



MODEL ANSWER
SUMMER– 17 EXAMINATION

Subject Title: EMBEDDED SYSTEM

Subject Code:

17658

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 anyequivalent 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. No.	Sub Q.N.	Answer	Marking Scheme																		
Q.1	a)	Attempt any THREE of the following:	12-Total Marks																		
	(i)	State the interrupts used in 89C51? Give their priorities and vector addresses.	4M																		
	Ans:	<table border="1"><thead><tr><th>Interrupt</th><th>Vector Address</th><th>Priority</th></tr></thead><tbody><tr><td>IE0 / (External interrupt 0 , INT0)</td><td>0003H</td><td>1</td></tr><tr><td>TF0 / (Timer 0 Interrupt)</td><td>000BH</td><td>2</td></tr><tr><td>IE1 / (External interrupt 1 , INT1)</td><td>0013H</td><td>3</td></tr><tr><td>TF1 / (Timer 1 Interrupt)</td><td>001BH</td><td>4</td></tr><tr><td>TI or RI (serial port interrupt)</td><td>0023H</td><td>5</td></tr></tbody></table>	Interrupt	Vector Address	Priority	IE0 / (External interrupt 0 , INT0)	0003H	1	TF0 / (Timer 0 Interrupt)	000BH	2	IE1 / (External interrupt 1 , INT1)	0013H	3	TF1 / (Timer 1 Interrupt)	001BH	4	TI or RI (serial port interrupt)	0023H	5	4M
Interrupt	Vector Address	Priority																			
IE0 / (External interrupt 0 , INT0)	0003H	1																			
TF0 / (Timer 0 Interrupt)	000BH	2																			
IE1 / (External interrupt 1 , INT1)	0013H	3																			
TF1 / (Timer 1 Interrupt)	001BH	4																			
TI or RI (serial port interrupt)	0023H	5																			
	(ii)	Write difference between synchronous and asynchronous data communication.	4M																		
	Ans:		(Any four points 1 M each)																		



Sr. No.	Synchronous	Asynchronous
1	Same clock pulse is required at transmitter and receiver	Different clock pulse is required at transmitter and receiver
2	Used to transfer group of character	Used to transfer one character at a time
3	Synchronous character is required.	Synchronous character is required.
4	No start and stop signals are required	Start and stop signals are required.
5	Data transmission rate is greater then or equal to 20Kbps	Data transmission rate is less then or equal to 20 Kbps.
6	It is less reliable	It is more reliable

(iii) List the software development tools in an embedded system and state the function of compiler and debugger.

4M

Ans: Software development tools:

- Compiler
- Cross assembler
- Cross compiler
- Locators
- Loaders
- Simulators
- Debugger
- Integrated development environment (IDE)

The function of compiler

- 1) **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.

The function of debugger.

- 2) **Debugger:** - Allows you to download your code to the emulator's memory and the control all of the functions of the emulator from a PC. Common debugging features include the capability to examine and modify the microcontroller's on-chip registers, data- and program-memory; pausing or stopping program executing at defined program locations by setting breakpoints; single-stepping (execute one instruction at a time) through the code and looking at a history of executed code (trace).

(List: 2M,
Explanation compiler:
1M,
Debugger:1
M)

(iv) Draw interfacing diagram of 4*4 matrix keyboard with 89C51 micro controller (No program)

4M

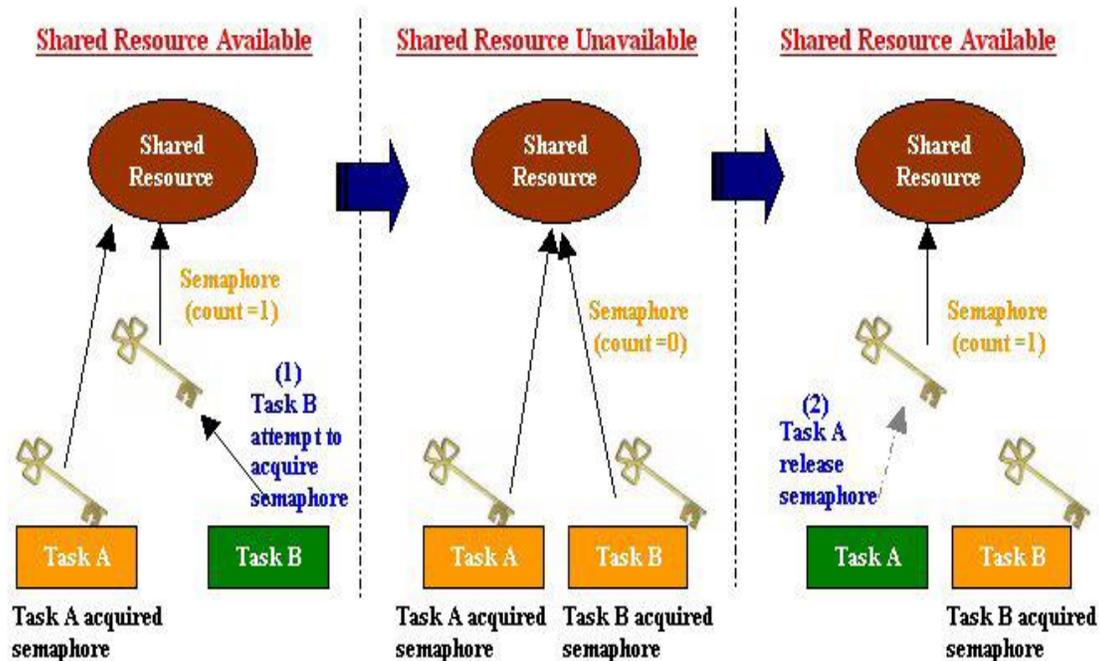
Ans:	<p style="background-color: #000080; color: white; padding: 2px; display: inline-block;">Matrix Keyboard Connection to ports</p>	(Correct labelled diagram-4 M)
b)	Attempt any <u>ONE</u> of the following:	6M
(i)	Describe the methods of task synchronization and explain any one in details.	6M
Ans:	<p><u>The methods of task synchronization are:</u></p> <p style="margin-left: 20px;">Synchronization primitives</p> <ul style="list-style-type: none"> • Semaphore: counting semaphore and binary semaphore A semaphore is created with initial count, which is the number of allowed holders of the semaphore lock. (initial count=1: binary sem) . Sem wait will decrease the count; while sem_signal will increase it. A task can get the semaphore when the count > 0; otherwise, block on it. • Mutex: similar to a binary semaphore, but mutex has an owner. A semaphore can be “waited for” and “signaled” by any task, while only the task that has taken a mutex is allowed to release it. • Spinlock: lock mechanism for multi-processor systems, A task wanting to get spinlock has to get a lock shared by all processors. • Read/write locks: protect from concurrent write, while allow concurrent read Many tasks can get a read lock; but only one task can get a write lock. Before a task gets the write lock, all read locks have to be released. • Barrier: to synchronize a lot of tasks, They should wait until all of them have reached a certain “barrier.” <p>Semaphores: It is a system of sending message by using flags. Multiple concurrent threads of execution with an application must be able to synchronize their execution & co-ordinate mutually exclusive access to shared resources. To fulfill this requirement RTOS kernel provides a semaphore object that one or more threads of execution can acquire or release for the purpose of synchronization or mutual exclusion. Semaphore is</p>	(Task synchronization Methods: 3 M and Any one explanation : 3M)

like a key that allows a test to carry out some operation or to access a resource A kernel supports many different types of semaphores

Binary: Binary semaphores are used for both mutual exclusion and synchronization purposes. A binary semaphore is used to control sharing a single resource between tasks..

