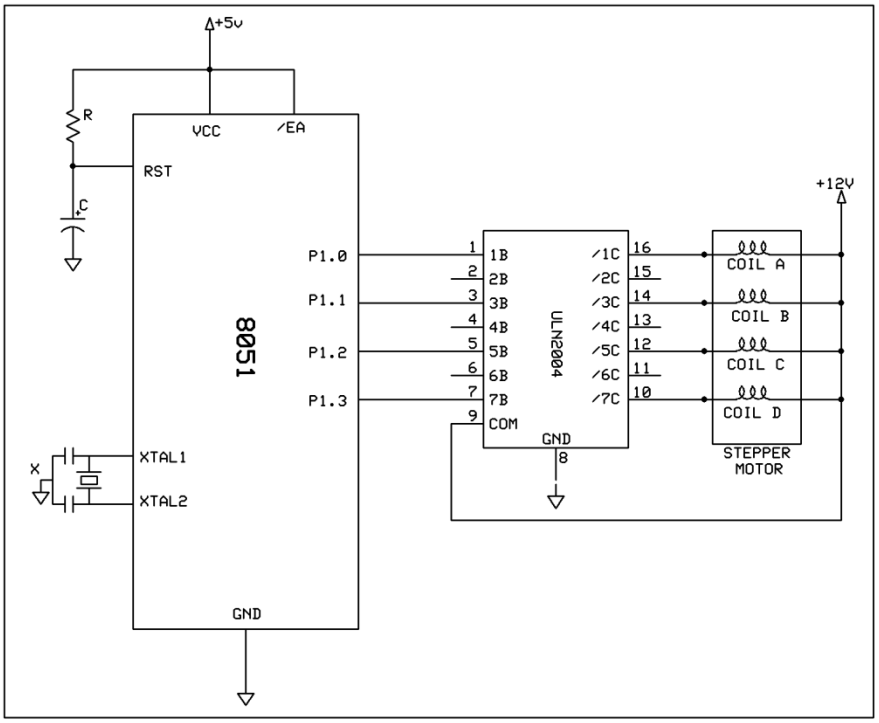
[NOTE: Program may change but Calculations will not change.

Student can also use the other logic. Please check the logic and understanding of students.]

(ii) Draw the labeled diagram to interface stepper motor to port 1 of 8051. Write assembly or 'C' language program to rotate the motor in clockwise direction continuously.

(Interfacing diagram- 3 Marks, Program - 3 Marks)

Ans: Stepper Motor Interfacing Diagram:



Program:

;PROGRAM : SIMPLE STEPPER MOTOR DRIVER **USING ROTATE INSTRUCTION**

```
        ORG     0000H     ;START THE PROGRAM
        MOV     A,#66H    ;LOAD THE CODE TO DRIVE THE MOTOR
UP:     MOV     P1,A      ;SEND CODE TO P1
        LCALL   DELAY     ;DELAY
        RR      A         ;ROTATE THE CODE
HERE:   SJMP    HERE      ;STOP PROGRAM AFTER 180^O ROTATION

DELAY:                    ; DELAY LOOP WITIN LOOP
```

```
        MOV     R7,#0FFH
UP2:    MOV     R6,#0FFH
UP1:    DJNZ    R6,UP1
        DJNZ    R7,UP2
        RET
```

OR

We are sending 4 pulses as follow

;	A	B	C	D		
;	1	0	0	1	=09	COIL A & D ON
;	1	1	0	0	=0C	COIL A & B ON
;	0	1	1	0	=06	COIL B & C ON
;	0	0	1	1	=03	COIL C & D ON

```
;PROGRAM : STEPPER MOTOR DRIVER USING LOOKUP TABLE
        ORG     0000H       ;START THE PROGRAM
RPT:    MOV     DPTR,#0400H ;LOAD THE STARTING ADDRESS OF LOOKUP
TABLE
        MOV     R7,#04H     ;LOAD THE COUNTER
H:      CLR     A           ;CLEAR THE A
        MOVC    A,@A+DPTR   ;TAKE THE CODE FROM LOOKUP TABLE
        MOV     P1,A        ;SEND THE CODE TO P0
        INC     DPTR        ;INCREMENT DPTR FOR NEXT CODE
        LCALL   DELAY       ;DELAY
        DJNZ    R7,H        ;DECREMENT THE COUNTER
        SJMP    RPT         ;STOP PROGRAM AFTER 180O ROTATION

        ORG     0400H       ;STARTING ADDRESS OF LOOKUP TABLE
        DB      09H         ;0400H = 09H
        DB      0CH         ;0401H = 0CH
        DB      06H         ;0402H = 06H
        DB      03H         ;0403H = 03H

DELAY:                      ;DELAY LOOP WITIN LOOP
        MOV     R0,#0FFH
UP2:    MOV     R1,#0FFH
UP1:    DJNZ    R1,UP1
        DJNZ    R0,UP2
        RET
```

OR

// C language program Stepper motor interfacing

```
#include <Intel\8052.h>
#include <standard.h>
void delay (unsigned int);

/*
COIL   A = P1.0
COIL   B = P1.1
COIL   C = P1.2
COIL   D = P1.3
*/
void main ()
{
        P1 = 0x00;                  //MOTOR OFF
        While(1)
        {
                P1 = 0x09;          // COIL A & D ON
                delay (100);
                P1 = 0x0C;          // COIL A & B ON
                delay (100);
                P1 = 0x06;          // COIL B & C ON
                delay (100);
                P1 = 0x03;          // COIL C & D ON
                delay (100);
        }
}


        void delay(unsigned int t)

        {

                Unsigned int x, y;

                for(x=0; x<=t; x++)

                for(y=0; y<=675; y++);

        }
```

[NOTE: Program may change. Student can also use the other logic.

Please check the logic and understanding of students.]

2. Attempt any FOUR of the following: Marks 16

a) State the salient features of microcontroller 8051.

(Any four each- 1 Marks)

Ans:

1. 4 KB on chip program memory (ROM or EPROM)).
2. 128 bytes on chip data memory(RAM).
3. 8-bit data bus
4. 16-bit address bus
5. 32 general purpose registers each of 8 bits
6. Two -16 bit timers T_0 and T_1
7. Five Interrupts (3 internal and 2 external).
8. Four Parallel ports each of 8-bits (PORT0, PORT1,PORT2,PORT3) with a total of 32 I/O lines.
9. One 16-bit program counter and One 16-bit DPTR (data pointer)
10. One 8-bit stack pointer
11. One Microsecond instruction cycle with 12 MHz Crystal.
12. One full duplex serial communication port.

b) State various addressing modes available in 8051, with two examples each.

