

(13)

(11) Nov-23

## Scoring Indicators

COURSE NAME : EMBEDDED SYSTEM AND REAL TIME OPERATING SYSTEM

COURSE CODE : 5131

QID : 2109230280

## PART A

I. Answer all the following questions in one word or sentence.

(9 x 1 = 9 Marks)

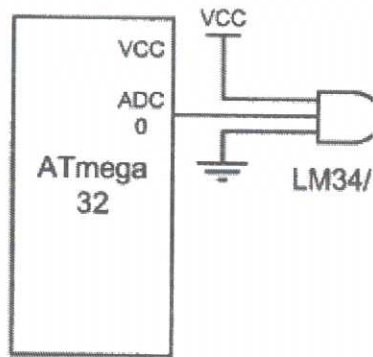
Max. marks

Q.No	Scoring Indicators	Split score	Sub Total	Total score
	<b>PART A</b>			9
I.1	An embedded system is an electrical/electro-mechanical system designed to perform a specific function and is a combination of both hardware and firmware (software).	1	1	
I.2	PINx, PORTx	0.5 x 2	1	
I.3	DDRB = 0b0000 0100 or 0x04	1	1	
I.4	AND (&), OR( ), Inverter (~), Shift operators (<<, >>)	1	1	
I.5	0001 0100	1	1	
I.6	RS	1	1	
I.7	The resolution of an A/D converter is defined as the smallest change in the input that can be detected by ADC.	1	1	
I.8	A TCB is used for holding the information corresponding to a task.	1	1	
I.9	A process is a program or part of it in execution.	1	1	
	<b>PART B</b>			24
II.1	<ul style="list-style-type: none"> <li>● Stores the program instructions.</li> <li>● Retains its contents even after the power is turned off.</li> <li>● It is generally known as Non volatile storage memory.</li> <li>● ATmega32 has 32K x 8bytes program ROM.</li> <li>● Depending on the fabrication, erasing and programming techniques they are classified as Masked ROM, EPROM, EEPROM etc.</li> </ul>	3	3	

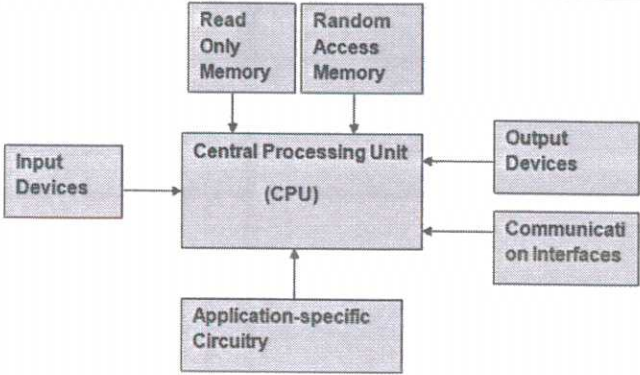
II.2	<b>General Purpose Computing System</b>	<b>Embedded System</b>	Any 3 comparison x 1 mark	3																		
	For executing a variety of applications.	For executing a specific set of applications.																				
	Contains a General Purpose Operating System (GPOS)	May or may not contain an operating system for functioning.																				
	Applications are programmable by the user.	The firmware is pre-programmed and it is non-alterable by the end user.																				
	Performance is the key deciding factor in the selection of the system .	Application-specific requirements are the key deciding factors.																				
	Response requirements are not time-critical.	For certain category, response time is highly critical.																				
	Need not be deterministic in execution behaviour.	Execution behaviour is deterministic for certain types of embedded systems.																				
	Reduced operating power requirements.	Highly power saving modes supported by the hardware and Operating system.																				
II.3	<p>AVR has a flag register to indicate arithmetic conditions. The flag register in AVR is called Status register (SReg). Status register is an 8-bit register.</p> <p>The bits C, Z, N, V, S and H are called conditional flags, they indicate some conditions that results after an instruction is executed.Each conditional flag perform a conditional branch (jump).</p> <table><tr><td>Bit</td><td>D7</td><td></td><td></td><td></td><td></td><td></td><td></td><td>D0</td></tr><tr><td>SREG</td><td>I</td><td>T</td><td>H</td><td>S</td><td>V</td><td>N</td><td>Z</td><td>C</td></tr></table> <p>C – Carry flag                      S – Sign flag Z – Zero flag                        H – Half carry N – Negative flag                  T – Bit copy storage V – Overflow flag                  I – Global Interrupt Enable</p>		Bit	D7							D0	SREG	I	T	H	S	V	N	Z	C	Listing all flags (1 mark) + expln.2	3
Bit	D7							D0														
SREG	I	T	H	S	V	N	Z	C														