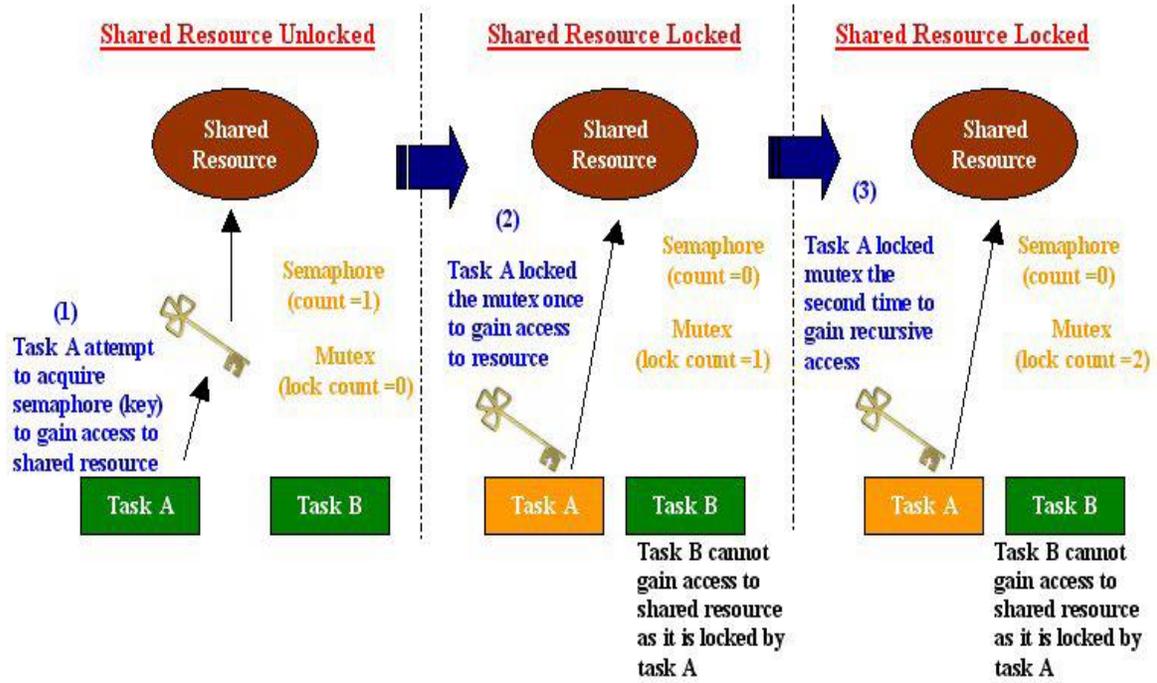
Counting: it is a semaphore that increments when an IPC is given by a task. It decrements when a waiting task unblocks and starts running.

Counting Semaphore



- **Mutex:** Mutexes are binary semaphores that include a priority inheritance mechanism. Mutexes are the better choice for implementing simple mutual exclusion (hence 'MUT'ual 'EX'clusion). A **mutex** allows exclusive access to the resource. The long form is Mutually Exclusion Semaphores (semaphore value of 0 or 1 but lock count can be 0 or greater for recursive locking). A mutex is intended to protect a critical region. Mutex is similar to a binary semaphore, but mutex has an owner. The main difference is that a semaphore can be "waited for" and "signaled" by any task, while only the task that has taken a mutex is allowed to release it.

Mutual Exclusion (Mutex) Semaphore



Example of using semaphores for Synchronization:

Assume two concurrent process P1 and P2 having statements S1 and S2. We want in any case S1 should execute first. this can be achieved easily by initialize Sem=0;

In process P1

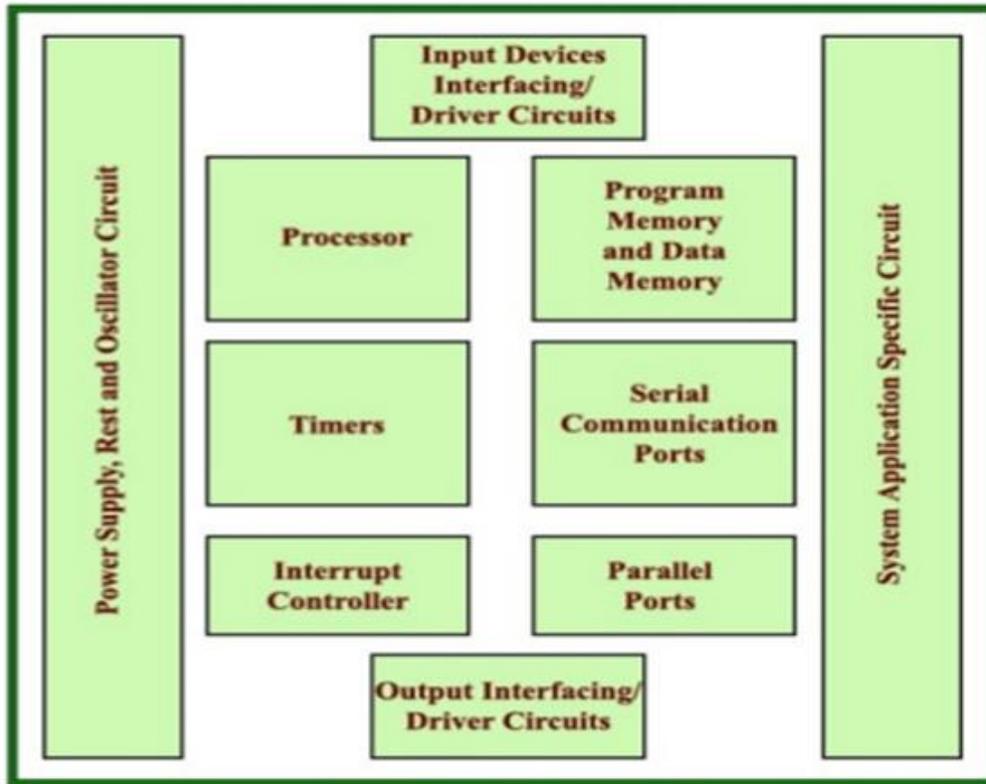
```
{
// Execute whatever you want to do
// before executing P2
```

```
S1;
signal(Sem);
}
```

in process P2

```
{
wait(Sem);
S2;
}
```

Ans:



(Draw-2M,
Explanation-
1/2 M each
unit)

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

2. Power supply, reset & oscillator circuit:

- Most of the systems have their own power supply. Some embedded systems 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.
- 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



		<p>interrupts simultaneously pending for service.</p> <p>6. I/O ports: I/O ports are used to interface external devices like sensors, key buttons, transducers, LEDs, LCD actuators, alarms, motors, valves, 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.</p> <p>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</p> <p>8. System Application specific circuits: These are the circuits that can control specific target circuits. They consist of ADC, DAC, relays, sensors etc.</p>									
Q. 2		Attempt any <u>FOUR</u> of the following:	16M								
	a)	State any four advantages of an embedded systems.	4M								
	Ans:	<p><u>Advantages of an embedded systems-</u></p> <p>1. Design and Efficiency: The central processing core in embedded system is generally less complicated, making it easier to design. The limited function required of embedded system allows them to design to most efficiently perform their function.</p> <p>2. Cost: The streamline make-up of most embedded system allows their parts to be smaller less expensive to produce.</p> <p>3. Accessibility: If something goes wrong with certain embedded systems they can be too inaccessible to repair. This problem is addressed in the design stage, so by programming an embedded system. So that it will not affect related system negatively when malfunctioning.</p> <p>4. Maintenance: Embedded systems are easier to maintain because the supplied power is embedded in the system and does not required remote maintenance.</p> <p>5. Redundancies: Embedded system does not involve the redundant programming.</p>	(Any four-1 M each)								
	b)	Draw the format of TMOD SFR and write significance of each bit.	4M								
	Ans:	<div style="display: flex; justify-content: space-between; width: 100%;"> MSB LSB </div> <table border="1" style="margin: 10px auto; border-collapse: collapse; text-align: center;"> <tr> <td style="padding: 5px;">GATE</td> <td style="padding: 5px;">C/T</td> <td style="padding: 5px;">M1</td> <td style="padding: 5px;">M0</td> <td style="padding: 5px;">GATE</td> <td style="padding: 5px;">C/T</td> <td style="padding: 5px;">M1</td> <td style="padding: 5px;">M0</td> </tr> </table> <div style="display: flex; justify-content: space-around; margin-top: 10px;"> <div style="text-align: center;"> <p>← Timer 1 →</p> </div> <div style="text-align: center;"> <p>← Timer 0 →</p> </div> </div>	GATE	C/T	M1	M0	GATE	C/T	M1	M0	(Format-2M, Description - 2M)
GATE	C/T	M1	M0	GATE	C/T	M1	M0				



GATE : Starts and stops Timer / Counter by means of a signal provided to the pin $\overline{\text{INT1}}$ / $\overline{\text{INT0}}$
1 – Timer / counter operates only if the bit $\overline{\text{INT1}}$ / $\overline{\text{INT0}}$ is set