(Any four Addressing modes - 2 Marks, Example- 2 Marks)

Ans:

1) Immediate addressing mode

Ex 1 MOV A, #6AH

Ex 2 MOV R1,#55 H

2) Direct addressing mode

Ex1 MOV A, 04H

Ex2 MOV P1 , A

3) Register direct addressing mode

Ex1 MOV A, R4

Ex2ADD A, R7

4) Register Indirect address mode

Ex1 MOV A, @R0

EX2 MOV B,@R1

5) Indexed addressing mode.

Ex1 MOVC A, @A+DPTR

EX2 MOVC A, @A+PC

(Note : any other correct example can write)

c) List the alternate function of 8051 port 3 pins. Also write the instruction to set all the port 1 pins as input.

(Alternate function-3 Marks, Instruction - 1 Mark)

Ans:

Port Pin	Alternate Function
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
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)

Instruction :

MOV P1, # FF h make P1 input port

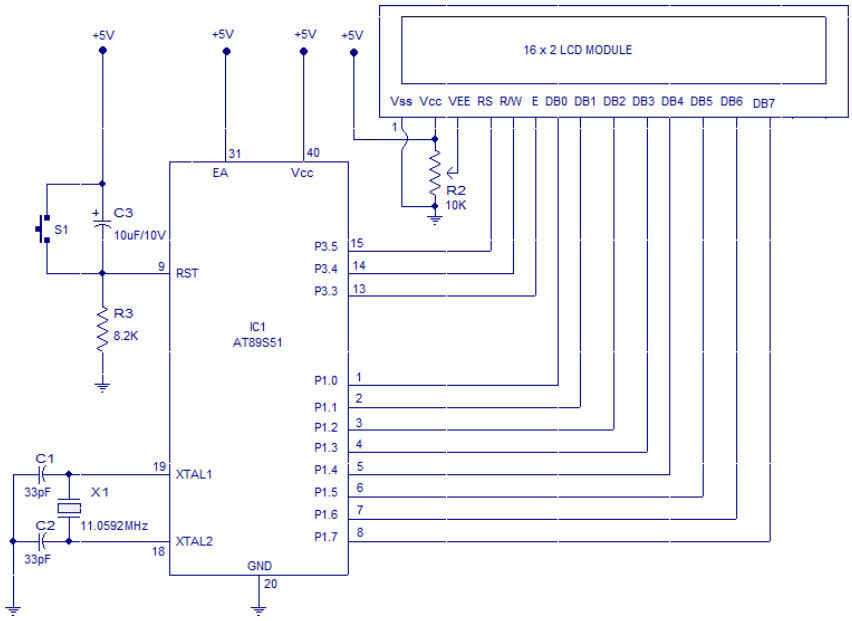
OR

P1=0xFF; //make P1 input port

d) Draw the labeled diagram to interface 16 X 2 LCD to microcontroller 8051

(Diagram -3 Marks, Label - 1 Mark)

Ans:



[Note: student can draw any other correct diagram]

e) State various advantages and disadvantages of embedded system.

(Advantages-2 Marks, Disadvantages- 2 Marks)

Ans: Advantages:

1) Design and Efficiency

2) Cost

3) Accessibility

4) Maintenance

5) Redundancies

Disadvantages:

1) It is designed to do a specific task only, other task cannot be executed.
2) It is difficult to upgrade the machine
3) Harder to carry files from one machine to another

f) What is task in an embedded system? What are various states of a task?

(Task-3 Marks, States - 1 Mark)

Ans: Task:

An application program can also be said to be a program consisting of the tasks and task behaviors in various states that are controlled by OS. A task is like a process or thread in an OS. Task term used for the process in the RTOS for the embedded systems.

A task consists of executable program (codes), state of which is controlled by OS,

The state during running of a task: - represented by information of process status (running, blocked, or finished),

process-structure—its data, objects and resources, and task control block (PCB) Runs when it is scheduled to run by the OS (kernel), which gives the control of the CPU on a task request (system call) or a message. • Runs by executing the instructions and the continuous changes of its state takes place as the program counter (PC) changes .Task is that executing unit of computation, which is controlled by some process at the OS scheduling mechanism, which lets it execute on the CPU and by some process at OS for a resource-management mechanism that lets it use the system memory and other system-resources such as network, file, display or printer

The task─ can send signal (s) or message(s) that can let another task run. The OS can only block a running task and let another task gain access of CPU to run the servicing codes

States of a Task in a system

(i) Idle state [Not attached or not registered]

(ii) Ready State [Attached or registered]

(iii) Running state

(iv) Blocked (waiting) state

(v) Delayed for a preset period

3. Attempt any FOUR of the following: Marks 16

a) Draw and explain the RAM structure of 8051.

(Diagram- 2 Marks, Explanation- 2 Marks)

Ans: Explanation :

Internal data memory of 8051 is 256 bytes, which is divide in two parts. The lower 128 bytes

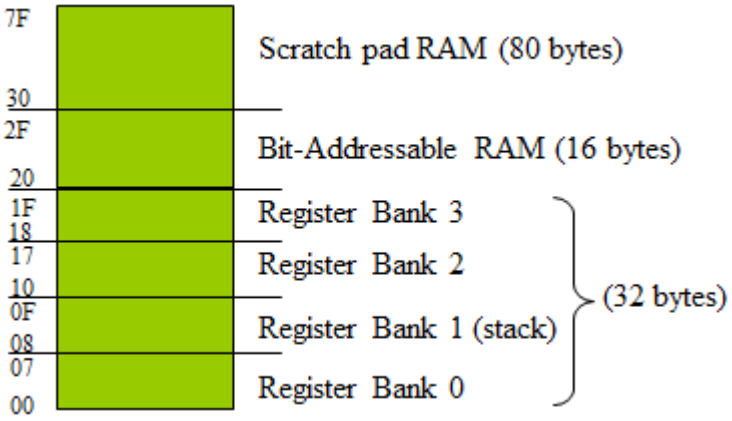
(00h to 7Fh) as internal Data RAM and upper 128 bytes consists of SFRs. Internal 128 bytes of RAM can be addressed as memory locations or as registers. 22 special function registers occupy data memory space from 80H to F8H

- The lowest 32 bytes(00H-7FH) of the on-chip RAM form 4 banks of 8 registers each.
- The next 16 bytes – locations 20H to 2FH – form a block that can be addressed as either bytes or individual bits.