II.4	<pre>#include&lt;avr/io.h&gt;  int main(void) {     DDRA = 0xFF;     While(1) {         PORTA = ~PORTA : }     }</pre>	Output port setting (1 mark) + Logic using inverter (2 marks)	3	
II.5	<p>In Normal Mode, the timer counts upward from 0 to its maximum value (usually 0xFF for an 8-bit timer or 0xFFFF for a 16-bit timer) and then wraps around to 0.</p> <p>In CTC Mode, the timer counts up from 0 to a specified compare value (OCRn), and then it automatically resets back to 0.</p> <p>The WGM bits in the TCCRx register is used to set the mode.</p>	1.5 x 2	3	
II.6	<p>The SEI (Set Global Interrupt Flag) instruction enables all interrupts globally by setting the Global Interrupt Flag (I) in the Status Register (SREG). When the I bit is set, interrupts are enabled.</p> <p>The CLI (Clear Global Interrupt Flag) instruction disables all interrupts globally by clearing the Global Interrupt Flag (I) in the Status Register (SREG). When the I bit is cleared, interrupts are disabled</p>	1.5 x 2	3	
II.7	<ul style="list-style-type: none"> <li>● Low Power Consumption</li> <li>● Compact and Thin Design</li> <li>● Wide Range of Sizes and Resolutions</li> <li>● Cost-Effective</li> <li>● Longevity and Reliability</li> </ul>	Any 3 x 1 mark	3	
II.8	<p>LM35 is a temperature measuring device having an analog output voltage proportional to the temperature. It provides output voltage in Centigrade (Celsius). It does not require any external calibration circuitry.</p>	Use 1 mark + Diagram 2 mark	3	

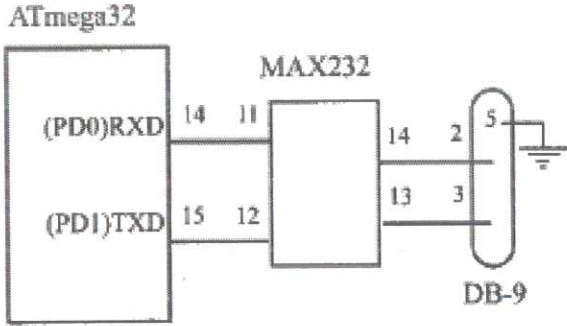




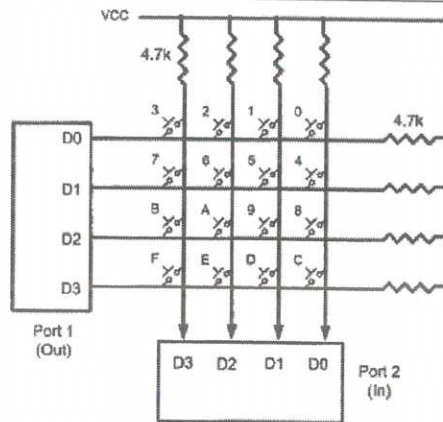
II.9	<p>A deadlock is a situation in which two or more processes (or threads) are unable to proceed because each is waiting for the other to release a resource. In a deadlock, none of the processes can make progress, and the system becomes effectively stuck. Deadlocks are common in multi-process or multi-threaded systems and can lead to significant issues if not properly managed.</p> <p>Example:</p> <p><u>Process A:</u></p> <p>Acquires Resource X.</p> <p>Attempts to acquire Resource Y.</p> <p><u>Process B:</u></p> <p>Acquires Resource Y.</p> <p>Attempts to acquire Resource X.</p>	2 + 1 mark for example	3	
II.10	<p>Multi-processing involves the execution of multiple independent processes by utilizing multiple CPUs or CPU cores concurrently</p> <p>Multi-tasking is the capability of an operating system to execute multiple tasks (also known as threads or processes) within a single CPU or CPU core by rapidly switching between them.</p>	1.5 x 2	3	
	<b>PART C</b>			42
III	<ul style="list-style-type: none"> <li>● Consumer electronics - Camcorders, camera, etc.</li> <li>● Household appliances - Television, DVD players, washing machine, fridge, microwave oven, etc.</li> <li>● Home automation and security system - Air conditions, sprinklers, intruder detection alarms, closed circuit television cameras, fire alarms, etc...</li> <li>● Automotive industry - Anti-lock braking systems (ABS), engine control, ignition systems, automatic navigation system, etc.</li> <li>● Telecom - Cellular telephones, telephone switches, handset</li> </ul>	Any 4 applicatio ns	7	

