TED UNIVERSITY

Department of Computer Engineering

CMPE453-Embedded Systems

Fall 2023, Midterm Exam

Date: 30/11/2023, Thursday - Time: 19:00-20:30

Duration: 90 minutes

QUESTIONS	POINTS	
Q1	30	
Q2	40	
Q3	30	
TOTAL	100	

FULL NAME:

STUDENT ID:

SECTION:

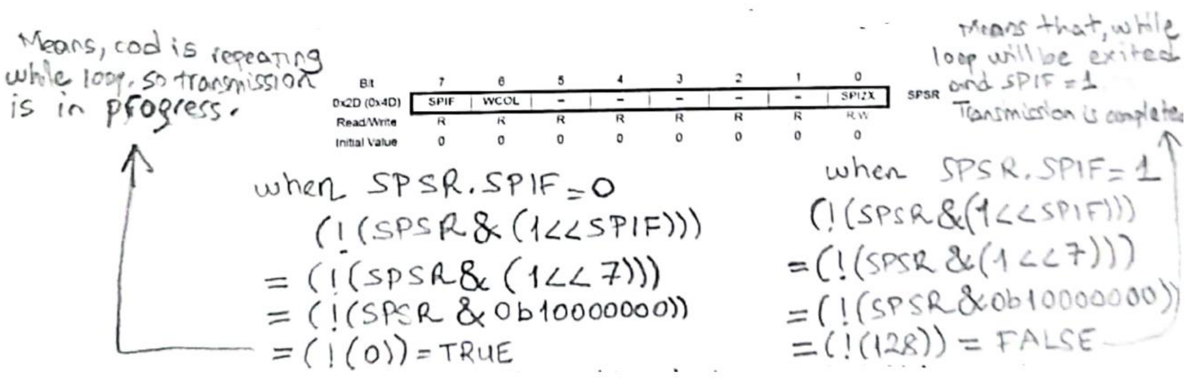
1- Answer the following questions.

a) (SPI). SPSR register for SPI communication is given below. Explain how the code line:

while(!(SPSR & (1<<SPIF)))

is used for "waiting until SPI transmission is completed." Show your calculations and work clearly.(15pts)

Bit	7	6	5	4	3	2	1	0	
0x2D (0x4D)	SPIF	WCOL	–	–	–	–	–	SPI2X	SPSR
Read/Write	R	R	R	R	R	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	



Means, cod is repeating while loop. so transmission is in progress.

Bit	7	6	5	4	3	2	1	0	
0x2D (0x4D)	SPIF	WCOL	–	–	–	–	–	SPI2X	SPSR
Read/Write	R	R	R	R	R	R	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Means that, while loop will be exited and SPIF = 1 Transmission is completed

when SPSR.SPIF = 0

(!(SPSR & (1<<SPIF)))

= (!(SPSR & (1<<7)))

= (!(SPSR & 0b10000000))

= (!(0)) = TRUE

when SPSR.SPIF = 1

(!(SPSR & (1<<SPIF)))

= (!(SPSR & (1<<7)))

= (!(SPSR & 0b10000000))

= (!(128)) = FALSE

b) (Creating Delay). An AVR Microcontroller has a clock frequency of 8Mhz. The CPU uses a derived frequency from master clock with a division factor of 4. Write a C method which takes the required delay (in ms.) as argument and returns the delay.(15 pts)

F = 8x10^6 Hz.
T = 1/F = 1/(8x10^6)
div.factor = 4
T' = 4*T = 4/(8x10^6) sec. → time passes for 1 cycle.

#code begins here

```
int delayms (int delay) {

    int cycles = (delay / 1000) / (4/(8x10^6))

    int i;
    for (i=0; i<cycles;i++) {

    }

    return delay

}
```

2. (UART). Following registers are associated with UART operation.

