The University of Alabama in Huntsville
ECE Department
CPE 323 01 – Introduction to Embedded Computer Systems
Sample Final Exam
Spring 2015

Name: _____

1. **(15 points)** An MSP430-based system interfaces 4 external devices (ED0, ED1, ED2, ED3), each capable of generating an interrupt request. If multiple requests occur at the same time, ED0 should have the highest priority and ED3 the lowest priority. The external devices place a request by setting the request line (a transition from a logic one to a logic zero). The request lines are connected on port 1 pins P1.5, P1.4, P1.3, and P1.2, respectively. A request line is kept active as long as the interrupt request is pending, until the request is serviced. Answer the following questions. Assume that processing request from each peripheral takes ~4 ms (e.g., if two request are active at the same time, the P1_ISR will take 8 ms to execute). The processor is in a low-power mode when not executing the P1_ISR. The following table describes a sequence of events in time. Fill in the table by answering what happens with the MSP430 on each relevant event if we know the following: ED3 raises an interrupt request at 35 ms, ED2 at 37 ms, and ED1 at 48 ms, and ED0 at 55 ms.

Time	**Event**	**MSP430 status**
0 ms	SW initialization	Active (run initialization software)
10 ms	Go to a low-power mode	Sleep
35 ms	Request from ED3 is received	P1_ISR (ED3 portion executed)
37 ms	**Request from ED2 is received**	**P1_ISR (ED3 portion executed, ED2 pending)**
39 ms	**P2_ISR ED3 portion finished**	**P1_ISR (ED2 portion executed)**
43 ms	**Return from P1_ISR**	**Sleep**
48 ms	**Request from ED1 received**	**P1_ISR (ED1 portion executed)**
52 ms	**Return from P1_ISR**	**Sleep**
55 ms	**Request for ED0 is received**	**P1_ISR (ED0 portion is executed)**
59 ms	**Return from P1_ISR**	**Sleep**

2. **(10 points)** Consider an MSP 430, a microcontroller with 64KB address space divided between code memory (flash), RAM memory, and input/output peripherals. It has 4 KBytes of RAM memory starting at the address 0x0320, and 64 Bytes of address space reserved for special purpose registers and 8-bit input/output peripherals (starting at the address 0x0000), followed by 256 Bytes reserved for 16-bit input/output peripherals. It also has a flash memory of 2048 Bytes placed at the top of the address space. Determine the address map by filling in the following table.

Address	Address [hexadecimal]	What?
Last Flash address	**0xFFFF**	Flash Memory
First Flash address	**0xF800**	
Last RAM address	**0x131F**	RAM Memory
First RAM address	**0x0320**	
Last I/O address (16-bit per.)	**0x013F**	I/O address space
First I/O address (16-bit per.)	**0x0040**	
Last I/O address (8-bit per.)	**0x003F**	I/O address space
First I/O address (8-bit per.)	**0x0000**	

3. **(2 points)** _True_ (True or False) Internally generated clock signals by digitally-controlled oscillators tend to vary as a function of the power supply and temperature.

4. **(3 points)** List 3 clock signals the MSP430 FLL+ module provides to the rest of the system.

   1)_MCLK_ 2)_SMCLK_ 3)_ACLK_

5. **(2 points)** List two counting modes of TimerA/TimerB peripherals that require the use of the CCRx block.

   1)_UP_ 2)_UP/DOWN_

6. **(2 points)** _True_ (True or False) I^2C is a 2-wire synchronous serial communication protocol.

7. **(4 points)** You plan to interface an external device using handshaking through parallel ports. You need to send 4-bit data to the device (simplex link). Sketch the interface between your MSP430 microcontroller and the device assuming you can use parallel port 3 for this interfacing. Name the signals and their function.

   4 Data lines, Output
   1 RDY, Output
   1 ACK, Input

8. **(6 points)** Complete the following MSP 430 assembly language program to implement the following C statements, assuming all variables are 16-bit integers.

```
int i, smallest, largest, temp;

smallest = 0;
largest = 0;
for (i= 1; i < n; i++)
{
  if (arr[i] < arr[smallest])
    smallest = i;
  if (arr[i] > arr[largest])
    largest = i;
} // end for
temp = arr[largest];   // Swap smallest found with largest
arr[largest] = arr[smallest];
arr[smallest] = temp;
```

```
            org   0F000h
n:          DC16  10
arr:        DC16  -245, 3876, -9999, -45, 20000, 25000, -30000, -732, 3876, -99
largest:    DS16  1
smallest:   DS16  1
i:          DS16  1
            org   0FF00h
            CLR   smallest          ; smallest = 0
            CLR   largest           ; largest = 0
            MOV   #1, i             ; i = 0
for:        CMP   n, i              ; if i >= n, we are done
            JGE   swap              ; go to swap
            MOV   #arr, R9          ; add 2* smallest to &arr
            ADD   smallest, R9      ; to access arr[smallest]
            ADD   smallest, R9
            MOV   #arr, R10         ; add 2* largest to &arr
            ADD   largest, R10      ; to access arr[largest]
            ADD   largest, R10
            MOV   #arr, R8          ; add 2* I to &arr
            ADD   i, R8             ; to access arr[i]
            ADD   i, R8
            CMP   0(R9), 0(R8)      ; if arr[i] < arr[smallest]
            JL    small_up          ; go change smallest
            JMP   large_cmp         ; if not, check for largest
small_up:   MOV   i, smallest       ; update smallest
large_cmp:  CMP   0(R8), 0(R10)     ; if arr[largest] < arr[i]
            JL    large_up          ; go change largest
            JMP   for_update        ; if not, done with loop
large_up:   MOV   i, largest        ; update largest
for_update: INC   I                 ; i++
            JMP   for               ; next iteration of for loop
```

swap:	MOV	#arr, R9
	ADD	smallest, R9
	ADD	smallest, R9
	MOV	#arr, R10
	ADD	largest, R10
	ADD	largest, R10
	MOV	0(R9), R8
	MOV	0(R10), 0(R9)
	MOV	R8, 0(R10)

```
done:       JMP   $
            END
```

9. **(20 points)** Consider using a 16-bit analog-to-digital converter to sample a periodic input signal y = 1.3 + sin(0.5*pi*f*t), f=1,000 Hz. Answer the following questions.

a. (2 points) What is the duration of one period of the signal y in milliseconds?

0.5 * pi *1,000 * t = 2 * pi, t = 2/(0.5*1000) = 2/500 = 4 ms

b. (2 points) What is the maximum and the minimum voltage at the analog input? Fill in the table below by specifying min/max values and times when those values are achieved.

Min/Max	Value [Volts]	Time [milliseconds]
Min	**0.3 v**	**3 ms, 7 ms, 11 ms…**
Max	**2.3 V**	**1 ms, 5 ms, 9 ms**

c. (2 points) What is a minimum change of the input signal that can be detected by the ADC16, if we use the reference voltages Vref- = -0.2 V and Vref+ = 2.6 V?

V_{LSB} = 2.4/2^16 = 0.0000366 V = 36.6 uV

d. (2 points) You want to sample the input signal y with the sampling rate of f_S = 20,000 Hz. How many samples do you receive from the ADC per one period of the input signal?

T_s = 1/20000 = 0.05 ms, T = 4 ms, T/T_s = 4 ms/0.05 ms = 80 samples

e. (6 points) What values would you read from the 16-bit ADC buffers if your sampling rate is 20,000 Hz. Fill in the following table with the values for the first 4 samples (assume the first sample is taken at t = 0 s). Assume Vref- = 0 V and Vref+ = 3 V.

Sample	t=? [ms]	sin(0.5*pi*f*t)	y = 1.3 + sin(0.5*pi*f*t)	Samples from ADC [unsigned value in decimal]
1	**0**	**0**	**1.3**	**28,398**
2	**0.05**	**0.078**	**1.378**	**30,102**
3	**0.1**	**0.157**	**1.457**	**31,828**
4	**0.15**	**0.233**	**1.533**	**33,488**

f. (2 points) If we use a TimerB to trigger an ADC conversion, how would you configure TimerB assuming its input clock, SMCLK, is set to 3,000,000 Hz?

3,000,000/20,000 = 150 => set TimerB in UP mode and TB_CCR0=149

g. (2 points) Is it possible to improve the resolution (V_{LSB}) of the analog-to-digital conversion described in this problem? If yes, explain what changes would you make?

