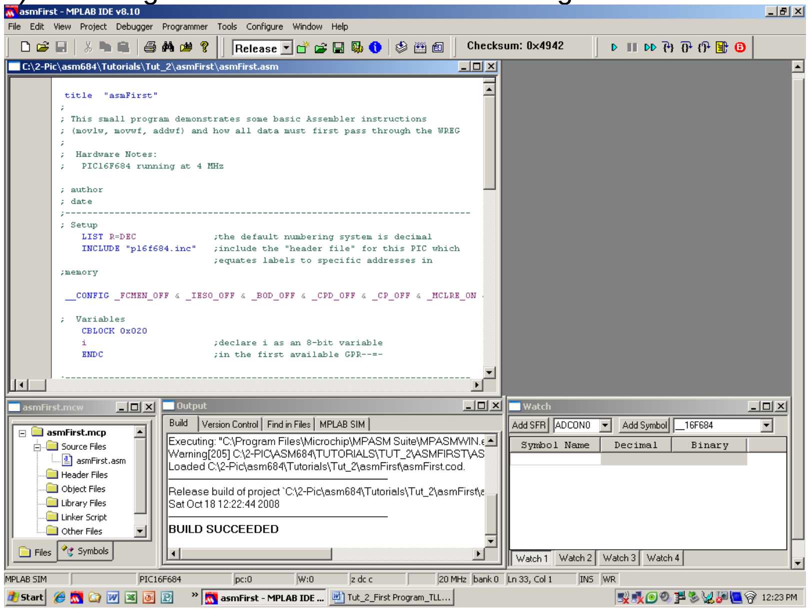
ICE 4M0 TUTORIAL #2: "My First Program!" Name:_______________

Objectives:

- Create your first program in MPLAB and observe how data moves through *WREG.*
- Simulating code with WATCH window to observe data move through *WREG.*
- Use basic assembly language instructions (*movlw, movwf, addlw, clrf, goto*).

Procedure:

1. Create a new project and workspace:
    a) Select *Project>New…* Name the project *asmFirst*.
    b) Create a new folder called *asmFirst* within the *Pic Programming* folder in your student drive.
    c) Save the project *asmFirst* into the folder *asmFirst.*
2. Create a new source code file – use *asmTemplate.asm*:
    a) Select *File>Open…* Open *asmTemplate.asm* that you created in *Tutorial #1.*
    b) Select *File>Save As…* Change the name from *asmTemplate.asm* to *asmFirst.asm.*
    c) Save it in the folder *asmFirst.*
3. Add source code to project and build:
    a) Right click on *Source Files* in the *Project* window (TIP: To open the *Project* window, click *View>Project* and the *Project* window should open up).
    b) Select *Add Files…* Add *asmFirst.asm* to the project.
    c) Build project (Ctrl + F10); ensure you get "BUILD SUCCEEDED!".
4. Open *MPLAB SIM* and *Watch* window:
    a) Select *Debugger>Select Tool>MPLAB SIM*. You should notice a new set of tools on the taskbar have been added. These are the controls for the simulator.
    b) Select *View>Watch* to open up the *Watch* window.
    c) Rearrange the four windows to match Fig 2.1.



```
 title  "asmFirst"
;
; This small program demonstrates some basic Assembler instructions
; (movlw, movwf, addwf) and how all data must first pass through the WREG
;
;  Hardware Notes:
;   PIC16F684 running at 4 MHz

; author
; date
;---------------------------------------------------------------------------
; Setup
   LIST R=DEC                ;the default numbering system is decimal
   INCLUDE "p16f684.inc"     ;include the "header file" for this PIC which
                             ;equates labels to specific addresses in
;memory

 __CONFIG _FCMEN_OFF & _IESO_OFF & _BOD_OFF & _CPD_OFF & _CP_OFF & _MCLRE_ON

;  Variables
   CBLOCK 0x020
   i                         ;declare i as an 8-bit variable
   ENDC                      ;in the first available GPR--=-

;---------------------------------------------------------------------------
```

```
Executing: "C:\Program Files\Microchip\MPASM Suite\MPASMWIN.e
Warning[205] C:\2-PIC\ASM684\TUTORIALS\TUT_2\ASMFIRST\AS
Loaded C:\2-Pic\asm684\Tutorials\Tut_2\asmFirst\asmFirst.cod.
----------------------------------------
Release build of project `C:\2-Pic\asm684\Tutorials\Tut_2\asmFirst\a
Sat Oct 18 12:22:44 2008
----------------------------------------
BUILD SUCCEEDED
```