0 – Timer / Counter operates regardless of the state of the bit $\overline{\text{INT1}}$ / $\overline{\text{INT0}}$

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

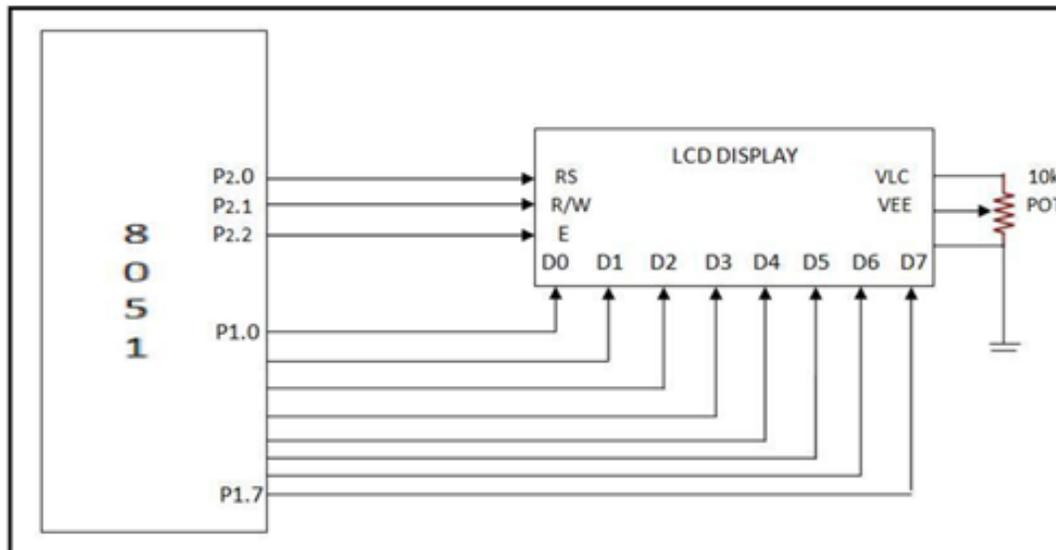
M1, M0 : These two bits selects the Timer operating modes.

M1	M0	Mode	Description
0	0	0	13-bit timer
0	1	1	16-bit timer
1	0	2	8-bit auto-reload
1	1	3	Split mode

- c) Draw the interfacing diagram of 16*2 LCD display with 89C51 and state the function of
- RS
 - EN
 - R/W

4M

Ans:



(Interfacing diagram: 1M, function of each pin: 1 M)



Function:

RS: - RS is used to make the selection between data and command register.

RS=0, command register is selected

RS=1 data register is selected.

RW: -R/W gives you the choice between writing and reading.

R/W=1, reading is enabled.

R/W=0 then writing is enabled.

EN: -Enable pins are used by the LCD to latch information presented to its data pins.

When data is supplied to data pins, a high to low pulse must be applied to this pin in-order for the LCD to latch in the data present at the data pins.

d) Write 89C51 "C" language program to toggle all bits of port P₂ continuously with 500 ms delay.

4M

Ans: *NOTE: Program may change. Student can also use the other logic. Please check the logic and understanding of students.*

(Correct program-4M)

```
#include <reg51.h>
void delay (unsigned int);
void main (void)
{
while(1) //repeat loop
{
P2=0xff; //toggle all bits of port2
delay (500); //add delay
P2=0x00; //toggle all bits of port2
delay (500); //add delay
}
}
void delay (unsigned int i)
{
Unsigned int x, y;
for(x=0; x< i; x++)
for (y=0; y<1275; y++);
}
```

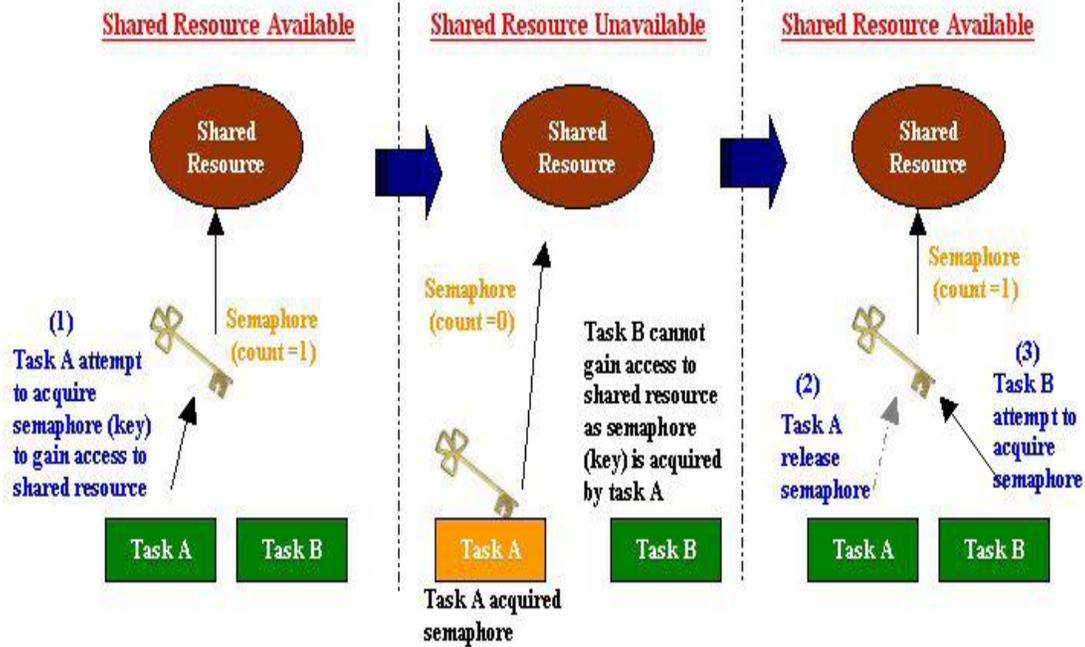
e) Differentiate between CAN and I²C protocols with respect to

- i. Data transfer rate**
- ii. Number of fields**
- iii. Addressing bits**
- iv. Application**



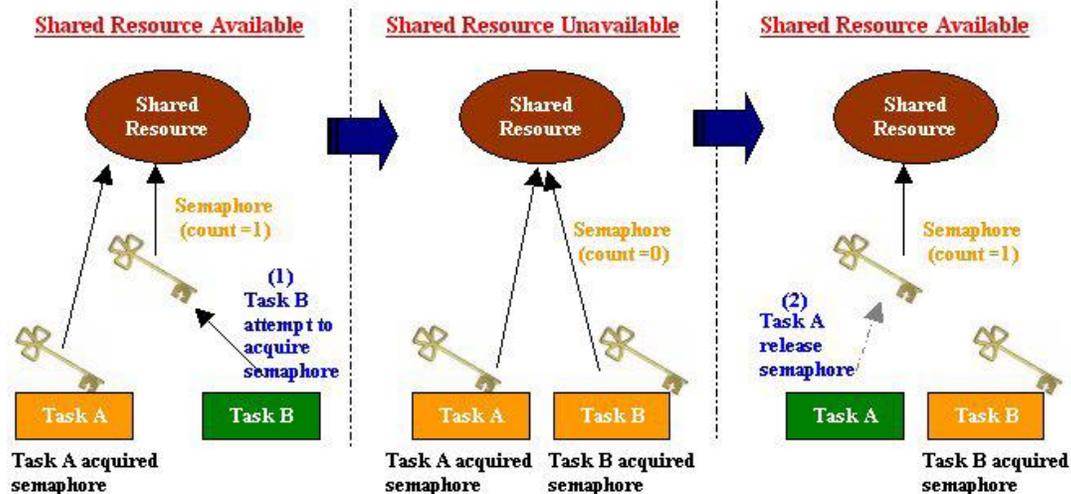
Ans:	<table border="1"><thead><tr><th></th><th>I2C</th><th>CAN</th></tr></thead><tbody><tr><td>Data transfer</td><td>Synchronous with 3 speeds 100kbps, 400kbps and 3.4mbps</td><td>Asynchronous with 250kbps upto 1mbps.</td></tr><tr><td>Number of field</td><td>07</td><td>08 (including 7 bits of frame end and 3 bits of inter frame gap)</td></tr><tr><td>Addressing bit</td><td>7-bit or 10-bit address</td><td>11 bit</td></tr><tr><td>application</td><td>To interface devises like watch dog, flash &RAM memory, Real time clock, Microcontrollers</td><td>elevator controllers, copiers, telescopes, production-line control systems, and medical instruments</td></tr></tbody></table>		I2C	CAN	Data transfer	Synchronous with 3 speeds 100kbps, 400kbps and 3.4mbps	Asynchronous with 250kbps upto 1mbps.	Number of field	07	08 (including 7 bits of frame end and 3 bits of inter frame gap)	Addressing bit	7-bit or 10-bit address	11 bit	application	To interface devises like watch dog, flash &RAM memory, Real time clock, Microcontrollers	elevator controllers, copiers, telescopes, production-line control systems, and medical instruments	(1M each point)
		I2C	CAN														
	Data transfer	Synchronous with 3 speeds 100kbps, 400kbps and 3.4mbps	Asynchronous with 250kbps upto 1mbps.														
	Number of field	07	08 (including 7 bits of frame end and 3 bits of inter frame gap)														
	Addressing bit	7-bit or 10-bit address	11 bit														
application	To interface devises like watch dog, flash &RAM memory, Real time clock, Microcontrollers	elevator controllers, copiers, telescopes, production-line control systems, and medical instruments															
f)	Describe semaphore with suitable example.	4M															
Ans:	<p>Semaphores: It is a system of sending message by using flags. Multiple concurrent threads of execution with an application must be able to synchronize their execution & co-ordinate mutually exclusive access to shared resources. To fulfill these requirement RTOS kernel provides a semaphore object that one or more threads of execution can acquire or release for the purpose of synchronization or mutual exclusion. Semaphore is like a key that allows a test to carry out some operation or to access a resource A kernel supports many different types of semaphores</p> <p>Binary: Binary semaphores are used for both mutual exclusion and synchronization purposes. A binary semaphore is used to control sharing a single resource between tasks. Its internal counter can have only the values of 1 (available) and 0 (unavailable). A semaphore test passes if the count is 1, in which case, the current task is allowed to proceed.</p>	(Description of semaphore - 3M, Example-1 M)															

