Problem #1. (20 points). ADC, DAC

A (5 points). A sensor device gives an analog voltage that is directly proportional to atmospheric pressure. The sensor output voltage is always in the range of 0.5 V (min) to 1.5 V (max). How would you configure the MSP430’s ADC12 to perform analog-to-digital conversion with maximum resolution? What is the digital value that corresponds to 1.25 V input voltage? Explain.

B (5 points). What is the analog voltage at the output of the MSP430’s DAC if we use 2.5 V reference voltage and 8-bit resolution? The data register DAC12_DATA = 0x007A.
C (10 points). Analyze the following code. Add missing comments in the code below. What does this program do?

```c
// ACLK = 32kHz, MCLK = SMCLK = default DCO 1048576Hz, ADC12CLK = ADC12OSC
//
// MSP430xG461x
// -----------------
// |              XIN|
// Vin0 -->|P6.0/A0          | 32kHz
// Vin1 -->|P6.1/A1      XOUT|
// Vin2 -->|P6.2/A2      |
// Vin3 -->|P6.3/A3          |
// |
//***************************************************************************
#include "msp430xG46x.h"
volatile unsigned int A0results[8];
volatile unsigned int A1results[8];
volatile unsigned int A2results[8];
volatile unsigned int A3results[8];
unsigned int Index;

void main(void)
{
    WDTCTL = WDTPW + WDTHOLD;                 // Stop watchdog
    P6SEL |= 0x0f;                            // Enable A/D inputs
    ADC12CTL0 = SHT0_8 + MSC + ADC12ON;       //
    ADC12CTL1 = SHP + CONSEQ_3;               //
    ADC12IE = 0x08;                           //
    ADC12MCTL0 = INCH_0;                      //
    ADC12MCTL1 = INCH_1;                      //
    ADC12MCTL2 = INCH_2;                      //
    ADC12MCTL3 = EOS + INCH_3;                //
    while (1)
    {
        ADC12CTL0 |= ENC;                       //
        ADC12CTL0 |= ADC12SC;                   //
        __bis_SR_register(LPM0_bits + GIE);     // LPM0
    }
}
#pragma vector = ADC12_VECTOR
__interrupt void ADC12_ISR(void)
{
    A0results[Index] = ADC12MEM0;             //
    A1results[Index] = ADC12MEM1;             //
    A2results[Index] = ADC12MEM2;             //
    A3results[Index] = ADC12MEM3;             //
    Index = (Index + 1) & 0x7;                //
    __no_operation();                         // SET BREAKPOINT HERE
    __bic_SR_register_on_exit(LPM0_bits);     // Exit LPM0
    }
```

This program initializes an MSP430 microcontroller to perform analog-to-digital conversions on four analog inputs connected to P6.0/A0, P6.1/A1, P6.2/A2, and P6.3/A3. The ADC12 module is configured for single-shot mode with automatic conversion and no carrier select. The program enters LPM0 low-power mode after each ADC conversion and waits for the next conversion to start. The results of each conversion are stored in separate arrays for each channel.
Problem #2. (25 points) ADC, DAC.
Your task is to write a program that samples an analog signal \( a0 \) as illustrated below using the MSP430’s ADC12 device. The samples are then forwarded to the MSP430’s DAC12 device. Answer the following questions.

A. (2 points) What is the maximum and minimum input voltage of the input signal \( a0 \)?

B. (2 points) What is duration of one period of the input signal \( a0 \) in milliseconds?

C. (6 points) Let us assume that we configure the MSP430’s device to sample analog input \( a0 \) with the sampling frequency \( f_{\text{sample}} = 10 \) KHz. How many samples we have per one period of the input analog signal? Fill in the following table (please not that the number of rows does not reflect the number of samples per one period). Assume that our sampling is synchronized with the \( a0 \) (i.e., the first sample is taken at the very beginning of an \( a0 \) period).

<table>
<thead>
<tr>
<th>Sample number</th>
<th>( a0 ) [V]</th>
<th>Sample value [hexadecimal]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

D. (5 points) Sketch the output of the analog signal that is driven by the MSP430’s DAC12 device (as you would see it on the oscilloscope).
E. (5 points) Give a short description of your program that performs the ADC and DAC conversions. We assume that clocks are initialized as follows: $f_{MCLK}=f_{SMCLK}=4$ MHz. What should be done to initialize devices, what is done in the main program loop, and what is done in corresponding interrupt service routines?