Yes. The signal is bounded between 0.3 and 2.3 V, and we can use these as reference voltages Vref+ = 2.3 V and Vref- = 0.3 V (provided externally or from a DAC). This way, we can improve resolution as follows:
V_{LSB} = 2/2^16 = 0.0000305 V = 30.5 uV

h. (2 points) What would be the minimum ADC16 internal clock rate under the following assumptions: the ADC16 aperture time (the time the ADC is sampling the input signal) is 12 clock cycles and the conversion time is 14 clock cycles.

If we are taking 20,000 samples in a second, the budget for sampling and conversion of one input sample is 1/20,000 = 50 us. So we need at least 28 clock cycles in 50 micro seconds => 28/0.000005 = 560,000 Hz.

10. **(5 points)** Consider the following subroutine MySub(void). What does the subroutine MySub do? USCIA0 is configured in the UART mode in your MS430 board is connected to a hyper terminal application on a workstation.

```
unsigned int sec=0; // count seconds
unsigned char tsec=0; // count (1/10)th of a second
char Msg[7]; //

void MySub(void) {
   sprintf(Msg, "%05d:%01d", sec, tsec);
   for(int i=0;i<7;i++)  {
      while (!(IFG2 & UCA0TXIFG));
      UCA0TXBUF = Msg[i];
    }
    while (!(IFG2 & UCA0TXIFG));
    UCA0TXBUF = 0x0D; // Carriage Return
}
```

The subroutine MySub creates a 7-character message that contains 5 digit sec variable and 1 digit tsec variable, separated by a column character. The message is then sent to USCI0 through using a UART interface. After that, the carriage return is sent. The program is using polling. On the hyperterminal we will see running timer that counts seconds and 1/10th of a second.

11. **(4 points)** You are designing an MSP430-based system that monitors presses and releases of an SW1 switch. The SW1 switch is connected to P1.0. When a switch is pressed, the supply voltage is connected to the corresponding input pin (a logic '1'). Your program should send a message to a hyper-terminal on a workstation after each "press and release" cycle reporting the sequence number of the cycle (1, 2, 3....65,535) and the duration of the press in milliseconds (assume that presses do not take more than 65,535 ms). The messages should appear in the hyper-terminal as follows (e.g., the press number #1 lasted 45 ms, the second press lasted 122 ms):

```
#     1 @    45 ms
#     2 @   122 ms
#     3 @ 65535 ms
```

Formatting note: each message starts with a '#' character followed by a space character; the sequence number occupies 5 digits and is followed by a space, '@', and another space character; finally, the duration occupies 5 digits followed by a space and "ms" (the total message is 19 characters + new line + carriage return characters). Specify the peripherals that should be initialized in the main program to configure the system.

Port 1: P1.0 interrupt enabled input, active edges (rising for the press event, falling for the release event)
TimerB: configure timer in UP mode to count ms
UART: configure for sending messages

12. **(15 points)** Consider the following assembly language program.

```
#include "msp430.h"                    ; #define controlled include file

00            NAME    main                   ; module name
01            PUBLIC  main                  ; make the main label visible
02                                          ; outside this module
03            ORG     0FFFEh
04            DC16    main                    ; set reset vector to 'init' label
05            ORG     0F000h
06  n:        DC16    8
07  a:        DC16    9
08  i:        DS16    1
09  arr1:     DC16    -22, 837, -2000, -962, 3400, 25, -8, 42
10  arr2:     DS16     8
11  main:     NOP                            ; main program
12            MOV.W   #WDTPW+WDTHOLD,&WDTCTL  ; Stop watchdog timer
13            MOV     #arr1, R8
14            MOV     #arr2, R12
15            CLR     i
16  for:      CMP     n, i
17            JGE     done
18            MOV     @R8+, R9
19            CLR     R10
20            MOV     a, R11
21            ADD     #0, R11
22            JZ      next
23  again:    ADD     R9, R10
24            DEC     R11
25            JNZ     again
26  next:     MOV     R10, 0(R12)
27            ADD     #2, R12
28            INC     i
29            JMP     for
30  done:     JMP     $
31            END
```

a. (10 points) What does this code segment do? Explain your answer.

This code loops initializes i to 0, the starting address of arr1 in R8, and the starting address of arr2 in R12 and loops until i = n terminating the loop. Each array element is loaded into R9 and the pointer advanced to the next element. R10 is set to 0, then a is copied into R11. If a is nonzero, we add R9 to R10, a times, effectively multiplying it by a. When R11 gets down to 0, the result in R10 is stored in to the element of arr2 pointed to by R12. R12 is advanced to the address of the next element of arr2. This code does a scalar multiply of an array arr2 = a*arr1.

b. (5 points) Calculate the total number of instructions for the code sequence from above (count 30 as executing once).

c.

Lines 11-15, 30 execute one time	6
Lines 16-17 execute 9 times	18
Lines 18-22, 26-29 execute 8 times	72
Lines 23-25 execute 8*9 times	216
Total = 312	

13. **(12 points)** Floating Point Number Conversion

(a) **(6 points)** Convert the following number from decimal to the IEEE 32-bit floating point. Show your work. -8.625_{10}

S = 1, negative number, 8 = 1000, 0.625 = 0.101
8.625 = 1000.101 = 1.000101 x 2^3
Exponent + bias = 3 + 127 = 130 = 1000 0010, Fraction = 00101, Putting them together,
1100 0001 0000 1010 0000 0000 0000 0000 = 0xC10A 0000

(b) **(6 points)** Convert the following number from the binary IEEE 32-bit floating point to decimal. Show your work. 2359_7000_{16}

0x2359 = 0010 0011 0101 1001 0111 0000 0000 0000
S = 0, Number is positive
Exponent + bias = 0100 0110 = 70, Exponent = 70 -127 = -57
Fraction = 1011 0010 1110 0000 0000 000 = 2^{-1} + 2^{-3} + 2^{-4} + 2^{-7} + 2^{-9} + 2^{-10} +2^{-11} =
(1024+256+128+16+4+2+1)/2048
Number = 1.6987 x 2^{-57} = 1.1787 x 10^{-17}

Department of Electrical and Computer Engineering
University of Alabama in Huntsville

CPE 323 – Introduction to Embedded Computer Systems

Final Exam Solutions

Instructor: Dr. Aleksandar Milenkovic

Date: December 10, 2012

Place: EB 207

Time: 3:00 PM – 5:30 PM

Note: Work should be performed systematically and neatly. This exam is closed books and closed neighbor(s). Allowable items include exam, pencils, straight edge, calculator, and materials distributed by the instructor. Best wishes.

Question	Points	Score
1	20	
2	20	
3	20	
4	20	
5	20	
6(bonus)	7	
Sum	100+7	

Please print in capitals:

Last name:_____

First name: _____

1. (20 points) MSP430 System Architecture/Miscellaneous

Answer the questions or circle the correct answers when appropriate.

A. (6 points) The MSP430552x is a 16-bit microcontroller with 64KB of address space divided between code memory (flash), RAM memory, and input/output peripherals. It has 32 KB of flash memory placed at the top of the address space, 4KB of RAM memory starting at the address 0x2400, and 4KB of address space reserved for input/output peripherals starting at the address 0x0000. Determine the address map by filling in the following table.

Address	*Address [hexadecimal]*	*What?*
Last Flash address	***0xFFFF***	Flash Memory
First Flash address	***0x8000***	
Last RAM address	***0x33FF***	RAM Memory
First RAM address	***0x2400***	
Last I/O address	***0x0FFF***	I/O address space
First I/O address	***0x0000***	

B. (2 points) What is the maximum number of long words that can be allocated in the RAM memory from above?
4KB/4B = 2^10 long words = 1024

C. (2 points) (**True** | False) The MOV.W #40, &0x2800 will actually write the constant 0x28 into a memory location at the address 0x2800.

D. (2 points) (True | **False**) The program downloaded into the flash memory through a JTAG port will remain in the flash memory only until the power is turned on (i.e., will be lost when the device is turned off).

E. (2 points) (**True** | False) The content of the RAM memory remains intact if a MSP430 goes into a low-power mode.

F. (2 points) (**True** | False) To exit a low-power mode we can execute an instruction in an ISR to clear a copy of the status register on the stack.

G. (2 point) (True | **False**) MSP430 instruction set includes a multiply instruction that finds a product two unsigned operands.

H. (2 point) (True | **False**) The MSP430 devices can read/write a byte from memory only if is placed at an even address in the address space.

2. (20 points) Interrupts

Answer the questions or circle the correct answers when appropriate.

A. (2 points) (True | **False**) Interrupt service routines must end with a RET (MOV @SP+, PC) instruction that retrieves the return address from the top of the stack.

