Reg. No.										

MANIPAL INSTITUTE OF TECHNOLOGY

MANIPAL

(A constituent unit of MAHE, Manipal)

DEPARTMENT OF MECHATRONICS ENGINEERING

VII SEMESTER B.TECH. (MECHATRONICS)

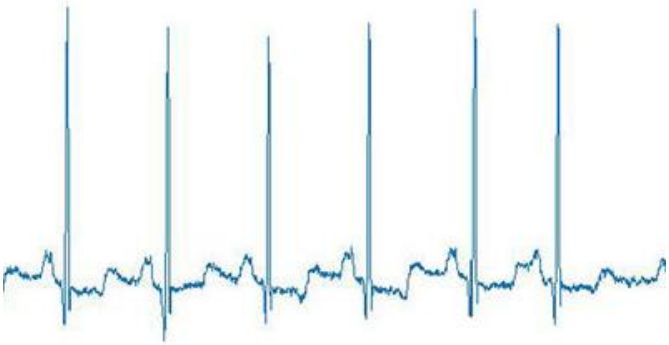
END SEMESTER EXAMINATIONS, Jan, 2021

SUBJECT: MECHATRONICS SYSTEM DESIGN [MTE 4101]

Time: 3 Hours **MAX. MARKS: 50**

Instructions to Candidates:

- Answer **ALL** the questions.
- Any data not provided may suitably assumed.

Q. No	Question	M	CO	PO	LO	BL
1A.	Mention the importance of design for embedded systems in automation.	**2**	**1**	**1**	**1**	**2**
1B.	Recall IEC regulators for equipped unit for control and address Functional Safety Management for Embedded System Design.	**3**	**1, 4**	**6**	**5**	**2**
1C.	Summarize the importance of Hardware Reliability Test (HRT) with Mean Time to Dangerous Failure and recall IEC 61000-6-2 in HRT.	**5**	**1**	**6**	**5**	**2**
2A.	Consider the digital filter as mentioned below to implement for real time applications such as detecting the peak of ECG signal. Write an assembly language code that calculates y[k] = 1*x[k+1] +0*x[k]-1*x[k-1]. Assume that the address 0x2000000 onwards stores the data x[] and the output is stored into the memory 0x20000100 onwards. Mention the outcome in the application mentioned.  Fig 2A. Sample ECG signal.	**6**	**2**	**1, 2**	**1, 2**	**3, 4**
2B.	What is nested interrupt. Mention the role of stack in the nested interrupt.	**4**	**3**	**1, 2**	**1, 2**	**2**

3A.	Write an Embedded C code to set clock frequency at 48MHz and map the frequency to T32 module. The handler function of T32 runs a code for rotating LIDAR in clockwise direction attached to an autonomous driving vehicle. Assume LIDAR rotation is connected via DC Motor (brushless) connected to P5.2 and P1.1 on MSP432P401R.	**6**	**3**	**1, 2**	**1, 2**	**3**
3B.	Write the difference between Handler and Thread Mode with associated registers.	**2**	**3**	**1, 2**	**1, 2**	**2**
3C.	Mention the clock sources available in MSP432PXXX and list of the steps required to map a block with 32KHz.	**2**	**3**	**1, 2**	**1, 2**	**2**
4A.	Explain the ADC circuit called Conversion by Successive Approximation integrated in the MSP432 launch pad with neat flow diagram.	**4**	**3**	**1, 2**	**1, 2**	**2**
4B.	MSP432P401R is used by a company to develop a Washing Machine. Develop an Embedded C code to address the manufacturers specifications. Use Timer A module for this operation. Condition 1. Power ON and OFF is connected to P2.4 2. Auto Button is connected to P5.1 3. Manual Button is connected to P3.5. 4. In Auto mode each operation is changed automatically after 3 seconds and each mode duration lasts for 120 seconds. 5. In manual mode next process waits for user to set and each mode lasts for 120 seconds, but if any switch is provoked the existing operation is terminated and follows the user command. 6. The motor is connected to P1.1 and P1.2 to spin clockwise and counterclockwise for each 3 seconds.	**6**	**4**	**1, 2**	**1, 2**	**3**
5A.	Establish communication for sending the information say “g” to glow the LED of another board, and “s” to stop the glowing. Use two MSP432 Launchpad boards. The first MSP432 Launchpad will be used for the button. The second will be used for the LEDs. Establish a digital communication link between these two boards using I2C communication mode. Use the respective cheat sheet given below.	**6**	**3**	**1, 2**	**1, 2**	**3**
5B.	Analyze the following Embedded C code for traffic signal management and rewrite the correct code. Each road has Red, Yellow and Green signals connected to: P1.0, P1.1, P1.2 Street 1 P2.0, P2.1, P2.2 Street 2 P3.0, P3.1, P3.2 Street 3 P4.0, P4.1, P4.2 Street 4 on MSP432P401R.	**4**	**4**	**1, 2**	**1, 2**	**3, 4**

