**Chapter 4**

**Questions and Assignments**

**4.1 What is a flowchart?**

It is a graphical representation of processes (tasks) to be performed and the sequence to be followed in solving computational problem.

**4.2 What is an assembler?**

An assembler is a program for converting instructions written in low-level symbolic code into machine code.

**4.3 What is the function of a header file in an assembly language program?**

According to what I could find, an .h or .inc file is like a library and when we want to use it we have to put i.e.: **#include <p18f452.inc>**. The header needs to be added to the project to be used unless it is included by default in the assembler.

**4.4 Explain the term pseudocode in an assembler and describe its function.**

Pseudocode is an informal high-level description of the operating principle of a computer program or other algorithm. In an assembler, pseudocode is used to represent the algorithm that the program is going to follow in a more readable way.

**4.5 What does a label (excluding labels used for equates) represent in a program?**

The label is assigned the current value of the active location counter and serves as an instruction operand.

**4.6 What is the function of the ORG statement in an assembly language program and can the program have more than one ORG statement?**

The ORG operand will set the location counter. In other words, it is an indication on where to put the next piece of code/data.

We can use more than one ORG statement in an assembly program.

**4.7 List the parts of an assembly language statement and indicate the parts that are optional.**

– Label (Optional)

– Opcode (operation code)

– Operand (data, register, or memory address to be operated on)

– Comment (Optional)

**4.8 List the number of files generated by an assembler after assembling a program.**

It depends on the assembler. If the program was assembled in PIC18 Simulator, it will create 2 extra files beside the .asm. The files created after assembled are: “.HEX”, “.lst”.

If the program was assembled in MPLAB it will create 7 extra files beside the .asm file. The files created after assembled are: “.map”, “.cof”, “.HEX”, “.mcs”, “.O”, “.lst”, “.err”.

**4.9 What is the purpose of the List file of an assembled program and what does it include?**

The list file is generated primarily for documentation. It includes seven columns:

– Memory addresses where binary code is stored

– Hex code

– Line numbers

– Contents of source file:

• Labels

• Opcode

• Operands

• Comments

**4.10 Specify the memory location where the program will be assembled if the address in ORG statement is as follows: ORG 0040**

The instructions after the ORG 0040 will start to be store after 40h (including 40h).

**4.11 START: MOVLW 0x67**

**ADDLW 0x33**

**SLEEP**

1. **Specify the result you expect in the W register****:**

0x9A

1. **Specify the flags that are set after the addition:**

N and OV.

1. **Specify the byte you would observe in the STATUS register:**

Bit 4 and 3.

**Assemble the program using the editor and the assembler of MPLAB starting at location 0020H. Build the project and execute the instructions using the single step and verify your answers.**

**4.12 START: MOVLW 0x42**

**SUBLW 0x33**

**SLEEP**

1. **Specify the result you expect in the W register:**

0xF1

1. **Specify the flags that are set after the subtraction:**

N and DC.

1. **Specify the byte you would observe in the Status register.**

Bit 4 and 1.

Note: *Why I didn’t have an overflow?*

**Assemble the program using the editor and the assembler of MPLAB starting at location 0020H. Build the project and execute the instructions using the single step and verify your answers.**

**4.13 The following set of instructions is expected to load two bytes (A7H and 92H ) in data registers 01 H and 02 H, add the bytes, and save the sum in register 03 H. Calculate the sum of these two bytes and identify the flags that are set.**

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | MOVLW | 0xA7 | ; |
| 2. | MOVWF | 0x01, 0 | ; |
| 3. | MOVLW | 0x92 | ; |
| 4. | MOVWF | 0x02, 0 | ; |
| 5. | ADDWF | 0x01, 1, 0 | ; |
| 6. | MOVWF | 0x03, 0 | ; |

The sum of both files (0xA7 and 0x92) will is 39 because an overflow. The flags are OV and C.

**4.14 In Q. 4.13, the add instruction sets overflow and carry flag. Explain why the overflow flag is set and interpret the result if the numbers are signed numbers.**

It has an overflow because the sum passed the value of FF. It also sets a carry flag because the process of the sum generate a carry when both value have a 1 in the same bit position.

**4.15 In Q. 4.13, explain why registers W and 03H have the byte 92H at the end of the program?**

Because in line 5 the instruction ADDWF has a 1 to store the result in the file instead of having a 0 to store the result in WREG. Line 5 should be “**ADDWF 0x01,0,0**”. In that way WREG and 0x03 will have 39h.

**4.16 In Q. 4.13, identify the location where the sum is saved.**

The sum is saved in 0x01.