Scoring Indicators

COURSE NAME : Embedded System and Real time Operating System

COURSE CODE : 5131 **QID :** 21092230284

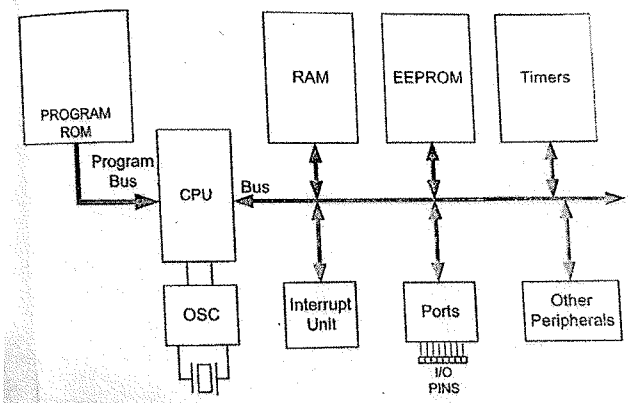
PART A

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

(9 x 1 = 9 Marks)

Q.No.	Scoring Indicators	Max. marks Split score	Sub Total	Total score
	PART A			9
I.1	An embedded system is an electronic/electro-mechanical system designed to perform a specific function and is a combination of hardware and software.		1	
I.2	PORTx Registers		1	
I.3	0xff or 0b11111111 or decimal 255 (any one)		1	
I.4	unsigned int,int, unsigned char,char ,unsigned long,long,float,double (any four)		1	
I.5	Normal mode , CTC mode,PWM mode (any two)		1	
I.6	Initialise LCD,send commands,Send Data to LCD		1	
I.7	Resolution,speed,dynamic range, resistance of load (any two)		1	
I.8	Job queue,Ready queue,Device queue		1	
I.9	Custom Developed or off the shelf, cost, ease of use,Development and debugging tool availability,After sales(Any three points)		1	
	PART B			24
II.1	Status Registers, timer registers, serial communication, I/O port ,ADC,DAC etc		3	
II.2	A sensor is a device that converts energy from one form to another for any measurement or control purpose. Actuators Actuator is a form of transducer device (mechanical or electrical) which converts signals to corresponding physical action (motion). Actuator acts as an output device		3	
II.3	Based on Generation Complexity and performance requirements Deterministic behavior Based on Triggering		3	
II.4	INT0, INT1,INT2		3	

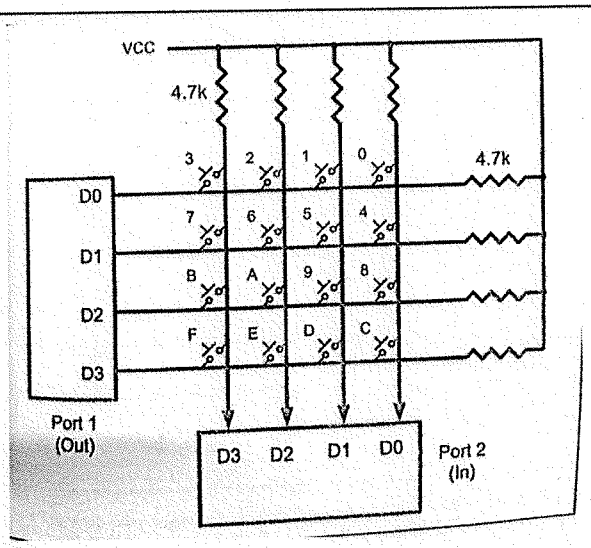
II.5	Sources of interrupts in the AVR 1. There are at least two interrupts set aside for each of the timers, one for over- flow and another for compare match. 2 Three interrupts are set aside for external hardware interrupts. Pins PD2 PORTD.2), PD3 (PORTD.3), and PB2 (PORTB 2) are for the external hard- ware interrupts INTO, INT1, and INT2, respectively. 3. Serial communication's USART has three interrupts, one for receive and two interrupts for transmit. 4. The SPI (serial peripheral Interface)interrupts. (any 3)	1x3	3	
II.6	For moving LSB first:- compare LSB is 1 or 0 by masking all other bits(AND operation with 0x01) if it is 1 ,set the value 1 at the pin where the serial output is taken else reset to 0 at the pin where the serial bit is taken shift the data by one bit right repeat the above steps 8 times (until a byte of data moved)		3	
II.7	Detect a valid key press Identify the key by scanning each row Send the scan code of pressed key		3	
II.8	Resolution speed reference Voltage conversion time accuracy Response		3	
II.9	Threads A thread is the primitive that can execute code. A thread is a single sequential flow of control within a process. "Thread' is also known as light- weight process. A process can have many threads of execution. Different threads, which are part of a process, share the same address space; meaning they share the data memory, code memory and heap memory area. Threads maintain their own thread status (CPU register values), Program Counter (PC) and stack.		3	

