Announcements:

- Midterm on March 27th
  - The exam will cover through Lecture notes, Part 6; Lab 3, Homework 6, and readings in Bates assigned in homework 1 - 6 (Chap 1, the appendices, Chap 12 and Section 6.8)
  - A past exam is posted for review. This exam can give a general sense of what the EE367 first exam may be like (a mix of multiple choice, analysis and long answer questions).

Reading:

- Lecture notes
  - Part 6, Section 4 to end
  - Chapter 6 (sections 6.6 and 6.8 have already been assigned)

Assignment:

1 Costs of including an instruction, consider adding an instruction

When designing a microprocessor, adding an instruction to the instruction set will have two types of cost for the microprocessor. Consider adding the new instruction

\[
\text{MOVLF } k, f, a \quad ;; \quad k \text{ is 8-bit data, } f \text{ is 8-bit address, } a \text{ is the access mode bit}
\]

to the PIC18. This instruction will move data \(k\) into file register \(f\), as does the two instructions:

\[
\begin{align*}
\text{MOV\textsubscript{LW} } &\quad k \quad ;; \quad k \rightarrow [\textbf{W}] \\
\text{MOVE\textsubscript{WF} } &\quad f, a \quad ;; \quad [\textbf{W}] \rightarrow [f]
\end{align*}
\]

(Note: \text{MOVLF} k,f,a and the two instructions above are not exactly equivalent, since the two instructions also have the side effect of over-writing the data previously stored in the W register.)
A. (Essay question) Describe in one to two sentences each of the two types of cost associated with building the instruction into the microprocessor. Mention the nature of the cost and why it is important.

(1)

(2)

2  Laying out a new instruction for the PIC18

Every instruction needs an Op Code and a layout for the bits of the machine code. Assuming MOVLF is a two-word instruction and considering the existing Op Codes of the PIC18, assign an op code and make a layout for the bits of the machine code for instruction that doesn’t interfere with any instruction already in the microprocessor.

Note: The instructions are listed in order of increasing op code in Part 5b of the lecture notes.

A. What are the two gaps in the already assigned op codes? Note that operands such as d, a, f, k, n can take any pattern of zeros and ones.

B. How many bits of operand will the new instruction need to have?

C. Considering the fact that in the PIC18, for all two-word instructions, the first nibble of the second word is 1111, how many bits can the op code have? That is

\[ 32 \text{ bits} - (4 \text{ bits for the 1111 first-nibble of the second word (this is called a preamble)}) - \text{Total number of operand bits} \]

D. Select an op code within one of the gaps in the already assigned op codes, and layout the bits of the machine code for the MOVLF instruction. Show your machine code in the same form seen in table 1, below.

E. Considering that the action of MOVLF can be implemented as two instructions, if you were a Microchip designer, would you have included MOVLF in the PIC18 instruction set? Why or why not?

3  Costs of using an instruction in a program

When a programmer uses an instruction in a program, the instruction has two types of cost that are distinct from the costs of building the instruction into the microprocessor. For the costs to the programmer, if the programmer uses the instruction ten times, these costs are about 10x higher than using the instruction once.
A. (Essay question) Describe in one to two sentences each of the costs encountered by the programmer when using the instruction. Mention the nature of the cost and why it is important.

(1)

(2)

B. Details of several instructions are shown in table 1 and two code segments are shown below. The 'Cycles' is the number of instruction clock cycles required to execute the instruction.

(1) Determine the machine code for each code segment below. (An example of listing machine code is seen in the lecture notes, part 5, with example program CountFrontTen-TakeTwo.)

(2) How many words of instruction memory are used by the first code segment?

(3) How many words of instruction memory are used by the second code segment?

(4) If the PIC18 has an instruction clock rate of 1 mega-hertz, how many microseconds are required to complete one iteration of the first code segment? (That is, one pass through all instructions, including the branch back to Repeat).

(5) How many microseconds are required to complete one iteration of the second code segment?

<table>
<thead>
<tr>
<th>Mnemonic</th>
<th>Operands</th>
<th>Operation</th>
<th>Status Affected</th>
<th>Machine code</th>
<th>Cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADDLW</td>
<td>k</td>
<td>k + [W] -&gt; [W]</td>
<td>N, OV, C, DC, Z</td>
<td>0000 1111 kkkk kkkk</td>
<td>1</td>
</tr>
<tr>
<td>ADDWF</td>
<td>f, d, a</td>
<td>[W] + [f] -&gt; [dest]</td>
<td>N, OV, C, DC, Z</td>
<td>0010 0da ffff ffff</td>
<td>1</td>
</tr>
<tr>
<td>MOVF</td>
<td>f, d, a</td>
<td>[f] -&gt; [dest]</td>
<td>Z, N</td>
<td>0101 00da ffff ffff</td>
<td>1</td>
</tr>
<tr>
<td>MOVLW</td>
<td>k</td>
<td>k -&gt; [W]</td>
<td>None</td>
<td>0000 1110 kkkk kkkk</td>
<td>1</td>
</tr>
<tr>
<td>MOVWF</td>
<td>f,a</td>
<td>[W] -&gt; [f]</td>
<td>None</td>
<td>0110 111a ffff ffff</td>
<td>1</td>
</tr>
<tr>
<td>GOTO</td>
<td>n20</td>
<td>2*n20 -&gt; [PC]</td>
<td>None</td>
<td>1110 1111 nnnn nnnn</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1111 nnnn nnnn nnnn</td>
<td></td>
</tr>
<tr>
<td>BRA</td>
<td>n11</td>
<td>[PC]+2+2*n11 -&gt; [PC]</td>
<td>None</td>
<td>1101 0nnn nnnn nnnn</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 1: Details of several instructions. The number of instruction clock cycles to execute each instruction is shown. The complete list of instruction cycles is included in PIC18_InstructionSetSummary.pdf.

