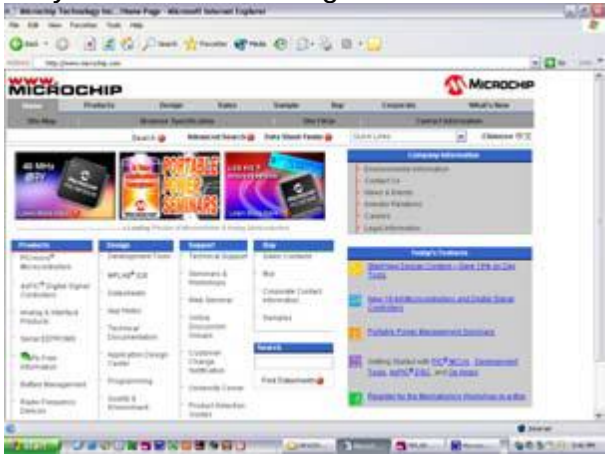
TEJ4M0 *TUTORIAL #3: PIC SCAVENGER HUNT* Name: _____

Objective:

To educate oneself on how to find answers about the *P16F684* and *Assembler*

Procedure

• Go to www.microchip.com and download the latest full version of ***MPLAB IDE*** onto your PC at home. Also load *PICC-Lite* when it prompts you after loading *MPLAB.* This is the *'C'* compiler you can use if writing in *'C'*.



• Use MPLAB IDE, and "The PIC Package" to answer the following questions:

1. How many available GPRs are there in Bank 0 of the *16F684*?
2. What is the first available address (in hex) in the GPR? The last?
3. Which programmers can you use with the *P16F684*? (hint: see "Configure" menu in MPLAB IDE)
4. How large is the instruction set in MPASM (Microchip PIC Assembler)?
5. For the instruction ***movlw k*** , what does k represent?
6. Where must all data first be written to before going to any other register?
7. For the instruction ***addwf f,d*** , what does ***f*** represent?
8. For the instruction ***addwf f,d*** , what does ***d*** represent?
9. Using *Notepad*, open the header file *p16f684.inc* (run a search). What values are assigned (EQU) to ***W*** and ***F*** registers. Why?
10. What are the two address locations of the Status Register?
11. What would have to happen in order to set (logical 1) the **Z** bit of the Status Reg?
12. What would have to happen in order to set (logical 1) the **C** bit of the Status Reg?
13. Which bit in the STATUS Register is responsible for bank selecting?
14. Which bank is being selected if the Status Reg had a value of 00100110?
    Is the value of the Digit Carry bit set (logical 1) or cleared (logical 0)?
15. What is the address of the "Reset Vector"?
16. What do bits 6 and 7 represent in both PORTA and PORTC?
    Why?
17. ***asmSecond***
    Make a new project and name it *asmSecond.mcp*, saving it in a folder called *asmSecond*
    Write a program that adds the contents of 2 different *GPR* registers, subtracts this sum from the contents of another register, and finally stores the result in the *WREG* (i.e. (10 + 13) - 5 =18). Remember to utilize the template when coding and use *MPLAB SIM* to view the results of your code in the *Watc*