F. (5 points) If you know that less than 80 clock cycles is spent for processing one sample (read the sample for the ADC12 and write the digital value of the sample to the DAC12 data register), what would be the maximum sampling frequency we could have without oversubscribing our processor time? Elaborate your answer.
Problem #3 (20 points) Communication
A (5 points). What is synchronous and asynchronous serial communication? Give an example for each, and sketch the block diagram to illustrate main connections between the transmitter and receiver.

B (5 points). Your task is to initialize the USART0 device to carry out asynchronous serial communication with 9600 bit/sec using 32768 Hz as the source clock. Assume that we use 8-bit characters with a parity bit, and 2 stop bits. Will this communication work properly without modulation? How do you determine value in the modulation register? Use illustrations to support your answer including timing waveforms.
C (10 points). Illustrate the USART operation in SPI mode. Sketch the master, slave(s), and relevant registers in each module. Explain the sequence of steps in typical communication scenarios.

See reference manual for explanation.
Problem #4. (20 points) We are considering processing requirements of an application with the main loop that repeats every 200ms. Each application cycle invokes three interrupt service routines: isrA is invoked 10 times per application cycle, isrB 20 times, and isrC 20 times. We profiled the service routines and know that isrA takes 250 clock cycles, isrB takes 200 clock cycles, and isrC takes 400 clock cycles. The time spent in the main program during one application cycle is 20,000 processor clock cycles. Answer the following questions.

A. (4 points) How many clock cycles is required for the CPU to stay in the active mode during one application cycle?

B. (4 points) If the processor clock is $f_{MCLK}=4$ MHz, what portion of the application cycle can be spent in a low-power mode.

C. (4 points) Do we need to support nesting of interrupt routines for this application to function properly? Assume that interrupts isrA and isrB are generated periodically during an application cycle? Explain your answer. What is the maximum time one service routine may be pending before it gets serviced?

D. (4 points) What is the operating time of the application if we know the following: battery capacity is 660 mAh, current drawn by the microcontroller in active mode is 4 mA and 10 μA in a low-power mode?

E. (4 points) What is the minimum processor clock cycle that we could have and still meet the timing requirements for the application?
Problem #5 (25 points) DMA Controller

A (5 points). Sketch a block diagram illustrating main registers for a single channel in the MSP430. For each register describe its purpose and the initialization process.

Circle the correct answer for the following questions.

B (2 points). The DMA controller allows to:
(i) Move data from one location to another without CPU intervention
(ii) Increase the throughput of peripheral modules
(iii) Reduce system power consumption (CPU in low power mode)
(iv) All of above

C (2 points). The upper byte of a byte-to-word transfer:
(i) Remains with the previous value
(ii) Is cleared
(iii) Is loaded with the same values as the lower byte
(iv) None of above

D (2 points). When the DMA transfer mode DMAxDTx = 1 is selected:
(i) Each transfer requires a trigger
(ii) A complete block is transferred with one trigger
(iii) CPU activity is interleaved with a block transfer
(iv) None of above

E (2 points). NMI interrupts interrupt the DMA controller when:
(i) GIE bit is set
(ii) DMAIE is set
(iii) ENNMI bit is set
(iv) DMAEN is set

F (2 points). The DMA destination address when the DMADSTBYTE = 1 and DMDSTINCRx = 3:
(i) Remains unchanged
(ii) Increments by one
(iii) Decrements by two
(iv) Increments by two
G (10 points). Analyze the following code. What does this program do?

```c
#include "msp430xG46x.h"

const char String1[13] = "\nHello World";

void main(void)
{
    WDTCTL = WDT_ADLY_1000;                   // WDT 1000ms, ACLK, interval timer
    IE1 |= WDTIE;                             // Enable WDT interrupt
    P4SEL |= 0x03;                            // P4.0,1 = USART1 TXD/RXD
    ME2 |= UTXE1 + URXE1;                     // Enable USART1 TXD/RXD
    UCTL1 |= CHAR;                            // 8-bit characters
    UCTL1L = SSEL0;                           // BRCLK = ACLK
    UBR01 = 0x03;                             // 32k/9600=3.41
    UBR11 = 0x00;
    UMKCTL1 = 0x04A;                          // Modulation
    UCTL1 &= ~SWRST;                          // Release USART state machine
    DMACTL0 = DMA0TSSEL_10;                   //
    DMA0SA = (int)String1;
    DMA0DA = TXBUF1_;                         //
    DMA0SA = 0014;
    DMA0CTL = DMASRCINCR_3 + DMASDBB + DMALEVEL; //
    __bis_SR_register(LPM3_bits + GIE);       // Enter LPM3 w/ interrupts
}
```

#pragma vector = WDT_VECTOR
__interrupt void WDT_ISR(void)
{
    DMA0CTL |= DMAEN;
}