EE 337: Microprocessors Laboratory (Spring 2024)
Indian Institute of Technology Bombay
Lab 3: 20 points Date: January 19, 2024

1. [8 points] *Multiply and Accumulate (MAC)* operation is used extensively in digital signal processing and machine learning.
Consider the inputs to be $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ and the corresponding weights/coefficients to be $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$. The result after the MAC operation will be

$$\mathbf{x} = \mathbf{a}_1\mathbf{b}_1 + \mathbf{a}_2\mathbf{b}_2 + \mathbf{a}_3\mathbf{b}_3.$$

Write an assembly language program to generate the result $\mathbf{x}$, when $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$ are three inputs present in the memory locations `70H` to `72H` and $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ are 8-bit weights/coefficients present in the memory locations `73H` to `75H`.

Since the result $\mathbf{x}$ can be 18 bits long, store the result in memory locations `50H`, `51H`, `52H`. For $\mathbf{x} = x_{17}x_{16}x_{15}\ldots x_3x_2x_1x_0$, the memory location `50H` should have $000000x_{17}x_{16}$, the memory location `51H` should have the bits $x_{15}x_{14}\ldots x_8$, and the memory location `52H` should have the bits $x_7x_6\ldots x_0$.

Use the following program as a starting point. Use the `ADD16` subroutine written in the previous question to implement the MAC operation.

```
 // -- DO NOT CHANGE ANYTHING UNTIL THE **** LINE--//
ORG 0H
LJMP MAIN
ORG 100H
MAIN:
CALL MAC
HERE: SJMP HERE
ORG 130H
// *****************

MAC:
// MAC OPERATION, CALL THE ADDITION SUBROUTINE USING "CALL ADD16"
RET

ADD16:
// ADD16 SUBROUTINE FROM PREVIOUS LAB
RET
END
```

To reduce the effort involved in adding multiple items in memory locations, you can use the command window in Keil. Refer to Lab-2 lab-sheet.

TA Checkpoint 1

Check the following cases:

- $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3 =$ `18H, 24H, 0CH`, $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3 =$ `03H, 02H, 06H`.
- $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3 =$ `DFH, E4H, 01H`, $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3 =$ `00H, FFH, F1H`.
- $\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3 =$ `FFH, FFH, FFH`, $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3 =$ `FFH, FFH, FFH`.

2. [8 points] Write assembly program to generate 5 seconds delay. Use the following program as a starting point. Add your codes in the `DELAY_5s` subroutine.

```
// -- DO NOT CHANGE ANYTHING UNTIL THE **** LINE--//
ORG 0H
LJMP MAIN
ORG 100H
MAIN:
MOV P1, #01H
CALL DELAY_5s
MOV P1, #00H
HERE: SJMP HERE
ORG 130H
// *****************

DELAY_5s:
// ADD YOUR CODE HERE
RET
END
```

Use `delay_1ms` subroutine shown below that generates 1 ms delay to make a subroutine to generate delay of 5 seconds. Do NOT modify this snippet itself.

```
delay_1ms:
    push 00h
    mov r0, #4
    h2: acall delay_250us
    djnz r0, h2
    pop 00h
    ret

delay_250us:
    push 00h
    mov r0, #244
    h1: djnz r0, h1
    pop 00h
    ret
```

- To calculate the delay amount, refer the **8051 Instruction Set** to find number of cycles each instruction runs for and how correct amount delay is achieved.
- Port 1.0 is initially set to 1. Then the 5 second delay sub-routine is called and then Port 1.0 is set to 0.
- Use Logic Analyzer to check the delay in Port 1.0. Make sure you change the default clock frequency to 24MHz (as it is on the Pt-51 board).

TA Checkpoint 2

Check the following cases:

- Observe the delay in Port 1.0 on the Logic Analyzer.
- Explain how the required delay is generated.

3. [4 points] This will be an in-lab experiment, you will be demonstrating that your PT-51 kit works by running the test code for the kit. The test code also checks the peripherals on the kit. Using this, you can also verify that all required software is correctly installed. Before coming to the lab please install the software mentioned below and go through the slide decks to get familiarize with the board.

   - Get familiar with PT-51 kit by going through the slides here: **User Manual**
   - Install FLIP (on Windows) or DFU Programmer (on Linux/Mac). This software is used to load programs into the 8051 microcontroller on the PT-51 kit. The steps are here: **FLIP/DFU Installations**

   In the in-lab session, verify the following in front of your TAs.

   - The procedure to load a hex file onto the PT-51 kit using Flip is shown in the video given here: **Running a Hex file on Pt-51**

     Follow the procedure and load the `led.hex` file to see that the LEDs on your kit toggle.
   - Download the file `pt51_test.hex` and load it on your PT-51 kit, as described here: **Testing the peripherals of Pt-51**
   - Follow the steps in the slide deck titled "PT-51 Test Program" to do the self-test of the kit. Check if all tests run successfully. If there are failed tests, inform your respective TA.
   - Please use the installation given here **JRE_Installer** if you are not able to find JRE or the installation with JRE fails.
   - Please put your board in boot mode before installing drivers and flashing code. Detach → Press boot → Press Reset → Release Reset → Release Boot → Attach