B. (2 points) (**True** | False) The GIE bit in the status register can be set or cleared by the programmer at any point of time.

C. (2 points) (**True** | False) If multiple interrupt requests are pending at the time of exception processing, the MSP430 selects the one with the highest priority.

D. (2 points) (True | **False**) The interrupt vector table is filled with the starting addresses of the ISRs at the time of manufacturing of the microcontroller and its content cannot be changed.

E (2 points) (True | **False**) The MSP430 automatically resets (in hardware) the relevant interrupt flags that record pending requests for both single-sourced and multi-sourced interrupt service routines.

An MSP430-based system interfaces 3 external devices (ED1, ED2, ED3), each capable of generating an interrupt request. The external devices place a request by setting the request line (a transition from a logic zero to a logic one). The request lines are connected on port 2 pins P2.4, P2.5, P2.6, respectively. A request line is kept active as long as the interrupt request is pending, until the request is serviced. Answer the following questions.

F (3 points) Outline the main steps that need to be taken in the main program to initialize the system for accepting the interrupts from the devices ED1-ED3.

Main:
Configure ports P2.4, P2.5, and P2.6 as interrupt request lines
(input direction, sensitive to the rising edge, enable them):
```
P2IE=0x70;          ; enable interrupts from P2.4, P2.5, P2.6
P2IES=P2IES & 0x8F ; clear bits for rising edge
P2DIR=P2DIR & 0x8F;
```

G. (3 points) If multiple requests occur at the same time, ED1 should have the highest priority and ED3 the lowest priority. Only one request should be serviced at a time. Outline the main steps taken in the interrupt service routine.

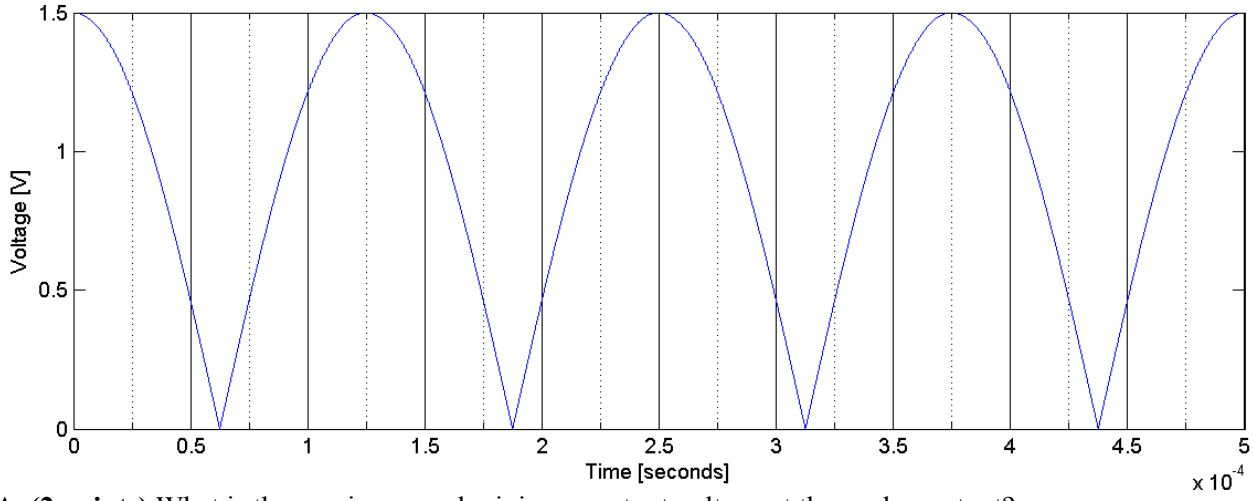
P2_ISR:
if (ED1 request is pending) then {Clear P2.IFG[4]; Process the interrupt for ED1}
else if (ED2 request is pending) then {Clear P2.IFG[5]; Process the interrupt for ED2 }
else if (ED3 request is pending) then {Clear P2.IFG[6]; Process the interrupt for ED3 }
RETI

H (4 points) Assume that the ISR from part G takes 5 ms to execute. The processor is in a low-power mode when not executing the ISR. The following table describes a sequence of events in time. Fill in the table by answering what happens with the MSP430 on each relevant event if we know the following: ED3 raises an interrupt request at 35 ms, ED2 at 37 ms, and ED1 at 48 ms.

Time	**Event**	**MSP430 status**
0 ms	*SW initialization*	*Active (run initialization software)*
10 ms	*Go to a low-power mode*	*Sleep*
35 ms	*Request from ED3 is received*	*P2_ISR (ED3 portion executed)*
37 ms	***Request from ED2 is received***	***P2_ISR (ED3 portion executed, ED2 pending)***
40 ms	***Return from P2_ISR/Enter P2_ISR***	***P2_ISR (ED2 portion executed)***
45 ms	***Return from P2_ISR***	***Sleep***
48 ms	***Request form ED1 received***	***P2_ISR (ED1 portion executed)***
53 ms	***Return from P2_ISR***	***Sleep***

3. (20 points) Embedded Software Design (Digital to Analog Conversion).

To drive an external actuator we need to provide an analog periodic signal y, defined as follows, y = 1.5*|cos (2*pi*f*t)| (see below), f=4,000 Hz. Your task is to generate this signal using the MSP430's digital-to-analog converter peripheral (DAC12). Answer the following questions.



A. (2 points) What is the maximum and minimum output voltage at the analog output?
Max: 1.5 V (x=0, cos(x)=1)
Max: 0 V (x=π/2, cos(x)=0)

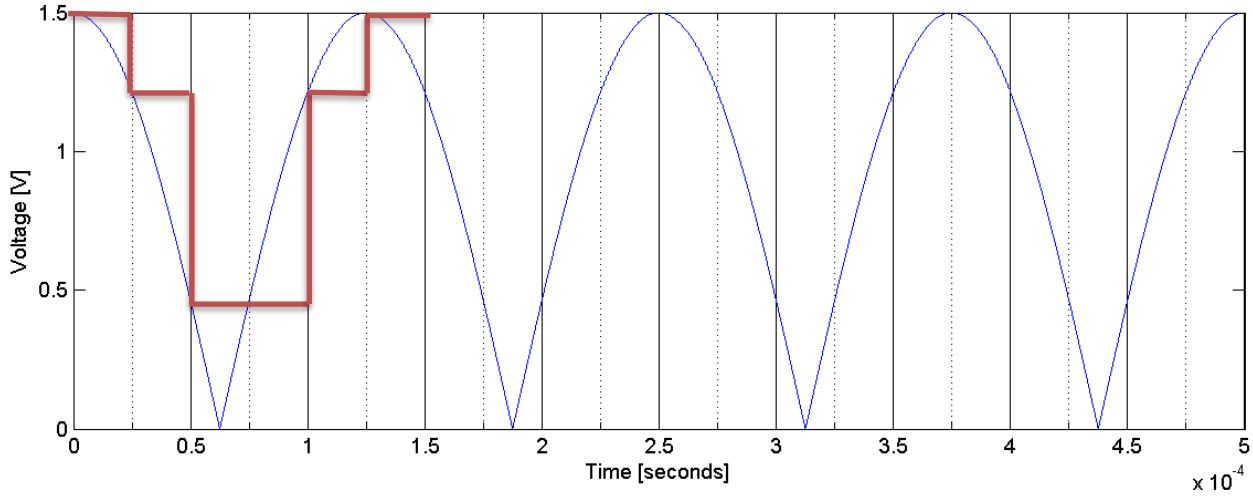
B. (2 points) What is duration of one period of the signal y in milliseconds?
T = 0.25 ms/2 = 0.125 ms

C. (2 points) What is frequency of the signal y?
Fy = 1/0.125 ms = 8,000 Hz

D. (6 points) Let us assume that we provide a lookup table with only 5 samples for a single period of the signal y? Fill in the following table (assume the first sample starts at t=0).

Sample	t=? [ms]	cos(2*pi*f*t)	y=1.5*|cos(2*pi*f*t)|	Lookup table [12-bit unsigned value]
1	***0***	***1***	***1.5***	***4095***
2	***0.025***	***0.809***	***1.21***	***3313***
3	***0.05***	***0.309***	***0.46***	***1265***
4	***0.075***	***-0.309***	***0.46***	***1265***
5	***0.1***	***0.809***	***1.21***	***3313***

E. (4 points) Sketch the output analog signal as you would see it on the oscilloscope. Use the lookup table from part C.



F. (2 points) If we use a TimerA ISR to control sending the next value to the DAC, how many interrupts per second TimerA should generate?