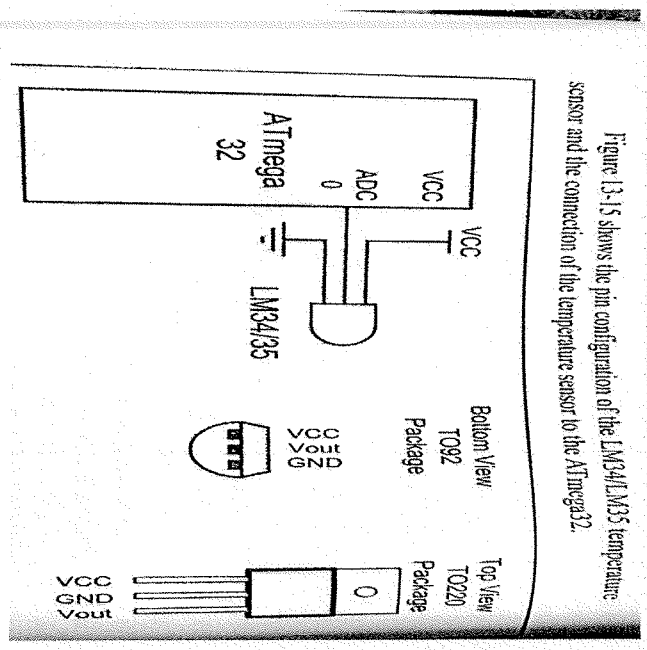
II.10	Co operative multitasking pre emptive multitasking Non pre emptive multitasking brief description		3	
	PART C			42
III.11	 1-2. Simplified View of an AVR Microcontroller **Explanation**	figure 4 3	7	
III.2.	**Purpose** **1 Data collection/Storage/Representation** **2. Data communication** **3. Data (signal) processing** **4. Monitoring** **5. Control** **6. Application specific user interface** **Applications of embedded systems:** **Home automation systems, industrial control systems, medical devices such as pacemakers and insulin pumps.autonomous vehicle control systems, navigation systems, communication systems etc(any 8)**	3 4	7	

III.3	**Solution:**
	`#include <avr/io.h>` `//standard AVR header`
	`int main(void)`
	`{`
	`unsigned char x, y;`
	`unsigned char mybyte = 0x29;`
	`DDRB = DDRC = 0xFF;` `//make Ports B and C output`
	`x = mybyte & 0x0F;` `//mask upper 4 bits`
	`PORTB = x | 0x30;` `//make it ASCII`
	`y = mybyte & 0xF0;` `//mask lower 4 bits`
	`y = y >> 4;` `//shift it to lower 4 bits`
	`PORTC = y | 0x30;` `//make it ASCII`
	`return 0;`
	`}`
	Note:
	Sample value here is 0x29
	They can use any value
			7	