– The bytes have addresses 20H to 2FH.
– The bits have addresses 00H to 7FH.

- Locations 30H to 7FH are general purpose RAM.

Diagram:



b) Describe any four assembler directives used in assembly language programming.

(Any four assembler directives with example -1 Mark each)

Ans: Various 8051 directives are:

1. ORG
2. EQU
3. END
4. DB
5. CODE
6. DATA

1. ORG : ORIGIN

The ORG directive is used to indicate the beginning of the address. The number that comes after ORG is the address from where program will begin. The number can be either in hex and decimal

Ex: ORG 1000H

It indicates that program shall start from memory address 1000h

2. EQU: EQUATE

It is used to define a constant without occupying a memory location The EQU directive assigns a constant value to a label. When the label appears in the program, its constant value will be substituted for the label.

Ex: COUNT EQU 10
MOV R2, #COUNT
When the instruction is executed , register R2 is loaded with value 10.

3. **END:** This indicates to the assembler the end of the source (asm) file . The END directive is the last line of an 8051 program. Means that in the program anything after the END directive is ignored by the assembler

Ex: MOV A , #20
ADD A , #10
END

4. DB : Define Byte
It is used to define the 8-bit data. It is used to write the value after DB , into the program memory. When DB is used to define data, the numbers can be in decimal, binary, hex, ASCII formats

Ex: ORG 1000H
MYDATA: DB 20,21
After execution of this , location 1000h=20 & 1001h = 2**1**

5. CODE : This directive assigns a specific name to the specified program memory location**.**

Ex: LIST CODE 1020H
Program Memory location 1020H is now referred to as **LIST**

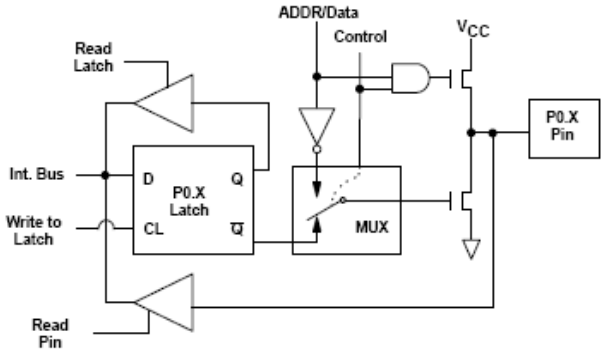
6. DATA : This directive assigns a specific name to the specified internal data memory location**.**

Ex: LIST DATA 20H
Internal data Memory location 20H is now referred to as **LIST**

c) Draw the structure of port 0 of 8051. Also state the need of pull up resistors.

(Diagram -2 Marks, Need- 2 Marks)

Ans: Diagram:



Need of PULL UP Register:

The pins of P0 are connected internally to an "open drain" circuit. Port 0 does not have internal pull up, therefore, it must be connected to an external pull-up resistor to operate properly as an **INPUT** port.

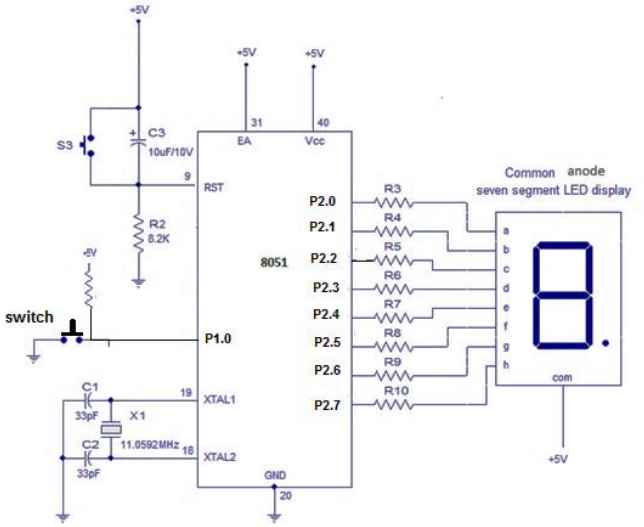
If pull up resister is not connected then , the pin status is undefined, i.e. neither logic '0' nor logic '1'.

But when we connect pull up register, the pin status is defined as logic '1'. By using SETB or CLR instruction we can make the status of PIN as logic '1' or logic '0'.

d) Draw the labeled diagram to interface switch to P1.0 and seven segment display to port 2 of 8051.

(Neat labeled diagram - 4 Marks)

Ans: (Students can draw common anode or common cathode 7 seg.display)



e) Explain the terms:

(i) Device driver

(ii) In circuit imulator

**** [Note: Instead of imulator, Emulator is considered]**

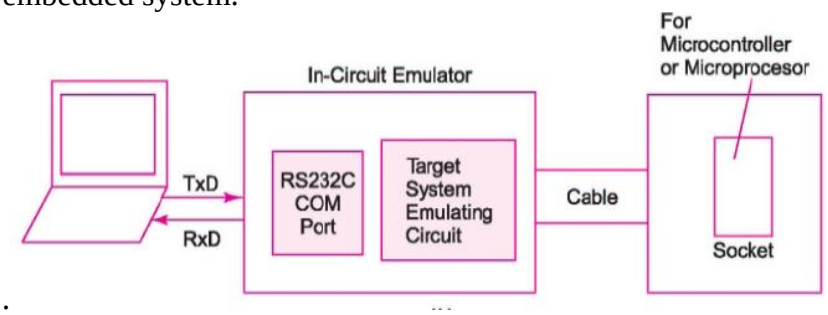
(Device driver-2 Marks,For in circuit Emulator explanation- 1 Mark, Diagram-1 Mark)

Ans: (i) ***Device driver***

Adevice driver is a software interface to a hardware devicenthat handles requests from the kernel regarding the use of particular device. A device driver has a set of routines (functions) used by a high-level language programmer, which does the interaction with the device hardware, sends control commands to the device, communicates data to the device and runs the codes for reading device data. Each device in a system needs device driver routine with number of device functions. An ISR relates to a device driver command (device-function). Device driver can be considered as software layer between an application program and the device.