- Two code segments

```plaintext
;; Constants section
Increment EQU .10 ;; Increment by 10T
;; Variables section
```

(Revised: Feb 27, 2019) Page 3
udata_acs
MyVar01 RES 1

;; First code example, addition done in the W register
code 0x00000
Repeat:
  MOVF MyVar01,W,A
  ADDLW Increment
  MOVWF MyVar01,A
  GOTO Repeat

;; Second code segment, addition done in Reg f, BRA used
code 0x00000
Repeat
  MOVLW Increment
  ADDWF MyVar01,F,A
  BRA Repeat
4 **Complete the state table below**

Complete the state table below by carrying out these steps

A. For the registers, show each value where it changes. Show the PC at the end of the instruction (ready for the next instruction).

B. Show 0, 1 or '-' for flags on each line, **pay careful attention to the flags**. Flags touched by each instruction can be looked up in any one of: Lecture notes part 6, the exam reference booklet (on the web site), or PIC18_InstructionSetSummary_2Pages.pdf (also on the web site).

C. Show final values for all registers and the flags

- Symbols are defined by:

  ;; Constants section
  Val01: EQU 0x25
  Val02: EQU 0xA7

  ;; Variables section
  udata_acs 0x00
  Var01: RES 1
  Var02: RES 1

- State table:

<table>
<thead>
<tr>
<th>State Table:</th>
<th>W</th>
<th>PC</th>
<th>SP</th>
<th>0x00</th>
<th>0x01</th>
<th>0x02</th>
<th>N</th>
<th>Ov</th>
<th>Z</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Initial Values:</strong></td>
<td>$12</td>
<td>$0020</td>
<td>$00</td>
<td>$BB</td>
<td>$CC</td>
<td>$DD</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>MOVLW Val02</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOVWF Var02</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOVLW Val01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOVWF Var01</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOVF Var02,W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Final Values:**

Symbol values: Var01: 0x000, Var02: 0x001
5 Complete the state table below

Complete the state table below by carrying out these steps

A. For the registers, show each value where it changes. Show the PC at the end of the instruction (ready for the next instruction).

B. Show 0, 1 or '-' for flags on each line.

- Pay special attention to the flags touched by the MOVF instruction.

C. Show final values for all registers and the flags

- Symbols are defined by:

  ;; Constants section
  Val01: EQU 0x25
  Val02: EQU 0xA7

  ;; Variables section
  udata_acs 0x00
  Var01: RES 1
  Var02: RES 1

- State table:

<table>
<thead>
<tr>
<th>State Table:</th>
<th>W</th>
<th>PC</th>
<th>SP</th>
<th>0x00</th>
<th>0x01</th>
<th>0x02</th>
<th>N</th>
<th>Ov</th>
<th>Z</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Values:</td>
<td>$85</td>
<td>$0020</td>
<td>$00</td>
<td>$BB</td>
<td>$CC</td>
<td>$DD</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>MOVF Var01,W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDWF Var02,W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ADDWF Var02,F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MOVLW Val02</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SUBWF Var02,F</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final Values:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

  Symbol values: Var01: 0x000, Var02: 0x001

- Note: Var01 is address 0x00, Var02 is address 0x01
6 8-bit Direct Mode Addressing

A. How many bytes of memory can be accessed in 8-bit direct mode addressing?

B. In an instruction such as ADDWF, what bit in the instruction word controls whether 8-bit access mode addressing is used? (Details of ADDWF are seen in table 1.)

C. With 8-bit direct mode addressing, the accessible bytes of memory fall into two groups, what are the two groups?

   (1) Which group includes the registers PORTA and TRISA?

   (2) Which group includes address 0x01?

D. When we use 8-bit direct mode addressing in an instruction such as ADDWF, where does the 8-bit file-register address come from?

E. Figure 2 of Lecture Notes Part 5 is a block diagram of the PIC18. In the block diagram, what is the name given to the bus carrying the 8-bit address of 8-bit direct mode addressing?

F. How many bytes of RAM are available to the programmer when using 8-bit direct mode addressing?

7 Assembler Directives

Assembler directives are used to setup constants, reserve memory variables, set the memory address for program material, and other tasks. Considering example CountFromTen-TakeTwo in lecture notes part 5, the constants block assigns symbolic names to constant data, while the variables block allocates memory and assigns memory addresses to symbolic names.

Write the top portion of an assembly language program using assembler directives to setup the following conditions:

A. Define two constants, InitialVal01 and InitialVal02, and set them to $12 and $34, respectively.

B. Create memory variables Var03 and Var04. Make Var03 3 bytes and make Var04 2 bytes.

C. Start the program at line $00020

8 Comments

A. Name five items that must be in the banner comment.

B. (Essay question) What is a side effect?