



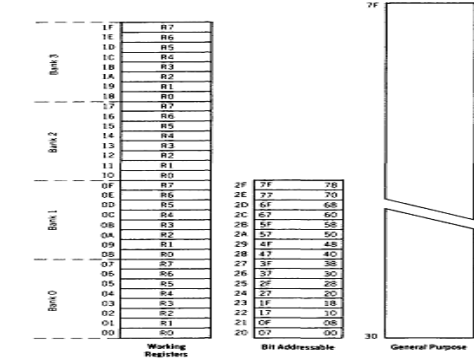
**MODEL ANSWER**  
**SUMMER- 18 EXAMINATION**

**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 any equivalent figure drawn.
- 5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
- 6) In case of some questions credit may be given by judgement on part of examiner of relevant answer based on candidate's understanding.
- 7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q. No.	Sub Q.N.	Answer	Marking Scheme
Q.1	a)	Attempt any <b>THREE</b> of the following;	12-Total Marks
	i)	Draw internal RAM organisation of 89C51 microcontroller. Explain register banks in it.	4 Marks
	Ans:	<p style="text-align: center;">Internal RAM Organization</p>  <p>The 89c51 has 128 byte internal RAM and divided into 3 parts/area.</p> <p><u>In first part :</u></p> <p>32 bytes from addresses 00H to 1FH i.e. 32 registers organised in 4 banks. Each bank contains 8 registers (each of 8 bit) and available at each address.</p> <p>Registers are named as:</p>	<p><b>Labeled diagram 2 Marks</b></p> <p><b>Description of Registration bank 2 Marks</b></p>



	<p>Bank 0 –R0, R1, R2, R3, R4, R5, R6, R7</p> <p>Bank 1 –R0, R1, R2, R3, R4, R5, R6, R7</p> <p>Bank 2 –R0, R1, R2, R3, R4, R5, R6, R7</p> <p>Bank 3 –R0, R1, R2, R3, R4, R5, R6, R7</p> <p>If register banks are not selected, it can be used as general purpose area.</p> <p>On reset Bank 0 is selected.</p> <p>RS1 and RS0 bits (D3 and D4) of PSW register are used to select the register bank.</p> <p><b>Note : Others parts of RAM not expected .</b></p>	
ii)	<b>State the function of simulator, linker compiler and debugger.</b>	<b>4 Marks</b>
Ans:	<p><b>Simulator:-</b></p> <ol style="list-style-type: none"><li>1. It is the software that functions like the hardware without actual hardware.</li><li>2. Allows simulation of peripherals and I/O devices.</li><li>3. Allows checking of software before the hardware is available to the user.</li></ol> <p><b>Linker :-</b></p> <ol style="list-style-type: none"><li>1. Linker is used to link with the library and generation of executable file.</li><li>2. It is used for relocation process.</li><li>3. It is done during compilation also it can be done at run time by a relocating loader.</li><li>4. It is a program that takes one or more objects generated by compiler and combines them into a single executable program.</li></ol> <p><b>Compiler :-</b></p> <ol style="list-style-type: none"><li>1. It is program which converts high level language program to machine language.</li><li>2. It also indicates the syntax errors in the program if any.</li><li>3. It generates object file corresponding to the target device.</li></ol> <p><b>Debugger :-</b></p> <ol style="list-style-type: none"><li>1. Debugger is used to find the error and it keeps the control over the system environment and ability to test or follow the execution of the program.</li><li>2. Debugger allows the user to load program in to the system memory, executes the</li></ol>	<p><b>Only one function of each :</b></p> <p><b>1 Mark</b></p>



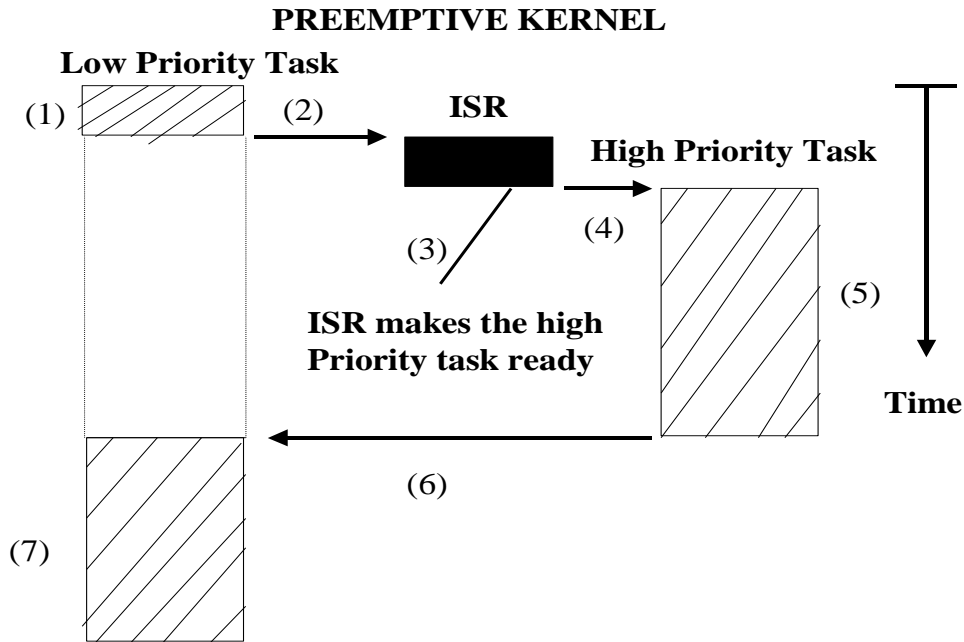
	program by single stepping and detect logical errors in the program.	
iii)	<b>Describe parallel communication protocols.</b>	<b>4 Marks</b>
Ans:	<p><b>Parallel Protocols : PCI and PCI-X</b></p> <p><b>PCI :-</b></p> <p>PCI stands for Peripheral Component Interconnect/Interface bus. It is popular for higher 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 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><b>PCI-X :-</b></p> <p>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>	<p><b>Description or any two features of :</b></p> <p><b>PCI</b> (2 Marks)</p> <p><b>PCI-X</b> (2 Marks)</p>
iv)	<b>Draw a labelled interfacing diagram of ADC 0808 with 8951 microcontroller.</b>	<b>4 Marks</b>
Ans:	<p><b>Note:</b> Any labeled diagram showing handshaking signal of ADC (Data bus, Channel Select, ALE, START, end of conversion, output enable) Any port of 89C51 is considered for interfacing.</p>	<b>4 Marks for complete labeled diagram.</b>



b)	Attempt any <u>ONE</u> of the following:	06 Marks
i)	Classify embedded system. Describe any two of them in short.	6 Marks
Ans:	<div><div><div><div><div>Embedded Systems</div><div>Based On Performance &amp; Functional Requirements</div><div>Real Time</div><div>Stand Alone</div><div>Networked</div><div>Mobile</div></div><div>Based On The Performance Of The Microcontroller</div><div>Small Scale</div><div>Medium Scale</div><div>Sophisticated</div></div></div></div> <p><b>Small scale:-</b> 8 bit or 16 bit microcontroller is used. These types of embedded systems are designed with a single 8 or 16-bit microcontroller that may even be activated by a battery. For developing embedded software for small scale embedded systems, the main programming tools are an editor, assembler, cross assembler and integrated development environment (IDE).</p> <p><b>Medium scale:-</b> 16 bit or 32 bit microcontroller or microprocessor is used. Such as DSP, RISC. These types of embedded systems design with a single or 16 or 32 bit microcontroller, RISCs or DSPs. These types of embedded systems have both hardware and software complexities. For developing embedded software for medium scale embedded systems, the main programming tools are C, C++, and JAVA, Visual C++, and RTOS, debugger, source code engineering tool, simulator and IDE.</p> <p><b>Sophisticated embedded system:-</b> ASIP, ARM, IP processors are used. These types of embedded systems have enormous hardware and software complexities, that may need ASIPs, IPs, PLAs, scalable or configurable processors. They are used for cutting edge applications that need hardware and software Co-design and components which have to assemble in the final system.</p> <p><b>Reactive and real-time:-</b> A real time embedded system is defined as a system which gives a required o/p in a particular time. These types of embedded systems follow the time deadlines for completion of a task. Real time embedded systems are classified into two types such as soft and hard real time systems .Example is autopilot system in a flight.</p> <ul style="list-style-type: none"><li>▶ Hard Real Time: System gives absolute Guarantee. Strictly adhere to each deadline.</li><li>▶ Soft Real Time Statistical Guarantee .Dead line are mostly met.</li></ul>	<b>Classification</b> <b>2 Marks</b>  <b>Description of</b> <b>each 2 marks</b> <b>( any two)</b>



