

ECE 3724 Test #1 – Summer 2006 – Reese -- there 5 pages (3 pages front/back)

Net ID: _____ (no names please)

You may NOT use a calculator. You may use only the provided reference materials. For problems in part II, assume initial memory contents as shown below at the start of EACH instruction. Explain what register/memory location is modified, and give the FINAL HEX value of the modified register/memory location, and the final status of the C, Z flags. Recall that the 'd' bit in a machine word indicating a destination is '0' if the destination is W, is '1' if the destination is the file register. For the 'a' bit, use the assumptions we have used in class ('a' bit is '0' if address in ACCESS RAM, 'a' bit is '1' if not in ACCESS RAM).

Part I: (20 pts)

a. Give the machine code in HEX for the instruction *bcf 0x17A, 3*

b. Circle any statement below that is true about the PIC18 Program memory. More than one statement can be circled.

1. Is NON-VOLATILE
2. Is VOLATILE.
3. Address supports a maximum of 4K bytes.
4. Address supports a maximum of 2M bytes.
5. Contains instructions.
6. Contains the Special Function Registers (SFRs).

c. The machine code 0x1BAF represents instruction? (use 'w' or 'f' for the destination, and 'ACCESS' or BANKED to represent the value of the *a* bit). You need to give the complete instruction including all of the operands.

d. How many instruction cycles does it take to execute the following instructions? How many clock cycles? With a 20 MHz clock, how long does it take to execute the following instructions? (give the answer in **nanoseconds**). For reference, 1 MHz has a period = 1 us = 1000 ns.

```
incf 0x3A0,f
goto 0x0130
```

e. Assume that you could add another instruction to the PIC18 that had the format:

```
addff srcA, srcB, dest ; dest ← (srcA) + (srcB)
```

where dst, srcA, srcB are all locations in data memory. What is the minimum number of instruction WORDS (1 word = 2 bytes = 16 bits) would it take to encode this instruction? You MUST defend your answer or you will get no credit.

Location	Contents
0x061	0x02
0x062	0x30
0x063	0x91
0x064	0x01

Assume the W register has the value 0xC9 in it, and that initial values of C, Z are both '0'.

Part II. (35 pts) Assume the above memory contents, W register value, initial C,Z values at the START of each instruction.

a. bsf 0x063,3

Circle one: W dest. Reg. file dest.

New value (hex) ____ C_flag : ____, Z flag: __

b. subwf 0x061, f

Circle one: W dest. Reg. file dest.

New value (hex) ____ C_flag : ____, Z flag: __

c. andwf 0x062, f

Circle one: W dest. Reg. file dest.

New value (hex) ____ C_flag : ____, Z flag: __

d. rrcf 0x064,w

Circle one: W dest. Reg. file dest.

New value (hex) ____ C_flag : ____, Z flag: __

e. movlw 0x63

Circle one: W dest. Reg. file dest.

New value (hex) ____ C_flag : ____, Z flag: __

(45 pts) PART III. Convert the following C code fragments to PIC18 assembly.

UNLESS otherwise stated in a particular problem, assume all variables are in locations 0x000 to 0x07F.

If you use a temporary memory location, use temp and assume it is in bank 0. When writing code, you **must use** symbolic names for variable names, register names, and bit names for (i.e, use: bsf STATUS, C instead of bsf 0xFD8, 0x0). You do not have to show the CBLOCK declaration for variables.

Hint: A common mistake in these problems is to write code that modifies variables to the right of the '=' sign (i.e, for 'a = b - c;' the code you write somehow modifies *b*, or *c*, as well as *a*). This is incorrect; make sure that your code only modifies variables to the left of the '=' sign.

Also, recall that 'k++' is the same as 'k=k+1;', 'j- -' is the same as 'j = j - 1', that "i = j" is true if *i* is equal to *j*, that "i != j" is true if *i* is not equal to *j*, "<<" is a left shift, ">>" is a right shift, '|' is bitwise logical OR, '&' is a bitwise logic AND, '^' is a bitwise logical XOR.

unsigned char i,j,k,p,q,r,s,t;

a. (7 pts)

```
j = p + q + 2;
```

b. (9 pts)

```
if ( (q == 5) && (j != 0) {  
    //if-body – just write a placeholder here  
} else {  
    //else-body – just write a placeholder here  
}
```

c. (6 pts) Write the following in assembly language
 $i = (k \mid 0xF0) \ll 1;$

d. (10 pts)

```
while ( (p > q) || (r == 0) ) {  
  //loop-body – just write a placeholder here  
}
```

- e. (6 pts) Assume that r is in bank 1, s is in bank 2, and t is in bank 3. Implement the following code in assembly language:

```
t = r - s ;
```

- f. (7 pts) Write a PIC18 instruction sequence that does the following.

```
do {  
    //loop body, place holder  
} while (p >= q)
```