	multimedia applications, etc.			
IV	 <p>Building blocks of an embedded system are :</p> <ol style="list-style-type: none"> <li>1. Input Devices</li> <li>2. Output Devices</li> <li>3. Central Processing Unit</li> <li>4. Communication Interfaces</li> <li>5. Memory (ROM and RAM)</li> <li>6. Application Specific Circuit</li> </ol> <p>The controller can be a microprocessor or a microcontroller or a field programmable gate array or a digital signal processor or an application specific integrated circuit.</p> <p>Embedded hardware/software systems are designed to regulate a physical variable by sending control signals to the actuators connected to the o/p port of the system.</p> <p>Keyboards, push button, switches, etc. are Examples of common user interface input devices.</p> <p>The memory of the system is responsible for holding the code.</p>	Listing 2 marks + Expln. 5 marks	7	
V	<pre>#include&lt;avr/io.h&gt;w #include&lt;util/delay.h&gt; int main(void) { DDRA = 0xFF; While(1) {     for( z=0;z&lt;8;z++)     {         PORTA = PORTA &lt;&lt; z ;         _delay_ms(1000);     } }</pre>	Correct port setting and Syntax - 2 marks + Logic using shift operator - 4 marks + Delay function	7	

	<pre> } } </pre>	- 1 mark		
VI	<pre> #include&lt;avr/io.h&gt;  int main(void) {     DDRB = DDRA = 0xFF;     Unsigned char x,y;     Unsigned char bcdbyte = 0x34;     X=bcdbyte &amp; 0x0F;     PORTA = X   0x30 ;     Y = bcdbyte &amp; 0xF0;     Y = Y&gt;&gt;4;     PORTB = Y   0x30 ; } </pre>	<p>Correct ports setting and Syntax - 2 marks</p> <p>+ Logic for conversion - 5 marks</p>	7	
VII	<p><u>Steps to program Timer0 in normal mode</u></p> <ol style="list-style-type: none"> <li>1. Load TCNT0 register with initial count value.</li> <li>2. Load TCCR0 register, indicating which mode (8 or 16 bit) is to be used and the prescaler option.</li> <li>3. When the clock source is selected, the timer starts to count and each tick causes the content of the timer to increment by one.</li> <li>4. Keep monitoring the Timer Overflow flag bit (TOV0) to see if it is raised. Get out of the loop when TOV0 is high.</li> <li>5. Stop the timer by disconnecting the clock source using instructions.</li> <li>6. Clear the TOV0 flag for the next round.</li> <li>7. Go back to step 1 to load TCNT0 again.</li> </ol>	7	7	
VIII	<p>An Interrupt Service Routine (ISR), also known as an Interrupt Handler or Interrupt Handler Routine, is a specialized subroutine designed to handle interrupts generated by hardware devices or software conditions that require immediate attention from the CPU.</p> <p>Depending on which peripheral is incorporated into the chip, widely used sources of interrupts in the AVR are:</p> <ul style="list-style-type: none"> <li>● Timer/Counter Overflow interrupts</li> <li>● Three external hardware interrupts INT0 ,INT1, and INT2.</li> <li>● Serial communication's USART has three interrupts - one for</li> </ul>	<p>ISR (2 marks)</p> <p>+ Sources (5 marks)</p>		

