LAB NO. 2

7 SEGMENT DISPLAY WITH PIC

OBJECTIVES: -

- To write a code for PIC in assembly for 7 segment LED.
- To simulate the circuit on Proteus.

APPARATUS: -

- Laptop
- MPLAB X
- Proteus Circuit Simulator

CLASS TASK: -

- Write a code for PIC to display digits from 0-9 on 7-segment LED using RETLW method.

THEORY: -

SEVEN SEGMENT DISPLAY: -

A seven-segment display is a group of seven LED packed in a package in such a way that digits from 0 to 9 can be displayed in it but turning LED combination on or off. The fact that it is named as 7-sgement is number of its segments or LEDs which are also known as segments.

And 8th LED is also present some time which indicates the decimal point to display decimal numbers. More than one LEDs to display multi digited number. A 7-segment display is shown below.

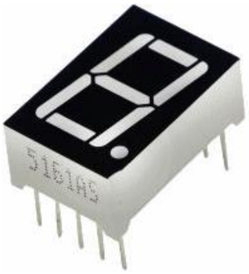


Figure 2.1: Seven Segment Display Module

The digits are displayed as follows: -

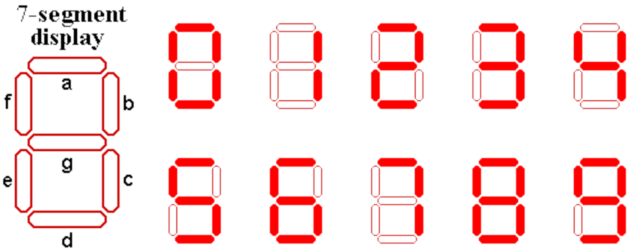


Figure 2.2: Digits from 0 to 9 displaying

Now there are two types of 7-segment depending upon their nature: -

- Common Cathode
- Common Anode

COMMON CATHODE -

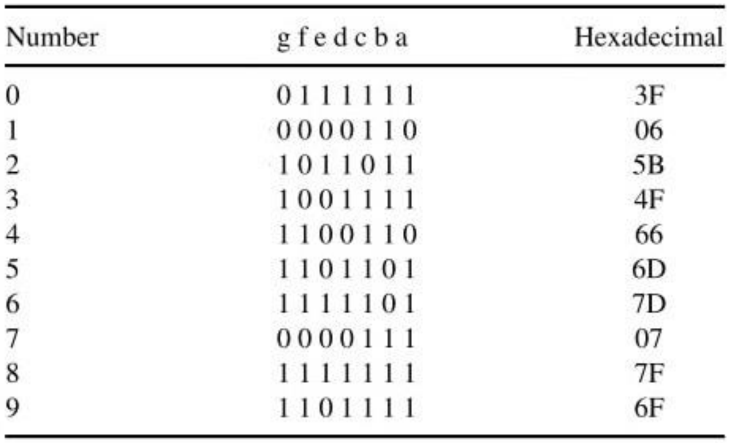
In this type all the LEDs are commonly grounded, means their cathodes are common and can be connected to the negative terminal of the battery using a single pin, the LEDs are tuning on or off using anode. One can apply HIGH to the respective pin to display the respective digit on display.

COMMON ANODE: -

In this type all the LEDs are commonly connected to supply, means their anodes are common and can be connected to the positive terminal of the battery using a single pin, the LEDs are tuning on or off using cathode. One can apply LOW to the respective pin to display the respective digit on display.

BCD NUMBERS & 7-SEGMENT: -

BCD (Binary Coded Decimal) numbers can be displayed on the 7-segment display with some combination of codes applied on the pins of the seven-segment.



Number	g f e d c b a	Hexadecimal
0	0 1 1 1 1 1 1	3F
1	0 0 0 0 1 1 0	06
2	1 0 1 1 0 1 1	5B
3	1 0 0 1 1 1 1	4F
4	1 1 0 0 1 1 0	66
5	1 1 0 1 1 0 1	6D
6	1 1 1 1 1 0 1	7D
7	0 0 0 0 1 1 1	07
8	1 1 1 1 1 1 1	7F
9	1 1 0 1 1 1 1	6F

To display the respective digit on the display respective code can be applied on the pins. One can apply Hex or Binary code for that purpose.

METHODS FOR BCD DISPLAY: -

There are commonly two methods to display BCD numbers on the 7-segment. Two of the algorithms are same to some extent just the different is in fetching the value from the PIC.

- Using RETLW Instruction
- Using Look-Up-Table

USING RETLW INSTRUCTION: -

RETLW instruction is an assembly instruction similar to the RETURN, but instead of only returning the code from the Sub-Routine to the location where it was called it returns a literal value to the WREG. Thus, it does two tasks at a time, returning code from sub-routine and returning literal value to working register.

In this algorithm PORTB is initialized as OTUPUT port and 7-segment is connected to the PORTB pins, the BCD is incremented and converted to the code for 7-segment. The conversion process is being done using RETLW and Program Counter PCL register.

