



**MANIPAL INSTITUTE OF TECHNOLOGY**

**MANIPAL**

(A constituent unit of MAHE, Manipal)

**DEPARTMENT OF MECHATRONICS**

**IV SEMESTER B.TECH. (MECHATRONICS)**

**END-TERM EXAMINATION**

**SUBJECT: Microcontroller-based System Design**

**Subject Code: MTE 2225**

**Date: 5-5-2024**

**Time: 3 Hrs**

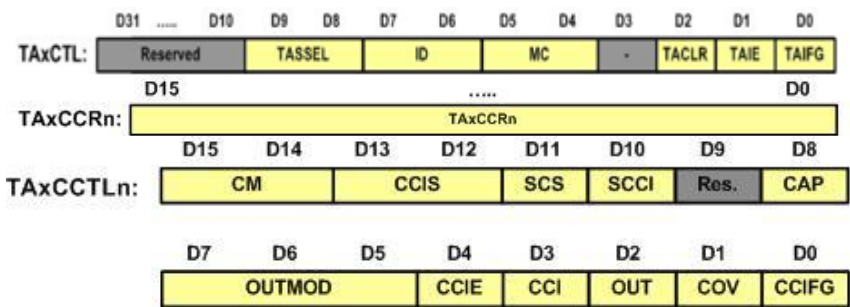
**Exam Time 2:30 PM – 5:30 PM**

**MAX. MARKS: 50**

❖ Answer **ALL** the questions. Assume data suitably if missing.

Q. No.		M	CO	PO	LO	BL
1(a)	You have been tasked with designing an encryption algorithm to securely transmit large amounts of data between two electronic devices. In this algorithm, each transmitted signal requires a new key generated by summing the two previous keys. You aim to develop an assembly language program to implement this encryption algorithm and test its effectiveness. The program should accept two initial values of Keys 10 and 30 and should encrypt data up to a length of twenty. Additionally, all generated keys should be stored at memory location 0x20000000. Assume that the electronic devices have limited processing power and memory and that the encryption and decryption processes must be efficient. Also, assume that the transmitted data is sensitive and requires high security. Furthermore, the steps to perform the encrypted algorithm must be written.	5	2	5	3	5
1(b)	Analyze the risk and ethical evaluation of autonomous road vehicle controllers.	3	4	8	9	4
1(c)	A 3-stage pipelined CPU runs a 3-iteration of the following code.  MOV R1, #2 L1 AND R2, R2, #1 B L1 MOV R3, #3 MOV R4, #4 (a). Determine the number of instruction cycles the CPU requires to execute the program with the help of a pipelining diagram.  (b). Also, calculate the number of branch penalties.	2	1	1	1	3
2(a)	You are tasked with programming a microcontroller to read temperature data from a sensor. The program should precisely determine whether the current temperature reading is even or odd. Develop an assembly language program to find whether the temperature is even or odd. Sum all the even and odd temperatures	5	2	5	3	3

