USAF ACADEMY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ECE 382 GRADED REVIEW #1
FALL 2016

Name: _____ **Section:** _____

Academic Security	This examination is not released from academic security until **1630** on **30 September 2016**. Until this time, you may not discuss the examination contents or the course material with anyone other than your instructor.
Integrity	Your honor is extremely important. This academic security policy is designed to help you succeed in meeting academic requirements while practicing the honorable behavior our country rightfully demands of its military. Do not compromise your integrity by violating academic security or by taking unfair advantage of your classmates.
Authorized Resources	1. A clean copy of the MSP 430 blue book 2. Any calculator without ECE 382 equations or concepts
Instructions	▪ Show all work for full credit ▪ Box or circle your final answer. ▪ For all numerical answers, use engineering notation and include units. ▪ Completely label all your diagrams, drawings, graphs, etc. for full credit. ▪ You have **53 minutes** to complete this exam.

Problem	Value	Earned
1	20	
2	20	
3	20	
4	20	
5	20	
Total	100	

This page intentionally left blank

Problem 1 (20 points) *General Knowledge*

Answer the following questions based on your knowledge of the MSP430 and its assembly language.

a) (4 pts) The _____ takes an _____ file (s) and creates an executable program.

The _____takes an assembly language program and creates _____object code.

b) (4 pts) What is the purpose of the following line of code?

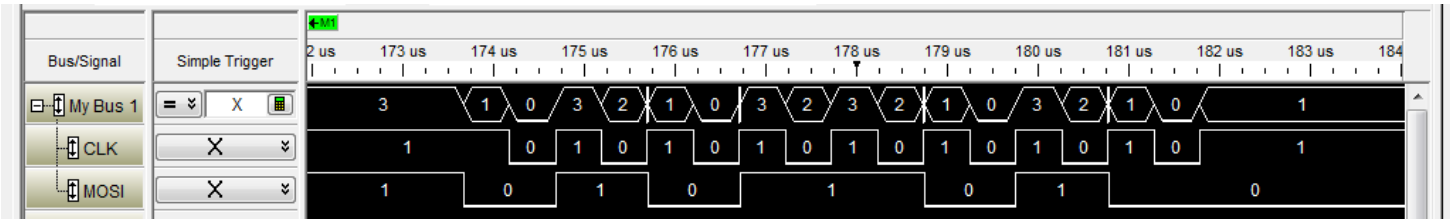
```
mov.w   #WDTPW|WDTHOLD, &WDTCTL
```

c) (4 pts) Circle the correct answer.

- The x86 is a(n): Instruction Set Architecture Microarchitecture
- The MSP 430 is: CISC RISC

d) (4 pts) Name one advantage and one disadvantage of using serial over parallel communication.

e) (4 pts) What is the hex value of the byte being sent via 8-bit SPI in the below logic analyzer output? Note: the clock phase bit and the MSB first select bit are set to 1.



Clock Polarity value = _____

Value Sent = 0x_____

Problem 2 (20 points) *Addressing Modes, Machine code , Status Register*

a) (6 pts) For each of the following assembly instructions, identify the addressing mode(s) used and convert the instruction to machine code. Only convert the instruction – don't worry about any extension words. **Write your answer in hex – don't convert to Little Endian.**

mov.w @r3+, r5

Machine Code: 0x_____

Addressing Mode(s):

b) (3 pts) You discover the hex number 0x9503 and think it is an assembly instruction. Write the line of assembly code that assembles to 0x9503.

c) (6 pts) This is the current state of the status register. Circle the jumps that will take place.

V	SCG1	SCG0	OSCOFF	CPUOFF	GIE	N	Z	C
0	0	0	0	0	1	1	0	1

JHS JL JMP JEQ

Use the following snippet in part d) for part e) as well.

d) (3 pts) What are the contents of r6 after the following code is assembled and executed?

```
        .data
yourMsg:    .byte   0xED, 0xCB
        .text
myMsg:      .byte   0xBA, 0xDA          r6 = 0x _____
        mov     #myMsg, r6
```

e) (2 pts) At which address is yourMsg stored?

