

Circle one: JONES section / REESE section

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

Student 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 *xorwf 0x3AF, f*

b. Circle the statement that is a true statement about a 4K x 16 memory

1. Has 16 address lines and 4K locations
2. Has 12 address lines and 4 bits per location
3. Has 4 K address lines and 16 bits per location
4. Has 12 address lines and 16 bits per location
5. Has 4 K locations and 12 bits per location

c. The machine code 0x313A 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 5 MHz clock, how long does it take to execute the following instructions? (give the answer in **microseconds, do NOT round to the nearest microsecond!**). For reference, 1 MHz has a period = 1 us = 1000 ns.

```
movff 0x1F0, 0x2A0
addwf 0x3A0, f
decf 0x020, w
```

e. Circle the statement that is a true statement about a *Program Counter* inside of a computer:

1. Counts the number of instructions in a program
2. Is connected to the data lines of Program Memory
3. Is connected to the address lines of Program Memory
4. Implements the counting function for a program
5. Is connected to the inputs of the ALU

Location	Contents
0x05A	0x02
0x05B	0xF0
0x05C	0x90
0x05D	0x00

Assume the W register has the value 0xA4 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. `addlw 0x5C`

Circle one: W dest. Reg. file dest.
 New value (hex) ____ C_flag : ____, Z flag: __

b. `subwf 0x05A, w`

Circle one: W dest. Reg. file dest.
 New value (hex) ____ C_flag : ____, Z flag: __

c. `andwf 0x05B, f`

Circle one: W dest. Reg. file dest.
 New value (hex) ____ C_flag : ____, Z flag: __

d. `btg 0x05B,4`

Circle one: W dest. Reg. file dest.
 New value (hex) ____ C_flag : ____, Z flag: __

e. `rlcf 0x05C,w`

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)

```
k = i - (k << 1);
```

b. (9 pts)

```
if ( (i == 0) || (j != k) {
    //if-body – just write a placeholder here
} else {
    //else-body – just write a placeholder here
}
```

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

d. (10 pts)

```
do {  
  //loop-body – just write a placeholder here  
}  
while ( ( i <= k ) || j )
```

- 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 >> 1) + s;
```

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

```
while (k > (j+5)) {  
    //loop body, place holder  
}
```