Binary Semaphore



Counting: it is a semaphore that increments when an IPC is given by a task. It decrements when a waiting task unblocks and starts running.

Counting Semaphore



Mutex: Mutexes are binary semaphores that include a priority inheritance mechanism. Mutexes are the better choice for implementing simple mutual exclusion (hence 'MUT'ual 'EX'clusion). When used for mutual exclusion the mutex acts like a token that is used to guard a resource. When a task wishes to access the resource it must first obtain ('take') the



token. When it has finished with the resource it must 'give' the token back - allowing other tasks the opportunity to access the same resource.

Example of using semaphores for Synchronization:

Assume two concurrent processes P1 and P2 having statements S1 and S2. We want in any case S1 should execute first. this can be achieved easily by initialize Sem=0;

In process P1

```
{  
// Execute whatever you want to do  
// before executing P2  
S1;  
signal(Sem);  
}
```

in process P2

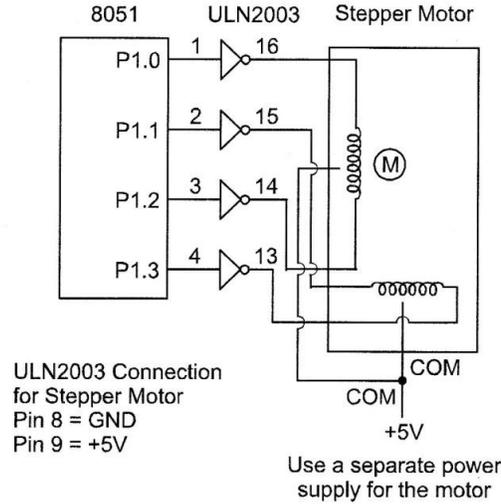
```
{  
wait(Sem);  
S2;  
}
```

Q. 3	Attempt any FOUR of the following:	16M
a)	Find the content of Accumulator after execution of the following code i. ACC = 0*94>>5; ii. ACC = 0*5A<<2;	4M
Ans:	i) ACC= 0*94>>5 Ans: ACC= 0*04 ii) ACC= 0*5A<<2 Ans: ACC= 0*68	2 M each
b)	Write any four feature of RTOS.	4M
Ans:	<ul style="list-style-type: none">• Reliability: A reliable system is the system which is available all the time. A system that does not fail is a reliable system. RTOS is reliable but it does not guarantee the reliability of an embedded system. The reliability of an ES depends on the hardware, the application code.• Predictability: RTOS have predictable behavior in which the completion of OS calls occurs within known time frame.• Performance: The system must perform fast enough to fulfil the timing requirements. The performance of RTOS can be measured on a call-by-call basis. Time stamps are produced when a system call starts and when it completes. This method of analyzing is useful in designing stage but the true performance is measured as whole.• Compactness: RTOS used in embedded system are extremely constraint as far as resources are concerned. The design requirements limit the system memory which limits the size of the application and the operating system.• Scalability: RTOS are most used in variety of embedded system they must be able to scale up/down to suit the application.	1M each
c)	Write 89C51 "C" language program to rotate stepper motor by 180 ⁰ in clockwise direction motor has step angle 1.8 ⁰ . Use stepper moter of 4 step pulse sequence.	4M



Correct
program-
4M

Ans:



Step Angle= 1.8°

Total steps required= $\frac{180}{1.8} = 100$

**NOTE: Program may change. Student can also use the other logic.
Please check the logic and understanding of students.**

```
#include <reg51.h>
#define stepper P1
Void delay (unsigned int);
Void main ()
{
  Unsigned int i;
  for (i=1; i<=25;i++)
  {
    stepper = 0X0C;
    delay (10);
    stepper = 0X06;
    delay (10);
    stepper = 0X03;
    delay (10);
    stepper = 0X09;
    delay (10);
  }
}
Void delay (unsigned int k)
{
  Unsigned int x, y;
  for(x=0; x< k; x++)
  for (y=0; y<1275; y++);
}
```

d) Write any four characteristics of an embedded system.

4M

Ans: 1. **Processing Power:** Selection of processor is based on the amount of processing power to get the job done and also on the basis of register width required.

(Any 4
1M each)



		<ol style="list-style-type: none"> 2. Throughput: The system may need to handle a lot of data in a short time. 3. Response: The system has to react to the changing events quickly. 4. Memory: Hardware design must make the best estimate of the memory requirement and must make the provision for expansion. 5. Power consumption: Systems generally work on battery and design of both software and hardware must take care of power saving techniques. 6. Number of units: The number of units expected to be produced and sold will dictate the trade-off between production cost and development cost. 7. Expected life-time: Design decisions like selection of components to system development cost will depend upon on how long the system is expected to run. 8. Program Installation: Installation of software on to the embedded system needs special development tools. 9. Testability and Debug ability: Setting up test conditions and equipment will be difficult and determining what is wrong with the software will become a difficult task without a keyboard and usual display. 10 Reliability: It is always required that the system designed must give the output for which it is designed. 	
	e)	Write any four feature of USB.	4M
	Ans:	<ol style="list-style-type: none"> 1. Multiple device connection: Upto 127 different devices can be connected on single USB bus. 2. Transfer rate: The initial USB supported 12 MBps transfer rate where USB 2.0 supports higher rate currently 60 MB/sec. 3. Support for large range of peripherals: Low bandwidth devices such as keyboard, mouse, joystick, and game -port, FDD. 4. Hub architecture: The devices are not daisy chained. Each device is connected to an USB hub. The USB hub interacts with PC on one side and peripheral on other side. 5. Plug ability: The USB device can be connected without powering off a PC i.e. plug and play feature in BIOS together with the device takes care of detection, handling and device recognition. 6. Power allocation: USB controller in the PC detects the presence or absence of the USB devices and does the allocation of power. 7. Ease of installation: There is only one cable. A 4-pin cable carries signals like power signal (-), signal (+), ground. 8. Host centric: The CPU software initiates every transaction on the USB bus. Hence the overhead on the PC increases when there are large number of peripherals involving large number of transactions. 	(Any 4 1 M each)
Q. 4	A)	Attempt any <u>THREE</u> of the following:	12M
	i)	Explain I²C protocols with suitable diagram.	4M

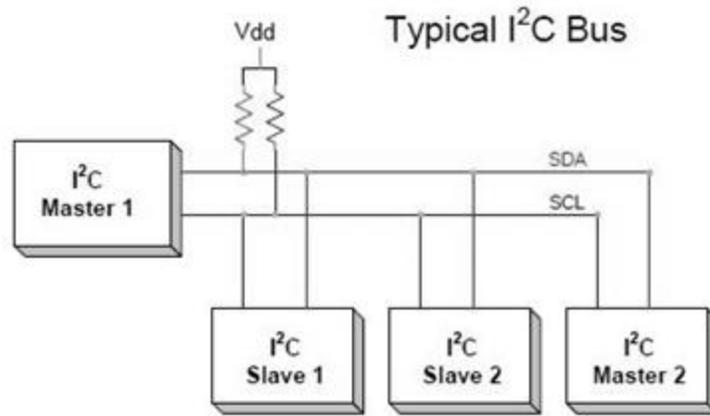
Ans: **I²C Bus:** In any process plant there are n numbers of device circuits that are used for measurement of temperature, pressure and it is required to connect these ICs through a common bus. For this I²C has become a standard. There are three standards:

- Industrial 100 kbps I²C.
- 100 kbps SM I²C.
- 400 kbps I²C.