8,000*5 = 40,000

G. (2 points) If $f_{MCLK}=f_{SMCLK}=4{,}194{,}304$ Hz, how would you configure TimerA?

Use SMCLK as a source clock for TimerA. Initialize TimerA in up mode with n=4,194,304 Hz/40,000 = 105 => TACCR0=104

4. (20 points) Time, Timers, Operating time

Answer the questions or circle the correct answers when appropriate.

A. (2 points) (**True** | False) The MSP430's clock module produces multiple clock signals that can be configured in software independently from each other.

B. (2 points) (True | **False**) The MSP430 clock signals are active all time regardless of the operating mode.

C. (2 points) (True | **False**) The watchdog timer (WDT) can be configured in such a way to timestamp an external hardware event (e.g., rising edge of a signal) without intervention in software.

D. (2 points) (**True** | False) The watchdog timer supports only several predefined time intervals.

E. (2 points) An MSP430 FLL+ is configured as follows: MCLK = 2,097,152 Hz and ACLK = 32,768 Hz. How many MCLK clocks occur in one ACLK clock?

2^21/2^15 = 64

Let us consider a program that samples a data from a 3-dimensional accelerometer. The ADC12 samples acceleration with frequency of 40 samples per second. When all 3 samples are ready, the MSP430 reads the samples from the ADC12 in the ISR and then sends the samples over a UART to a workstation in the main program. The processor is active when executing ADC12_ISR (40 processor clock cycles) and while sending samples in the main program (3,960 processor clock cycles for one triplet x, y, and z). After sending one triplet, the processor goes into a low-power mode (TimerA and ADC12 remain active). The MCLK in the active mode is 1 MHz. The platform draws 1.2 mA when the MSP430 is in the active mode and 0.2 mA when in the low-power mode.

F. (3 points) What is the total active and sleep time during one application cycle?

We take 40 samples per second => One application cycles is equivalent to 1,000,000/40 = 25,000 CPU clock cycles

Active time: 40 + 3,960 = 4,000 clock cycle in active mode and 21,000 – 4,000 = 21,000 clock cycles in a LPM

Active time percentage: 4,000/25,000 = 0.16, LPM = 0.84

G. (2 points) Calculate the average current drawn by the MSP430.

I_{AVG} = (4/25)*1.2mA + (21/25)*0.2mA = 0.36mA

H. (2 points) Calculate the total power P consumed (in milliWatts) if we power the platform by two AA batteries (V=3 V).

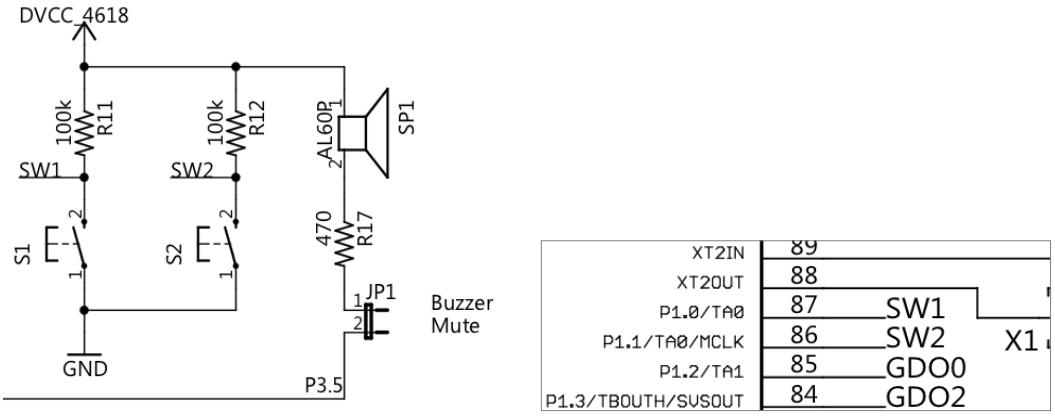
P = V*I_{AVG} = 3*0.36mA = 1.08mW

I. (2 points) Determine the system operating time in days if we know that the battery capacity is 2500 mAh.

OT = BC/I_{AVG} = 2500/0.36 ≈ 6,944 hours ≈ 289 days

5. (20 points) Misc. peripherals

Consider the following schematic that describes switches (S1 and S2) and a buzzer interface to an MSP430.



A. (2 points) S1 is pressed and S2 is released. What register will reflect the status of these switches? What is the value observed in that register?

P1IN, P1IN.BIT0 should have a '0', and P1IN.BIT1 should have a logic '1'.

B. (4 points) Write a C code segment that detects when a switch is pressed (including debouncing) and sets a flag named sw1p to 1. The code segment should end by detecting that a switch is released.

```
#define SW1 BIT0&P1IN // SW1 - P1.0
...
if ((SW1) == 0) {
  for(i = DEBOUNCE; i > 0; i--); // debounce
  if((SW1) == 0) sw1p = 1;       // set local flag
  while(SW1==0) {}; // wait for release
  // do something if needed
}
```

The buzzer SP1 is connected to P3.5 of the MSP430. To sound the buzzer a pulse-width modulated signal (a square wave) with frequency of 2,500 Hz must be provided. The port P3.5 is multiplexed with TB4 – an output coming from the TimerB4 (capture and compare module 4 of TimerB).

C. (4 points) Describe how TimerB should be configured to drive the buzzer as described above. Outline steps taken in software during initialization and in steady-state operation. Assume SMCLK = 1MHz.

Configure TimerB to give a signal with requested period.
Use SMCLK as the source clock: 1,000,000/2,500 = 400 => TBCCR0 = 400 – 1 = 399, UP mode
To have a square wave at the output of the TB4, TBCCR4 = 199, use set/reset output mode.

An MSP430 development platform is connected to a workstation over a RS232 serial interface. Assume that MSP430 uses the following UART configuration: baud rate 38,400; 8-bit characters; 1 stop bit, parity bit is included (even parity). Note: ignore multiprocessing bit.

D. (2 points) What is the maximum number of characters you can send in 1 second from the MSP430 to the workstation?

the number of bits per character: 1 (start) + 8 + 1 (parity) + 1 (stop) = 11 bits per character.
the number of characters is less than 38,400 bits/sec / 11 bits = 3490

E. (2 points) (True | **False**) The number of characters your program can send to the workstation will dramatically decrease if the workstation starts sending characters to the MSP430 at the same time.

F. (2 points) (True | **False**) When sending ascii character for '0' (ascii('0')=48), the parity bit P = 1.
48 = 0x30 = 0011_0000 => even parity means that the total number of logic '1's should be even => P=0.

G. (2 points) Two devices C and D are communicating using a serial peripheral interface link. How many signals need to be routed between the devices excluding the common ground? Sketch the connection between the devices, including signal names.
3 or 4, depending on setting.
MOSI – master out slave in (from master to slave)
MISO – master in slave out (from slave to master);
SCLK – clock signal
SS – slave select (enable) (from master to slave)

H (2 points) Two devices E and F are communicating using a parallel bidirectional link with 8 data lines (half-duplex link, i.e., E and F cannot send data at the same time). In addition to data lines, what other control signals are needed to implement proper data communication? Sketch the connections.
8 data lines, e.g. use parallel port P3 on both devices
Handshaking signals, 4 in total
E2Frdy, F2Eack
F2Erdy, F2Eack

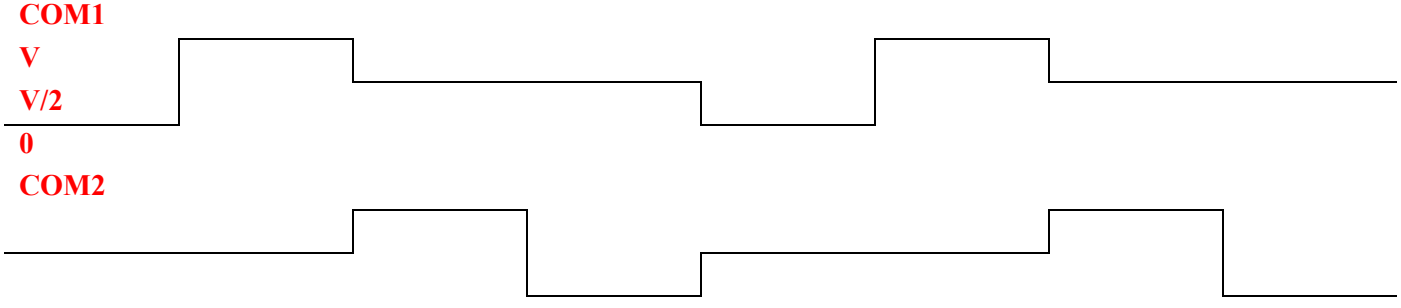
6. (7 points) BONUS QUESTION, DMA, LCD

Answer the questions or circle the correct answers when appropriate.
A. (1 points) (**True** | False) DMA controller can be initialized to perform a transfer of a block of data from a memory buffer to a serial communication interface (USCI peripheral).