III.4	`#include <avr/io.h> // standard AVR header`
	`void our_delay();`
	`;int main(void)`
	`{unsigned int i;`
	`DDRB = 0xff; // setting PORTB as output`
	`For(i=0;i<=255;i++) //for loop`
	`{`
	`PORTB = i; // setting value for PORTB`
	`our_delay((1000);`
	`}`
	`void our_delay()`
	`{`
	`TCNTO = 0x20; //load TCNT with initial count value`
	`TCCRO = 0x01; //Timer0 Normal mode`
	`while ((TIFR & 0x01)==0); //wait until timer0 //overflows`
	`TCCR0=0; //Stop timer`
	`TIFR = 0; //Clear TF0`
	`}`			

III.5	TCCR(Timer/Counter Control Register) TCNTR (Timer/Counter Register) OCR(Output Compare register) **TIFR(Timer/Counter Interrupt Flag Register)** **Explain each**		7	

III.6	**Steps in executing an interrupt activation of an interrupt, the microcontroller goes through the following steps** **1. It finishes the instruction it is currently executing and saves the address of the next instruction (program counter) on the stack.** **2 It jumps to a fixed location in memory called the interrupt vector table The interrupt vector table directs the microcontroller to the address of the interrupt of the service routine (ISR)** **3. The microcontroller starts to execute the interrupt service subroutine until it reaches the last instruction of the subroutine, which is RETI (return from inter rupt).** **4. Upon executing the RETI instruction, the microcontroller returns to the place where it was interrupted.**			
III.7	 **Explanation**	4 3	7	

III.8	 Figure 13-15 shows the pin configuration of the LM34/LM35 temperature sensor and the connection of the temperature sensor to the ATmega32.	3	7	
	Explanation	4		
III.9	**The mechanism through which processes/tasks communicate with each other is known as Inter Process/Task Communication (IPC). Inter Process Communication is essential for process coordination. The various types of Inter Process Communication (IPC) mechanisms adopted by processes are kernel (Operating System) dependent.** **In a multitasking system, multiple tasks processes run concurrently in pseudo parallelism and processes may or may not interact between. Based on the degree of interaction, the processes in an OS are classified as** **Co-operating Processes: In the co-operating interaction model one process requires the inputs from other processes to complete its execution** **Competing Processes: The competing processes do not share anything among themselves but they share the system resources. The competing processes compete for the system resources such as file, display device, etc. Co-operating processes exchanges information and communicate through the following methods** **Co-operation through Sharing: The co-operating process exchange data through some shared resources** **Co-operation through Communication: No data is shared between the processes. But they communicate for synchronisation.** **Some of the important IPC mechanisms adopted by various kernels are** 1. **mail box** 2. **message queue**		7	

III.10	**The selection of a scheduling criterion/algorithm should consider the following factors** **CPU Utilisation: The scheduling algorithm should always make the CPU utilisation high. tion is a direct measure of how much percentage of the CPU is being utilised.** **Throughput: This gives an indication of the number of processes executed per unit of time.The throughput for a good scheduler should always be higher.** **Turnaround Time: It is the amount of time taken by a process for completing its execution the time spent by the process for waiting for the main memory, time spent in the ready queue on completing the I/O operations, and the time spent in execution. The turnaround time s minimal for a good scheduling algorithm.** **Waiting Time: It is the amount of time spent by a process in the 'Ready queue waiting to time for execution. The waiting time should be minimal for a good scheduling algorithm.** **Response Time: It is the time elapsed between the submission of a process and the first response. For a good scheduling algorithm, the response time should be as least as possible.** **To summarise, a good scheduling algorithm has high CPU utilisation, minimum Turn around time and response time and max throughput**			
III.11	**Task synchronization issues:** **Deadlock** **Mutual exclusion** **Hold and wait** **No resource pre-emption** **Circular wait** **Explain each**	7		
III.12	**In the operating systems context multiprocessing describes the ability to execute multiple processes simultaneously. Systems which are capable of performing multiprocessing, are known as multiprocessor systems. Multiprocessor systems possess multiple CPUs and can execute multiple processes simultaneously** **The ability of the operating system to have multiple programs in memory, which are ready for execution, is referred to as multiprogramming. In a uniprocessor system, it is not possible to execute processes simultaneously.**			