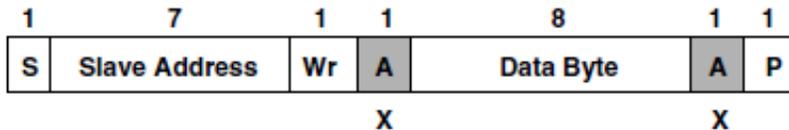
(Explanation: 2M, format: 2M)

The I²C bus has two lines that carry its signals. One line is for the clock and another is for the bi-directional data. Serial data line [SDA] and Serial Clock [SCL]. The voltages used are +5 V and +3.3 V.

There is a protocol for the I²C bus.



The format of bits at the I²C bus is as shown:



- A** Acknowledge (this bit position may be 0 for an ACK or 1 for a NACK)
- P** Stop Condition
- Rd** Read (bit value of 1)
- S** Start Condition
- Sr** Repeated Start Condition
- Wr** Write (bit value of 0)
- X** Shown under a field indicates that that field is required to have a value of X
- ... Continuation of protocol
- Master-to-Slave
- Slave-to-Master



Field and its length	Description
1 st field of 1 bit	It is like a start bit in UART
2 nd field of 7 bits	It is address field. It defines the slave address, which is being sent the data frame by the master.
3 rd field of 1 control bit	It defines whether a R/W cycle is in progress.
4 th field of 1 control bit	Defines whether the present data is an acknowledgement.
5 th field of 8 bits	It is for the IC device data byte.
6 th field of 1-bit	It is a NACK (No Acknowledgement). If active then ACK is not needed from the slave, else ACK is needed.
7 th field of 1 bit	Similar to STOP bit in an UART.

I²C Bus reference design has 7 bit address space with 16 reserved addresses, so a maximum 112 nodes can communicate on the same bus.

Each device has a unique address using which the data transfer is carried out. The master can address have 127 slaves at an instance.

It has a processing element functioning as a bus controller or a microcontroller with I²C bus interface circuit.

So each slave must have an I²C bus controller and processing element. A number of masters can also connect to bus but there can be only one master which can initiate data transfer on SDA line and which can transmit the SCL pulses.

Common I²C Bus speeds are 100 Kbps standard mode and 10 Kbps low speed mode.

Advantages:

1. Multiple slave devices can be accessed with only 2 wires.
2. Low cost to implement.
3. Implemented in hardware and software.
4. Ease to implement.
5. Supports multi-master configuration.

Dis-Advantages:

1. Short distance.
2. Slow speed.
3. Limited device addresses.

Applications of I²C Bus: I²C Bus is used for peripherals where simplicity and low manufacturing cost are more important than speed.

1. Supporting systems management for PCI cards.
2. Accessing NVRAM chips that keep user settings.
3. Accessing low speed DAC's and ADC's.
4. Changing contrast, hue and colour balance settings in monitors.
5. Changing sound volume in intelligent speakers.

Controlling OLED/OLCD displays in cell phones

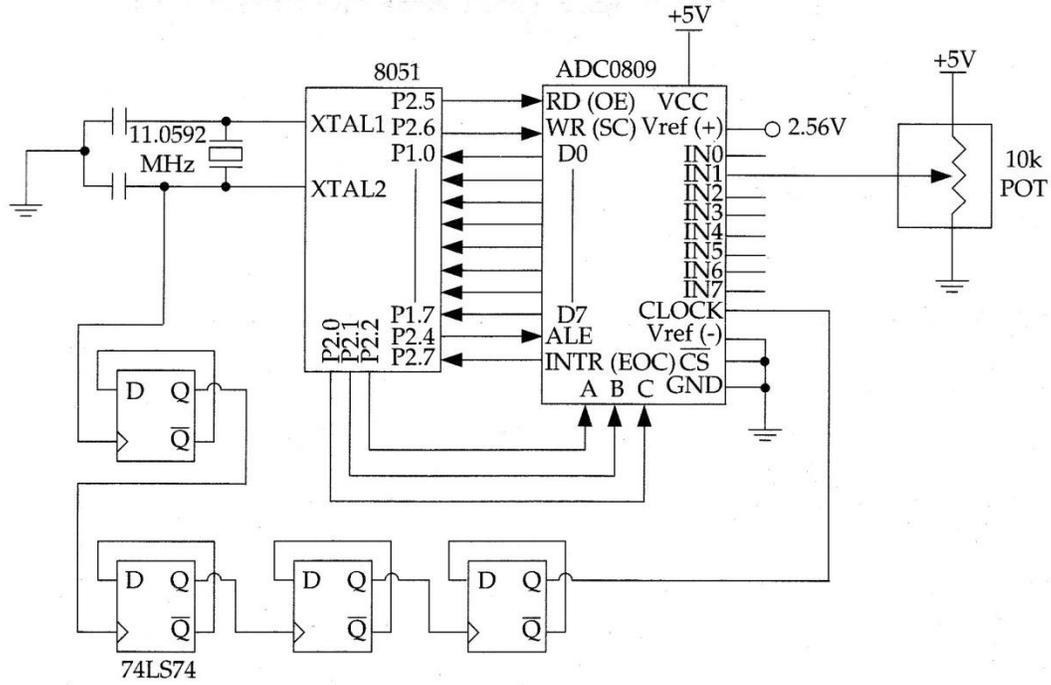
ii)	Differentiate between General purpose operating system and RTOS (any four points).	4M
-----	---	-----------