B. (2 points) List DMA controller registers that enable transfer of a block of the data in part A and explain how they should be initialized.
DMAxSA – source address register
DMAxDA – destination address register
DMAxSZ – size register
DMAxCTRL – control register

C. (2 points) Could you use a DMA controller to transfer 128 samples coming from the ADC12 controller to a buffer in RAM memory? If yes, explain what steps would you take to configure the DMA properly? How do you know when the transfer has been completed?
Yes, a DMA channel can handle this transfer (DMA0SA=ADC12buffer; DMA0DA=membuffer; DMA0SZ=128, word transfer). The DMA will generate an interrupt request when the transfer has been completed.

D. (2 points, bonus) LCD_A controller is configure in the 2-MUX mode. How many common signals are provided by the controller in this mode? Sketch the waveform(s) for the common signal(s).
Two common signals.



Department of Electrical and Computer Engineering
University of Alabama in Huntsville

CPE 323 – Introduction to Embedded Computer Systems

Final Exam Solutions

Instructor: Dr. Aleksandar Milenkovic

Date: May 02, 2012

Place: EB 207

Time: 3:00 PM – 5:00 PM

Note: Work should be performed systematically and neatly. This exam is closed books and closed neighbor(s). Allowable items include exam, pencils, straight edge, calculator, and materials distributed by the instructor. The bonus question is not mandatory. Good luck!

Question	Points	Score
1	20	
2	20	
3	20	
4	20	
5	20	
6	10 (bonus)	
Sum	100+10(b)	

Please print in capitals:

Last name:_____

First name: _____

1. (20 points) MSP430 System Architecture/Miscellaneous

Answer the questions or circle the correct answers when appropriate.

A. (6 points) The MSP430FG4618 device has an address space of 128 KB (0x0000-0x1FFFF). Fill in the table below if we know the following. The first 16 bytes of the address space (starting at the address 0x00000) is reserved for special function registers (IE1, IE2, IFG1, IFG2, etc.), the next 240 bytes is reserved for 8-bit peripheral devices, and the next 256 bytes is reserved for 16-bit peripheral devices. The RAM memory of 8 Kbytes starts at the address 0x01100. Finally, right after the RAM memory starts the flash memory that occupying the rest of address space (the last byte address is 0x1FFFF).

What	Start Address	End Address
Special Function Registers (16 bytes)	0x00000	0x0000F
8-bit peripheral devices (240 bytes)	0x00010	0x000FF
16-bit peripheral devices (256 bytes)	0x00100	0x001FF
RAM memory (8 Kbytes)	0x01100	0x030FF
Flash Memory	0x03100	0x1FFFF

B. (2 points) What is the size of the flash memory (in Kilobytes) in the example above? *115.75 KB*

C. (2 points) (True | **False**) The MOV.W #34, &0x1000 will actually write the constant 34 into a flash memory location at the address 0x1000.
(Note: Cannot write into the flash memory using MOV.)

D. (2 points) (True | **False**) The program downloaded into the flash memory through a JTAG port will remain in the flash memory only until the power is turned on (i.e., will be lost when the device is turned off).

E. (2 points) (True | **False**) The content of the RAM memory is lost when a system goes into a low-power mode.

F. (2 points) (**True** | False) The content of the general-purpose and special-purpose registers is preserved while the processor is in a low-power mode.

G. (2 points) (**True** | False) An instruction is executed to set certain bits in the status register to cause a transition from the active mode into a low-power mode.

H. (2 points) (True | **False**) An instruction is executed to clear certain bits in the status register to cause a transition from a low-power mode into the active mode.
(Note: False, when in a sleep mode, the CPU is not executing instructions. Changing the copy of the status register on the stack to exit a LPM after ISR is completed does not fit this description (we modify the SR copy on the stack, not the SR itself.)

2. (20 points) Interrupts

Answer the questions or circle the correct answers when appropriate.

A. (2 points) (True | **False**) When a new interrupt request is received, the processor immediately stops processing the currently executing instruction (before finishing it) and starts the exception processing.

B. (2 points) (**True** | False) The RETI (return from interrupt) instruction retrieves both the status register and the program counter from the stack.

C. (2 points) (True | **False**) If multiple interrupt requests are pending, the processor selects the one that arrived first.

D. (2 points) (**True** | False) The GIE (Global Interrupt Enable) bit in the status register can be set inside an interrupt service routine explicitly by the programmer to enable nesting of interrupts.

E. (2 points) (**True** | False) In case that multiple interrupt request sources share a single entry in the interrupt vector table, programmers need to explicitly clear interrupt flag bits in the corresponding service routine.

Your task is to design an MSP430-based system that can interface 2 external devices (ID1, ID2), each capable of generating an interrupt request. The external devices place a request by setting the request line (a transition from a logic zero to a logic one). This line is kept active as long as the interrupt request is pending, until the request is serviced. Answer the following questions.

F. (2 points) How would you connect the interrupt request lines coming from the devices to the MSP430 platform?

The requests links ID1.irq and ID2.irq need to be connected to any two pins that belong to the parallel ports P1 and P2 that include support for external interrupts (PxIFG, PxIE, PxIES registers).

G. (2 points) In a section of the program you would like to service requests coming from the device ID1 but not the requests coming from the device ID2. What would you do to support this option?

We can set/reset appropriate bits in PxIE register to enable selective masking.

H. (2 points) Outline the main steps that need to be taken in the main program to initialize the system for accepting the interrupts from the devices ID1 and ID2.

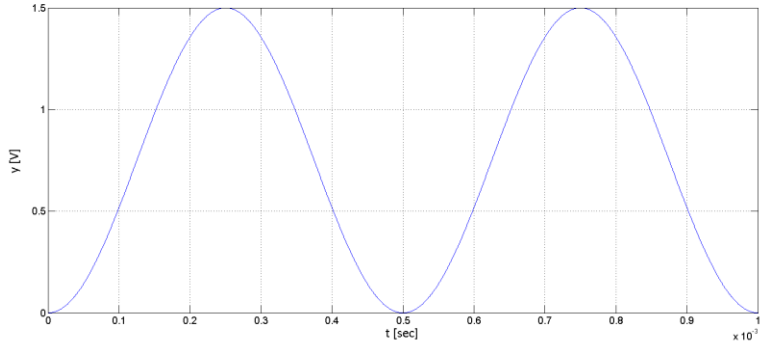
Main:
Configure ports P2.0 and P2.1 as interrupt request lines (input direction, sensitive to the rising edge, enable them);

I. (4 points) Outline the main steps that need to be taken inside the ISR for the interrupts that come from the devices. If both requests are pending concurrently, ID1 is processed first. Only one request can be processed at a time in the ISR. Outline the key steps in the main program.

P2_ISR:
if (ID1 is pending) then {Clear P2.IFG[0]; Process the interrupt for ID1; RETI}
else if (ID2 is pending) then {Clear P2.IFG[1]; Process the interrupt for ID2; RETI}

3. (20 points) Embedded Software Design (Digital to Analog Conversion).

To drive a sensor you need to provide the following wave signal y, $y = 1.5*\sin^2(x)$, as shown below (x=2*pi*f*t, f=1000 Hz, t is time in seconds). To accomplish this task we will use the MSP430's digital-to-analog converter (DAC12) peripheral and a lookup table prepared in memory, similarly to the laboratory exercise.



A. (2 points) What is the maximum and minimum output voltage at the analog output? What is the reference voltage you would use to generate the required signal?

Max: 1.5 V (x=pi/2, sin(x)=1)
Max: 0 V (x=0 , sin(x)=0)
Vref=1.5 V

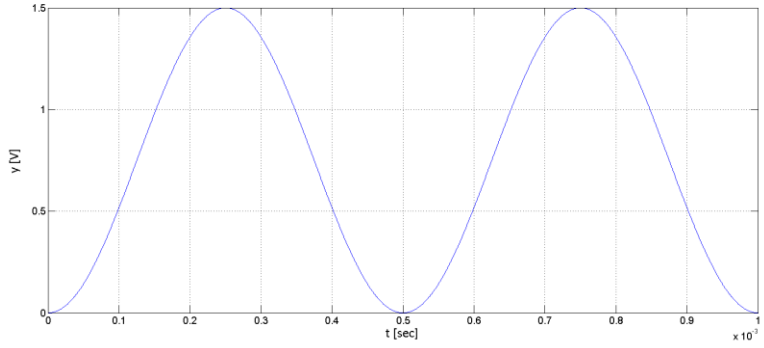
B. (2 points) What is duration of one period of the wave signal y in miliseconds?