	<p><b>Networked :-</b> These types of embedded systems are related to a network to access the resources. The connected network can be LAN, WAN or the internet. The connection can be any wired or wireless. This type of embedded system is the fastest growing area in embedded system applications. The embedded web server is a type of system wherein all embedded devices are connected to a web server and accessed and controlled by a web browser.</p> <ul style="list-style-type: none"><li>▶ Example for the LAN networked embedded system is a home security system wherein all sensors are connected and run on the protocol TCP/IP Embedded System</li></ul> <p><b>Mobile Embedded Systems:-</b></p> <ul style="list-style-type: none"><li>▶ Mobile embedded systems are used in portable embedded devices like cell phones, mobiles, digital cameras, mp3 players and personal digital assistants, etc. The basic limitation of these devices is the other resources and limitation of memory.</li></ul> <p><b>Stand-alone :-</b> This embedded systems do not require a host system like a computer, it works by itself. It takes the input from the input ports either analog or digital and processes, calculates and converts the data and gives the resulting data through the connected device-Which either controls, drives or displays the connected devices. Examples for the stand alone embedded systems are mp3 players, digital cameras, video game consoles, microwave ovens and temperature measurement systems.</p>	
ii)	<b>Explain pre-emptive scheduling and round-robin scheduling algorithms in RTOS.</b>	<b>6 Marks</b>
Ans:	<p><b>Preemptive scheduling :-</b> Preemptive scheduling ensures that every task will get CPU time for execution . The allotment of CPU time depends on the preemptive scheduling algorithm. It allows a process to be interrupted in the middle of its execution, taking the CPU away and allocating it to another process. The scheduling algorithm has high overhead. System is costly and complex in design.</p> <p><b>Shortest Job first:-</b> Shortest Job first is the example of this preemptive scheduling. If a new process arrives with a CPU burst length less than the remaining time of the current executing process, pre-empt. This scheme is known as the Shortest-Remaining-Time-First (SRTF).</p> <p><b>Priority scheduling in pre-emptive :-</b> A <b>preemptive</b> approach will pre-empt the CPU if the priority of the newly-arrived process is higher than the priority of the currently running process. (diagram)</p>	<b>3 Marks to each</b>



#### Round-robin scheduling:-

In the round robin algorithm, each process gets a small unit of CPU time (a time quantum), usually 10-100 milliseconds. After this time has elapsed, the process is pre-empted and added to the end of the ready queue. If there are 'n' processes in the ready queue and the time quantum is 'q', then each process gets '1/n' of the CPU time in chunks of at most 'q' time units at once. No process waits more than '(n-1)q' time units.

Performance of the round robin algorithm

q large then FCFS

q small then q must be greater than the context switch time; otherwise, the overhead is too high

One rule of thumb is that 80% of the CPU bursts should be shorter than the time quantum

**OR**

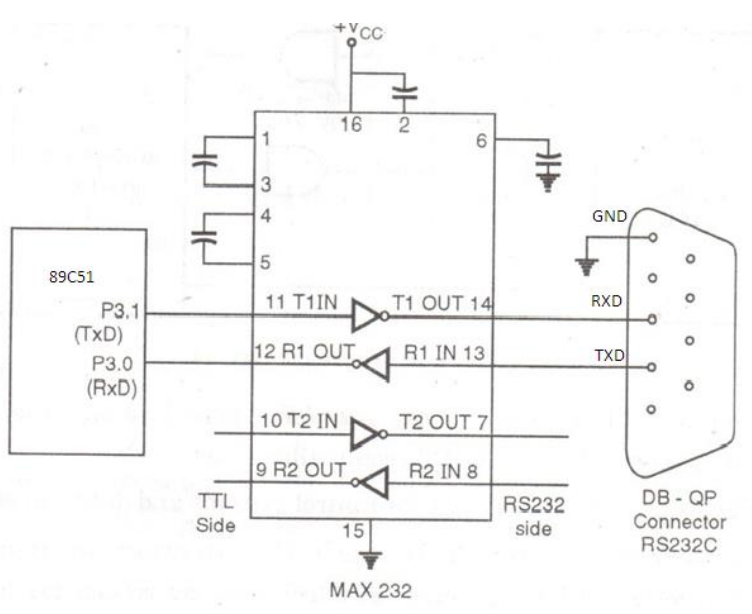
In the round robin algorithm, the kernel allocates a certain amount of time for each task waiting in the queue. The time slice allocated to each task is called quantum. As shown in fig. if three tasks 1,2, 3 are waiting in the queue the CPU first executes task1 then task2 then task 3 and the again task1 in round robin algorithm each task waiting in the queue is given a fixed time slice. The kernel gives control to the next task if the current task has completed its work within the time slice or if the current task has completed its allocated time. The kernel gives control to the next task if

- a) the current task has completed within the time slice
- b) the current task has no work to do
- c) the current task has completed its allocated time slice



		<p>This algorithm is very simple to implement but there is no priorities for any task. All tasks are considered of equal importance . If time critical operation are not involved then this algorithm will be sufficient . Digital millimeter , microwave oven has this algorithm.</p> <div><table><tr><td>Task1</td><td>Task2</td><td>Task3</td><td>Task 1</td><td>Task2</td><td>Task3</td></tr><tr><td>Running</td><td>Running</td><td>Running</td><td>Running</td><td>Running</td><td>Running</td></tr></table><p>Time</p><p>Fig. Round robin scheduling algorithm</p></div>	Task1	Task2	Task3	Task 1	Task2	Task3	Running	Running	Running	Running	Running	Running					
Task1	Task2	Task3	Task 1	Task2	Task3														
Running	Running	Running	Running	Running	Running														
Q 2		Attempt any <b>FOUR</b> of the following:	16 Marks																
	a)	Compare RISC and CISC architectures with any four points.	4 Marks																
	Ans:	<table><tr><th>CISC</th><th>RISC</th></tr><tr><td>Emphasis on hardware</td><td>Emphasis on software</td></tr><tr><td>Multiple instruction sizes and formats</td><td>Instructions of same set with few formats</td></tr><tr><td>Less registers</td><td>Uses more registers</td></tr><tr><td>More addressing modes</td><td>Fewer addressing modes</td></tr><tr><td>Extensive use of microprogramming</td><td>Complexity in compiler</td></tr><tr><td>Instructions take a varying amount of cycle time</td><td>Instructions take one cycle time</td></tr><tr><td>Pipelining is difficult</td><td>Pipelining is easy</td></tr></table>	CISC	RISC	Emphasis on hardware	Emphasis on software	Multiple instruction sizes and formats	Instructions of same set with few formats	Less registers	Uses more registers	More addressing modes	Fewer addressing modes	Extensive use of microprogramming	Complexity in compiler	Instructions take a varying amount of cycle time	Instructions take one cycle time	Pipelining is difficult	Pipelining is easy	One mark for each point
CISC	RISC																		
Emphasis on hardware	Emphasis on software																		
Multiple instruction sizes and formats	Instructions of same set with few formats																		
Less registers	Uses more registers																		
More addressing modes	Fewer addressing modes																		
Extensive use of microprogramming	Complexity in compiler																		
Instructions take a varying amount of cycle time	Instructions take one cycle time																		
Pipelining is difficult	Pipelining is easy																		
	b)	Explain the use of assembly language in C language with suitable example.	4 Marks																
	Ans:	<p>This is known as inline assembly which does not require separate assembly and link steps. It is more convenient than separate a separate assembler. This uses the functions and variables of ‘c’ language. This helps to integrate the assembly language program and ‘c’ language program easily.</p> <p>Its uses following syntax</p> <pre># include &lt;reg51.h&gt; void main (void) { C Instructions; # pragma asm Assembly instructions # pragma endasm C Instructions;</pre>	Uses(advantag ed +) 2 Marks Example 2 marks																