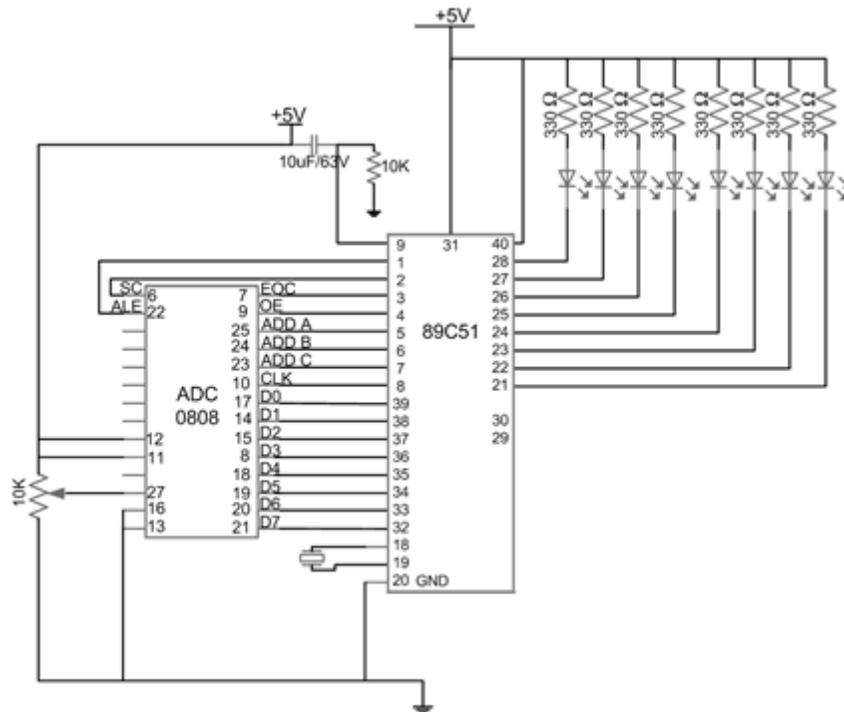
Ans:	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Sr no</th> <th style="width: 40%;">OS</th> <th style="width: 50%;">RTOS</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Non-Deterministic time behaviour.</td> <td>Deterministic time behaviour.</td> </tr> <tr> <td>2</td> <td>Used in general desktop computer system.</td> <td>Used in embedded system</td> </tr> <tr> <td>3</td> <td>Generalized Kernel.</td> <td>Real time kernel.</td> </tr> <tr> <td>4</td> <td>OS services can inject random delays into application software, may cause slow responsiveness of an application at unexpected time.</td> <td>OS services consumes only known and expected amounts of time regardless the number of services.</td> </tr> <tr> <td>5</td> <td>Ex: Windows XP, MS-DOS</td> <td>Ex: Windows CE.</td> </tr> </tbody> </table>	Sr no	OS	RTOS	1	Non-Deterministic time behaviour.	Deterministic time behaviour.	2	Used in general desktop computer system.	Used in embedded system	3	Generalized Kernel.	Real time kernel.	4	OS services can inject random delays into application software, may cause slow responsiveness of an application at unexpected time.	OS services consumes only known and expected amounts of time regardless the number of services.	5	Ex: Windows XP, MS-DOS	Ex: Windows CE.	(Any 4 point:1M for each point)
Sr no	OS	RTOS																		
1	Non-Deterministic time behaviour.	Deterministic time behaviour.																		
2	Used in general desktop computer system.	Used in embedded system																		
3	Generalized Kernel.	Real time kernel.																		
4	OS services can inject random delays into application software, may cause slow responsiveness of an application at unexpected time.	OS services consumes only known and expected amounts of time regardless the number of services.																		
5	Ex: Windows XP, MS-DOS	Ex: Windows CE.																		
iii)	Differentiate between RISC and CISC computer.	4M																		
Ans:	<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">Sr no.</th> <th style="width: 40%;">RISC: Reduced Instruction Set Computer</th> <th style="width: 50%;">CISC: Complex Instruction Set Computer</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Emphasis on software.</td> <td>Emphasis on hardware</td> </tr> <tr> <td>2</td> <td>Single clock, reduced instruction.</td> <td>Multiclock, complex instruction set.</td> </tr> <tr> <td>3</td> <td>Register-Register, load and store are independent of instruction.</td> <td>Memory-Memory, load and store are incorporated.</td> </tr> <tr> <td>4</td> <td>Large code size.</td> <td>Small code size.</td> </tr> <tr> <td>5</td> <td>Spends more transistors on memory registers.</td> <td>Transistors used for storing complex instructions.</td> </tr> </tbody> </table>	Sr no.	RISC: Reduced Instruction Set Computer	CISC: Complex Instruction Set Computer	1	Emphasis on software.	Emphasis on hardware	2	Single clock, reduced instruction.	Multiclock, complex instruction set.	3	Register-Register, load and store are independent of instruction.	Memory-Memory, load and store are incorporated.	4	Large code size.	Small code size.	5	Spends more transistors on memory registers.	Transistors used for storing complex instructions.	(1M for each point)
Sr no.	RISC: Reduced Instruction Set Computer	CISC: Complex Instruction Set Computer																		
1	Emphasis on software.	Emphasis on hardware																		
2	Single clock, reduced instruction.	Multiclock, complex instruction set.																		
3	Register-Register, load and store are independent of instruction.	Memory-Memory, load and store are incorporated.																		
4	Large code size.	Small code size.																		
5	Spends more transistors on memory registers.	Transistors used for storing complex instructions.																		
iv)	Write the definition of an embedded system? How it is classified?	4M																		
Ans:	<p>An embedded system is a computer system with a dedicated function within a larger mechanical or electrical system, often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts. Embedded systems control many devices in common use today.</p> <div style="text-align: center;"> <p>Types Of Embedded Systems</p> <pre> graph TD A[Types Of Embedded Systems] --> B[Based On Performance & Functional Requirements] A --> C[Based On The Performance Of The Microcontroller] B --> D[Real Time] B --> E[Stand Alone] B --> F[Networked] B --> G[Mobile] C --> H[Small Scale] C --> I[Medium Scale] C --> J[Sophisticated] </pre> </div>	(Define:1 M Classification :3M)																		
b)	Attempt any ONE of the following:	6M																		
i)	Draw interfacing diagram of ADC 0808 with 89C51 micro controller and write 'C' language program to read data from ADC 0808.	6M																		



Ans:



OR



(Diagram:
3 M
Program:
3M)



```
#include <reg51.h>
sbit ALE = P2^4;
sbit OE = P2^5;
sbit SC = P2^6;
sbit EOC = P2^7;
sbit ADDR_A = P2^0;
sbit ADDR_B = P2^1;
sbit ADDR_C = P2^2;
sfr MYDATA = P1;
void main()
{
    unsigned char value;
    MYDATA = 0xFF; //make P1 an input
    EOC = 1; //make EOC an input
    ALE = 0; //clear ALE
    OE = 0; //clear OE
    SC = 0; //clear SC
    while(1)
    {
        ADDR_C = 0; //C=0
        ADDR_B = 0; //B=0
        ADDR_A = 1; //A=1 (Select Channel 1)
        MSDelay(1); //delay for fast DS89C4x0
        ALE = 1;
        MSDelay(1);
        SC = 1;
        MSDelay(1);
        ALE = 0;
        SC = 0; //start conversion
        while(EOC==1); //wait for data conversion
        while(EOC==0);
        OE = 1; //enable RD
        MSDelay(1);
        value = MYDATA; //get the data
        OE = 0; //disable RD for next round
    }
}
```

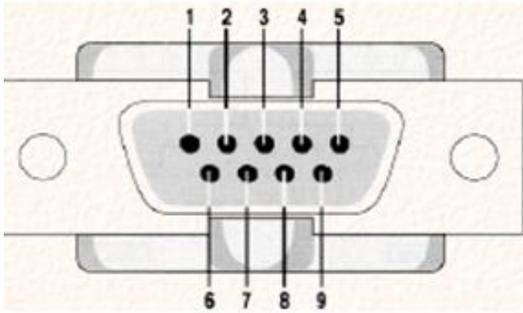
ii) Write 89C51 'C' program to transfer 'YES' serially at baud rate 9600 continuously
Use 8 bit data and 1 stop bit. Assume crystal frequency 11.0592MHz.

6M



	Ans:	$clock = \frac{11.0592 \times 10^6}{12} = 921.6KHz$ $UART = \frac{921.6 \times 10^3}{32} = 28800Hz$ <p>For Baud rate of 9600: $n = \frac{28800}{9600} = 3 = (-3) = FDH$</p> <p>program</p> <pre>#include <reg51.h> void SerTx(unsigned char); void main(void) { TMOD=0x20; // use Timer 1, mode 2 TH1=0xFD; // 9600 baud rate SCON=0x50; // 01010000 Serial mode1 TR1=1; // start timer while (1) { SerTx('Y'); SerTx('E'); SerTx('S'); } // End of While } void SerTx(unsigned char x) { SBUF=x; // place value in buffer while (TI==0); // wait until transmitted TI=0; // Clear TI Flag }</pre>	(Calculation: 2M Program: 4M)
Q.5		Attempt any <u>FOUR</u> of the following:	16M
	a)	Draw the pin out of RS232 and describe the function of TXD, RXD, DTE and DCE.	4M
	Ans:		(2M for Diagram with labels ½ M for

RS232 Connector DB-9



RS232 DB-9 Pins

Pin	Description
1	Data carrier detect (-DCD)
2	Received data (RxD)
3	Transmitted data (TxD)
4	Data terminal ready (DTR)
5	Signal ground (GND)
6	Data set ready (-DSR)
7	Request to send (-RTS)
8	Clear to send (-CTS)
9	Ring indicator (RI)

each functions)

TXD:It is used to transmit the data.

RXD:It is used to receive the data.

DTE: Data Terminal equipment which send and receive the data

DCE: Data communication equipment such as modem which is responsible for transferring the data.

b) Write 'C' language program to generate a triangular waveform of DAC 0808.