T = 0.5 ms

C. (4 points) Let us assume that we provide a lookup table with 10 samples for a single period of the signal y? Fill in the following table with the first 5 samples in the table (sample #1 has been provided in the table).

Sample	***t=?***	***sin(2*pi*f*t)***	***y=?***	***Lookup table***	
				decimal	***hex***
1	**0**	**0**	**0**	**0**	**0x0000**
2	$0.05*10^{-3}$	0.309	0.143	4095*y/Vref = 390	0x0186
3	$0.1*10^{-3}$	0.588	0.518	1,404	0x057C
4	$0.15*10^{-3}$	0.809	0.654	1,785	0x06F9
5	$0.2*10^{-3}$	0.951	1.358	3,707	0x0E77

D. (2 points) Sketch the output analog signal for the first 0.5 ms (as you would see it on the oscilloscope).



E. (2 points) If we use a TimerA ISR to control sending the next value to the DAC12, how many interrupts per second TimerA should generate?

0.5 ms : 10 = 1 sec : n => n = 10*2,000= 20,000

F. (2 points) If $f_{MCLK}=f_{SMCLK}$= 1 MHz, how would you configure TimerA?

Use SMCLK as a source clock for TimerE. Initialize TimerA in UP mode with TACCR0=1 MHz/20,000 - 1= 49

G. (2 points) You wrote the TimerA ISR that reads an entry in the lookup table, increments the table pointer and checks its range, and sends the value to the DAC12. By profiling you determined that the ISR requires 20 processor clock cycles? Is it possible to implement the sine wave signal generator under these conditions? Elaborate your answer?

The TimerA generates an interrupt every 50 CPU clock cycles. The time required for the ISR is only 20 cpu clock cycles. So, the answer is yes, we can implement the sine wave generator without any problems.

H. (4 points) How would you improve this design? Elaborate your answer. What needs to be changed in the code?

1) ***We can reduce the size of the lookup table and exploit the symmetry of the signal (smaller footprint in flash).***
2) ***We can consider a finer resolution of the required signal. E.g., we can provide 40 samples per one period (every 25 cpu clock cycles). As we spend 20 clock cycles in the DAC12 ISR, this can still work properly. Alternatively, we can increase the processor clock cycle.***
3) ***Use DMA to transfer samples from the memory to the DAC12.***

4. (20 points) Time, Timers

Answer the questions or circle the correct answers when appropriate.

A. (2 points) (**True** | False) The MSP430's clock module can internally generate clocks used by the processor and peripherals or use external crystal oscillators.

B. (2 points) (True | **False**) MCLK, SMCLK, and ACLK are fixed and cannot be changed in software.

C. (2 points) (True | **False**) To prevent a controlled reset of the systems, the watchdog timer in the watchdog mode requires a hardware signal to clear its control bit before the time period expires.

D. (2 points) (**True** | False) TimerA can be configured to generate pulse-width modulated signals without intervention from software.

E. (2 points) What does it involve capturing an external event in TimerA (what happens inside TimerA)?

When an external signal changes, the current content of the TimerrA counter (TAR) is latched into the one of the capture and control registers (CCRx) which can later be read from software.

Let us consider a program that displays time on the LCD of the TI's Experimenter platform. The time is displayed with resolution of one tenth of a second (100 ms, deciseconds). The processor wakes up periodically, updates local time variables, performs necessary format conversions, and displays the time on an LCD in the following format (hh.mm.ss.d). These steps take N=2,000 processor clock cycles during which the processor is in the active mode; after that the processor goes back into a low-power mode. The MCLK in the active mode is 1 MHz. The platform draws 1 mA when the MSP430 is in the active mode and 0.1 mA when in the low-power mode.

F. (3 points) What is the total active and sleep time during one application cycle? Calculate the average current drawn by the platform.

Application cycle is 100 ms.

I_{ACTIVE} = 2000*(1/1*10^6) = 2 ms, $T_{LOW\text{-}POWER}$ = 100 – 2 = 98ms

I_{AVG} = (2/100)*1.0mA + (98/100)*0.1mA = 0.02 mA + 0.098 mA = 0.118 mA

G. (3 points) Calculate the total power P consumed (in milliWatts) if we power the platform by two AA batteries (V=3 V). Determine the system operating time in days if we know that the battery capacity is 2500 mAh.

$P = V*I_{AVG}$ = 3*0.118mA = 0.354 mW

$OT = BC/I_{AVG}$ = 2500/0.118 ≈ 21186 hours ≈ 882 days

H. (4 points) Outline briefly organization of the program that displays time (list peripherals needed, initialization, main loop, interrupt service routines).

We will need TimerA and LCD Controller.

Initialization: Set TimerA to generate an interrupt every decisecond (100 times in a second).E.g., UP mode with MCLK/8 clock on TimerA (125,000 counts in a second) and 125,000/10 = 12,500 => 12,499 in CCR0

LCD_A: Mux2 or Mux4 mode,…

TimerA ISR: update local variables (hours, minutes, seconds, , exit LPM.

Main loop: Enter LPM. Convert variables into digits to be displayed on LCD_A, update LCD_A buffer.

5. (20 points) Communication

Answer the questions or circle the correct answers when appropriate.

A. (2 points) Two devices A and B are communicating using a bidirectional asynchronous serial link. How many signals need to be routed between the devices excluding the common ground?

2 (TxD and RxD)

B. (2 points) What is the total number of bits transferred when a single character is sent from device A to device B using the asynchronous serial link? Assume the following: 8-bit character, parity is on, 2 stop bits.

1+8+1+2 = 12 bits

C. (2 points) How long does it take in seconds to transfer the character from above, assuming that the devices are configured for 57,600 bits per second?

12 bits/57,600 = 0.208 ms

D. (2 points) What happens when a new character is received in the receive buffer before the previous one is read by the software?

The original character is lost, but the buffer overrun error condition bit is set. The error conditions may be investigated in the program.

E. (2 points) What does this interrupt service routine do?

```
#pragma vector=USCIAB0RX_VECTOR
__interrupt void USCIA0RX_ISR (void)
{
  while(!(IFG2&UCA0TXIFG));  //
  UCA0TXBUF = UCA0RXBUF;     //
}
```

Echo a received character (send it back).

F. (4 points) Two devices C and D are communicating using a serial peripheral interface link. How many signals need to be routed between the devices excluding the common ground? Sketch the connection between the devices, including signal names.

3 or 4, depending on setting.
MOSI – master out slave in (from master to slave)
MISO – master in slave out (from slave to master);
SCLK – clock signal
SS – slave select (enable) (from master to slave)

G. (4 points) Two devices E and F are communicating using a parallel bidirectional link with 8 data lines (half-duplex link, i.e., E and F cannot send data at the same time). In addition to data lines, what other control signals are needed to implement proper data communication? Sketch the connections.

8 data lines, e.g. use parallel port P3 on both devices
Handshaking signals, 4 in total
E2Frdy, F2Eack
F2Erdy, F2Eack

H. (4 points) What needs to be done in software during initialization and what during data communication assuming the following protocol: E starts exchange by sending one byte and F responds back by sending one byte.