	<pre>}  Example ( any suitable combination) # include &lt;reg51.h&gt; void main (void) { while(1) {     P0 = ~ P0; # pragma asm MOV TMOD,#10H MOV TH1,#0xFFH MOV TL1,#0xFFH SETB TR1 JNB TF1,\$ # pragma endasm } }</pre>	
c)	<b>Draw a labelled interconnection diagram between RS232 and 8951 microcontroller.</b>	<b>4 Marks</b>
Ans:	 <p>Any labeled diagram showing line driver /Receiver( MAX232/233 etc)</p>	<b>4 Marks</b>
d)	<b>A 230V AC bulb is connected through a relay at P2.2. A light sensor is connected at P3.4. A light sensor produces logic high in dark condition. Write a 'C' program to switch 'ON' the bulb in 'DARK' condition and switch it OFF in 'LIGHT' condition.</b>	<b>4 Marks</b>
Ans:	<pre># include&lt;reg51.h&gt; sbit relay = P2^2;</pre>	<b>4 Marks</b> <b>Any suitable</b>

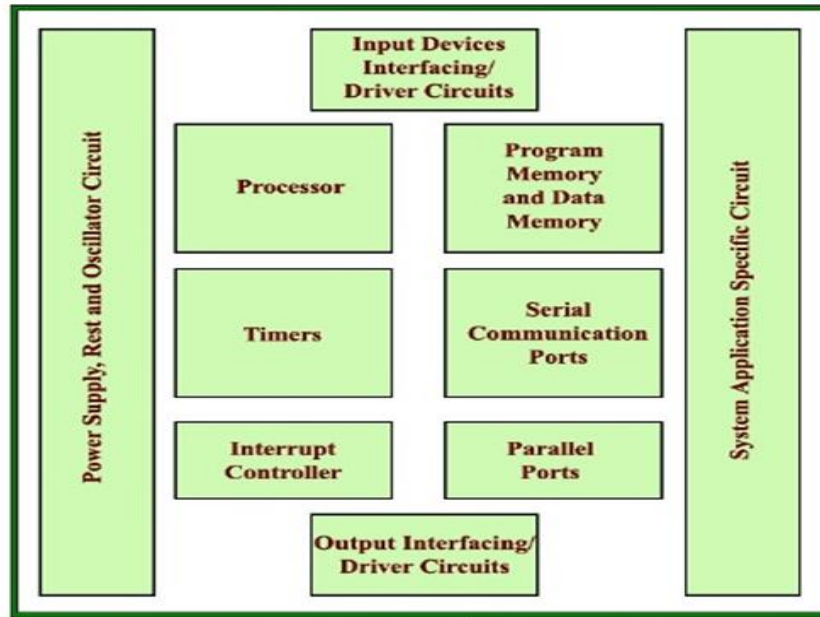




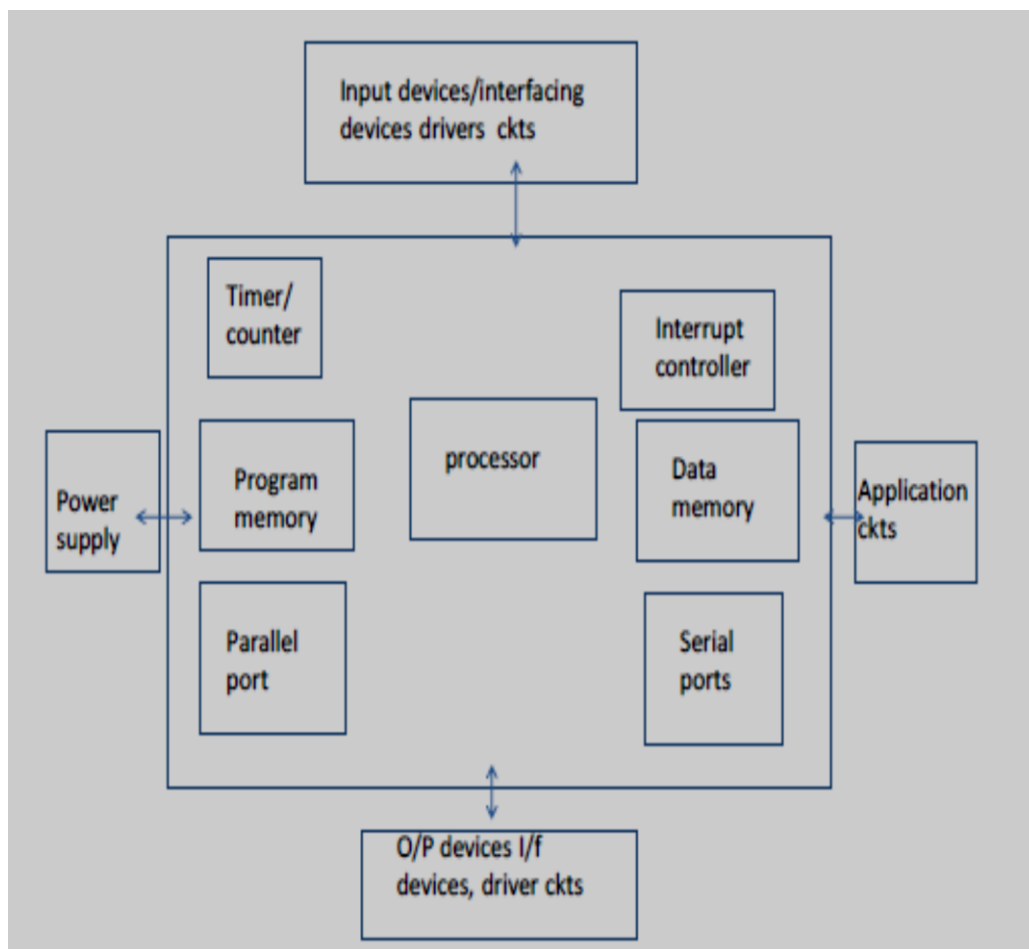
	<pre>sbit sensor = P3^4; void main (void) {   sensor=1;          //P3.4 as input line   relay = 0;          // Initially OFF    while (1)   {     while ( sensor == 0);    // check sensor if no dark wait//     relay = 1;              // dark ON bulb//     while ( sensor == 1);    // wait till dark//    } }</pre> <p>OR</p> <pre># include&lt;reg51.h&gt; sbit relay = P2^2; sbit sensor = P3^4; void main (void) {   sensor=1;          //P3.4 as input line   relay = 0;          // Initially OFF    while (1)   {     if ( sensor == 1)    // check sensor if dark //       relay = 1;          // ON bulb//     else       relay=0;    } }</pre>	<p>logic may be used. Only program is expected NO Marks for algorithm and flow chart. Appropriate Marks for step may be awarded .</p>
e)	<b>Describe inter task communication in RTOS.</b>	<b>4 Marks</b>
<b>Ans:</b>	<p>Software is the basic building block of RTOS. Task is a simply subroutine. Task must be able to communicate with one another to coordinate their activities or to share the data. Kernel object is used for inter task communication. Kernel objects uses message queue, mail box and pipes, Shared Memory, Signal Function and RPC a for inter task communication.</p> <p><b>Message queue:-</b> A message queue is a buffer like data structure, through which tasks and ISRs communicate with each other by sending and receiving messages and synchronize with</p>	<p><b>1 Marks for basics of inter task communication</b> <b>3 marks for its any three methods ( short</b></p>