Problem 3 (20 points) *Software Delays, Subroutines, and Coding*

Use the program snippet below to answer the subsequent questions.

```
      ; Main                                  # cycles per instruction
; initialize P1.3 for the button input                    ;P1.3 is button
; initialize P1.6 as the LED output                       ;P1.6=green LED
...
check_btn:  bit.b     #BIT3, &P1IN
            jnz       check_btn
            call      #mystery
            jmp       check_btn
mystery:      mov     #3, r6                            ; 3 big loops
do_again:     mov     #8, r7                            ; 8 waiting loops
              mov     #2, r8                            ; 2 waitmore loops
              bic.b   #BIT6, &P1OUT
waiting:      call    #delay1
              dec     r7
              jnz     waiting
              bis.b   #BIT6, &P1OUT
waitmore:     call    #delay1
              dec     r8
              jnz     waitmore
              dec     r6
              jnz     do_again
              bic.b   #BIT6, &P1OUT
              ret
delay1:       push    r5          _____________________
              mov     #0x9000, r5 _____________________
go_again:     dec     r5          _____________________ ;sub #1, dst – tricky
              jnz     go_again    _____________________
              pop     r5          _____________________ ;mov @SP+, dst
              ret                 _____________________ ;mov @SP+, PC
```

a) (4 pts) In one short sentence, describe to my grandma what this program does with respect to I/O. Be specific.

b) (10 pts) Fill in the number of cycles per instruction in the delay loop in the chart above. How many cycles does the delay loop last? Given a 8MHz clock, how long does the loop delay?

Delay1 subroutine total cycles: _____ Total time: _____

c) (6 pts) Think about the coding skills and standards we've used in this class. What are <u>two</u> **different** ways you could improve this code (improve coding technique, make it more efficient, etc.)? **Be specific** about the problem you'd fix, which lines would you alter and exactly how would you change those lines.

Problem 4 (20 points) *SPI and Polling*

You want to use your MSP430 to connect to an LCD screen. In fact, you have found some code that will initialize the LCD screen for you, but it does not initialize your MSP430. Fill in the blanks in the code that will allow you to use the USCI_A0 module in 3-pin SPI mode to communicate with the LCD. Be sure to address each of the comments given below. The use of labels is encouraged, but hex values are allowed.

```
; Main
            call   #initMSP
;-------------------------------------------------------------------------------
;      Name: initMSP
;      Inputs: none
;      Outputs: none
;      Purpose: initialize the SPI on the USCI_A0 module of the MSP430
;      Registers: none
;-------------------------------------------------------------------------------
initMSP:
a) (2 pts)  _____________________________          ; step 1 of initializing USCI

b) (4 pts)  mov #_______________, &UCA0CTL0        ; data captured on first edge
                                                   ; inactive state low
                                                   ; transmit the MSB first
                                                   ; transmit 8 bits at a time
                                                   ; our MSP430 is the master
                                                   ; 3-pin SPI
                                                   ; synchronous mode

c) (2 pts)  bis.b #0x80, _________________          ; SMCLK is the UCA clock source

d) (2 pts)  mov #_____________, &UCA0BR0           ; use the max bit clock BRCLK, not
                                                   ;a scaled version (i.e. div
            _____________________ &UCA0BR1         ;by 1), like in lab3

                                                   ; for these next signal, prevent
                                                   ;glitches by defining the value of
                                                   ;the pin before you define the
                                                   ;direction of the pin

e) (4 pts)  bis.b #_____________, &_____________   ; initialize P2.5 as the chip
                                                   ;select signal for the LCD
            bis.b #_____________, &_____________

f) (4 pts)  ________ #_____________, &P1SEL        ; expose UCA0MOSI, UCA0CLK signals

            ________ #_____________, &P1SEL2

g) (2 pts)  ______ #UCSWRST, &UCA0CTL1             ; last step of initializing USCI
                                                   ;without interrupts
            ret
```

Problem 5 (20 points) *The Stack*

Use the following program with its instruction addresses and initial register values for this problem.

```
main:
0xC100: push    r8
0xC102: push    r7
0xC104: call    #mySub
0xC108: forever:   jmp   forever

mySub:
0xD000: push.b  r9
0xD004: mov     6(SP), r7
0xD006: pop.b   r8
0xD008: pop.b   r10
0xD00A: ret
…
```

```
;Initial Register Values
R7:  0xC100
R8:  0xF35A
R9:  0xAF22
R10: 0xCAFE
R11: 0xDFEC
```

a. (10 pts) Assuming the stack pointer was at 0x400 before the program started, what does stack look like when the program is executing the instruction at 0xD00A? Fill in the chart below. Assume the stack was initially filled with 0xBEEF.

0x3F6	
0x3F7	
0x3F8	
0x3F9	
0x3FA	
0x3FB	
0x3FC	
0x3FD	
0x3FE	
0x3FF	
0x400	

b. (2 pts each) After the program has executed the instruction at 0xD00A,
   a. What is the value of the stack pointer? _____
   b. What is the value of R7? _____
   c. What is the value of R8? _____

c. (4 pts) This code will compile, but it contains an error that will not allow the program to run as intended. Describe the error and specifically how you would fix it.