E.P3 <=> F.P3 (bidirectional 8-bit link)
E.P4.0 -> F.P4.0 (E2Frdy)
E.P4.1 <- F.P4.1 (F2Eack)
E.P4.2 <- F.P4.2 (F2Erdy)
E.P4.3 -> F.P4.3 (E2Fack)

Device A Initialization:

Configure P3 initially as output P3DIR=0xFF; // (A initiates communication)

Configure P4.0 and P4.3 as output pins;

Configure P4.1 and P4.2 as input pins;

Device B Initialization:

Configure P3 initially as input (A initiates communication);

Configure P4.0 and P4.3 as input pins;

Configure P4.1 and P4.2 as output pins;

Device A	***Device B***

P3OUT = sbyte1;
Set P4OUT.0; // set E2Frdy
Wait for P4IN.1 to become set; //wait for F2Eack.high
Clear P4.OUT.0; // clear E2Frdy
Wait for P4IN.1 to become clear; // wait for F2Eack.low
P3DIR = 0x00; //set P3 to be input, change direction
Wait for P4IN.2 to become set; // F2Erdy.high
rbyte1=P3IN;
Set P4IN.3; // E2Fack
Wait for P4IN.2 to become clear; // F2Erdy.low
Clear P4IN.3; // clear E2Fack
	Wait for P4IN.0 to become set; // E2Frdy.high
	rbyte=P3IN;
	Set P4OUT.1; // F2Eack = 1
	Wait for P4IN.0 to become clear; // E2Frdy.low
	Clear P4OUT.1; // end of first transfer
	P3DIR = 0xFF; // change direction
	P3OUT=sbyte1;
	Set P4OUT.2; // F2Erdy
	Wait for P4IN.3 to become set;
	Clear P4OUT.2;

6. (10 points) BONUS QUESTION, DMA, LCD

Answer the questions or circle the correct answers when appropriate.

A. (2 points) (**True** | False) DMA controller can be initialized to perform a transfer of a block of data from one place in memory to another place in memory.

B. (3 points) List DMA controller registers that enable transfer of a block of data and explain their function.

DMAxSA – source address register

DMAxDA – destination address register

DMAxSZ – size register

DMAxCTRL – control register

C. (3 points) Could you use a DMA controller to transfer 128 samples coming from the ADC12 controller to a buffer in RAM memory? If yes, explain what steps would you take to configure the DMA properly? Use illustrations. How would you know when the task has been completed?

Yes, a DMA channel can handle this transfer (DMA0SA=ADC12buffer; DMA0DA=membuffer; DMA0SZ=128, word transfer). The DMA will generate an interrupt request when the transfer has been completed.

D. (2 points, bonus) What segments need to be ON to display a digit 5 on an LCD display?

a, c, d, f, g

Department of Electrical and Computer Engineering
University of Alabama in Huntsville

CPE 323 – Introduction to Embedded Computer Systems
Final Exam Solutions

Instructor: Dr. Aleksandar Milenkovic

Date: April 30, 2014

Place: SC 109

Time: 3:00 PM – 5:30 PM

Note: Work should be performed systematically and neatly. This exam is closed books and closed neighbor(s). Allowable items include exam, pencils, straight edge, calculator, and materials distributed by the instructor. Best wishes.

Question	Points	Score
1	20	
2	20	
3	20	
4	20	
5	22	
Sum	102	

Please print in capitals:

Last name:_____

First name: _____

1. (20 points) MSP430 System Architecture/Miscellaneous

Answer the questions or circle the correct answers when appropriate.

A. (6 points) The MSP430FG4617 is a microcontroller with address space divided between code memory (flash), RAM memory, and input/output peripherals. It has 8 KBytes of RAM memory starting at the address 0x01100, and 256 Bytes of address space reserved for special purpose registers and 8-bit input/output peripherals (starting at the address 0x0000), followed by 256 Bytes reserved for 16-bit input/output peripherals. The flash memory of 92 KB placed at the address that follows the RAM memory.
Determine the address map by filling in the following table.

Address	Address [hexadecimal]	What?
Last Flash address	**0x1A0FF**	Flash Memory
First Flash address	**0x03100**	
Last RAM address	**0x030FF**	RAM Memory
First RAM address	**0x01100**	
Last I/O address (16-bit per.)	**0x001FF**	I/O address space
First I/O address (16-bit per.)	**0x00100**	
Last I/O address (8-bit per.)	**0x000FF**	I/O address space
First I/O address (8-bit per.)	**0x00000**	

B. (2 points) The interrupt vector table includes 32 entries and contains 16-bit addresses (i.e., interrupt service routines can reside in the first 64 KByte of address space). What is the address range of the interrupt vector table?

Interrupt Vector Table	First address	Last address
--	**0x0FFC0**	**0x0FFFF**

C. (2 points) What is the maximum size of the program stack in the MSP430FG4617 from A?
___**4096**_____WORDS

D. (2 points) (True | **False**) The content of the RAM memory remains intact if the power is turned off?

E. (2 points) (True | **False**) The content of the RAM memory is lost when the processor goes into a low-power mode (clocks are turned off).

F. (2 points) (True | **False**) The RETI instruction that returns from the interrupt service routine retrieves only the program counter from the top of the stack as follows: PC <= M[SP]; SP <= SP + 2;

G. (2 points) (**True** | False) The content of the FLASH memory remains intact when the MSP430 goes into a low-power mode.

H. (2 point) (**True** | False) Instruction MOV.B 0x01100, R7 reads a byte from RAM memory from address 0x01100 and stores it into register R7.

2. (20 points) Interrupts

Answer the questions or circle the correct answers when appropriate.

A. (2 points) (True | **False**) During exception processing in hardware, the R0 (PC, program counter) and R1 (SP, stack pointer) registers are pushed on the stack.

B. (2 points) (True | **False**) The GIE bit in the status register retains its original value during exception processing in hardware.

C. (2 points) (True | **False**) *An interrupt enable bit* associated with a peripheral is always automatically cleared upon accepting the corresponding interrupt request.

D. (2 points) (**True** | False) An interrupt flag bit remains set as long as the corresponding interrupt request is pending (waiting to be serviced) and can be cleared automatically in hardware upon accepting the interrupt request or explicitly in software.

E. (2 points) (**True** | False) If multiple interrupt requests are pending at the time of exception processing, the fixed priority is used to select the one that is processed first.

An MSP430-based system interfaces switches SW1 and SW2 connected to P1.0 and P1.1, respectively. When a switch is pressed, the ground is connected to the corresponding input pin (a logic '0'). When SW1 is pressed LED1 (connected to P2.1) should be turned on (a logic '1' on the port). When SW2 is pressed LED2 (connected on P2.2) should be turned on. When switches are released the leds should be turned off.

F. (3 points) Specify registers that need to be initialized in the main program to configure the system. Fill in the table below. Note: to specify interrupts active on the falling edge the edge-selection bits should be set to 1.

Register	Full Name	Content after initialization (in binary)
P1DIR	Port 1 Direction register (input for P1.0 and P1.1);	????_??00 b;
P2DIR	Port 2 Direction register (output for LED1 & LED2);	????_?11? b
P1IES	Port 1 Interrupt Edge Selection (switch is pressed)	????_??11 b
P1IE	Port 1 Interrupt Enable	0000_0011 b
SR	Status register	GIE=1

G. (2 points) How many interrupt service routines is needed to handle SW1 and SW2?

1 (P1_ISR)

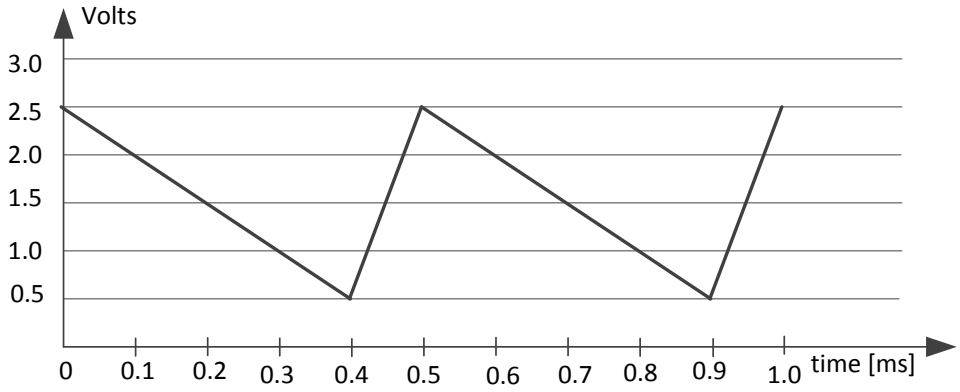
H. (5 points) Outline the interrupt service routine(s) and the main loop for handling requests from SW1 and SW2 as described above.

```
P1_ISR:
if (SW0 == 0 or SW ==0 ) {
        debounce;
        if (SW1 == 0) { set LED1 on; clear P1.IFG[0];}
        if (SW2 == 0) { set LED2 on; clear P1.IFG[1];}
}

Main loop:
for (;;) {
        while (SW1 == 0 and SW2 == 0);
        if (SW1 == 1) set LED1 off;
        if (SW2 == 1) set LED2 off;
}
```

3. (20 points) Embedded Software Design (Digital to Analog Conversion).

We are using a 10-bit analog-to-digital converter to sample a periodic input signal shown below. Answer the following questions.



