Lab Sheet 3

PIC16F84A Instructions

1. Overview

The instructions set of the PIC16F84A is divided into:

1. Byte-oriented instructions that operate on bytes of data and further divided into (tested in the second week):
    a. Data transfer instruction,
    b. Mathematical and logical instructions
2. Bit-oriented instructions that operate on a single bit.
3. Literal and control instructions. Tested in the first week.

2. Testing Bit-oriented Instructions

Program 1: First of all, create a project for this lab as was shown in the first week. In order to implement an *if* statement, we use the BTFSS or BTFSC instructions. Write the following program in the source code file (after defining a variable X at 0x0C and setting it in the file register window to the value 0xBD):

```
        MOVLW   0xBD
        XORWF   X, 0
        BTFSS   STATUS, Z
        GOTO    NO
        GOTO    YES
NO      BSF     X, 0
        GOTO    FINAL
YES     BCF     X, 7
FINAL   NOP
```

Code 1 – Program 1

Simulate the program and fill in the table below.

Instruction	PCL	W	X	Z (flag)
MOVLW 0xBD				
XORWF X, 0				
BTFSS STATUS, Z				

Table 1 Results of the second bit-oriented instructions program (X = 0xBD).

Set the value of X to 0xDB, re-simulate the program and fill in the table below.

Instruction	PCL	W	X	Z (flag)
MOVLW 0xBD				
XORWF X, 0				
BTFSS STATUS, Z				

Table 2 Results of the second bit-oriented instructions program (X = 0xDB).

Exercise 1: Write a program that:

- Defines two variables X and Y at addresses 0x0C and 0x0D. Set the starting value of X to 125 in the file register window,
- Multiplies X by 2 (*Hint:* use the RLF instruction),
- If there is an overflow, set Y to 0xFF, otherwise, set Y to 0x00.

Simulate your program, fill in the table below, and verify the results.

Instruction	PCL	X	Y	C (flag)

Table 3 Results of the exercise program (X = 125).

Set the value of X to 130, re-simulate the program, fill in the table below, and verify the results.

Instruction	PCL	X	Y	C (flag)

Table 4 Results of the exercise program (X = 130).

Program 3: In order to implement a loop, we use the DECFSZ or INCFSZ instructions. Write the following program in the source code file (after defining two variables X and COUNTER at addresses 0x0C and 0x0D):

```
        MOVLW   0x03
        MOVWF   COUNTER         ; Initializing the loop counter
        CLRF    X
LOOP    INCF    X               ; Body of the loop
        DECFSZ  COUNTER         ; Decrement the loop counter
        GOTO    LOOP
        NOP
```

Code 2 – Program 2

Simulate your program, fill in the table below and verify the results.

Instruction	PCL	COUNTER	X
MOVLW 0x03			
MOVWF COUNTER			
CLRF X			

Table 5 Results of the loop implementation program.

3. Report Exercises

Exercise 2: Write the following program in the source code file (after defining a variable X at 0x0C):

1.
```
MOVLW   0x0F
MOVWF   X
BSF     X, 6
BCF     X, 0
NOP
```

Code 3 – Program 3

Simulate the program and fill in the table below.

Instruction	W	X
MOVLW 0x0F		
MOVWF X		
BSF X, 6		
BCF X, 0		

Table 6 Results of the first bit-oriented instructions program.

Exercise 3:

Write a program that does the following steps:

- Define three variables X, Y and K at addresses 0x1C, 0x1D, and 0x1E,
- Set the starting values of X and Y to 130 and 120, respectively,
- Move X into W,
- Subtract Y from W,
- Move the result into K.

Exercise 4: Write and simulate a program that continuously multiplies a variable X by two until an overflow is detected. Test your program using different values for X (32, 100, and 150).