Introduction to the PIC Simulator

EE 310/EE310L - Mcirocontrller - Spring 2023

NAME______Ryan Chan_______ DATE: ___2/11/25____

Assignment # 2

https://docs.google.com/document/d/1LuMeqpsKCXaNShAmm1XbZFNdQlsesl8l/edit?usp=sharing&ouid=111229422470150013614&rtpof=true&sd=true

1. Assignment Overview

In this experiment we learn about the general architecture of a PIC18 microcontroller and how to use the PIC simulator.

2. Learning Objectives

By the end of this lab, you will

- Understand PIC architecture
- Learn about PIC simulator

3. Review

Consider reviewing the following before you start this assignment (must be logged on SSU to open the links):

- Tutorial on how the simulator works - Please make sure you review this before you start this assignment.
- Handout - Appendix F (read up to Section F.1.2)
- Review PIC18 Architecture - up to Section 2.2.1

4. Materials

You need the following to complete this assignment:

- PIC Simulator - Download OshoSoftware PIC Simulator. This is free software but you can only open it 30 times. The duration of each session is 1 hour. NOTE: The simulator works only on PC. You can either use the machines in teh lab or use Virtualization for MacOS. We will be using the simulator for only the first two weeks of the course.

5. Background Information

PIC is an abbreviation used for Peripheral Interface Controller. PIC microprocessors were initially designed to support PDP (programmed data processor) computers, for controlling the peripheral devices. It is based on RISC architecture.

PIC was invented in 1989 by the Microchip Technology Corporation. PICF18 is an 8-bit microcontroller.

We know a microcontroller is nothing but a combination of processor, memory, and peripherals in a single chip. In a similar way, PIC microcontroller consists of data RAM with some hundred bytes of ROM (Flash Memory) for storing the desired program, and some I/O ports. The figure below shows the interaction between CPU, Memory, GPIO ports, and different peripherals.