(ii) In circuit Emulator:

- An **in-circuit emulator** (ICE) is a hardware device used to debug the software of an embedded system.
- The ICE is temporarily installed between the embedded system and an external terminal or personal computer so that the programmer can observe and alter what takes place in the embedded system.



f) What is task synchronization? How is it achieved?

(Task synchronization- 1 Mark, Methods to achieve - 3 Marks)

Ans: Task Synchronization

Synchronization is essential for tasks to share mutually exclusive resources (devices, buffers, etc) and/or allow multiple concurrent tasks to be executed (e.g. Task A needs a result from task B, so task A can only run till task B produces it).

Task synchronization is achieved using two types of mechanisms:

· **a) Event Objects b) Semaphores**

a) **Event objects** : Event objects are used when task synchronization is required without resource sharing. They allow one or more tasks to keep waiting for a specified event to occur. Event object can exist either in triggered or non-triggered state. Triggered state indicates resumption of the task.

b) **Semaphores :**A semaphore functions like a key that define whether a task has the access to the resource. A task gets an access to the resource when it acquires the semaphore. A semaphore has

an associated resource count and a wait queue. The resource count indicates availability of resource. The wait queue manages the tasks waiting for resources from the semaphore.

There are three types of semaphore:

- Binary Semaphores
- Counting Semaphores
- **Mut**ually **Ex**clusion(Mutex) Semaphores

4 a) Attempt any THREE of the following: Marks 12

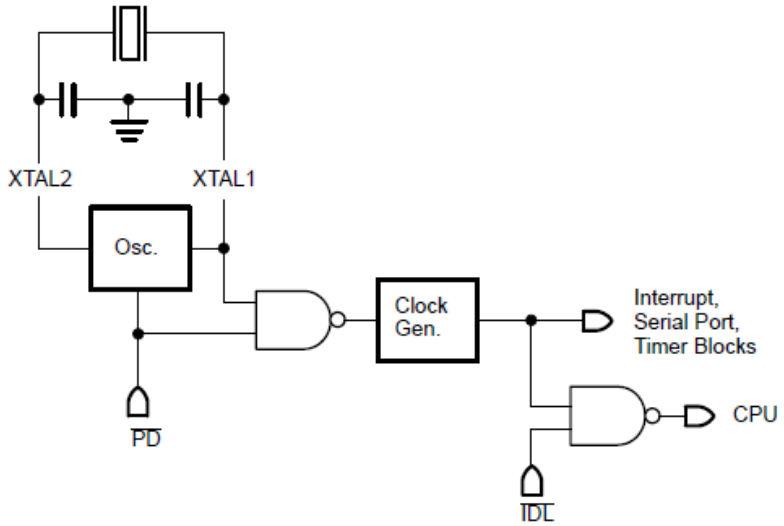
(i) Explain various power saving modes of microcontroller 8051

(For each explanation of idle and power down mode- 1 Mark, Diagram- 1 Mark, PCON format-1 Mark)

Ans. 8051 has two power saving modes:

1. IDLE MODE
2. POWER DOWN MODE

Idle and Power Down Hardware



1.IDLE MODE: It is selected by setting IDL bit in PCON register. In idle mode the internal clock is gated off to CPU but not to interrupt , timer and serial port functions. PC, SP , PSW ,

Accumulator and all other registers maintain their data during idle mode. Port pins hold their state, they had at time idle mode was activated.Idle mode can be terminated either by enabling an interrupt which causes IDL bit of PCON register to be cleared or by hardware reset

2.POWER DOWN MODE:

It is selected by setting PD bit in PCON register. In power down mode the on-chip oscillator is stopped. With the clock frozen , all functions are stopped. Contents of on chip RAM & SFRs are maintained. Port pins output the values they held by their respective SFRs. ALE & PSEN are held low. In this mode VCC can be reduced as low as 2Volts. Power down mode can be terminated only by hardware reset

POWER CONTROL (PCON) REGISTER :

MSB							LSB
SMOD	—	—	—	GF1	GF0	PD	IDL

(ii) Explain the following 8051 instructions:

1) SETB C

2) SWAP A

3) MOV 80h, 90h

4) MUL AB

(For each-1 Mark)

Ans:

1) SETBC

Set the carry flag. After the execution CY=1.

2) SWAP A

SWAP nibbles within the accumulator .This instruction interchanges the low order and high order nibbles of the accumulator.

EG.A = 05H

SWAP A

A = 50H

3) MOV 80H,90H

Move the content of internal RAM location 90h into internal RAM location 80h

E.G. 80H=02H , 90H = 04H

AFTER EXECUTION 80H=04H , 90H =04H

4) MUL AB

This instruction multiplies content of register A with content of register B, and the 16 bit product is stored in A &B , Such that lower byte of product is stored in register A & higher byte of product is stored in register B

BA = A X B

EG.: A= 02H, B=04H,

AFTER EXECUTION , A= 08H , B=00H

(iii) Write a program in assembly language or C to generate square wave of 10 KHz on pin p2.7 of 8051 using timer 0.

(Count calculation -1 Mark, Program -3 Marks)

Ans:

Frequency = 10 khz
Therefore Time period T = 1 / 10KHZ = 0.1ms
Therefore TON = TOFF = 0.1ms / 2 = 0.05ms

Required time delay = (12 / Fosc) x number of increments (N)
0.05 ms= (12 / 11.0592MHZ) x number of increments (N)
0.05 ms= 1.085usec. x N
N = 46

Using TIMER 0 in MODE 1,

COUNT = 2^{16} - N

COUNT =65536 – 46 =65490 = FFD2 H

(Count can be calculated using Fosc= 12MHZ also)

Assembly language Program :

```
        MOV TM0D,#01H ;TIMER 0, MODE 1
AGAIN:MOV TH0,#0FFH ;Load higher byte of count
        MOV TL0,#0D2H  ; load lower byte of count
         SETB TR0              ;start timer 0
HERE: JNB TF0,HERE ;CHECK IF TF0 IS SET
        CPL P2.7        ; Complement P2.7
         CLR TR0      ; IF TF0 =1 , STOP TIMER 0
         CLR TF0     ; CLEAR TF1
        SJMP AGAIN  ;REPEAT AGAIN
         END
```