	<p>data. It temporarily stores the message from a sender until the intended receiver is ready to read them. A message queue has queue control block, queue name, unique ID, memory buffers, a queue length. Kernel allocates the memory for message queue, ID, control block etc.</p> <p><b>Mail box:-</b> In general, mailboxes are similar to message queues. Mail box technique for inter task communication in RTOS based system used for one way messaging. The task/thread creates mail box to send the message. The receiver task can subscribe the mail box. The thread which creates the mail box is known as mailbox server. The others are known as client. RTOS has function to create, write and read from mail box. No of messages (limited or unlimited) in mail box have been decided by RTOS.</p> <p><b>Pipes :-</b> Pipes are kernel objects used for unstructured data exchange between tasks facilities synchronization among tasks. Pipe provides a simple data transfer facility.</p> <p><b>Shared Memory :-</b> Shared memory is simplest way of inter process communication. The sender process writes data into shared memory and receiver process reads data.</p> <p><b>Signal Function:-</b> Operating system provides the signal function for messaging among task (process).</p> <p><b>Remote Procedure Call(RPC) and Sockets:-</b> RPC is a mechanism used by process(task) to call the procedure of another process running on same or different CPU in the network. Sockets are used for RPC communication and establishes full duplex communication between tasks.</p>	<b>description is expected)</b>
f)	<b>Draw and explain block diagram of embedded system.</b>	<b>4 Marks</b>
<b>Ans:</b>		<b>Diagram - 2 marks</b> <b>Functional description of any four - 2 Marks</b>



OR



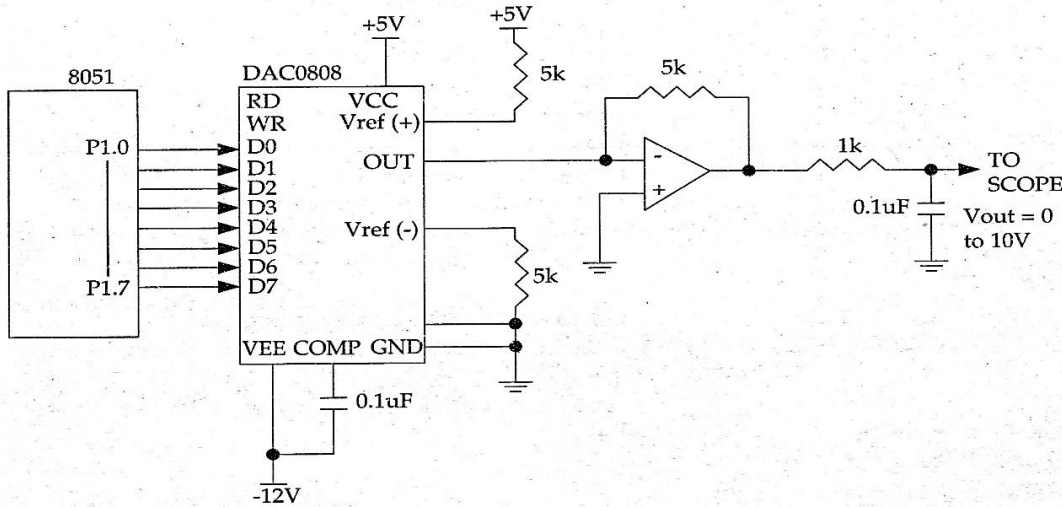


		<p><b>Processor:-</b> The processor is the heart of embedded system and selected as per application of interest.</p> <p><b>Power supply</b> is also called as “charge pump”. Power supply source or charge pump is essential in every system .</p> <p><b>Clock Oscillator Circuit and Clocking Units :-</b> For processing unit, highly stable oscillator is required and microprocessor clock out signal provides clock for synchronizing all system unit with the processor. Reset and watch dog timer is also available.</p> <p><b>Memory :-</b> A system embeds either in the internal flash or ROM, PROM or in an external flash or ROM or PROM of the microcontroller.</p> <p><b>Input, output and I/O Ports, I/O Buses and I/O Interfaces</b> A system connects to external physical devices and systems through parallel or serial I/O ports. De-multiplexers and multiplexers facilitate communication of signals from multiple channels through a common path. A system often n/w to other devices and systems through an I/O bus. E.g. I<sup>2</sup>C, CAN, USB, ISA, PCI, etc. For automatic control and signal processing applications, a system provides necessary interfacing circuit and software for DAC unit and ADC unit.</p> <p><b>Keyboard/keypad :-</b> For inputs, a keypad or keyboard may interface to a system. The system provides necessary interfacing circuit and s/w to receive I/p's directly from the keys or through a controller. The system may need the necessary interfacing circuit and s/w for the o/p to the LCD display controller and the LED interfacing ports or for the I/O's with the touch screen.</p> <p><b>Real Time Operating System (RTOS):-</b> It supervises the software application running on hardware and organizes access to resource according to priorities of task in the system. It provides a mechanism to run the process as per schedule and context switching between various processes.</p>	
<b>Q.3</b>		<b>Attempt any <u>FOUR</u> of the following:</b>	<b>16 Marks</b>
	<b>a)</b>	<b>List wireless communication protocols and state four features of zigbee protocol.</b>	<b>4 Marks</b>
	<b>Ans:</b>	<p><b><u>Wireless Communication Protocols:-</u></b></p> <ol style="list-style-type: none"> <li>1. Infrared [IrDA]</li> <li>2. Bluetooth</li> <li>3. Zigbee</li> <li>4. Wi-Fi</li> </ol> <p><b><u>Features of Zigbee:-</u></b></p> <ol style="list-style-type: none"> <li>1. Support for multiple network topologies such as point to point, point to multipoint and mesh networks, cluster tree.</li> </ol>	<b>List-2marks any 4 features -2marks</b>



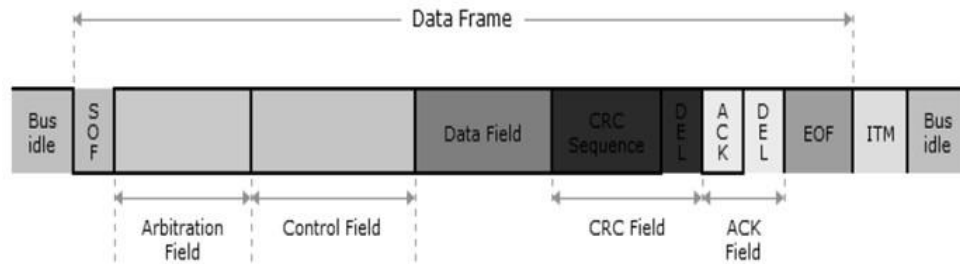
	<div>2. ZigBee operates in the industrial, scientific and medical (<u>ISM</u>) radio bands: 2.4 GHz in most jurisdictions worldwide.</div> <div>3. Low duty cycle provides long battery life.</div> <div>4. 250 Kbps data rate.</div> <div>5. Scalable and easy deployment.</div> <div>6. Highly reliable and secure.</div> <div>7. Low latency.</div> <div>8. Cost effective.</div> <div>9. Up to 65000 nodes per network.</div> <div>10. 128 bit AES encryption for secure data connections.</div> <div>Collision avoidance, retries and acknowledgments.</div>																
b)	Write a ‘C’ program to toggle P2.1 continuously with 100 ms delay. (Use simple delay subroutine).	4 Marks															
Ans:	<pre>#include &lt;reg 51.h&gt; void add_delay (unsigned int); sbit data_bit = P2^1; void main (void) { while (1) { data_bit =1; // set P2.1 bit add_delay (100); data_bit =0; // reset P2.1bit add_delay (100); } }  void add_delay (unsigned int delay_time) { unsigned int x,y; for (x = 0; x &lt;delay_time; x ++) for (y =0; y&lt;1275; y++); }</pre>																
c)	Compare desktop operating system with RTOS with any four points.	4 Marks															
Ans:	<table><tr><td>Sr no</td><td>OS</td><td>RTOS</td></tr><tr><td>1</td><td>Non-Deterministic time behavior.</td><td>Deterministic time behavior.</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</td><td>OS services consumes only known and expected amounts of time regardless the</td></tr></table>	Sr no	OS	RTOS	1	Non-Deterministic time behavior.	Deterministic time behavior.	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	OS services consumes only known and expected amounts of time regardless the	Any four points-1Mark each
Sr no	OS	RTOS															
1	Non-Deterministic time behavior.	Deterministic time behavior.															
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	OS services consumes only known and expected amounts of time regardless the															