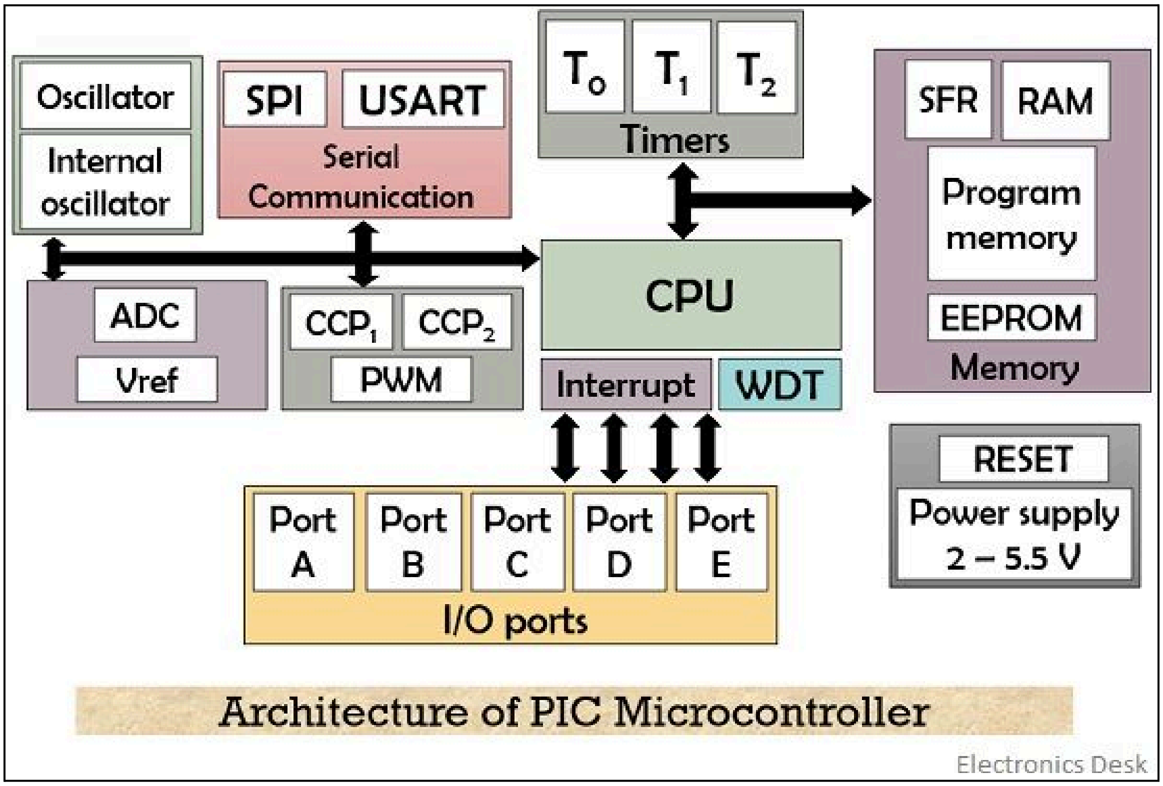


Figure from: https://electronicsdesk.com/pic-microcontroller.html

6. Experimental Procedures

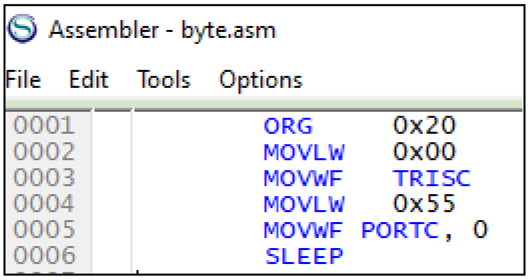
Complete the following steps.

6.1 Simulator Overview

Open the Simulator.

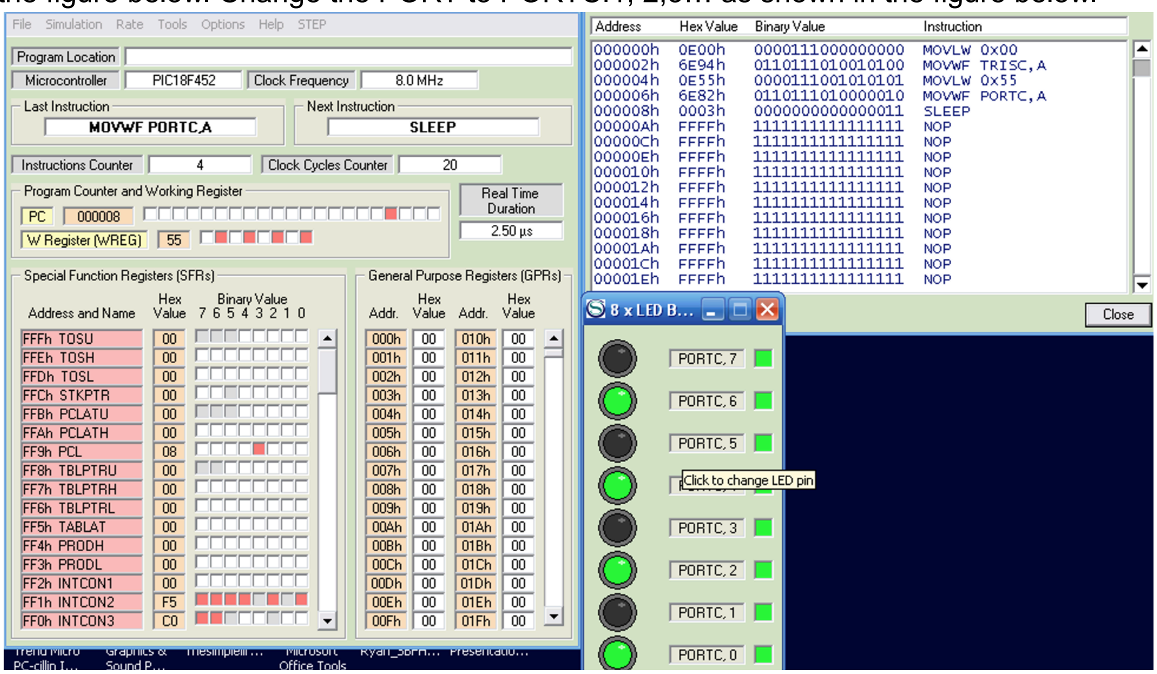
- Review the Special Function Registers (SPR) and the General Purpose Registers (GPR).
- Find the working register, also called the W register or WREG.
- Find the program counter.

Go to *Tools→ Assembler*. You can see a new window being opened. This window is used to write the assembly instructions. Type the following assembly instructions in the Assembler window. Save it as `Byte.asm`. For more information, you can review the instructions on the handout (Appendix F). Note that all instructions must start with a tab press the tab key on each new line and then start typing!)