	<p>receive and two interrupts for transmit.</p> <ul style="list-style-type: none"> <li>● Analog Comparator Interrupt</li> <li>● Watchdog Timer Interrupt</li> <li>● The SPI interrupts.</li> <li>● The ADC (analog-to-digital converter) conversion complete interrupt.</li> </ul>			
IX	<p><b>ATmega32</b></p>  <p><b>40-Pin DIP Package ATmega32</b></p> <ul style="list-style-type: none"> <li>● RS232 is one of the most widely used serial I/O interfacing standards.</li> <li>● In RS232, a 1 bit is represented by -3 to -25V and a 0 bit is represented by +3 to +25 volts which is not compatible with TTL logic levels.</li> <li>● MAX232 is used to convert TTL logic levels to RS232 voltage levels and vice versa.</li> <li>● The ATmega32 has two pins that are used specifically for transferring and receiving data serially.</li> <li>● These two pins are called TX and RX and are part of the PortD group (PD0 and PD1) of the 40-pin package.</li> <li>● Pin15 of the ATmega32 is assigned to TX and pin 14 is designated as RX.</li> </ul>	<p>Diagram (3 marks) + Expln. (4 marks)</p>		





### Keyboard interfacing

- The rows are connected to an output port and the columns are connected to an input port.
- If no key has been pressed, the input port will yield 1s for all columns since they are all connected to high(VCC).
- If all the rows are grounded and a key is pressed, one of the column will have 0 since the key pressed provides the path to ground.

### Steps to detect a key press:

X

- To detect a pressed key, the microcontroller grounds all rows by providing 0 to the output latch, and then it reads the columns.
- If the data read from the columns is D3-D0 = 1111, no key has been pressed and the process continues until a key press is detected.
- If one of the column bits has a zero, this means that a key press has occurred.
- After a key press is detected, the microcontroller will go through the process of identifying the key.
- Starting with the top row, the microcontroller grounds it by providing a low to row D0 only; then it reads the columns.
- If the data read is all 1s, no key in that row is activated and the process is moved to the next row.
- It grounds the next row, reads the columns, and checks for any zero. This process continues until the row is identified.
- After identification of the row in which the key has been pressed and find out which column the pressed key belongs to.
- After identifying the row and column, the microcontroller will identify which key has been pressed.

Expln.

(4 marks)

+

Diagram

(3 marks)

7

XI

### 1. Process Management

- Manages processes/tasks.

Any 4  
functions

7



	<ul style="list-style-type: none"> <li>● Manages the memory space for process, load the process code into memory, allocate system resource etc</li> </ul> <p><b>2. Primary memory management</b></p> <ul style="list-style-type: none"> <li>● Primary memory is the volatile memory where the processes are loaded and variables and shared data associated with each process shared.</li> <li>● The MMU(Memory Management Unit) is responsible for Keeping track of which part of memory used by which process</li> <li>● Allocating and deallocating memory space</li> </ul> <p><b>3. File system Management</b></p> <ul style="list-style-type: none"> <li>● The file system management responsible for creation, deletion and alteration of files and directories</li> <li>● Saving file in secondary storage</li> <li>● Provide automatic allocation of file space</li> </ul> <p><b>4. I/O System management</b></p> <ul style="list-style-type: none"> <li>● Responsible for routing the I/O requests coming from different user applications to the appropriate I/O devices of the system</li> </ul> <p><b>5. Secondary storage management</b></p> <ul style="list-style-type: none"> <li>● Manages the secondary storage memory devices connected to the system and deals with Disk storage allocation, Disk scheduling, Free Disk space management.</li> </ul> <p><b>6. Protection systems</b></p> <ul style="list-style-type: none"> <li>● Support multiple users with different levels of permission</li> <li>● Protection deals with implementing the security policies to restrict access to both user and system resources by different applications or processes or users</li> </ul> <p><b>7. Interrupt handler</b></p> <ul style="list-style-type: none"> <li>● Provides mechanism for all external/internal interrupts generated by the system.</li> </ul>			
XII	<p>There should be some mechanism in place to share the CPU among the different tasks and to decide which process/task is to be executed at a given point of time. Determining which task /process is to be executed at a given point of time is known as task/process scheduling. The kernel service/application, which implements the scheduling algorithm is known as Scheduler.</p>	<p>Defn. 3 marks + Algorithm with Example 4 marks</p>	7	