15	14	13	12	11	10	9	8	
–	–	–	–	UBRRn[11:8]				UBRRnH
UBRRn[7:0]								UBRRnL
7	6	5	4	3	2	1	0	

7	6	5	4	3	2	1	0	
RXCn	TXCn	UDREn	FEn	DORn	UPEn	U2Xn	MPCMn	UCSRnA
R	R/W	R	R	R	R	R/W	R/W	
0	0	1	0	0	0	0	0	

7	6	5	4	3	2	1	0	
RXB[7:0]								UDRn (Read)
TXB[7:0]								UDRn (Write)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
0	0	0	0	0	0	0	0	

7	6	5	4	3	2	1	0	
RXCIEn	TXCIEn	UDRIEn	RXENn	TXENn	UCSZn2	RXB8n	TXB8n	UCSRnB
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	

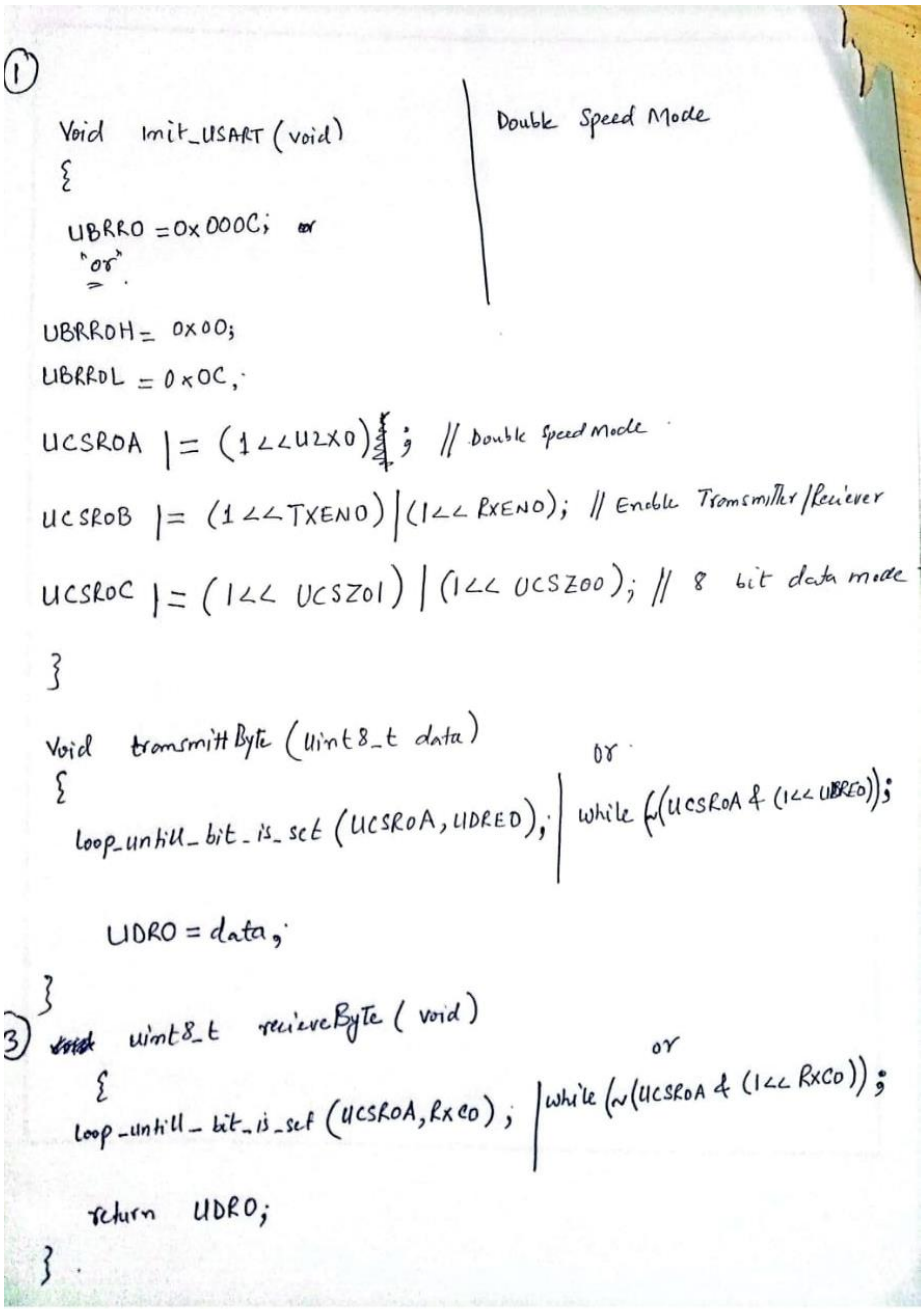
7	6	5	4	3	2	1	0	
UMSELn1	UMSELn0	UPMn1	UPMn0	USBSn	UCSZn1	UCSZn0	UCPOLn	UCSRnC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	

Write a complete C code which perform the following tasks. (40 pts.)