'C' Language program:

```
#include <reg51.h>
void T0M1Delay(void);
sbitmybit=P2^7;
void main(void)
{
while (1)
{
mybit=~mybit;
T0M1Delay();
}
}
void T0M1Delay(void)
{
TMOD=0x01;
TL0=0xFF;
TH0=0XD2;
TR0=1;
while (TF0==0);
```

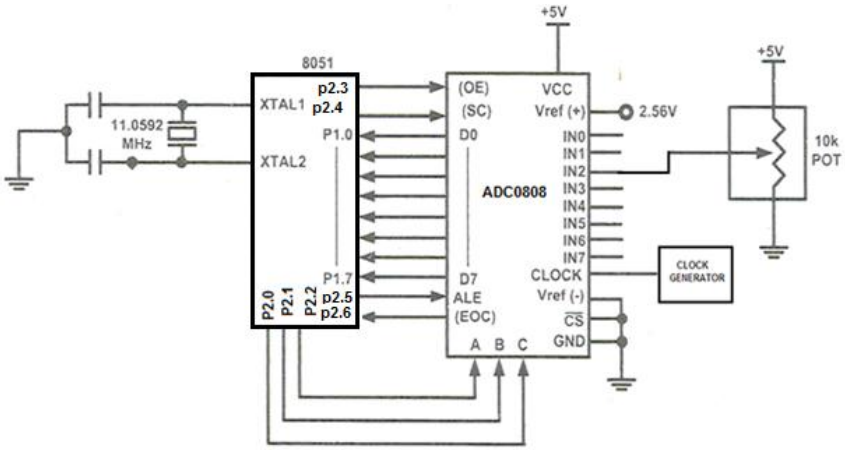
```
TR0=0;
TF0=0;
}
```

[Note: Student can write any other correct program ,any other timer mode can be used]

(iv)Draw labeled diagram to interface analog to digital converter ADC0808 to 8051.

(Neat labeled diagram -4 Marks)

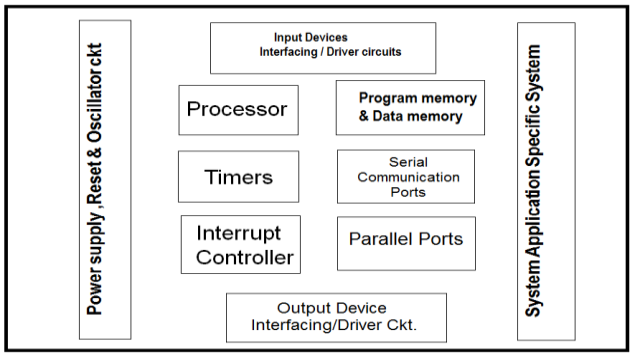
Ans:



b) Attempt any ONE of the following: Marks 06

(i) Draw the block diagram of embedded system. Explain various hardware units.
(Diagram- 2 Marks, Explanation- 4 Marks)

Ans:



Various hardware units of embedded system are:

1. Embedded processor:

It is the heart of the embedded system. It has two essential units : control unit and execution unit. Control unit fetches instructions from memory and execution unit includes ALU and circuits to perform execution of the instructions for a program control task An embedded system processor chip can be one of the following:

2. Power supply, reset & oscillator circuit:

- Most of the systems have their own power supply. Some embedded system do not have their own power supply.These embedded systems are powered by external power supply e.g. USB based embedded system, network interface card, Graphics Accelerator etc. are powered by PC power supply.
- Reset means that processor begins processing of instructions from starting address set by default in program counter on power up.

Watchdog Timer reset.

- The clock circuit controls execution time of instructions , CPU machine cycles.

3. Timers :

Timer circuit is suitably configured as system clock or RTC (Real time clock). To schedule various tasks and for real time programming an RTC (Real Time Clock), or system clock is needed.

4. Program & data memory:

In embedded system, secondary memory like disk is avoided. Most of the embedded processors have internal memory such as ROM, RAM, flash/EEPROM, EPROM/PROM for storing program and data.

5. Interrupt controller:

It is an interrupt handling mechanism which must exist in embedded system to handle interrupts from various processes and for handling multiple interrupts simultaneously pending for service.

6. I/O ports :

I/O ports are used to interface external devices like sensors, key buttons, transducers, LEDs, LCD actuators, alarms, motors, values, printer etc.There are two types of ports ,parallel and serial port. The parallel ports are used in short distance communication while serial ports are used in long distance communication.

7.Input& output device interfacing/driver circuits:

Some I/O devices like motors, actuators, valves, sensors are not compatible with the processor. Hence the I/O interface circuits are designed to drive such input and output devices interfaced to the embedded processor

8.System Application specific circuits

These are the circuits that can control specific target circuits. They consist of ADC,DAC, relays, sensors etc.

(ii) Explain the features of RTOS. State how it differs from general operating system.

(Features - 3 Marks, Difference - 3 Mark)

Ans: Features of RTOS:

1. **Multithreading & pre-emptibility:** The scheduler should be able to preempt any task in the system and allocate the resource to the thread that needs it most even at peak load.

2. **Thread Priority:** All tasks are assigned priority level to facilitate the preemption.

3. **Inter task communication and synchronization:** Multiple tasks pass information among each other in a timely fashion and ensuring data integrity

4. **Priority inheritance:** RTOS should have large number of priority levels and should prevent priority inversion using priority inheritance

5. **Short latencies:** The latencies are short and predefined.
   Task switching latency, interrupt latency, interrupt dispatch latency

6. **Control to memory management:** To ensure predictable response to an interrupt , an RTOS should provide way for task to lock its code and data into real memory

Difference between RTOS & general OS(any three)

Sr. No.	Desktop OS	RTOS
1.	Applications are compiled separately from the OS.	Applications are compiled and linked together with the RTOS.
2.	As you turn on your desktop, only OS starts.	At boot up time, application usually gets controlled first and then it starts the RTOS.
3.	It is a less reliable system	It is a more reliable system
4.	It is not able to customize dependency on applications.	It is able to customize dependency on applications.
5.	It does not have deterministic response.	It has deterministic response.
6.	Memory required depends on the version.	Memory required (footprint) is very less.
7.	It protects itself very carefully from applications.	It does not protect itself as carefully from applications.
8.	e.g. Windows, Linux.	e.g. RT Linux, Vx Works.