4M

Ans: *Note : Any ports pin can be defined for handshaking signals. Xfer/wr signal may be combined .*

(2M for +ve ramp and 2M for -ve ramp.. Any logic can be used.)

```
#include<reg51.h>
unsigned char d;
void main(void)
{
while(1)
{
for(d=0; d<255; d++)
{
P1 = d;
}
for(d=255; d>0; d--)
{
P1 = d;
}
}
}
```



OR

```
#include <reg51.h>
Sbit cs = P3^3 ;
Sbit wr = P3^4 ;
Sbit xfer = P3^5 ;
void main( )
{
    unsigned Char x,y;
    cs = 0 ;
    while (1)
    {
        for ( x=0; x<255;x++)
            { wr = 1 ;
            xfer = 1 ;
            P1 = x ;
            xfer = 0 ;
            wr = 0 ;
            }

        for ( y=255; y>0;y--)
            { wr = 1 ;
            xfer = 1 ;
            P1 = y ;
            xfer = 0 ;
            wr = 0 ;
            }
    }
}
```

c) **Describe deadlock in RTOS with suitable example.**

4M

Ans: A deadlock consists of a set of blocked processes, each holding a resource and waiting to acquire a resource held by another process in the set
A deadlock, also called as deadly embrace, is a situation in which two threads are each unknowingly waiting for resource held by other.

**(Description- 3 M,
Any one example- 1 M)**

Example #1

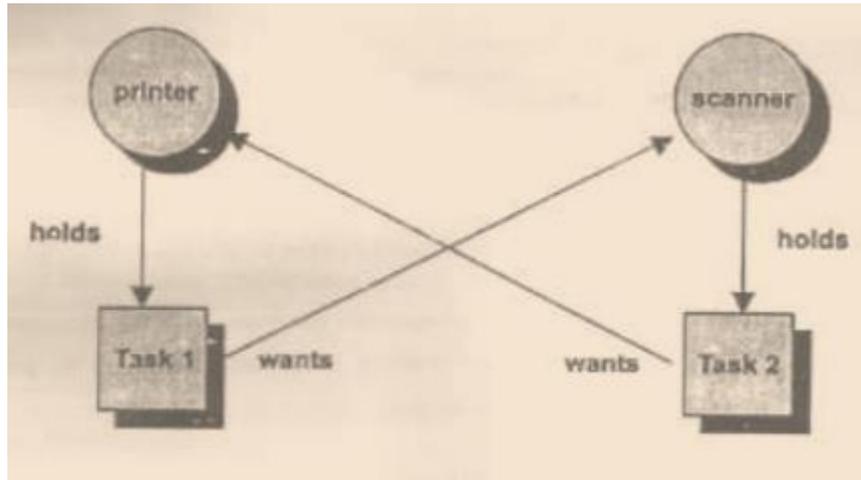
A system has 2 disk drives
 P_1 and P_2 each hold one disk drive and each need the other one.

Example #2

Semaphores A and B, initialized to 1

P_0		P_1
wait (A);		wait(B)
wait (B);		wait(A)

Example 3

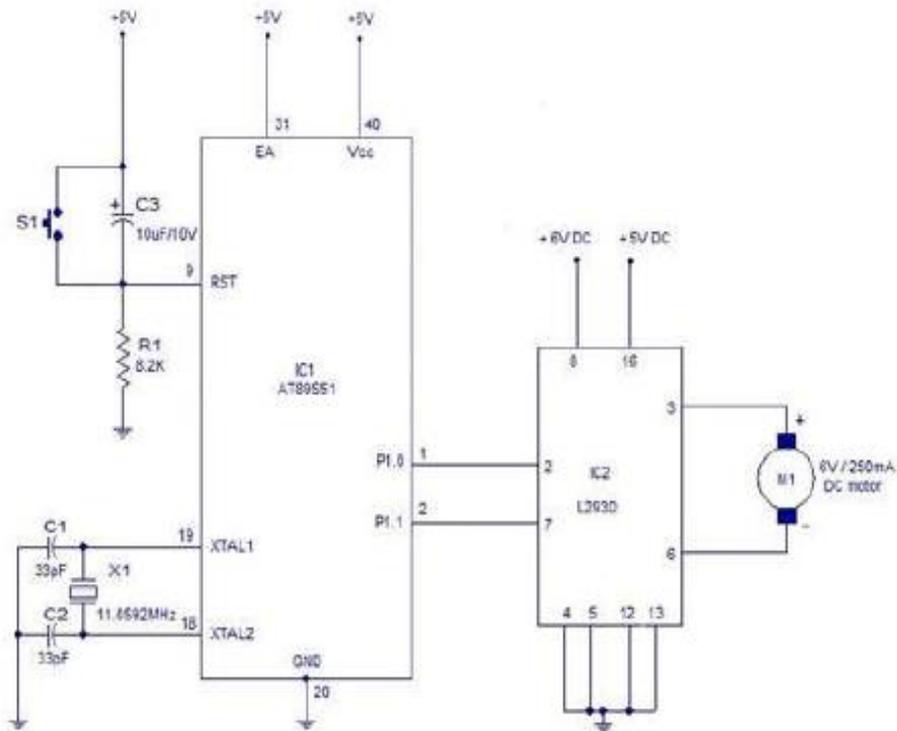


- Task #1 wants the scanner while holding the printer. Task #1 cannot proceed until both the printer and the scanner are in its possession.
- Task #2 wants the printer while holding the scanner. Task #2 cannot continue until it has the printer and the scanner.

d) Draw labelled interfacing diagram of DC Motor with 89C51 microcontroller.

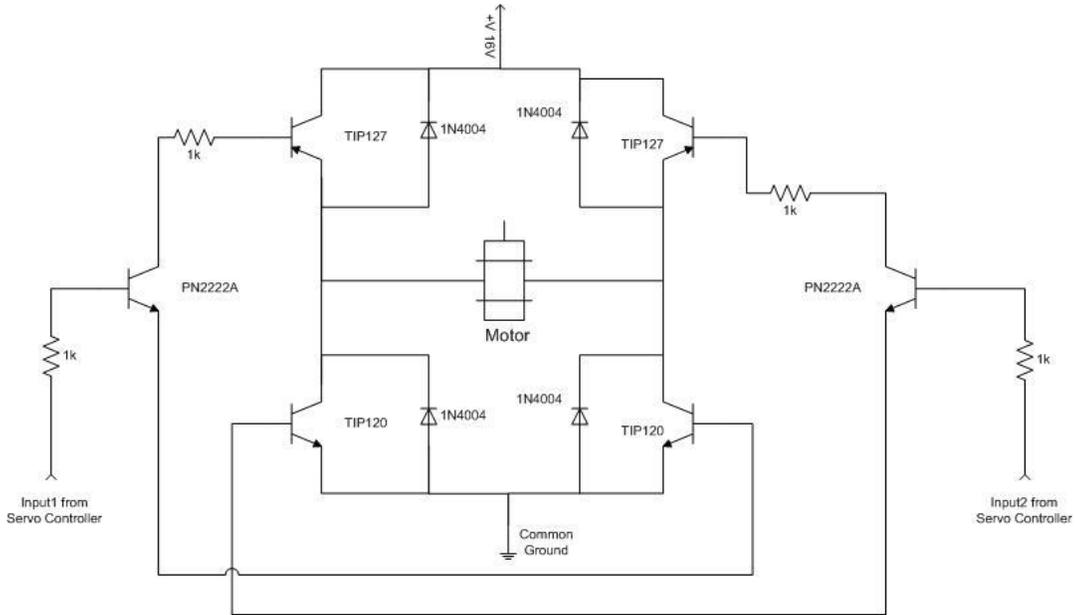
4M

Ans:



(Labelled diagram - 4M)

OR
H bridge with Transistor



e) Write 'C' language program to mask the lower 4 bits of port P₀ and upper 4 bits of port P₂ using logical operator.

4M

Ans:

```
#include <stdio.h>
#include <reg51.h>
void main( )
{
    P0 = P0 &0xF0;
    P2=P2&0x0F;
    While(1);
}
```

(2M for each correct syntax of masking.)

f) Describe program downloading tools. ISP and IAP.