PROGRAM: -

✚ CODE: -

```
#include "P18F452.inc"
LIST P=18F452, F=INHX32, MM=OFF

    CONFIG OSC=XT
    CONFIG WDT=OFF
    #define LEDs PORTB

    CBLOCK 0x00
    TEMP_0, TEMP_1, T1, T2, T3
    ENDC

    ORG 0x00
    GOTO MAIN
    ORG 0x200

MAIN:
    MOVLB 0x01
    MOVLW 0x00
    MOVWF TRISB
    CLRF TEMP_0, 1
    CLRF TEMP_1, 1
    CLRF LEDs

ENDLESS_LOOP
    MOVF TEMP_0, W, 1
    CALL BCD_SevnSeg
    MOVFF WREG, LEDs
    INCF TEMP_0, F, 1
    MOVF TEMP_0, W, 1
    DAW
    ANDLW 0x0F
    MOVWF TEMP_0, 1
    CALL DELAY
    BRA ENDLESS_LOOP

BCD_SevnSeg
    MULLW 0x02
    MOVFF PRODL, WREG
    ADDWF PCL, F
    RETLW 0x3F
    RETLW 0x06
    RETLW 0x5B
    RETLW 0x4F
    RETLW 0x66
    RETLW 0x6D
    RETLW 0x7D
    RETLW 0x07
    RETLW 0x7F
    RETLW 0x6F
```

```
DELAY ;500ms Delay, F=10MHz
            MOVLW D'5'
            MOVWF T1
    LOP_1   MOVLW D'200'
            MOVWF T2
    LOP_2   MOVLW D'250'
            MOVWF T3
    LOP_3   NOP
            NOP
            DECF T3, F
            BNZ LOP_3
            DECF T2, F
            BNZ LOP_2
            DECF T1, F
            BNZ LOP_1
    RETURN
END
```

EXPLANATION: -

The code starts from the Main and TRISB, PORTB and some CBLOCK locations are cleared and Endeless_Loop is started. BCD to Seven-Segment is called which will convert the given BCD to a hex code. This is done by using RETLW instruction and Program Counter in this code.

In conversion sub routine number is multiplied by two and the value is added to the Program Counter. Upon value of the PCL the RETLW instruction s executed.

Then the value of the BCD is incremented and saved in a specific register, DAW instruction corrects the BCD if it is incorrect. Then a delay is called and loop starts again from the calling of the conversion sub routine.

The delay sub-routine is a 500ms delay which is called so the one can clearly see the number displaying on the display.

CIRCUIT: -

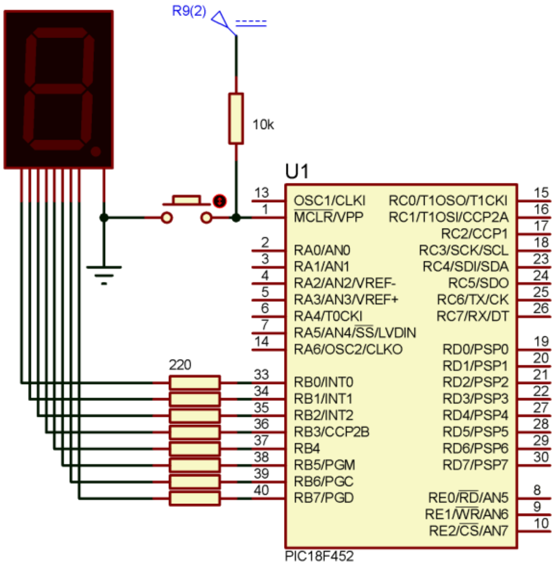


Figure 2.1: 7-segment display with PIC

Home Task: -

- Write a code for PIC to display digits from 0-9 on 7-segment LED using Look-Up-Table method.

Theory: -

Methods for BCD Display: -

A method is explained above which is using RETLW instruction and the other is explained below which is Loo-Up-Table method.

Using Look-Up-Table: -

Look-Up-Table is method to store data in Program ROM. The mode of addressing used to access pre saved data in the ROM is called Table Processing. Basically, in Look-Up-Table a combination of data is saved in the Program ROM and this data can be access using specific register.

DB *(Define Bit)* instruction is used to save data in the ROM, Using DB a bunch of data or a single character or a number can be burnt to the PIC ROM.

TBLPTR *(Table Pointer)* is a register used to access the data save din the ROM using DB instruction, it loads a register with location which is being sent to it, basically this pointer register points to the location of pre-saved data. It is 21-bit register to access all of the 2MB ROM of the PIC.

TABLAT *(Table Latch)* is a register used to read the location toward which TBLPTR points. This instruction fetches the data and loads it to another register or WREG.