		<table><tr><td></td><td>responsiveness of an application at unexpected time.</td><td>number of services.</td></tr><tr><td>5</td><td>Slow context switching</td><td>fast context switching</td></tr><tr><td>6</td><td>More memory requirements</td><td>Less memory requirements</td></tr><tr><td>7</td><td>Ex: Windows XP, MS-DOS</td><td>Ex: Windows CE, VxWorks</td></tr></table>		responsiveness of an application at unexpected time.	number of services.	5	Slow context switching	fast context switching	6	More memory requirements	Less memory requirements	7	Ex: Windows XP, MS-DOS	Ex: Windows CE, VxWorks	
	responsiveness of an application at unexpected time.	number of services.													
5	Slow context switching	fast context switching													
6	More memory requirements	Less memory requirements													
7	Ex: Windows XP, MS-DOS	Ex: Windows CE, VxWorks													
d)	State any two advantages and disadvantages of embedded system.		4 Marks												
Ans:	<p><b>Advantages :-</b></p> <ol style="list-style-type: none"><li>1. Cost is low.</li><li>2. Small in size.</li><li>3. Highly reliable.</li><li>4. Operation is fast.</li><li>5. Easy for mass production.</li><li>6. Less interconnection.</li><li>7. Improves product quality.</li></ol> <p><b>Dis-Advantages:-</b></p> <ol style="list-style-type: none"><li>1. Hard for maintenance as it is use and throw device.</li><li>2. No technological improvement.</li><li>3. Hard to back up of embedded files.</li><li>4. Less power supply durability if it is battery operated.</li></ol>		<p><b>Advantages any 2 – 2 Marks</b></p> <p><b>Disadvantages- any 2 – 2 Marks</b></p>												
e)	Draw a labelled interfacing diagram of DAC 0808 with 8951 microcontroller. Also write a ‘C’ program to generate triangular waveform using DAC.		4 Marks												
Ans:	<div><pre>#include&lt;reg51.h&gt;  unsigned char d; void main(void) { while(1)</pre></div>		<p><b>Diagram - 2Marks</b></p> <p><b>Program- 2Marks</b></p>												



		<pre>{ for(d=0; d&lt;255; d++) { P1 = d; } for(d=255; d&gt;0; d--) { P1 = d; } }</pre>																										
Q.4	a)	Attempt any <u>THREE</u> of the following:	12 Marks																									
	a)	Draw format of TMOD register. Find the value of TMOD register to operate timer 0 in mode 1.	4 Marks																									
	Ans:	<p>TMOD Format:</p> <div><div>MSB</div><table><tr><td>GATE</td><td>C/T</td><td>M1</td><td>M0</td><td>GATE</td><td>C/T</td><td>M1</td><td>M0</td><td>LSB</td></tr></table><div><div>TIMER 1</div><div>TIMER 0</div></div></div> <p>Value in TMOD to Operate Timer 0 in mode 1: 01 H</p> <table><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>Gate</td><td><math>C/\bar{T}</math></td><td>M1</td><td>M0</td><td>GATE</td><td><math>C/\bar{T}</math></td><td>M1</td><td>M0</td></tr></table>	GATE	C/T	M1	M0	GATE	C/T	M1	M0	LSB	0	0	0	0	0	0	0	1	Gate	$C/\bar{T}$	M1	M0	GATE	$C/\bar{T}$	M1	M0	<p>Format 2 marks</p> <p>Value 2 marks</p>
GATE	C/T	M1	M0	GATE	C/T	M1	M0	LSB																				
0	0	0	0	0	0	0	1																					
Gate	$C/\bar{T}$	M1	M0	GATE	$C/\bar{T}$	M1	M0																					
	b)	Explain CAN Bus protocol with the frame structure.	4 Marks																									
	Ans:	<p><u>Controlled Area Network [CAN]:-</u> Can is mainly used in automotive electronics. CAN bus is a standard bus in distributed network. It has a bi-directional serial line which receives or sends a bit at an instance by operating at maximum rate of 1Mbps. It employs a twisted pair connection to each node. The pair can run to a maximum length of 40m.</p> <div><div>Node 1</div><div>Node 2</div><div>.....</div><div>Node n</div></div> <div><div>120 Ω</div><div>CAN_H</div><div>CAN_L</div><div>120 Ω</div></div> <td><p>Frame format -2marks</p><p>Description -2 marks</p></td>	<p>Frame format -2marks</p> <p>Description -2 marks</p>																									



Field and its Length	Description of each field in CAN frame
1 <sup>st</sup> field of 12 bits	It is called arbitration field. It contains the packet 11-bit destination address and the RTR [Remote Transmission Request]. When this bit is at 1 this indicates the packet is for the destination address. If this packet for request for a data from a device defined by identifier. The device is at destination address specified in the field.
2 <sup>nd</sup> field of 6 bits	It is called a control field. The 1 <sup>st</sup> bit is the identifier extension. The 2 <sup>nd</sup> bit is always 1, and the last 4 bits are code for data length.
3 <sup>rd</sup> field of 0-64 bits	Its length depends on data length code in the control field.
4 <sup>th</sup> field of 16-bits { 3 <sup>rd</sup> if data field has no bit present }	It is the CRC word. The receiver node uses it to detect errors during transmission.
5 <sup>th</sup> field of 2 bits	1 <sup>st</sup> field is the ACK slot. The sender sends it as 1 and RX sends back 0 in this slot when the receiver detects an error in the reception. Sender after sensing 0 in the ACK slot transmits the data frame. The 2 <sup>nd</sup> bit is the ACK delimiter bit. It signals the end of the ACK field. If the transmitting node does not receive and ACK of data frame within a specified time slot it should retransmit.
6 <sup>th</sup> field of 7-bits	It is for end of the frame specification and has seven 0's.

c)	<b>State any eight design metrics of embedded system.</b>	<b>4 Marks</b>
<b>Ans:</b>	<ol style="list-style-type: none"> <li>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.</li> <li>2. Throughput or Performance : the execution time or throughput of the system. Instruction execution time in the system measures performance. Smaller execution time means higher performance. For eg. in mobile phone, voice</li> </ol>	<b>Any 8 Each 1/2Marks</b>



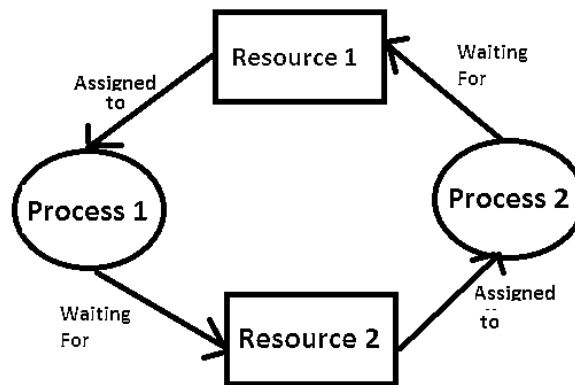


signals are processed between antenna and speaker in 0.1s shows phone performance. The system may need to handle a lot of data in a short time.

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.
11. Power Dissipation: For battery operated system this is important feature. Examples are mobile phone or digital camera where if power dissipation is small battery needs to be recharge less frequently.
12. Unit cost: the monetary cost of manufacturing each copy of the system, excluding NRE cost.
13. NRE cost (Non-Recurring Engineering cost): The monetary cost of designing the system. Once the system is designed, any number of units can be manufactured without incurring any additional design cost (hence the term “non-recurring”).
14. Size: the physical space required by the system, often measured in bytes for software, and gates or transistors for hardware.
15. Flexibility: the ability to change the functionality of the system without incurring heavy NRE cost. Software is typically considered very flexible. Flexibility in design enables, without significant engineering cost, development of different versions or product or to develop advanced version later on. For example software enhancement by adding extra functions.
16. Maintainability: Deals with support and maintenance to the end user or client in case of technical issues and product failure. A more reliable system means with less maintainability. As reliability of the system increases chances of failure and non-functioning also reduces.
17. Time-to-market: The amount of time required to design and manufacture the system to the point the system can be sold to customers. The main contributors are design time, manufacturing time and testing time. There may be multiple players in the embedded industry who develop products of the same category (like mobile phones, portable media players etc.). If you come with new product and time to market is high competitor may take advantage of it with their product.
18. Time-to-prototype: The amount of time to build a working version of the system, which may be bigger or more expensive than the final system implementation,