4M

- Ans:**
- In system Programming (ISP): Programming is done 'within the system' i.e. firmware (IEEE 1394 is wired isochronous high speed serial communication bus) is embedded into the target device without removing it from the target board.
 - The target device must have an ISP support. No additional hardware is required.
 - Chips supporting ISP generates the necessary programming signal internally using chip's supply voltage.
 - Target board is interfaced to utility program running on PC through Serial/Parallel/USB port
 - Serial Protocol for ISP: JTAG, SPI(serial peripheral interface).
 - In Application Programming (IAP):IAP is the technique running on the target device for modifying a selected portion of the code memory.

(2M for each)



		<ul style="list-style-type: none"> This technique is not used for first time embedding of user written firmware. It modifies the program code memory under the control of embedded applications Examples: Updating calibration data, look up tables ,Boot ROM etc. in code memory. 	
Q.6		Attempt any <u>FOUR</u> of the following:	16M
	a)	Describe parallel protocols PCI, PCI-X.	4M
	Ans:	<p><u>PCI:</u> PCI stands for Peripheral Component Interconnect/Interface bus. It is popular for higher bandwidth processor independent which can function as peripheral bus. It is introduced by Intel in 90's It is 32 bit local bus and extended up to 64 bit by processor it requires. It has high speed I/O subsystem performance. The PCI is designed to meet economically the i/o requirement of modern system. It supports ten i/o devices and provides 3 types of synchronous parallel interface It has two versions: 32 bit (33 MHz) ,64 bit (66 MHz) The data transfer rate for synchronous is 132 mbps and for asynchronous it is 528 mbps. The PCI driver can access hardware automatically or by programmer can assign address. The automatic detection and assignment of addresses of various devices simplifies the addition and removal of the system peripheral. PCI is designed to support variety of microprocessor best configuration including single and multi-processing system.</p> <p><u>PCI -X</u> It is an extension of PCI bus and supports 64 bit, 100MHz transfer. PCI – X is revised to double the maximum clock speed to improve the data exchange transfer between processor and peripherals. The data exchange rate is 1.06 gbps.</p>	(2 M for PCI and 2 M for PCI –X)
	b)	Draw labelled interfacing diagram of seven segment display with 89C51 micro controller.	4M
	Ans:	<p><i>Note: Student may draw the interfacing in multiplexed mode with more than one display. Common anode/cathode configuration can be used.</i></p>	(4M for correct label diagram)



c)	State all logical operators used in 'C' and explain any one with example.	4M																		
Ans:	<table border="1" data-bbox="321 262 1328 632"> <tr> <td>~</td> <td>Not</td> <td>P0=~P0 ;P0=0xff will give P0=0x00</td> </tr> <tr> <td>^</td> <td>XOR</td> <td>ACC ^ P1 = 0 x 02 ^ 0 xF3 = 0 x F1</td> </tr> <tr> <td> </td> <td>OR</td> <td>ACC P1 = 0 x 02 0 xF3 = 0 x F3</td> </tr> <tr> <td>&</td> <td>AND</td> <td>ACC & P1 = 0 x 02 & 0 xF3 = 0 x 02</td> </tr> <tr> <td>></td> <td>Right shift</td> <td>P0= P0>>1 ;P0=0x01 will give 0x00;</td> </tr> <tr> <td><</td> <td>Left shift</td> <td>P0= P1<<1 ;P0=0x01 will give 0x02;</td> </tr> </table>	~	Not	P0=~P0 ;P0=0xff will give P0=0x00	^	XOR	ACC ^ P1 = 0 x 02 ^ 0 xF3 = 0 x F1		OR	ACC P1 = 0 x 02 0 xF3 = 0 x F3	&	AND	ACC & P1 = 0 x 02 & 0 xF3 = 0 x 02	>	Right shift	P0= P0>>1 ;P0=0x01 will give 0x00;	<	Left shift	P0= P1<<1 ;P0=0x01 will give 0x02;	(Stating : ½ M each, Any one with example: 1 M)
~	Not	P0=~P0 ;P0=0xff will give P0=0x00																		
^	XOR	ACC ^ P1 = 0 x 02 ^ 0 xF3 = 0 x F1																		
	OR	ACC P1 = 0 x 02 0 xF3 = 0 x F3																		
&	AND	ACC & P1 = 0 x 02 & 0 xF3 = 0 x 02																		
>	Right shift	P0= P0>>1 ;P0=0x01 will give 0x00;																		
<	Left shift	P0= P1<<1 ;P0=0x01 will give 0x02;																		
d)	Draw interfacing diagram of LED to port pin P2.4 of 89C51 write 'C' language program to turn ON and OFF LED after 20 ms delay.	4M																		
Ans:	<pre data-bbox="232 705 690 1549"> #include<reg51.h> sbit led=P2^4; //make P2.4 as LED void delay (void); void main () { led=1; delay(); led=0; while(1); } void delay (void) { TMOD= 0x10; TH1=0xDC; TL1=0x00; TR1=1 while(TF1==0); TR1=0; TF1=0; } </pre> <p style="text-align: center;"><u>OR</u></p> <pre data-bbox="232 1696 690 1982"> #include<reg51.h> sbit led=P2^4; //make P2.4 as LED void delay(unsigned int); void main () { led=1; </pre>	(2M for Program ,2 M for interfacing only LED with Port pin is expected.)																		

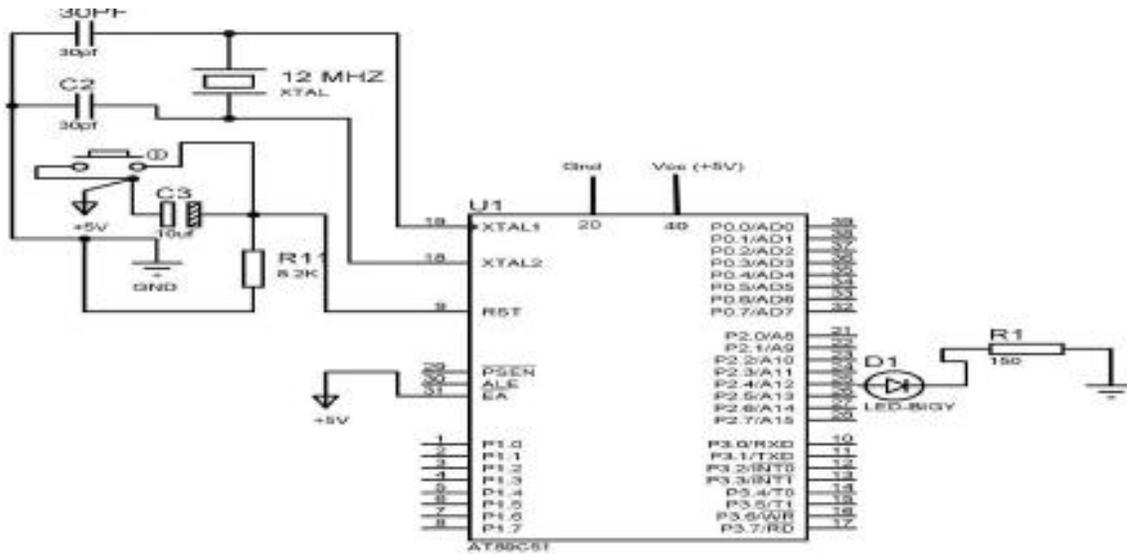


```
delay(20);  
led=0;  
while(1);  
}  
  
void delay (unsigned int itime)  
{  
  unsigned intx,y;  
  for(x=0; x<itime; x++)  
  for (y=0; y<1275; y++);  
}
```

OR

Note : If student has written program to on-off LED continuously, marks can be given

```
#include<reg51.h>  
sbit led=P2^4; //make P2.4 as LED  
void delay( unsigned int);  
  
void main ()  
{  
  While(1)  
  {  
    led=1;  
    delay(20);  
    led=0;  
    delay(20);  
  }  
}  
  
void delay (unsigned int itime)  
{  
  unsigned intx,y;  
  for(x=0; x<itime; x++)  
  for (y=0; y<1275; y++);  
}
```



e) List Data types used in 'C' with their values.

4M

Data Type	Size in bits	Data Range/ Usage
Unsigned Char	8	0 to 255
Signed Char	8	-128 to +127
Unsigned int	16	0 to 65536
Signed int	16	-32768 to +32767
float	32	+1.175494E-38 to 3.402823E+38
bit	1	0 or 1 bit addressable RAM
sbit	1	SFR bit addressable
sfr	8	RAM addresses 80 to FF only
sfr16	16	0 to 65536 (16 bit SFR only)

(Any four
Each -
1M)