	separately. Save the results in the separate memory locations 0x20000000 and 0x2FFFFFFF, respectively					
<b>2(b)</b>	Estimate the address space range of each of the following memory of an ARM chip: (a) 2 KB of EEPROM starting at address 0x80000000 (b) 16 KB of SRAM starting at address 0x90000000 (c) 64 KB of Flash ROM starting at address 0xF0000000	<b>3</b>	<b>2</b>	<b>1</b>	<b>1</b>	<b>3</b>
<b>2(c)</b>	Develop an assembly language program to calculate the factorial of 6 for MSP432P401R and save the result into the memory location 0x20000000.	<b>2</b>	<b>2</b>	<b>2</b>	<b>2</b>	<b>3</b>
<b>3(a)</b>	<p>Develop an Embedded C program that can blink an LED connected to a GPIO pin P.2.1 on a microcontroller using the SysTick timer. The program should be able to adjust the blink frequency of 3MHz. Instructions:</p> <ol style="list-style-type: none"> <li>1. Set up the GPIO pin connected to the LED as an output pin.</li> <li>2. Initialize the SysTick timer using the specified delay value of 1 millisecond.</li> <li>3. Establish a continuous loop to perpetually blink the LED by repeating step 3.</li> </ol> <p>whereas SysTick Registers are</p>	<b>5</b>	<b>3</b>	<b>5</b>	<b>3</b>	<b>5</b>
<b>3(b)</b>	<p>Develop an assembly code for adding numbers lying between 0x35 and 0x75 from the memory location.</p> <p>Saved numbers in memory: 0x27, 0x35, 0x76, 0x84, 0x33, 0x92, 0x43, 0x63, 0x97, 0x51.</p>	<b>3</b>	<b>2</b>	<b>2</b>	<b>2</b>	<b>3</b>
<b>3(c)</b>	<p>Estimate the contents of R2, R1, and memory location 0x20 after the following program:</p> <pre> MOV R2,#0x5 ;load R2 with 5 (R2 = 0x05) MOV R1,#0x2 ;load R1 with 2 (R1 = 0x02) ADD R2, R1,R2 ;R2 = R1 + R2 ADD R2,R1,R2 ;R2 = R1 + R2 MOV R5,#0x20 ;R5 = 0x20 STRB R2,[R5] ;store R2 into location pointed to by R5 </pre>	<b>2</b>	<b>2</b>	<b>1</b>	<b>1</b>	<b>3</b>
<b>4(a)</b>	You have a robotic wheel that needs to move forward and backward with a duty cycle of 30% for five milliseconds. Write an embedded C program that generates a 30% square wave and uses it to control the direction of the wheel. The robotic wheel should move forward	<b>5</b>	<b>3</b>	<b>5</b>	<b>3</b>	<b>4</b>

	<p>during the on-time of the square wave and backward during the off-time of the square wave.</p> 					
4(b)	<p>Examine the following code and give the result in R0, R1, and R2 registers.</p> <pre> MOV R1,#0 ; MOV R0,#0 ; LDR R2,=0x99999999 ; ADDS R0,R0,R2 ; BCC L1 ;if C = 0, ADDS R1,R1,#1 ; L1 ADDS R0,R0,R2 ; BCC L2 ;if C = 0, ADDS R1,R1,#1 ; L2 ADDS R0,R2 ; BCC L3 ;if C = 0, ADDS R1,R1,#1 ; L3 ADDS R0,R2 ; BCC L4 ;if C = 0, ADDS R1,R1,#1 ; L4 </pre>	3	2	1	1	3
4(c)	<p>Estimate the status of flags NZCV during the execution of the following program:</p> <pre> MOV R2,#4 ; MOV R3,#2 ; MOV R4,#4 ; SUBS R5,R2,R3 ; SUBS R5,R2,R4 ; </pre>	2	2	2	2	3
5(a)	<p>The following hardware and software requirements are needed for the MSP432P401R microcontroller:</p> <p><b>Hardware Components:</b></p>	4	3	2	2	3

	<ol style="list-style-type: none"> <li>1. Microcontroller with GPIO pins</li> <li>2. Red, green, and blue LEDs connected to GPIO pins 2.0, 2.1 and 2.2.</li> <li>3. Push buttons SW1 and SW2 connected to GPIO pins 1.1 and 1.4.</li> </ol> <p><b>Software Requirements:</b></p> <ol style="list-style-type: none"> <li>1. Continuously toggle the red LED on pin P2.0.</li> <li>2. When SW1 is pressed, blink the green LED on pin P2.1 three times.</li> <li>3. When SW2 is pressed, blink the blue LED on pin P2.2 three times.</li> <li>4. The main program should toggle the red LED while waiting for interrupts from SW1 or SW2.</li> </ol> <p>The register and functions definition is as follows</p> <p>NVIC_SetPriority()  NVIC_EnableIRQ()  NVIC_DisableIRQ()  __enable_irq()  __disable_irq()  IES: register for edge activation  IE: register for interrupt enable  IFG: Interrupt flag register  IRQ: for Port 1, is 35 or PORT1_IRQn</p> <p>Estimate the content of all the registers to set up the above requirements.</p>					
<b>5(b)</b>	Develop an embedded C code for the MSP432P401R microcontroller to manage LED states based on button inputs using the above question 5(a) specification.	<b>4</b>	<b>3</b>	<b>5</b>	<b>3</b>	<b>3</b>
<b>5(c)</b>	Evaluate the value for TAxCTL if we want to program Timer_A in continuous mode with no clock division. Use ACLK for the clock source.	<b>2</b>	<b>3</b>	<b>1</b>	<b>1</b>	<b>2</b>