16.317: Microprocessor-Based Systems I

Summer 2012

Exam 3 Solution

1. (20 points, 5 points per part) ***<u>Multiple choice</u>***

For each of the multiple choice questions below, clearly indicate your response by circling or underlining the single choice you think best answers the question.

a. Which of the following statements about microcontrollers are <u>true</u>?

A. A microcontroller typically contains a processor, on-chip storage, and additional peripheral devices.

B. Microcontrollers are ideal for workloads that require significant processing power.

C. Microcontrollers are always used by very small people to control very small remote-controlled cars.

D. Microcontrollers are typically low-power devices, allowing them to be widely used in battery-powered embedded systems.

i. A and C

ii. B and C

iii. ***<u>A and D</u>***

iv. B and D

b. Which of the following statements about the PIC16F684 microcontroller are <u>true?</u>

A. The PIC16F684 system stack is used to store function return addresses, function arguments, and registers that need to be saved before being overwritten by the function

B. The PIC16F684 uses a Harvard memory architecture, which merges program and data memory into a single module.

C. The RP0 bit in the PIC16F684 STATUS register determines which data memory bank is currently being accessed.

D. The only true data register in the PIC16F684 is the working register (W). All other "registers" are technically stored in data memory, which is divided into special function registers (SFRs) and general purpose registers (GPRs).

i. A and B

ii. <u>C and D</u>

iii. B and D

iv. A and C

c. You are given the PIC function shown below:

```
F:      movf   PORTA, W
        andlw  0x03
        addwf  PCL, F
        retlw  0xAA
        retlw  0xBB
        retlw  0xCC
        retlw  0xDD
```

What is the return value of this function if the current value of PORTA is 0x3F?

i. 0xAA

ii. 0xBB

iii. 0xCC

iv. <u>0xDD</u>

d. Choose your favorite statement(s) from the list below. Circle all that apply (but don’t waste too much time!):

    i. “I’m so glad I spent the last six weeks taking this course—that’s how I’ve always wanted to spend July and August!”

       (You may be more likely to agree with this statement if you imagine saying it with just a hint of sarcasm.)

    ii. “The exam ends here, right? Everything after this is just reference material … isn’t it? Please?”

    iii. “I would have been perfectly happy not doing Lab 5. Really. Sure you don’t want to reconsider that ‘extra credit’ idea?”

    iv. “I’d actually like to take this opportunity to leave some constructive feedback in the space below” *(Note: don’t do this unless you have time left at the end of the exam.)*

<u>All of the above are “true.”</u>

(40 points) ***Reading PIC assembly language***
Show the result of each PIC 16F684 instruction in the sequences below.

a. (22 points)

```
    cblock 0x20
        x, y
    endc

    movlw     0x34          W = 0x34

    addlw     0x33          W = W + 0x33 = 0x34 + 0x33 = 0x67

    movwf     x             x = W = 0x67

    comf      x, F          x = ~x = ~0x67 = ~(0110 0111) = 1001 1000 = 0x98

    subwf     x, F          x = x – W = 0x98 – 0x67 = 0x31

    btfsc     x, 7          Test bit 7 of x; skip following instruction if clear
                                x = 0x31 = 0011 0001 → bit 7 = 0, so skip goto

    goto      L1

    swapf     x, W          W = value in x with nibbles swapped = 0x13

    goto      L2            Go directly to L2—skip addlw

L1: addlw     0x11

L2: movwf     y             y = W = 0x13
```

2 (cont.)

b. (18 points)

```
cblock 0x20
    q
endc
movlw    0xF1          W = 0xF1

movwf    q             q = W = 0xF1

andlw    0xC3          W = W & 0xC3 = 0xF1 & 0xC3 = 0xC1

iorlw    0x18          W = W | 0x18 = 0xC1 | 0x18 = 0xD9

xorwf    q, F          q = q ^ W = 0xF1 ^ 0xD9 = 0x28

bcf      STATUS, C     Clear carry bit → C = 0

rrf      q, W          W = q rotated right through carry
                           (q, C) = 0010 1000 0 before rotate
                           After rotate, W = 0001 0100 = 0x14, C = 0

addlw    0xFF          W = W + 0xFF = 0x14 + 0xFF = 0x13
```

2. (40 points, 20 per part) ***Writing PIC assembly code***

For each of the following 80386 instructions, write a sequence of PIC 16F684 instructions that performs an equivalent operation. The operation is described in italics. **CHOOSE ANY TWO OF THE THREE PARTS** and fill in the space provided with appropriate code. **You may complete all three parts for up to 10 points of extra credit, but must clearly indicate which part is the extra one—I will assume it is part (c) if you mark none of them.**

Assume that variables with the same names are defined for all 8-bit 80386 registers (for example, "AL" and "BL"). If an operation uses a 16-bit register (e.g., AX), you can address each byte within that register (e.g. AH and AL). Also assume "TEMP" has been defined for cases where you may need an extra variable.

Finally, note that shift or rotate operations should not be done by simply writing copies of the PIC rotate instructions. Use the shift amount provided as a literal value that will help determine the number of times you shift or rotate.

Note that other solutions may be acceptable.

a. RCR AX, 6 *(Rotate 16-bit value AX right, through the carry, by 6 bits)*

Solution:

```
        movlw   6           ; Initialize loop counter
Loop:   rrf     AH, F       ; Rotate upper byte
        rrf     AL, F       ; Rotate lower byte
        addlw   -1          ; Decrement loop counter
        btfss   STATUS, Z   ; Check if decrement result is 0—if so, skip goto
                            ;    that returns to start of loop
        goto    Loop        ; Return to start of loop
```

b. SETC AL *(AL = 0xFF if carry bit is 1; AL = 0x00 otherwise)*

Solution:

```
        clrw                ; Clear working register
        btfsc   STATUS, C   ; Check carry—if 0, skip instruction to set W = 0xFF
                            ;    Note that clrw does not change carry, so C holds
                            ;    same value it did at start of sequence
        movlw   0xFF        ; Set W = 0xFF if C == 1
        movwf   AL          ; W holds appropriate instruction result—move to AL
```

c. NEG AX *(Perform two's complement negation of 16-bit value AX)*

Solution: *Note that, to perform a negation, we can flip all the bits of AX and add 1. Since we have to manipulate each byte separately, we have to account for the case in which a 1 gets carried from AL to AH after the add—if AL == 0 after adding 1, then we need to increment AH as well.*

```
        comf    AL, F       ; Flip all bits of AL
        comf    AH, F       ; Flip all bits of AH
        incf    AL, F       ; Increment AL
        btfsc   STATUS, Z   ; Check to see if AL == 0—if so, must increment AH
        incf    AH, F       ; Increment AH
```