A. (2 points) What is the maximum and the minimum voltage at the analog input? Fill in the table below by specifying min/max values and times when those values are achieved.

Min/Max	Value [Volts]	Time [milliseconds]
Min	0.5	0.4,0.9, 1.4, …
Max	2.5	0, 0.5, 1.0, 1.5, 2, …

B. (2 points) What is the duration of one period of the signal in milliseconds?
T = 0. 5 ms

C. (2 points) What is a minimum change of the input signal that can be detected by the ADC10, if we use reference voltages Vref- = 0 V and Vref+ = 3 V?
V_{LSB} = 3/2^10 = 0.0029 V = 2.9 mV

D. (6 points) What values would you read from the 10-bit ADC if your sampling rate is 10,000 Hz. Fill in the following table with the values for the first 5 samples (assume the first sample is taken at t = 0 s). Assume Vref- = 0 V and Vref+ = 3 V.

Sample	*t=? [ms]*	*Signal*	*Samples from ADC [unsigned value in decimal]*
1	**0.0**	**2.5**	**852**
2	**0.1**	**2.0**	**682**
3	**0.2**	**1.5**	**511**
4	**0.3**	**1.0**	**341**
5	**0.4**	**0.5**	**170**

E. (2 points) Describe steps taken during program initialization to configure the analog-to-digital conversion as described above.
Specify reference voltages (Vref+ =3V, Vref- = 0);
Specify input channels;
Specify sampling time;
Specify triggers (TimerB);
Enable ADC service routines if used;

F. (2 points) If we use a TimerB to trigger an ADC conversion, how would you configure TimerB assuming its input clock, SMCLK, is set to 2,000,000 Hz?

2,000,000/10,000 = 200 => set TimerB in UP mode and TB_CCR0=199

G. (2 points) Is it possible to improve resolution (V_{LSB}) of the analog-to-digital conversion described in this problem? If yes, explain what changes would you make?

Yes. The signal is bounded between 0.5 and 2.5 V, and we can use these as reference voltages Vref+ = 2.5V and Vref- = 0.5 V (provided externally or from DAC12). This way, we can improve resolution as follows: V_{LSB} = 2/2^10 = 0.019 mV

H. (2 points) Assume you want to generate a signal described above using a DAC12 and a lookup table with predefined samples for 50,000 Hz sampling frequency. What is the size of the lookup table in bytes you will need to prepare?

25 samples per period => 25*2 bytes = 50 bytes

4. (20 points) Time, Timers, Operating time

Answer the questions or circle the correct answers when appropriate.

A. (2 points) (**True** | False) The MSP430 FLL+ module can be configured in active mode as follows: ACLK = 32,767 Hz, MCLK = 4,194,304 Hz, and SMCLK = 1,048,576 Hz.

B. (2 points) (**True** | False) The MSP430 clocks can be sourced from an internal digital control oscillator or external crystal oscillators or resonators.

C. (2 points) (**True** | False) The MSP430's TimerA and TimerB are 16-bit peripherals that can be used to generate multiple pulse-width modulated signals.

D. (2 points) (True | **False**) A single capture and compare block of TimerA/TimerB can be configured in both capture and compare modes.

E. (2 points) How do we exit a low-power mode in the MSP430-based systems to continue processing in the main program?

Change the status register saved on the stack inside an ISR.

Consider the following code segment that utilizes the watchdog timer in in the interval mode with period set to 1 s (line 4 of the code).

```
1.  #include <msp430xG46x.h>
2.  void main(void) {
3.    int p = 0;
4.    WDTCTL = WDT_ADLY_1000; // 1 s interval timer
5.    P2DIR |= BIT2; // Set P2.2 to output direction
6.    for (;;) {
7.      if ((IFG1 & WDTIFG) == 1) {
8.        p++;
9.        if (p == 4) {P2OUT ^= BIT2; p=0;}
10.       IFG1 &= ~WDTIFG;
11.     }
12.   }
13. }
```

F. (5 points) What does the code segment do?

Using polling it checks when WDT counts down every 1 second. It increments local variable p modulus 4 (0, 1, 2, 3), and toggles the output every 4 seconds => it toggles the output port P2.2 every 4 seconds.

G. (5 points) How would you implement the given functionality using an interrupt service routine.

```
1.  #include <msp430xG46x.h>
2.  void main(void) {
3.    int p = 0;
4.    WDTCTL = WDT_ADLY_1000; // 1 s interval timer
5.    P2DIR |= BIT2; // Set P2.2 to output direction
6.    IE1 |= WDTIE;
7.   _BIS_SR(LPM0_bits + GIE);
8.  }
9.  #pragma vector = WDT_VECTOR
10. __interrupt void wdt(void) {
11.   static int p = 0;
12.   p++; if (p == 4) { P2OUT ^= BIT2; p = 0; }
13. }
```

5. (20 points +2 bonus) Misc. peripherals

A. (4 points) Consider two experimenter's boards, A and B. You need to create a half-duplex parallel link between A and B, so they can exchange 8-bits as follows: A sends one byte to B, and then B responds by sending two bytes to A. You have ports 4 and 5 (P4, P5) pins available on both boards for this purpose. Give a block diagram that specifies all the wires needed to carry out this communication. Fill in the table below with wire names and their purpose.

Wire Name	Direction (A2B or B2A)	Description
A2BData P4[7:0]	A2B / B2A	Data lines between A and B
A2Brdy P5.1	A2B	A placed data on the data lines
B2Aack P5.2	B2A	B read the data
B2Ardy P5.3	B2A	B placed data on the data lines
A2Back P5.4	A2B	A read the data
Ground	A2B, B2A	Common ground

<here comes a block diagram>

B. (2 points) What is the minimum number of wires (do not include the common ground) needed between A and B to carry out the communication from above?

Assuming that control signals can also be reconfigured as described above, the minimum number of wires is 10 (8 for data and two for control).

C. (4 points) Sketch a code that carries out the exchange on device A.

```
// configure: P4 output, P5.1 output, P5.2 input, P5.3 input
P4OUT = value_to_be_sent;
P5.1 = '1'; set A2Brdy
while (P5.2 == '0'); wait for B2Aack
P5.1 = '0'; clear A2Brdy
// configure P4 as input;
while (P5.3 == '0'); wait for B2rdy
received_value_1 = P1IN;
P5.4 = '1'; set A2B ack
while (P5.3 == '1'); wait for A2Brdy to clear
while (P5.3 == '0'); wait for B2rdy
received_value_1 = P1IN;
P5.4 = '1'; set A2B ack
```

Consider the following C source code.

```
1.  char gm1[] = "MSP430";

2.  void UART0_putdchar(char c) {
3.    while (!(IFG2 & UCA0TXIFG));
4.    UCA0TXBUF = c;
5.    while (!(IFG2 & UCA0TXIFG));
6.    UCA0TXBUF = '_';
7.  }
8.  …
9.  for(int i = 0; i < 6; i++) {
10.   ch = gm1[i];
11.   UART0_putdchar(ch);
12. }
```

D. (4 points) What does the code segment from lines 9 – 12 do? USCIA0 is configured in the UART mode.

It sends a message "M_S_P_4_3_0_" over a serial asynchronous link. The total number of characters sent is 12.

E. (3 points) USCI0 is configured in the UART mode to transfer 19,200 bits/second, 8-bit characters, no parity, and one stop bits. Estimate the time needed to execute the code segment from lines 9 to 12. You can ignore time needed to execute the non-waiting instructions.

Because we are sending 12 characters one by one using polling, the execution time of this procedure is limited by the communication speed. Thanks to double-buffering this procedure will at least take the time needed to send 11 characters or (11*10)/19,200 = 5.7 ms. Note: the entire transfer will obviously take at least 11*11/19,200 = 6.3 ms.

F. (3 points) Could you do the transfer described above using a DMA channel? If yes, explain what changes would you make and how you would initialize the DMA? Specify the content of relevant registers.

Yes, the transfer can be done using a DMA channel. We would initialize UART.
We will need to modify the message as follows: char gm1[] = "M_S_P_4_3_0_";

DMAxSA – source address register = gm1 (starting address of this string)
DMAxDA – destination address register = USCA0TXBUF
DMAxSZ – size register = 0x000C
DMAxCTRL – control register (increment SA, fixed DA, byte transfer, triggered by USCIA0 TxRDY bit)

G. (2 points, BONUS) You need to interface an LCD with four 8-segment digits from your platform that includes the LCD controller. If you are using 2-mux mode, how many port signals you need to run between your microcontroller and display?

2 COM signals + (8/2)*4 = 18 wires.