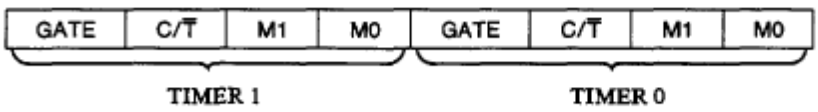
5. Attempt any FOUR of the following: Marks 16

a) Draw format of TMOD SFR. Explain the function of each bit.

(Format-2 Marks, Function of each bit- 2 Marks)

Ans:

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



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.

b) Explain step by step procedure to execute a program using any cross compiler like KEIL.
(Correct steps-4 Marks)

Ans: The step-by-step procedure to execute the program using KEIL cross compiler is as given below where the Project Manager makes it easy to create a new project and to design the embedded application:

1) Setup the Project to define the project file and select the microcontroller from the Device Database.
2) Software Components (MDK only) can be used to create applications with the Run-Time Environment.
3) Update Software Component Files allows you to exchange project files that are delivered with a Software Pack.
4) Add Source Files to Project shows ways of adding files to the project.
5) Create File Groups explains the reasoning behind grouping files and how this can be done.
6) Set Tool Options explains the Target option page and its parameters.
7) Configure Startup Code explains the configuration wizard options you may use in assembly or C startup files.
8) Build the Project explains the build commands and how errors and warnings can be tracked.
9) Create HEX File explains the configuration option required for creating a HEX file.

OR

1) Start Keil by double clicking on Keil icon.
2) To create new project, click on project and select **new project.**
3) Select appropriate location for new project,type project name , and click on save button.
4) Select device for target Target -1, window will open. It displays a list of manufacturers of micro controllers.
5) Double click on **ATMEL** or **Intel,** list of supported microcontrollers get displayed. Select **8051** from Intel or **AT 8951**(as per the target board) for Atmel, click **ok**.
6) Click file pull down menu, select new, a text editor window will open. Save this file in a same folder where project was stored. Give extension as "< file name>.c"
7) On left hand ,project workspace window will display Target 1 and source group 1.
8) **Right click** on source group ;select**Add files to source group 1**.
9) Select **file type as C file**. All file names with "C" will be displayed, select **appropriate file** ,click**ADD** and close.
10) Project workplace window will display 'Target 1' and 'source group 1'with added file name.
11) Type C language program. Save the file periodically to avoid loss of data .
12) Right click on source group, click on Build target or press F7. At the bottom, output window will display errors if any. If there are some errors then remove all errors and repeat from step number 11 until zero errors.
13) Start the simulation by ready to debug or execute.
14) Select appropriate ports from peripherals pull down menu,before executing the program.
15) On start of debug session, project window will display all internal registers of 8051 and their contents. To execute the program click on RUN button.
16) Observe the output program on port 3, by clicking on pins of port 2.
17) Stop the execution by clicking on the icon given.

c) Draw the format of IE SFR. Explain the function of each bit. Write an instruction to enable only timer interrupt.

(Format – 1 Mark, Explanation of each bit-2 Marks, Instruction-1 Mark)

Ans:

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.

Instruction to enable only timer interrupt –

MOV IE ,#10001010 b

or

MOV IE , #8A H

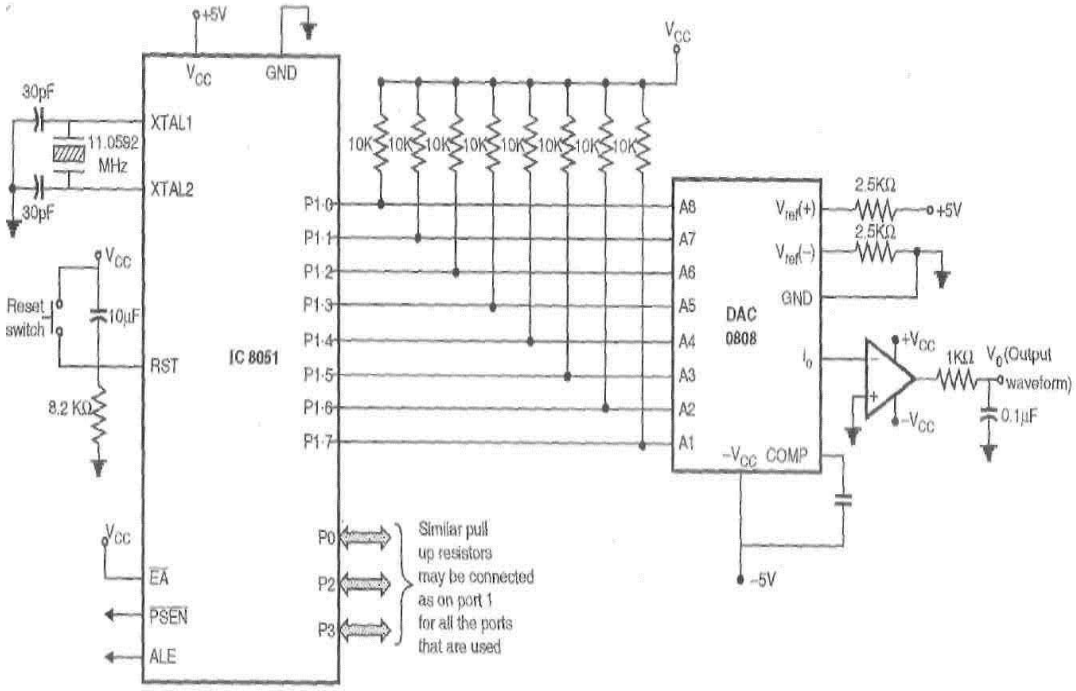
Or

IE=0X8A;

d) Draw the diagram to interface D to A converter DAC 0808 to 8051. Write the program in assembly or C to generate ramp wave.

(Diagram – 2 Marks, Program for ramp wave- 2 Marks)

Ans: Interfacing D to A convertor DAC 0808 to 8051.



Program for ramp wave.