1) Initialize the UART in Asynchronous Double Speed Mode with Baud Rate 9600, 8 bit character size, full duplex (Transmitter and Receiver Mode) enabled.
2) Write the transmit function which is able to transmit a String of characters to UART TX pin of microcontroller in pooling mode (i.e. transmit a character and wait for completion)
3) Write the receive function to receive a single character from RX pin of UART.
4) In the main function, write the code which should transmit "Hello world" message after every 1 sec and then receive the incoming character. And then display the ASCI (binary) equivalent of received character on 8 LEDs connected to a PORTB of microcontroller.

$$\mathrm{UBRRn} = \frac{f_{\mathrm{OSC}}}{8\mathrm{BAUD}} - 1$$

UBRR0 = 1M/(8*(9600))-1 = 12.02 ≅ 12 = 00001100b



```
①
  Void  init_USART (void)                 | Double Speed Mode
  {                                       |
   UBRR0 = 0x000C;  ~~or~~                |
   "or"                                   |
   => .                                   |
 UBRR0H = 0x00;
 UBRR0L = 0x0C;
 UCSR0A |= (1<<U2X0); // Double Speed Mode
 UCSR0B |= (1<<TXEN0)|(1<<RXEN0); // Enable Transmitter/Reciever
 UCSR0C |= (1<<UCSZ01)|(1<<UCSZ00); // 8 bit data mode

 }

 Void  transmittByte (uint8_t data)
  {                                                      or
   loop_untill_bit_is_set (UCSR0A, UDRE0); | while (~(UCSR0A & (1<<UDRE0));

     UDR0 = data;

 }
3) ~~void~~  uint8_t  recieveByte ( void)
     {                                                or
   loop_untill_bit_is_set (UCSR0A, RXC0); | while (~(UCSR0A & (1<<RXC0));

    return  UDR0;
 }.
```

(2)

```
Void printstring ( const char myString[] )
   {
      uint8_t   i = 0;
    while ( my_string[i] )
         {
          transmitByte (my_string[i]); // calling function
               i++;
         }
   }.
```

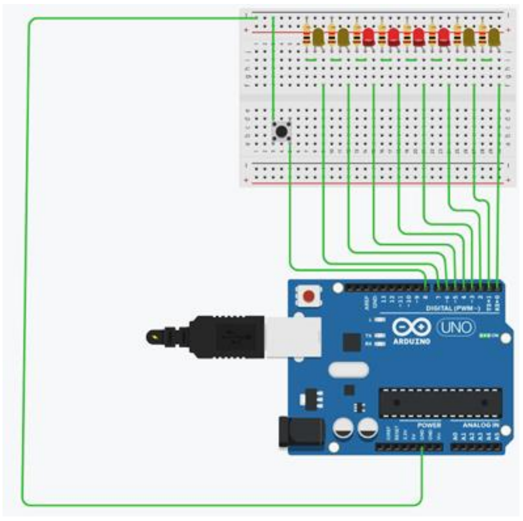
Main function.

(4)

```
 #include <Avr/io.h>
 # include <Avr/delay.h>

int main (void)
{  uint8_t rx_char;
   init_USART ( );
   DDRB = 0xff;     // declaring port B as output.
   while (1)
 {
   print string ("Hello world/n");
   rx_char = recieveByte ( );
   PORTB= rx_char;
  _delay_ms (1000);
 }
}.
```

3. **(Digital Input).** Consider that following circuit where a push button is connected with pin 0 of port B and 8 LEDs are connected with port D.



Program AVR C program to count the number of push in the button and display the number in binary using the LEDs. At the beginning, it displays 0 and when the number reaches the maximum, it restarts from 0. Assume that one press of the button consists of the press and release of the button and the leftmost LED indicates the MSB. (30 pts.)

```
// ------- Preamble -------- //
#include <avr/io.h>
#include <util/delay.h>
int main(void)
        {
        // -------- Inits --------- //
        PORTB |= (1 << PB0);
        DDRD = 0xff;
        bool old_state = false;
        bool current_state;

        // ------ Event ------ //
        int count = 0;
        PORTD = 0;
        while(1)
                {
                current_state = bit_is_clear(PINB, PB0);
                if(current_state && !old_state) {
                        PORTD = count++;
                        count = Count%256;
                }
                old_state = current_state;
                }
        return 0;
        }
```