Fig 2.1: Workspace arrangement

5. Write the program:
    a) Modify the 'title' and 'setup' sections.
    b) Add the code and programmer's comments in the mainline.
    c) Remember that only one minor syntax error will result in a "BUILD FAILED".

```
title "asmFirst"
;
; This small program demonstrates some basic Assembler
instructions
; (movlw, movwf, addwf) and how all data must first pass
through the WREG
;
; Hardware Notes:
; PIC16F684 running at 4 MHz
; author
; date
;-------------------------------------------------------------
--------------
; Setup
LIST R=DEC ;the default numbering system is decimal
INCLUDE "p16f684.inc" ;include the "header file" for this PIC
which
;equates labels to specific addresses in
                      ;memory
__CONFIG _FCMEN_OFF & _IESO_OFF & _BOD_OFF & _CPD_OFF &
_CP_OFF & _MCLRE_ON & _PWRTE_ON & _WDT_OFF & _INTOSCIO ;put
this all on one line in Editor window
; Variables
CBLOCK 0x020
i ;equate i to the first GPR
ENDC
;-------------------------------------------------------------
------------
PAGE
; Mainline of "asmFirst_one"
org 0 ;"origin" directive which indicates to start
                           ; program on first address of
                           program
                           ; memory(0x00)
movlw 6 ;initialize "i" register
movwf i
movlw 15 ;initialize "w" register
addwf i,f ;add contents of w to
;contents of i and
;place the result back in file register i
goto $ ;loop forever
     end
```

6. 'Rebuild' your program:
    a. *Ctrl + F10*
    b. Troubleshoot if necessary.
7. Setup *Watch* window (Fig 2.2):
    a. Right-click on the column entitled *Value* and select both *Binary* and *Decimal*.
    b. Deselect *Value* and *Address.*

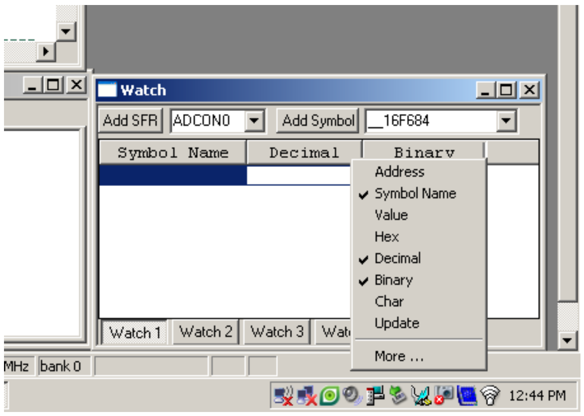


Fig 2.2: Value dropdown menu selections

c. Select the ***WREG*** from the pull down menu on the left.
d. Click the *Add SFR* button; ***WREG*** should appear in the window.

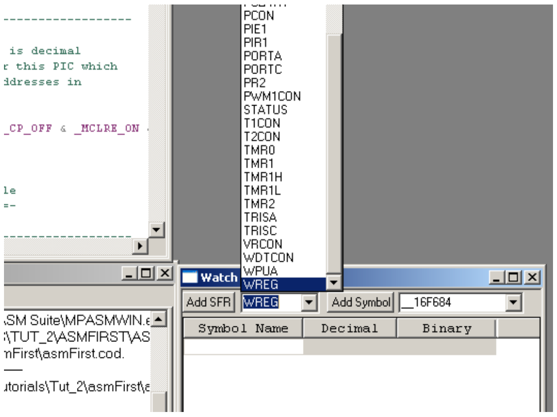
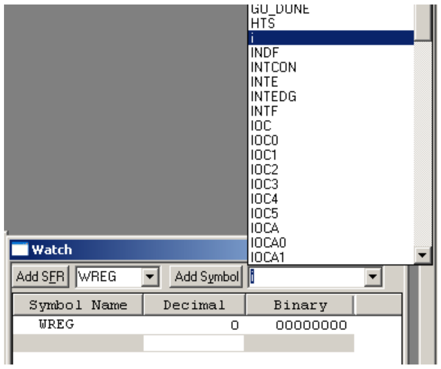