	<p><b>Algorithms</b></p> <p>1. FCFS scheduling</p> <ul style="list-style-type: none"> <li>- Allocates CPU time to the processes based on the order in which they enter the 'Ready' queue.</li> </ul> <p>2. SJF</p> <ul style="list-style-type: none"> <li>- The process with the shortest estimated run time is scheduled first followed by the next shortest process</li> </ul> <p>3. Round Robin scheduling etc</p> <ul style="list-style-type: none"> <li>- Each process in the Ready queue is executed for a predefined time slot</li> </ul> <p>The execution starts with picking up the first process in the Ready queue .</p> <p>It is executed for a predefined time and when the predefined time elapses or the process completes, the next process in ready queue is selected for execution.</p>			
XIII	<p><b>Functional Requirements</b></p> <p>1. Processor support</p> <ul style="list-style-type: none"> <li>- It is not necessary that all RTOS's support all kinds of processor architecture</li> </ul> <p>2. Memory Requirements</p> <ul style="list-style-type: none"> <li>- The OS requires ROM memory for holding OS files and it is normally stored in a non volatile memory like FLASH.</li> </ul> <p>3. Real-time capabilities</p> <ul style="list-style-type: none"> <li>- It is not mandatory that the OS for all embedded systems need to be Real time and all embedded OS are Real time in behaviour</li> </ul> <p>4. Kernel and Interrupt Latency</p> <ul style="list-style-type: none"> <li>- The kernel of the OS may disable interrupts while executing certain services and it may lead to interrupt latency</li> <li>- IPC and Task synchronization</li> <li>- It is OS kernel dependant</li> </ul> <p>5. Modularisation support</p> <ul style="list-style-type: none"> <li>- Developer can choose essential modules and recompile the OS image for functioning</li> </ul> <p>6. Support for Networking and Communication</p> <ul style="list-style-type: none"> <li>- Kernel may provide stack implementation and driver support for</li> </ul>			

	<p>a bunch of communication interfaces and networking</p> <p>6. Development Language Support</p> <p>- Support for development languages like java and C#. JVM required for running java applications.</p>			
XIV	<p>Task communication in a multitasking system refers to the methods and mechanisms through which different tasks or processes running concurrently on a computer or within an operating system can exchange data, coordinate their activities, and interact with each other.</p> <p>Some methods are :</p> <p><b>Shared Memory</b></p> <ul style="list-style-type: none"> <li>Processes share some area of the memory to communicate among them.</li> <li>Information to be communicated by the process is written to the shared memory area</li> </ul> <p><b>Message Passing</b></p> <ul style="list-style-type: none"> <li>Synchronous information exchange mechanism used for Inter process/ Thread Communication.w</li> <li>Direct message passing – The sender and the receiver of the messages are explicitly defined.</li> <li>Indirect message passing – The messages are placed in structures such as message queues or mailboxes and multiple tasks have read/write access.</li> </ul> <p><b>Signals:</b></p> <ul style="list-style-type: none"> <li>Signals are notifications sent by one task or process to another to indicate events or request specific actions. Tasks can send and receive signals to coordinate their activities.</li> </ul> <p><b>Remote Procedure Call</b></p> <ul style="list-style-type: none"> <li>Inter process communication mechanism used by a process to call a procedure of another process running on the same CPU or on a different CPU which is interconnected in a network.w</li> <li>Powerful technique for constructing distributed, client-server based applications.</li> </ul>	<p>Any 3 methods listing 1 mark + Expln. 6 marks</p>	7	