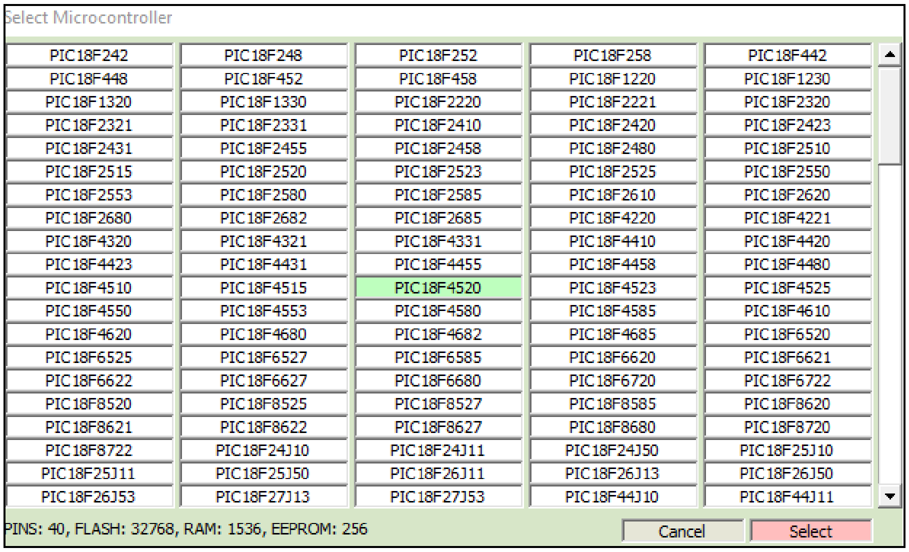


In the Assembler window, under Tools, select *Assemble and Load*.

Under the Tools tab select *8xLED Board*. You should see the LED board, as shown in the figure below. Change the PORT to PORTC.1, 2,3… as shown in the figure below.



Go to the *Simulator Window*. In the Simulator window select **PIC18F4520** as your microcontroller. Note that in the table below you can see the number of pins. It also shows the size of EEPROM (used for permanent storage), RAM, where Special Function Registers (SPR), and the General Purpose Registers (GPR) are stored, as well as the Flash Memory size, where the instructions (or program) is stored.



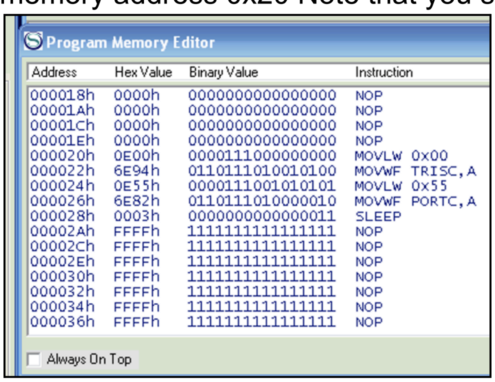
Under the *Rate tab*, select *step-by-step*.

Then, under the *Simulation tab* press *Start*. Then press F2 (or control F2). As you press F2, notice how the Next Instruction changes. Also, see how the real-time duration keeps changing. Pay attention to the Instruction Counter and how it is incrementing. Also, pay attention to the NEXT instruction and how it is changing.

Note that every time the register value changes, the bits turn RED.

Keep pressing F2 until you get to the SLEEP instruction.

Open the *Program Memory Editor* under the Tool tab. Scroll down until you get to memory address 0x20 Note that you should see your assembled code.



6.2 Adding a Break Point

Change your code, as shown in the figure below. Add a BREAKPOINT on line 6 (Edit→ BreakPoint or just press Cntl + K). Change the Rate to Normal and start the simulator.

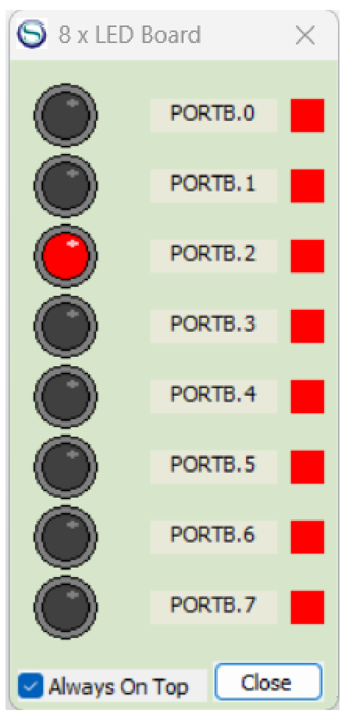
Note that the simulator stops at the BP. You can start the simulator by changing the Rate to Normal and pressing Step again.