Fig 2.3: Adding WREG to the Watch window

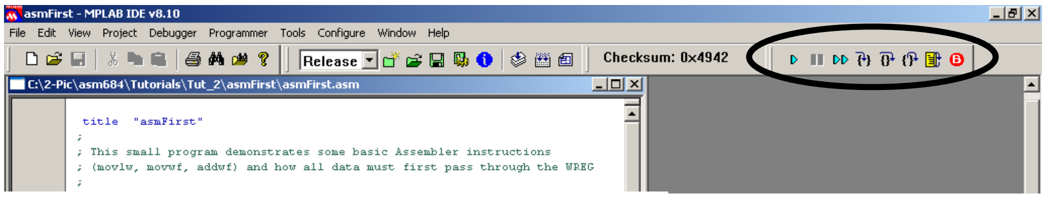
e. Select the ***i*** variable from the pull-down menu on the right.
f. Click the *Add Symbol* button; variable ***i*** should appear (Fig 2.4).



Symbol Name	Decimal	Binary
WREG	0	00000000

Fig 2.4: Adding the variable "i" to the Watch window

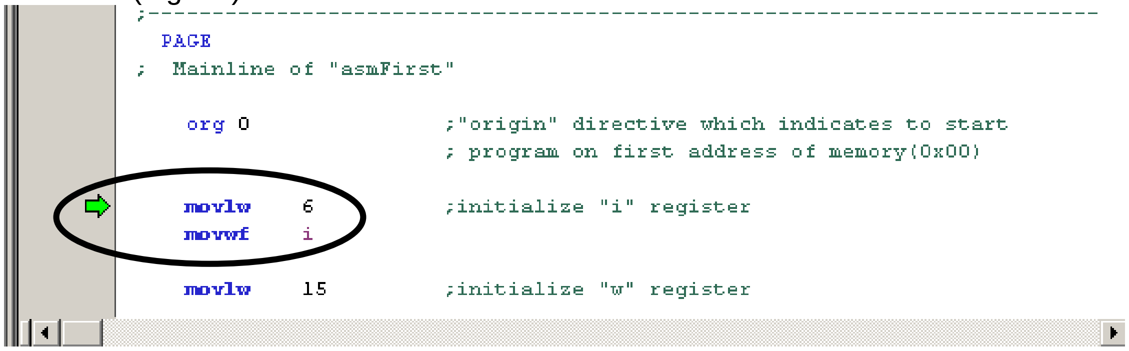
8. Start *MPLAB SIM:*
   a) Place your mouse cursor over each of the symbols on the simulator toolbar until you come across *Reset*. Click it (Fig 2.5).


asmFirst - MPLAB IDE v8.10
File Edit View Project Debugger Programmer Tools Configure Window Help
Release
Checksum: 0x4942
C:\2-Pic\asm684\Tutorials\Tut_2\asmFirst\asmFirst.asm

```
title  "asmFirst"
;
; This small program demonstrates some basic Assembler instructions
; (movlw, movwf, addwf) and how all data must first pass through the WREG
;
```

Fig 2.5: Simulator toolbar

   b) You will notice a green arrow (which represents the PC (Program Counter)) pop up on the screen, pointing at the first line of code after "org 0". This is where the simulator starts (Fig 2.6).