The code is having 70% practical correctness and 100% theoretical correctness. Strictly use the mentioned thread and handler modes.

```
#include"msp.h"
Int i=1;
Void main(void)
        {
        P1->DIR = BIT0|BIT1|BIT2;
        P2->DIR = BIT0|BIT1|BIT2;
        P3->DIR = BIT0|BIT1|BIT2;
        P4->DIR = BIT0|BIT1|BIT2;
        P1->OUT = 0x00;
        P2->OUT = 0x00;
        P3->OUT = 0x00;
        P4->OUT = 0x00;
        SysTick->CTRL |=
        SysTick_CTRL_CLKSOURCE_Msk|SysTick_CTRL
        _ENABLE_Msk;
        SysTick->LOAD = 0x5B8D80;
        SysTick->VAL = 0x01;
        SysTick->CTRL |= SysTick_CTRL_TICKINT_Msk;
        __enable_irq();
        P1->OUT &= BIT1;
        while(1)
        {
        if(i==5)
        {
        i=1;
        }
        }
}
voidSysTick_Handler(void)
{
        if(i== 1)
        {
                P1->OUT &= BIT1;
                P2->OUT &= BIT0|BIT2;
                P3->OUT &= BIT0|BIT2;
                P4->OUT &= BIT0|BIT2;
                i++;
        }
        if(i==2)
        {
                P1->OUT &= BIT0|BIT2;
                P2->OUT &= BIT1;
                P3->OUT &= BIT0|BIT2;
                P4->OUT &= BIT0|BIT2;
                i++;
        }
        if(i==3)
        {
```

```
                P1->OUT &= BIT0|BIT2;
                P2->OUT &= BIT0|BIT2;
                P3->OUT &= BIT1;
                P4->OUT &= BIT0|BIT2;
        i++;
        }
        if(i==4)
        {
                P1->OUT &= BIT0|BIT2;
                P2->OUT &= BIT0|BIT2;
                P3->OUT &= BIT0|BIT2;
                P4->OUT &= BIT1;
                i++;
        }
}
```

Data sheet: MSP432P401R

TIMER 32 REGISTERS

Bits	**7**	**6**	**5**	**3-2**	**1**	**0**
TIMER32_y->CONTROL	**ENABLE**	**MODE**	**IE**	**PRESCALE**	**SIZE**	**ONESHOT**
TIMER32_CONTROL_**XX**_x	1-enable, 0-disable	1- free running mode, 0- periodic	1-enable interrupt 0-disable interrupt	0=/1, 1=/16, 2=/256	1- 32 bits, 0-16 bits	1-oneshot 0-wrap

25	Timer32_INT1	T32_INT1_IRQn	T32_INT1_IRQHandler
26	Timer32_INT2	T32_INT2_IRQn	T32_INT2_IRQHandler

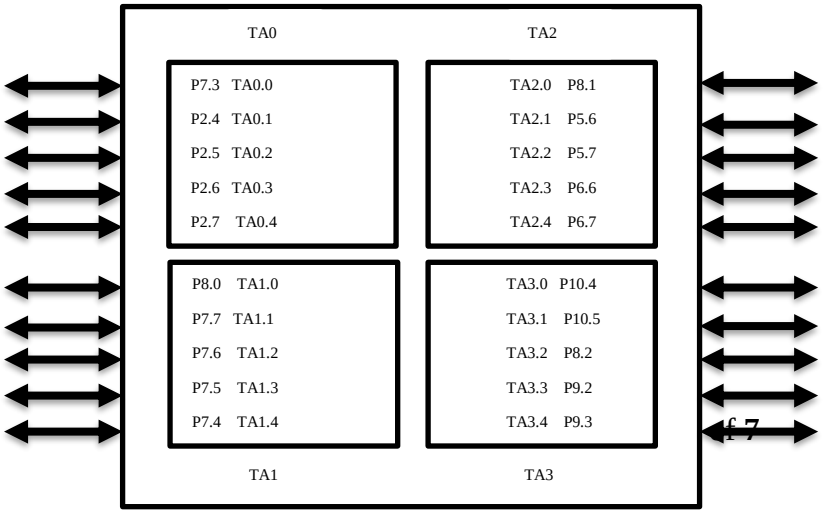
TIMER32_y->LOAD	Load register
TIMER32_y->INTCLR	Clear flag
TIMER32_y->VALUE	Check the counter value

TIMER A REGISTERS

Bits	**9-8**	**7-6**	**5-4**	**2**	**1**	**0**
TIMER_A**y**->CTL	**TASSEL**	**ID**	**MC**	TA**CLR**	TA**IE**	TA**IFG**
TIMER_A_CTL_**XX**_x	0-TACLK, 1-ACLK 2-SMCLK, 3-INCLK	0-/1, 1=/2, 2=/4, 3=/8	0-stop, 1-up 2- continuous, 3- up/down	1-clear counter	1-enable timer interrupt	1-timer overflow flag set