6.3 Exercise 1: Display Binary Numbers

Write the appropriate instructions so your 8x LED panel displays binary numbers 1-5 one by one as you step through your program.

```
MOVLW 0x00
MOVWF TRISB,0
MOVLW 0x01
MOVWF PORTB,0
MOVLW 0x02
MOVWF PORTB,0
MOVLW 0x03
MOVWF PORTB,0
MOVLW 0x04
MOVWF PORTB,0
MOVLW 0x05
MOVWF PORTB,0
NOP
SLEEP
```

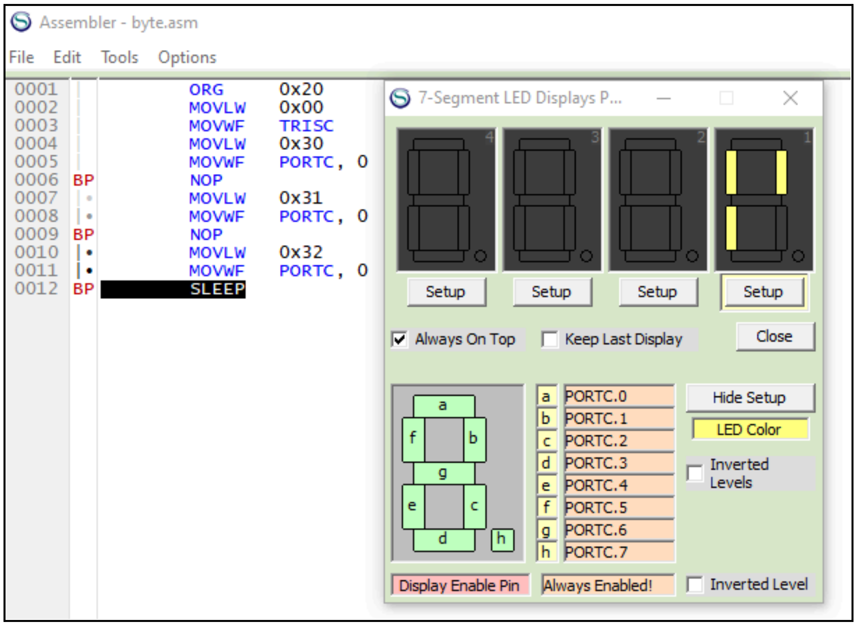
Show the snapshot of your code to display 1-5.



Show number FOUR using the LED board.

6.4 Using the Sevel Segment Display

Select the 7-Segment under the Tools tab, as shown below. Change the port to PORTC. Using the sample code below, you can display number 1 on one of the 7-segment displays.

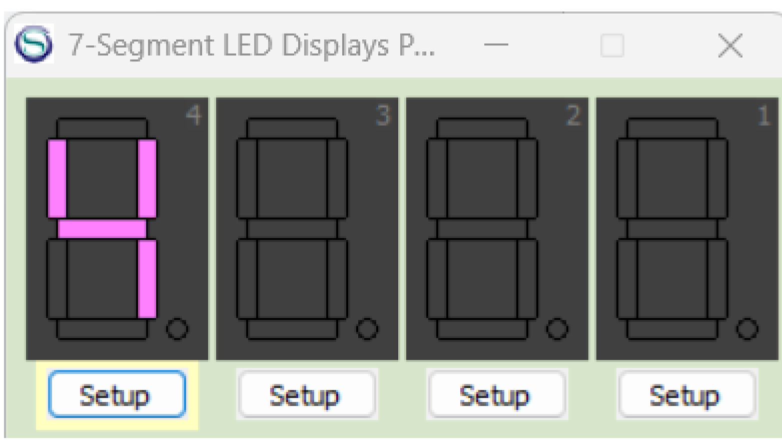


6.5 Exercise 2: Show Numbers on a 7-Segment Display

Write the appropriate instructions to display 0-9 (in decimal) on your 7-segment display, one number at a time. Your program must display 0, 1, 2, ….9 and then stop. **You must be able to demonstrate your project to receive a grade**.

```
MOVLW 0x00
MOVWF TRISB,0
MOVLW 0x3F
MOVWF PORTB,0
MOVLW 0x06
MOVWF PORTB,0
MOVLW 0x5B
MOVWF PORTB,0
MOVLW 0x4F
MOVWF PORTB,0
MOVLW 0x66
MOVWF PORTB,0
MOVLW 0x6D
MOVWF PORTB,0
MOVLW 0x7D
MOVWF PORTB,0
MOVLW 0x07
MOVWF PORTB,0
MOVLW 0x7F
MOVWF PORTB,0
MOVLW 0x6F
MOVWF PORTB,0
NOP
SLEEP
```

Show the snapshot of your code to write 0-9 on the 7-segment display.