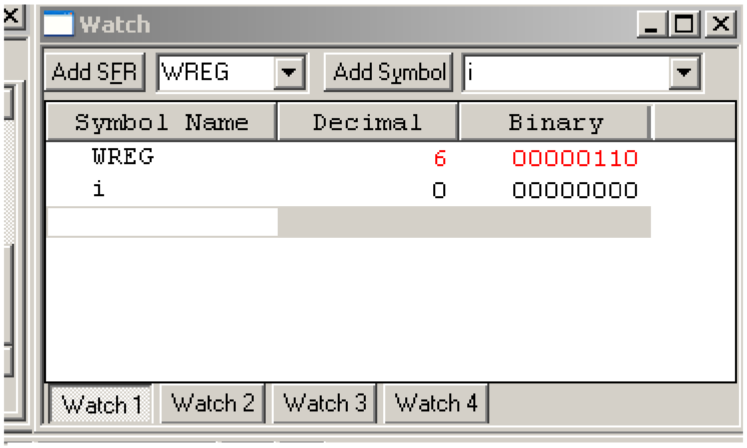
```
;-------------------------------------------------------------------------
  PAGE
;  Mainline of "asmFirst"

    org 0                    ;"origin" directive which indicates to start
                             ; program on first address of memory(0x00)

    movlw     6              ;initialize "i" register
    movwf     i

    movlw     15             ;initialize "w" register
```

Fig 2.6: Program Counter

9. Execute *MPLAB SIM*:
   a. Each instruction in *Assembler* has two distinct parts: The *mnemonic* and the *operand*. (refer to your "Instruction Set" table which lists all 35 instructions). For example, the instruction ***movlw k***, ***movlw*** is called the mnemonic and ***k*** is the operand. The mnemonic is simply an abbreviated form of the instruction. In this case, it's saying "move some value into the working register". The operand k, simply represents the value. Therefore, ***movlw 15*** would move a value of 15 into the working register.

b. Click on the *Step Into* button on the simulator taskbar.

c. Observe the change to ***WREG*** in the *Watch* window: ***6*** has been loaded into the working register. Hence, the instruction ***movlw*** (i.e. move a literal value into the working register) is visible in both binary and decimal. Notice also that it turned everything red, indicating a change has been made (Fig 2.7).

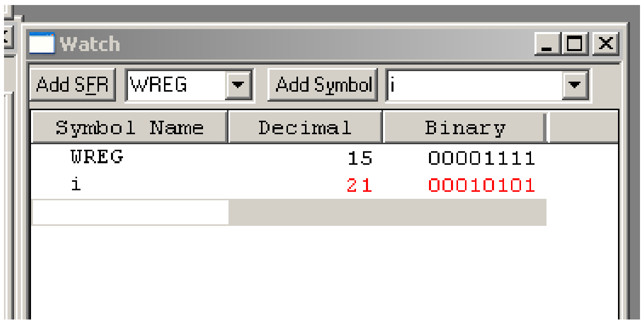


Symbol Name	Decimal	Binary
WREG	6	00000110
i	0	00000000

Fig 2.7: Moving literal values into the WREG

d. Click one more time and you will see that ***6*** has been moved from the ***WREG*** to ***i***. Hence, the instruction ***movwf*** (i.e. move the contents of w to a file register). **Remember: All data must pass though the working register <u>FIRST</u> before going to any other register.**

e. Step through until you reach the ***addwf i, f*** line of code. As you can probably guess, the ***addwf*** mnemonic is telling you that it will add the contents of the ***WREG*** with the contents of variable ***i***. The additional piece of code that is included is the letter ***f***. This is referred to as the *destination* bit, which can be either ***f*** or ***w***. This means you can store the result of the addition in either the variable you're using, or the ***WREG*** (Fig 2.8).

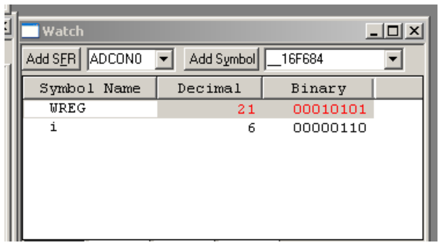


Symbol Name	Decimal	Binary
WREG	15	00001111
i	21	00010101

Fig 2.8: Adding the contents of WREG and file 'i'

f. Change the destination bit to ***w***, rebuild, reset the simulator and note the change (Fig 2.9).

g. The simulator is an essential tool for building successful programs and interfaces, particularly when building the robot!!!



Symbol Name	Decimal	Binary
WREG	21	00010101
i	6	00000110

Fig 2.9: Changing the location of the storage of the answer received from adding the contents of WREG and file 'i'.