Bits	**15-14**	**13-12**	**11**	**10**	**8**	**7-5**	**4**	**3**	**2**	**1**	**0**
TIMER_Ay->CCTL	**CM**	**CCIS**	**SCS**	**SCCI**	**CAP**	**OUTMOD**	**CCIE**	**CCI**	**OUT**	**COV**	**CCIFG**
TIMER_A_CCTLN_**XX_x**	0-no edge, 1- rising, 2-falling, 3- both	0-CCInA, 1-CCInB, 2-GND, 3-VCC	1-Synchronize timer clock	1-Observe synchronized input	1-capture 0-compare	0-output, 1-set, 2-toggle/reset, 3- set/reset, 4-toggle, 5-reset, 6- toggle/set, 7-reset/set	1-enable interrupt	Capture input value	Bit value		Set for capture in capture mode Set if compare is true in compare mode

Bits	**3-0**
TIMER_Ay->EX0	**TAIDEX**
TIMER_A_EX0_**XX**_x	**0-/1, 1=/2, 2=/3, 3=/4, 4=/5, 5=/6, 6=/7, 7=/8.**



8	Timer_A0	TA0_0_IRQn	TA0_0_IRQHandler	TA0CCR0-CCIFG
9	Timer_A0	TA0_N_IRQn	TA0_N_IRQHandler	TA0CCR1-6, TAIFG
10	Timer_A1	TA1_0_IRQn	TA1_0_IRQHandler	TA1CCR0-CCIFG
11	Timer_A1	TA1_N_IRQn	TA1_N_IRQHandler	TA1CCR1-6, TAIFG
12	Timer_A2	TA2_0_IRQn	TA2_0_IRQHandler	TA2CCR0-CCIFG
13	Timer_A2	TA2_N_IRQn	TA2_N_IRQHandler	TA2CCR1-6, TAIFG
14	Timer_A3	TA3_0_IRQn	TA3_0_IRQHandler	TA3CCR0-CCIFG
15	Timer_A3	TA3_N_IRQn	TA3_N_IRQHandler	TA3CCR1-6, TAIFG

I2C communication

Bits	15	14	13	11	10-9
EUSCI_Bx->CTLW0	UC**A10**	UC**SLA10**	UC**MM**	UC**MST**	UC**MODEx**
EUSCI_B_CTLW0_XX_x	1- select the own address length of the device 0-seven bit address	1-set the slave address	1-choose the master device number(>1) 0- one master in the system	1- used as master, 0- slave	3-I2C
8	7-6	4	3	2	1
UC**SYNC**	UC**SSELx**	UC**TR**	UC**TXNACK**	UC**TXSTP**	UC**TXSTT**
0-asynchronus 1-synchronous	0-UCLKI, 1-ACLK, 2-SMCLK	1-transmitter 0-receiver	1-NACK bit is generated 0-ACK bit occurs	Stop bit generated by master	Start bit generated by master
0 UCSWRST(XX_x=SWRST) 1- reset the module 0-operation mode					

Bits	6	5	3	2	1	0
EUSCIBx->IE	UC**BCNTIE**	UC**NACKIE**	UC**STPIE**	UC**STTIE**	UC**TXIE0**	UC**RXIE0**
EUSCI_B_IE_XX	1-enable byte counter interrupt	1-enable not acknowledge interrupt)	1-enable stop	1-enable start conditions	1-anable transmit interrupt	1-enable receive interrupt

Bits	6	5	3	2	1	0
EUSCIBx->IFG	UC**BCNTIFG**	UC**NACKIFG**	UC**STPIFG**	UC**STTIFG**	UC**TXIFG0**	UC**RXIFG0**
EUSCI_B_IFG_XX	1-set for byte counter interrupt	1-set not acknowledge interrupt)	1-set for stop conditions	1- set start conditions	1-set transmit interrupt	1-set receive interrupt

20	eUSCI_B0	ISER[0]-ICER[0]	0x00100000	EUSCIB0_IRQn	EUSCIB0_IRQHandler
21	eUSCI_B1	ISER[0]-ICER[0]	0x00200000	EUSCIB1_IRQn	EUSCIB1_IRQHandler
22	eUSCI_B2	ISER[0]-ICER[0]	0x00400000	EUSCIB2_IRQn	EUSCIB2_IRQHandler
23	eUSCI_B3	ISER[0]-ICER[0]	0x00800000	EUSCIB3_IRQn	EUSCIB3_IRQHandler

PIN	**PxSEL1=0, PxSEL0=1**
P6.5	UCB1SCL
P6.4	UCB1SDA
P1.6	UCB0SDA
P1.7	UCB0SCL
P6.6	UCB3SDA
P6.7	UCB3SCL
P3.6	UCB2SDA
P3.7	UCB2SCL