	<p>but can be used to verify the system's usefulness and correctness and to refine the system's functionality. If the prototype is developed faster, the actual estimated development time can be brought down.</p> <p>19. Correctness: our confidence that we have implemented the system's functionality correctly. We can check the functionality throughout the process of designing the system, and we can insert test circuitry to check that manufacturing was correct.</p> <p>20. Safety: the probability that the system will not cause harm. It deals with possible damages that can happen to the operators, public and the environment due to breakdown of embedded system, or due to the emission of radioactive or hazardous materials from embedded products. Safety analysis is a must in product engineering to evaluate the anticipated damages and determine best course of action.</p>	
d)	<b>Explain the concept of deadlock with suitable example.</b>	<b>4 Marks</b>
<b>Ans:</b>	<p>A deadlock is a situation where in two or more competing actions are each waiting for the other to finish, and thus neither ever does. In computer science, deadlock refers to a specific condition when two or more processes are each waiting for the other to release a resource, or more than two processes are waiting for resources in a circular chain.</p> <p>A deadlock, also called as deadly embrace, is a situation in which two threads are each unknowingly waiting for resource held by other.</p> <ul style="list-style-type: none"> <li>• Assume thread/process T1 has exclusive access to resource R1.</li> <li>• Thread/ process T2 has exclusive access to resource R2.</li> <li>• If T1 needs exclusive access to R2 and T2 needs exclusive access to R1,</li> <li>• Neither thread can continue.</li> <li>• They are deadlocked.</li> <li>• The simplest way to avoid a deadlock is for threads to: <ul style="list-style-type: none"> <li>➤ Acquire all resources before proceeding</li> <li>➤ Acquire the resources in the same order</li> <li>➤ Release the resource in the reverse order</li> </ul> </li> </ul> <p>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.</p>	<p><b>Explanation-2Marks</b></p> <p><b>Example-2 Marks</b></p>



In this example, process 1 wants the resource 2 ex; scanner while holding the resource 1 ex: printer. Process 1 cannot proceed until both the printer and the scanner are in its possession.

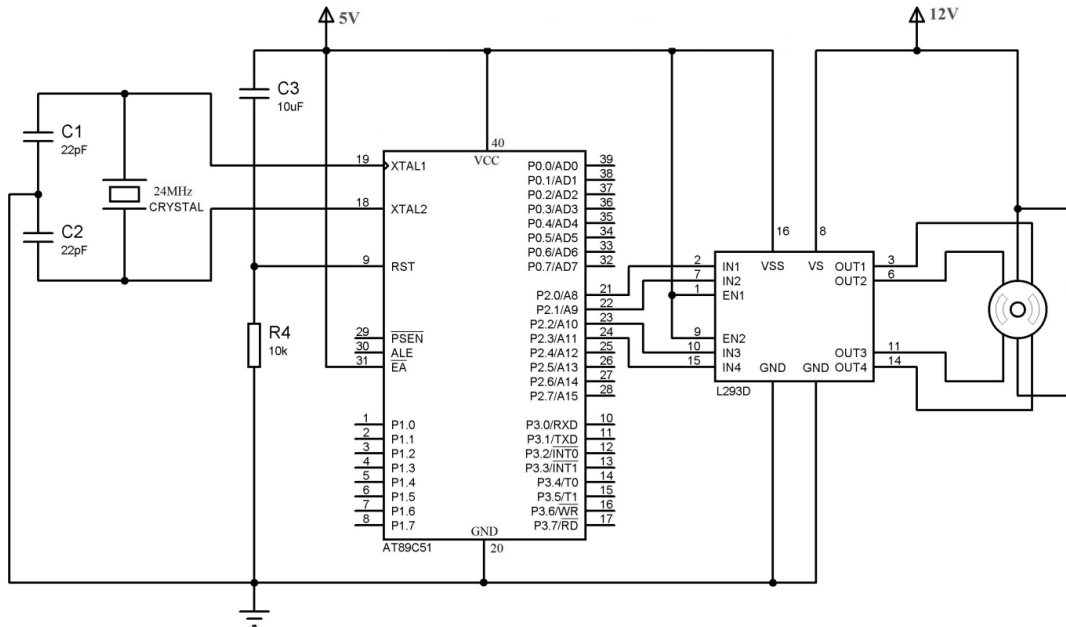
Process 2 wants the printer while holding the scanner. Process 2 cannot continue until it has the printer and the scanner.

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

b)	Attempt any <u>ONE</u> of the following:	06 Marks
a)	Write a 'C' program to generate a square wave of 5 kHz.(Operate timer 0 in mode 1).	6 Marks
Ans:	<p>Crystal frequency= 11.0592 MHz</p> $I/P \text{ clock} = \frac{1}{2} (\text{crystal frequency}) = \frac{1}{2} \times 11.0592 \text{ MHz} = 921.6 \text{ KHz}$ <p>Tin = 1.085 μsec For 5 kHz square wave Fout = 5 KHz</p> $\frac{1}{T_{out}} = 200 \mu \text{sec}$ $T_{out} = 5 \times 10^3$ <p>Tout = 200 μsec Consider half of it = Tout = 100 μ sec</p> $N = \frac{T_{out}}{T_{in}} = \frac{100 \mu}{1.085 \mu} = 92.16$ <p>65536-92 = (65444)D = FFA5 H</p> <p>Program:</p> <pre>#include&lt;reg51.h&gt; void delay(void); sbit p=P3^5; void main (void) {</pre>	<p>Calculation-2 Marks</p> <p>Program - 4 Marks</p>



	<pre>while (1) {   p=~p;   delay(); } } void delay() {   TMOD=0X01; //set timer 0 in mode 1 i.e. 16 bit number   TL0=0XA5H; //load TL register with LSB of count   TH0=0XFFH ; //Load TH register with MSB of count   TR0 =1 ; //Start timer 0   While(TF0= = 0); //wait until timer rolls over   TR0=0; //Stop timer 0   TF0=0; //Clear timer flag 0 }</pre>	
b)	<b>Draw labelled interfacing diagram of stepper motor with 8951. Write a ‘C’ program to rotate it in counterclockwise direction.</b>	<b>6 Marks</b>
Ans:	<div><p>ULN2003 Connection for Stepper Motor Pin 8 = GND Pin 9 = +5V</p><p>Use a separate power supply for the motor</p></div> <p style="text-align: center;"><b>OR</b></p>	<b>Diagram- 3Marks</b>  <b>Program- 3Marks</b>

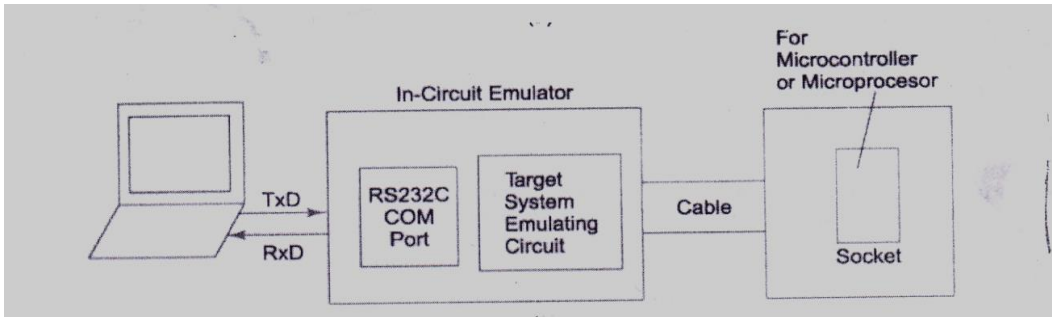


Instructions	comments
<pre>#include&lt;reg51.h&gt; void delay(unsigned int); void main (void) { while(1) { P2=0x33; delay(10); P2=0x66; delay(10); P2=0xcc; delay(10); P2=0x99; delay(10); } } void delay(unsigned int t) { unsigned int x,y; for(x=0;x&lt;=t;x++) for(y=0;y&lt;=675;y++); }</pre>	<p>Do forever</p> <p>Send appropriate data bytes to port-2</p> <p>Followed by delay to rotate stepper motor</p> <p>Delay function</p>