Show number **FOUR** on the seven-segment

6.6 Answer the Following Questions

1. What does PIC stand for?

   Peripheral Interface Controller

2. What is the difference between RAM and ROM?

   RAM is volatile memory used for temporary storage, while ROM is non-volatile that stores permanent data.

3. Does the programming instructions saved in RAM or ROM?

   ROM, so that they persist when the power is turned off.

4. Is GPR part of RAM or ROM?

   RAM

5. What is the size of each GPR in bits?

   8 bits (1 byte)

6. What is the size of each FSR in bits?

   12 bits

7. What happens if you change the frequency of the clock in the MCU?

   When clock frequency increases, it will increase execution speed, reduce instruction cycle time, but increase power consumption.

8. Is the working register (WREG) considered to be an SFR or GPR? What is its address in HEX?

   SFR. Its address is 0xFE8.

9. For the selected PIC device, how many GPRs are available in kilo-bytes?

   The PIC18F4520 has 3.75 KB of RAM, where a portion can be allocated to GPRs depending on the specific application and configuration.

10. Some of the SFRs are grayed out. Why do you think that is the case?

    Grayed-out SFRs are reserved or unused in the current configuration.

11. What is the purpose of the ORG instruction?

    ORG sets the starting memory address where the program will be placed in ROM.

12. What do you have to do to store the program starting at 0046 HEX?

    Use the ORG 0x0046 directive in assembly to set the program counter (PC) to 0x0046.

13. Can you start the program at 0x43? If so, how? If not, why?

    No. Program execution must begin at 0x0000 reset vector or the predefined interrupt vector at 0x0008 (high-priority) and 0x0018 (low-priority).

14. How many clock cycles does it take to execute one executable instruction?

    4 clock cycles

15. Assuming the clock frequency is 8 MHz, how long does it take to execute `MOVLW 0x05` instruction in **nanoseconds**? Briefly explain.

    Instruction cycle time = (1 / 8 MHz) × 4 = 500 ns

    The MOVLW 0x05 instruction takes 500 nanoseconds to execute.

16. What is the binary code for SLEEP instruction?

    0000 0000 0000 0011

17. What is the length of the binary code instruction for `MOVLW 0x55` in bits?

    16 bits

18. What is the HEX equivalent of the OP-code for `MOVLW 0x55` instruction?

    0x0E55

19. What do you think will be the HEX equivalent of the binary code for `MOVLW 0x50` instruction?

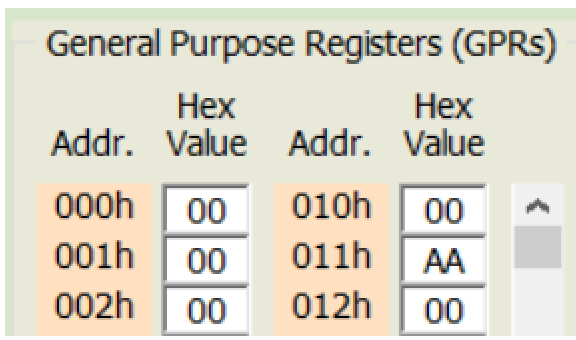
    0x0E50

20. Identify the following for the selected chip (don't forget the units):
    a. The number of pins:______40 pins________
    b. EEPROM size :______256 Bytes_______
    c. RAM size:______1.5 KB________
    d. Flash Memory size :______32 KB_______

21. What will be the command(s) to write AA into GPR number 0x11? Prove that your instructions work properly (complete the table below).

```
MOVLW 0xAA
MOVWF 0x11
NOP
SLEEP
```

Show the instructions to write AA into GPR number 0x11



Show that register 0x11 has the value of AA - take a snapshot of your GPR.

7. Survey Questions

Answer the following questions, please:

Survey question	Response
On a scale of 1-10 how did you like this exercise? (10 is the best, 1 is the worst)	10
On a scale of 1-10 how much did you learn as a result of completing this exercise? (10 = plenty; 1=very little)	10
How many hours did you spend completing this exercise?	5

8. References

[1] Complete Electronics Self-Teaching Guide with Projects | Earl Boysen, Harry Kybett. ISBN: 978-1-118-28232-8 July 2012