Using C LANGUAGE

```
#include<reg51.h>
void main (void)
{
P1=0x00;                                    // Initialise Port1 to all 0's
while (1)
{
```

```
while (P1!=0XFFH)                    // FFH increments to 00H
P1++;
}
}
```

OR

Assembly Language Program

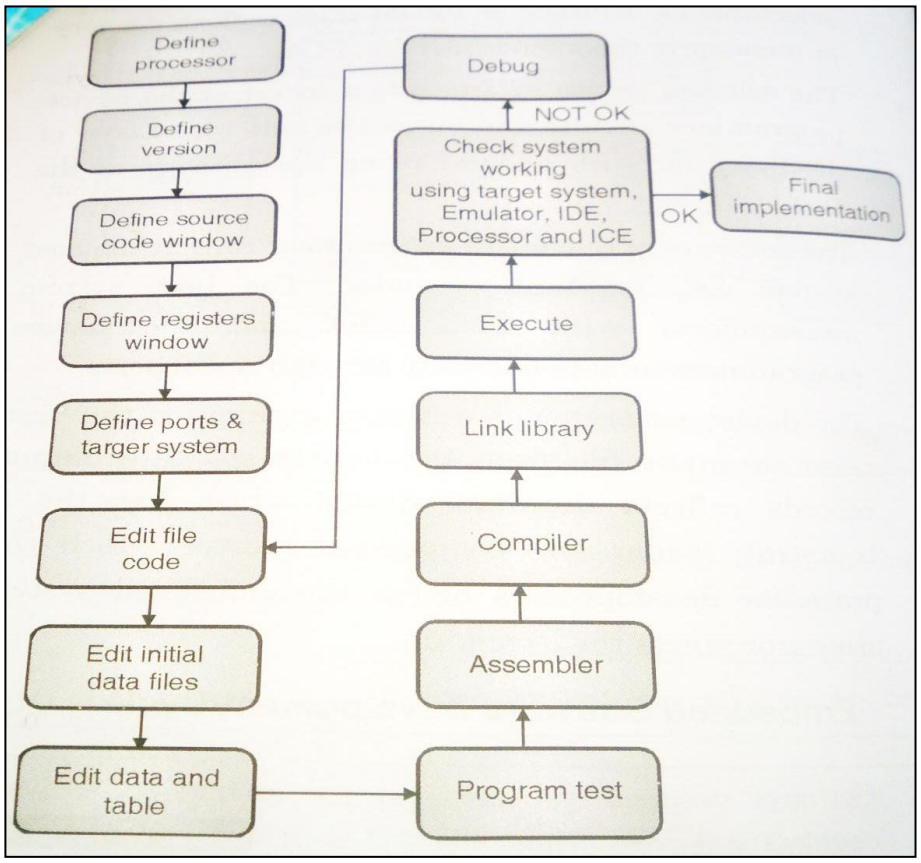
```
ORG 000H
LJMP START
ORG 0100H
START:
MOV A, #00H                          // Initialise Port 1 TO ALL 0's
HERE:MOV P1, A
INC A
CJNE A, # 0FFH, HERE                 // Increment the data in port 1
SJMP HERE
END
```

(NOTE: students can write any other correct program logic)

e) State various steps in software development cycle of an embedded system.

(Software development cycle -4 Marks)

Ans: Steps in software development cycle of an embedded system



OR

Steps

1) Define the processor /processing device (family and version) for the target system.

2) Defining the source code window with labels and symbolic arguments as execution goes on for each single step.

3) Define the processor registers for each step /module.

4) Define details of ports and target system.

5) Editor to edit source code files, initial data files , data and tables.

6) Define assembler/compiler for program test with link library.

7) Execute the source code to check the target system , else debug the source code.

8) For system working properly as per the specifications, then final implementation is carried out.

9) Finally application software is embedded in the system by using device programmer.

f) What is intertask communication? State various mechanisms to achieve it.

(Intertask communication-2 Marks, Any four Mechanisms- 2 Marks)

Ans: Intertask/Inter process communication is the set of techniques for exchanging the data among multiple threads in one or more processes. Inter process communication techniques are divided into methods for message passing, synchronization, shared memory and remote procedure calls. The method of IPC may vary based on bandwidth and latency of communication between the threads, and the type of data being communicated

Methods are as follows:

1. Message Queue

2. Pipeline

3. Remote procedure call

4. Semaphore

5. Signal

6. Attempt any FOUR of the following: Marks 16

a) State various interrupts available in 8051 alongwith their priority and vector locations. Also list various SFRS associated with interrupt related operation.

(Interrupts with priority & vector location - 3 Marks, Any two SFRs- 1 Mark)

Ans: Interrupts available in 8051 with their priority and vector locations is as shown below

Interrupt	Vector address	Priority
External interrupt 0	**0003h**	1 Highest
Timer/counter 0	000Bh	2
External interrupt 1	**0013h**	3
Timer/counter 1	001Bh	4
Serial port	0023h	5 Lowest

SFR's associated with interrupts-

1) **IP** (INTERRUPT PRIORITY REGISTER)
2) **IE** (INTERRUPT ENABLE REGISTER)
3) **TCON** (Timer Control D0 to D3)

b) Write a program using assembly language or 'C' to read data from port 0 and port 1. Multiply the data received send the result on port 2 and port 3 respectively.

(Proper program -4 Marks)

Ans: C language program

```
#include <reg51.h>
unsigned char x, y;
unsigned int z;
void  main( )
{
        P0=0xFF;            // configure ports as input ports
        P1=0xFF;
```

```
        x=P0;           //Read P0 in x
        y=P1;           //Read P1 in y
        z=x*y;          //Do Multiplication

        P2=z;           //store lower 8 bit at P2
        P3=z>>8;        //store upper 8 bit at P3

        while (1)
        {               //infinite loop
        }
}
```

OR

Assembly Language Program

```
        MOV   P0, #0FFH     ; configure ports as input ports
        MOV   P1, #0FFH
        MOV   A, P0         ; P0 data in A
        MOV   B, P1          ; P1 data in B
        MUL   AB            ; multiplication result in A
        MOV   P2, A         ; transfer result to P2
        MOV   P3, B         ; transfer result to P3
HERE: SJMP   HERE
```

[NOTE: Program may change. Student can also use the other logic.

Please check the logic and understanding of students.]

c) Write a program to toggle the LED connected to P1.7 on every occurance of external interrupt INTO.