Q.5	Attempt any <b>FOUR</b> of the following.	16 Marks
a)	State 'C' language logical operators for AND, OR, NOT and EX-OR operation. Give one example of each.	4 Marks



Ans:	<table><tr><td>Sr no:</td><td>Operator</td><td>Symbol</td><td>Example</td></tr><tr><td>1.</td><td>AND</td><td>&amp; is called Logical AND operator.</td><td>Y= A&amp;B</td></tr><tr><td>2.</td><td>OR</td><td>  is called Logical OR operator.</td><td>Y= A B</td></tr><tr><td>3.</td><td>NOT</td><td>~ is called NOT operator.</td><td>Y= ~A</td></tr><tr><td>4.</td><td>EX-OR</td><td>^ is called logical Ex-OR operator.</td><td>Y=A ^B</td></tr></table>	Sr no:	Operator	Symbol	Example	1.	AND	& is called Logical AND operator.	Y= A&B	2.	OR	is called Logical OR operator.	Y= A B	3.	NOT	~ is called NOT operator.	Y= ~A	4.	EX-OR	^ is called logical Ex-OR operator.	Y=A ^B	1 Mark each	
Sr no:	Operator	Symbol	Example																				
1.	AND	& is called Logical AND operator.	Y= A&B																				
2.	OR	is called Logical OR operator.	Y= A B																				
3.	NOT	~ is called NOT operator.	Y= ~A																				
4.	EX-OR	^ is called logical Ex-OR operator.	Y=A ^B																				
b)	Distinguish between synchronous and asynchronous communication with any four points.	4 Marks																					
Ans:	<table><tr><td>Sr. No.</td><td>Synchronous</td><td>Asynchronous</td></tr><tr><td>1</td><td>Same clock pulse is required at transmitter and receiver</td><td>Different clock pulse is required at transmitter and receiver</td></tr><tr><td>2</td><td>Used to transfer group of character</td><td>Used to transfer one character at a time</td></tr><tr><td>3</td><td>Synchronous character is required.</td><td>Synchronous character is required.</td></tr><tr><td>4</td><td>No start and stop signals are required</td><td>Start and stop signals are required.</td></tr><tr><td>5</td><td>Data transmission rate is greater then or equal to 20Kbps</td><td>Data transmission rate is less then or equal to 20 Kbps.</td></tr><tr><td>6</td><td>It is less reliable</td><td>It is more reliable</td></tr></table>	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	Any four points -1 Mark 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																					
c)	State number of port lines required for a keyboard matrix having following keys: (i) 16 (ii) 256 (iii) 64 (iv) 144	4 Marks																					
Ans:	Number of port lines required for a keyboard matrix having following keys: (i) 16 -16 keys can be arranged in a matrix of 4X4, so number of port lines required are 4+4 =8 (ii) 256 - 256 keys can be arranged in a matrix of 16X16, so number of port lines required are 16+16 =32 (iii) 64 - 64 keys can be arranged in a matrix of 8X8, so number of port lines required are 8+8 =16 (iv) 144 -144 keys can be arranged in a matrix of 12X12, so number of port lines required are 12+12 =24	1 Mark Each																					

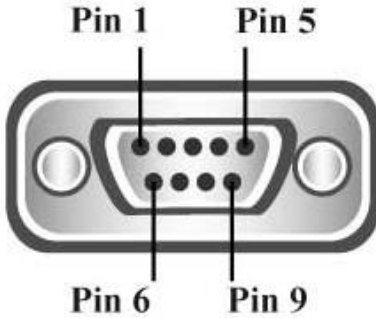
<b>d)</b>	<b>State four key specifications of RTOS.</b>	<b>4 Marks</b>
<b>Ans:</b>	<p>Key specifications of RTOS:-</p> <ol style="list-style-type: none"> <li>1. Reliability: A reliable system must be available (continue to provide service) and should not fail. The combination of all system elements determines the reliability of a system. ( Hardware, BSP, RTOS, and applications)</li> <li>2. Predictability: The RTOS used in this case needs to be predictable to a certain degree. The term deterministic describes RTOSes with predictable behavior, in which the completion of operating system calls occurs within known timeframes.</li> <li>3. Performance: An embedded system must perform fast enough to fulfill its timing requirements. Typically, the processor's performance is expressed in million instructions per second (MIPS). Throughput also measures the overall performance of a system, with hardware and software combined. One definition of throughput is the rate at which a system can generate output based on the inputs coming in.</li> <li>4. Compactness: In embedded systems, where hardware real estate is limited due to size and costs, the RTOS clearly must be small and efficient. In these cases, the RTOS memory footprint can be an important factor.</li> <li>5. Scalability: Because RTOSes can be used in a wide variety of embedded systems, they must be able to scale up or down to meet application-specific requirements. Depending on how much functionality is required, an RTOS should be capable of adding or deleting modular components, including file systems and protocol stacks.</li> </ol>	<p><b>Any 4</b> <b>1 Mark each</b></p>
<b>e)</b>	<b>Describe in-circuit emulator.</b>	<b>4 Marks</b>
<b>Ans:</b>	<p>In-circuit emulator:-</p>  <p>In-circuit emulator (ICE) is one of the oldest embedded debugging tools, and is still unmatched in power and capability. It is only tool that substitutes its own internal processor for the one in the target system Using one of a number of hardware tricks, the emulator can monitor everything that goes on in this on-board CPU , giving a complete visibility into the target code's operation. The emulator is bridge between the target and workstation giving an interactive terminal peering deeply into the target and a rich set of debugging resources. ICE uses another circuit with a card that connect to target processor (circuit) through a socket.</p>	<p><b>1 Mark</b> <b>Diagram,</b></p> <p><b>3 Marks</b> <b>Explanation.</b></p>



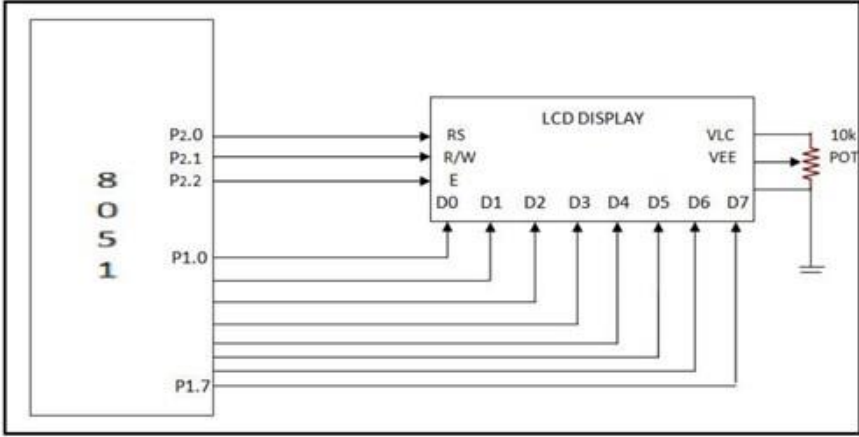


	<p style="text-align: center;"><b>OR</b></p> <ul style="list-style-type: none"><li>• An in-circuit emulator (ICE) is a hardware device used to debug the software of an embedded system. It was historically in the form of bond-out processor which has many internal signals brought out for the purpose of debugging. These signals provided information about the state of the processor.</li><li>• An in-circuit emulator provides a window into the embedded system. The programmer uses the emulator to load programs into the embedded system, run them, step through them slowly, and view and change data used by the system's software.</li><li>• More recently the term also covers JTAG based hardware debuggers which provide equivalent access using on-chip debugging hardware with standard production chips.</li><li>• ICE's attach a terminal or PC to the embedded system. The terminal or PC provides an interactive user interface for the programmer to investigate and control the embedded system.</li><li>• In usage, an ICE provides the programmer with execution breakpoints, memory display and monitoring, and input/output control</li></ul>	
f)	<b>Draw labeled interfacing diagram of 4 X 4 matrix keypad with 8951.</b>	<b>4 Marks</b>
Ans:	<b>Interfacing diagram of 4 X 4 matrix keypad with 8951</b> <p>The diagram shows a 4x4 matrix keypad interfaced to an 8951 microcontroller. The keypad has 4 rows (D0-D3) and 4 columns (0-3). Each intersection has a switch. Pull-up resistors (4.7k) are connected to VCC for each column. The rows are connected to Port 1 (Out) of the 8951. The columns are connected to Port 2 (In) of the 8951. The keypad is labeled with numbers 0-9, letters A-F, and a space key.</p>	<b>Diagram 4 Marks</b>
<b>Q.6</b>	<b>Attempt any <u>FOUR</u> of the following:</b>	<b>16 Marks</b>
a)	<b>Distinguish between assembly language and C language with reference to:</b> (i) Ease of programming (ii) Memory requirement (iii) Coding time (iv) Execution time	<b>4 Marks</b>