(Correct program-4 Marks)

Ans: C LANGUAGE PROGRAM

```
#include<reg51.h>
sbit LED=P1^7;
void external(void) interrupt 0
```

```
{
        LED = ~ LED;          // toggle pin
}

void main ( )
{
        LED=0;                //LED off
         IE = 0x81;           //enable External interrupt 0
        While(1)
        {                             //infinite loop
        }
}
```

OR

Assembly language program:

```
        ORG   0000H
        LJMP  MAIN

        ORG   0003H          ; Vector location of INT0
        CPL   P1.7           ; toggle LED on the P1.7
        RETI

        ORG    0030H
MAIN: MOV  IE, #81H          ; Enable INT0
HERE: SJMP  HERE
```

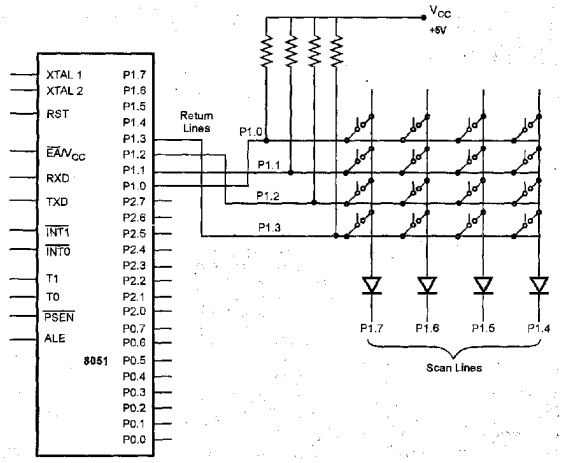
[NOTE: Program may change. Student can also use the other logic.

Please check the logic and understanding of students.]

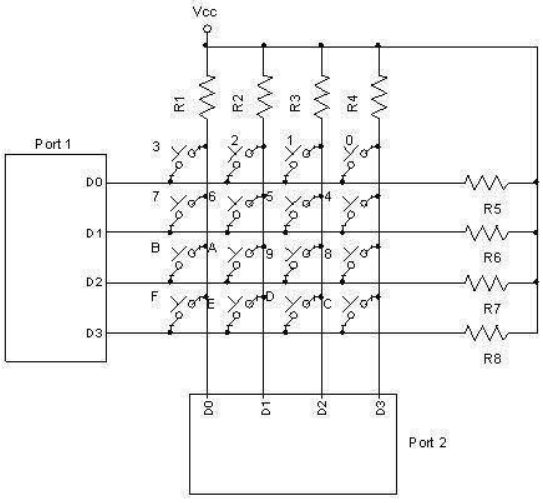
d) Draw labeled diagram to interface 4 X 4 keyboard to 8051.

(Correct Diagram- 4 Marks)

Ans:



OR



e) Explain in brief:

(i) Device programmer

(ii) Target board

(Device programmer -2 Marks, Target board-2 Marks)

Ans: i)Device programmer:

Also called as laboratory programmer, a programming system for a application device such as EPROM/ROM or Flash memory or microcontroller memory, PLA.

The device to be programmed is inserted into the socket at the device programmer and burns the code using software at the host. i.e. personal computer through serial port.

The device programmer software running on the host uses an input file from the locator software output records reflects the final design which has the bootstrap loader and compressed records which the processor decompresses before the embedded system processor starts the execution.

ii) Target board

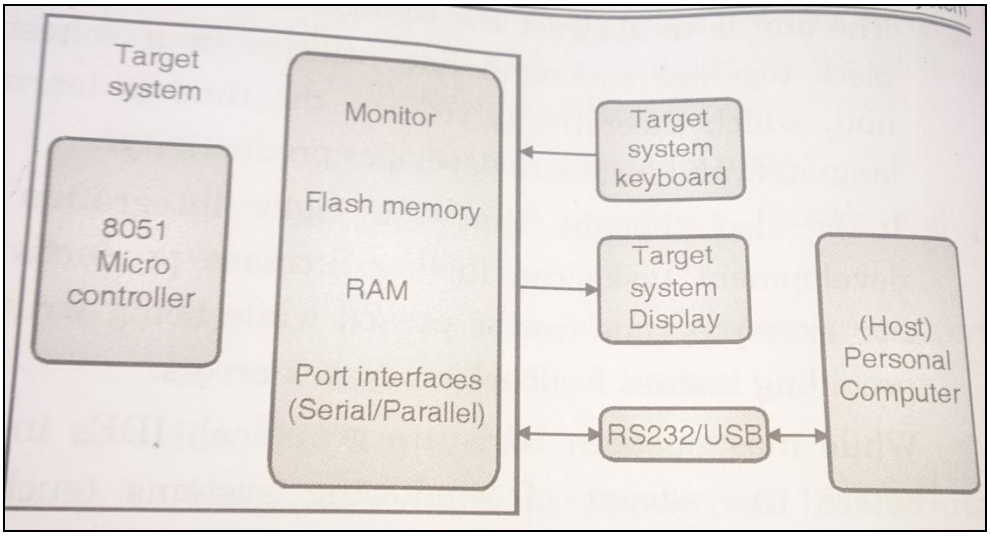
Target board or machine or system consists of-

1) A microprocessor or microcontroller,

2) ROM-memory of image of embedded system,

3) RAM- memory for implementation of stack, temporary variables and memory buffers
Peripheral devices and interfaces such as RS 232,10/100 base ethernet,parallel ports, USB etc.

Example –A simple sample target system is as shown-



The target board differs from the final system as it interfaces with personal computers as well as work as a standalone system which requires a repeated downloading of the codes during development phase in the flash memory. Also requires repeated modification, testing, simulation, debugging till it works according to final specifications. Once done with, the code is downloaded in ROM (instead of flash memory) in the target system.

f) Describe starvation with example.

(Starvation -2 Marks, Example -2 Marks)

Ans: Starvation:

Starvation: Starvation is a resource management problem where a process does not get the resources it needs for a long time because the resources are being allocated to other processes. Starvation generally occurs in a Priority based scheduling System. Where High Priority (Lower Number = Higher Priority) requests get processed first. Thus a request with least priority may never be processed.

For example: Consider priority based scheduling with scheduling criteria as the shortest process first. In a ready queue where all the processes are waiting for CPU, shortest process will be selected first. If there is a continuous supply of shortest process then the longer process may never get scheduled on CPU which leads to its starvation.