Ans:	<table><tr><td>Sr no</td><td>Parameters</td><td>Assembly Language program</td><td>Embedded C</td></tr><tr><td>1.</td><td>Ease of Programming</td><td>Less</td><td>More.</td></tr><tr><td>2.</td><td>Memory requirement</td><td>Less</td><td>More.</td></tr><tr><td>3.</td><td>Coding time</td><td>More time is required for coding.</td><td>Less time required for coding</td></tr><tr><td>4.</td><td>Execution Time</td><td>Faster [Less Execution time required]</td><td>Slower [More execution time required]</td></tr></table>	Sr no	Parameters	Assembly Language program	Embedded C	1.	Ease of Programming	Less	More.	2.	Memory requirement	Less	More.	3.	Coding time	More time is required for coding.	Less time required for coding	4.	Execution Time	Faster [Less Execution time required]	Slower [More execution time required]	1 Mark each									
	Sr no	Parameters	Assembly Language program	Embedded C																											
	1.	Ease of Programming	Less	More.																											
	2.	Memory requirement	Less	More.																											
	3.	Coding time	More time is required for coding.	Less time required for coding																											
4.	Execution Time	Faster [Less Execution time required]	Slower [More execution time required]																												
b) Draw pin diagram of DB9. connector, stating function of each pin.																															
Ans:	<p>Pin Diagram:</p> 			Pin diagram- 1Marks,  function - 3Marks																											
	<table><tr><td>Abbreviation</td><td>Full Name</td><td>Function</td></tr><tr><td>TD</td><td>Transmit Data</td><td>Serial Data Output (TXD)</td></tr><tr><td>RD</td><td>Receive Data</td><td>Serial Data Input (RXD)</td></tr><tr><td>CTS</td><td>Clear to Send</td><td>This line indicates that the Modem is ready to exchange data.</td></tr><tr><td>DCD</td><td>Data Carrier Detect</td><td>When the modem detects a "Carrier" from the modem at the other end of the phone line, this Line becomes active.</td></tr><tr><td>DSR</td><td>Data Set Ready</td><td>This tells the UART that the modem is ready to establish a link.</td></tr><tr><td>DTR</td><td>Data Terminal Ready</td><td>This is the opposite to DSR. This tells the Modem that the UART is ready to link.</td></tr><tr><td>RTS</td><td>Request To Send</td><td>This line informs the Modem that the UART is ready to exchange data.</td></tr><tr><td>RI</td><td>Ring Indicator</td><td>Goes active when modem detects a ringing signal from the PSTN.</td></tr></table>				Abbreviation	Full Name	Function	TD	Transmit Data	Serial Data Output (TXD)	RD	Receive Data	Serial Data Input (RXD)	CTS	Clear to Send	This line indicates that the Modem is ready to exchange data.	DCD	Data Carrier Detect	When the modem detects a "Carrier" from the modem at the other end of the phone line, this Line becomes active.	DSR	Data Set Ready	This tells the UART that the modem is ready to establish a link.	DTR	Data Terminal Ready	This is the opposite to DSR. This tells the Modem that the UART is ready to link.	RTS	Request To Send	This line informs the Modem that the UART is ready to exchange data.	RI	Ring Indicator	Goes active when modem detects a ringing signal from the PSTN.
	Abbreviation	Full Name	Function																												
	TD	Transmit Data	Serial Data Output (TXD)																												
	RD	Receive Data	Serial Data Input (RXD)																												
	CTS	Clear to Send	This line indicates that the Modem is ready to exchange data.																												
	DCD	Data Carrier Detect	When the modem detects a "Carrier" from the modem at the other end of the phone line, this Line becomes active.																												
	DSR	Data Set Ready	This tells the UART that the modem is ready to establish a link.																												
	DTR	Data Terminal Ready	This is the opposite to DSR. This tells the Modem that the UART is ready to link.																												
	RTS	Request To Send	This line informs the Modem that the UART is ready to exchange data.																												
RI	Ring Indicator	Goes active when modem detects a ringing signal from the PSTN.																													



c)	<b>Draw labelled interfacing diagram of 16 X 2 LCD with 8951 and state function of RS and R/W pin.</b>	<b>4 Marks</b>
Ans:	<p>Interfacing diagram of 16 X 2 LCD with 8951</p>  <p>Function:-</p> <p>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.</p> <p>RW: -R/W gives you the choice between writing and reading. R/W=1, reading is enabled. R/W=0 ,writing is enabled</p>	<b>labelled interfacing diagram- 2 Marks,</b> <b>functions- 1 Mark each.</b>
d)	<b>A key is connected at P3.2 and 8 LEDs are connected to P1 of 8951. Write a ‘C’ program to display 0 to 255 in binary on LEDs, when a key is pressed.</b>	<b>4 Marks</b>
Ans:	<p>(Assuming that P3.2 =0, when the key is pressed)</p> <pre>#include&lt;reg51.h&gt; sbit sw = P3^2; void delay (unsigned int); void main(void) { sw=1; unsigned char i; P1=0X00; while(1) { if(sw==0) { for(i=0;i&lt;=255;i++) { P1=i; delay(100);</pre>	<b>Program with correct logic – 4 Marks</b>



	<pre>} } else P1=0; } } void delay (unsigned int i time) {     unsigned int i,j;     for (i=0;i&lt;itime;i++)     for(j=0;j&lt;1275;j++); }</pre>																					
e)	<p>Manipulate the following table for data types used in ‘C’ language.</p> <table><tr><th>Sr. No.</th><th>Data type</th><th>Bit size</th><th>Data range</th></tr><tr><td>1.</td><td>Unsigned char</td><td>?</td><td>?</td></tr><tr><td>2.</td><td>Signed int</td><td>?</td><td>?</td></tr><tr><td>3.</td><td>Sbit</td><td>?</td><td>?</td></tr><tr><td>4.</td><td>Sfr</td><td>?</td><td>?</td></tr></table>	Sr. No.	Data type	Bit size	Data range	1.	Unsigned char	?	?	2.	Signed int	?	?	3.	Sbit	?	?	4.	Sfr	?	?	4 Marks
Sr. No.	Data type	Bit size	Data range																			
1.	Unsigned char	?	?																			
2.	Signed int	?	?																			
3.	Sbit	?	?																			
4.	Sfr	?	?																			
Ans:	<table><tr><th>Sr. No.</th><th>Data type</th><th>Bit size</th><th>Data range</th></tr><tr><td>1.</td><td>Unsigned char</td><td>8</td><td>0 to 255</td></tr><tr><td>2.</td><td>Signed int</td><td>16</td><td>-32768 to +32767</td></tr><tr><td>3.</td><td>Sbit</td><td>1</td><td>SFR bit addressable</td></tr><tr><td>4.</td><td>Sfr</td><td>8</td><td>RAM addresses 80 to FF only</td></tr></table>	Sr. No.	Data type	Bit size	Data range	1.	Unsigned char	8	0 to 255	2.	Signed int	16	-32768 to +32767	3.	Sbit	1	SFR bit addressable	4.	Sfr	8	RAM addresses 80 to FF only	1 Mark each
Sr. No.	Data type	Bit size	Data range																			
1.	Unsigned char	8	0 to 255																			
2.	Signed int	16	-32768 to +32767																			
3.	Sbit	1	SFR bit addressable																			
4.	Sfr	8	RAM addresses 80 to FF only																			