Program: -

Code: -

```
#include "P18F452.inc"
LIST P=18F452, F=INHX32, MM=OFF

    CONFIG OSC=XT
    CONFIG WDT=OFF
    #define LEDs PORTB

    CBLOCK 0x00
    TEMP_0, T1, T2, T3
    ENDC

 ORG 0x00
    GOTO MAIN
    ORG 0x200

MAIN:
    CLRF TRISB
    CALL START_AGAIN

ENDLESS_LOOP
    TBLRD*+
    TBLRD+*
    MOVF TABLAT, W
    MOVWF LEDs
    CALL DELAY
    DECF TEMP_0, F
    BNZ ENDLESS_LOOP
    CALL START_AGAIN
    BRA ENDLESS_LOOP

START_AGAIN
    MOVLW D'20'
    MOVWF TEMP_0
    MOVLW 0x00
    MOVWF TBLPTRL
    MOVLW 0x05
    MOVWF TBLPTRH
    RETURN

DELAY                   ; 500ms Delay, F=10MHz
            MOVLW D'5'
            MOVWF T1
    LOP_1   MOVLW D'200'
            MOVWF T2
    LOP_2   MOVLW D'250'
            MOVWF T3
    LOP_3   NOP
            NOP
            DECF T3, F
            BNZ LOP_3
            DECF T2, F
            BNZ LOP_2
            DECF T1, F
            BNZ LOP_1
            RETURN

LUT:
    ORG 0x500
    DATA1 DB 0X3F
    DATA2 DB 0X06
    DATA3 DB 0X5B
    DATA4 DB 0X4F
    DATA5 DB 0X66
    DATA6 DB 0X6D
    DATA7 DB 0X7D
    DATA8 DB 0X07
    DATA9 DB 0X7F
    DATA10 DB 0X6F
END
```

EXPLANATION: -

The code is similar to the above code the difference is that the hex codes is fetched from the ROM. These codes are written in to ROM using DB and is fetched using TBLRD instruction. In the program like above program a value is incremented and TBLPTR register is also incremented and the value of the TABLAT is called using TBLTD instruction. Here auto incrementation of the TABLAT is being used by using the instructions

- TBLRD*+ (Read the content and Increment by one)
- TBLRD+* (Increment the content by on and Read)

Due to the fact the PIC instructions are two bytes, so that every instruction is in memory with a difference of two bytes from the previous one. Thus, TABLAT is incremented two time in this code.

CIRCUIT: -

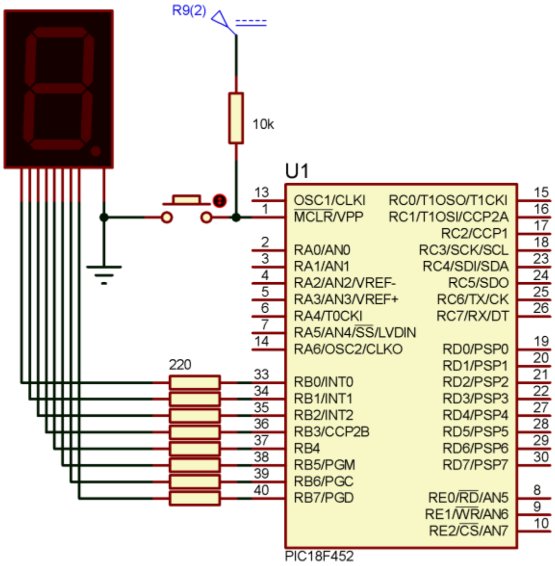
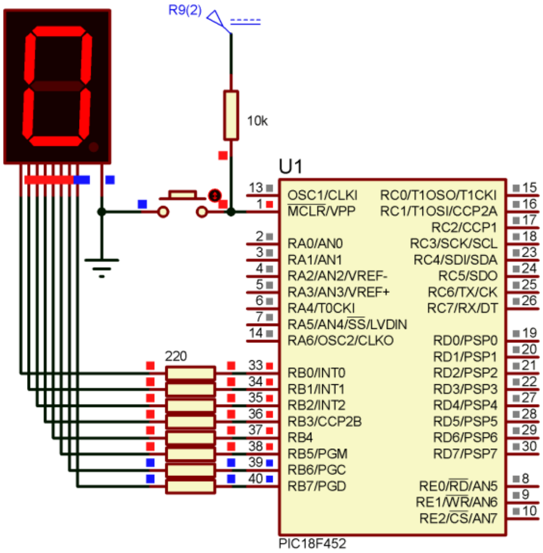
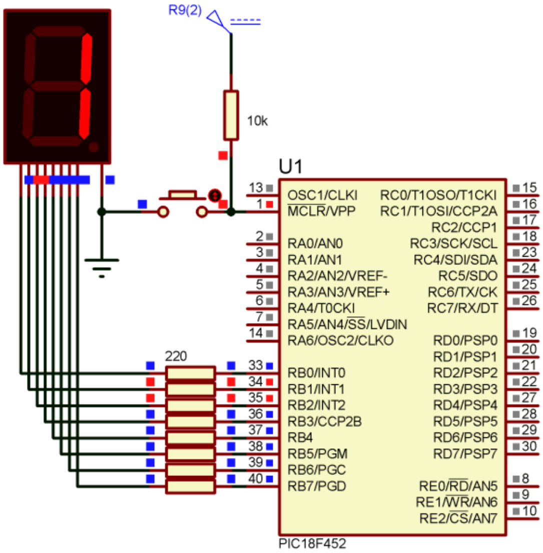


Figure 2.2: 7-segment display with PIC

DISPLAY IN WORK: -





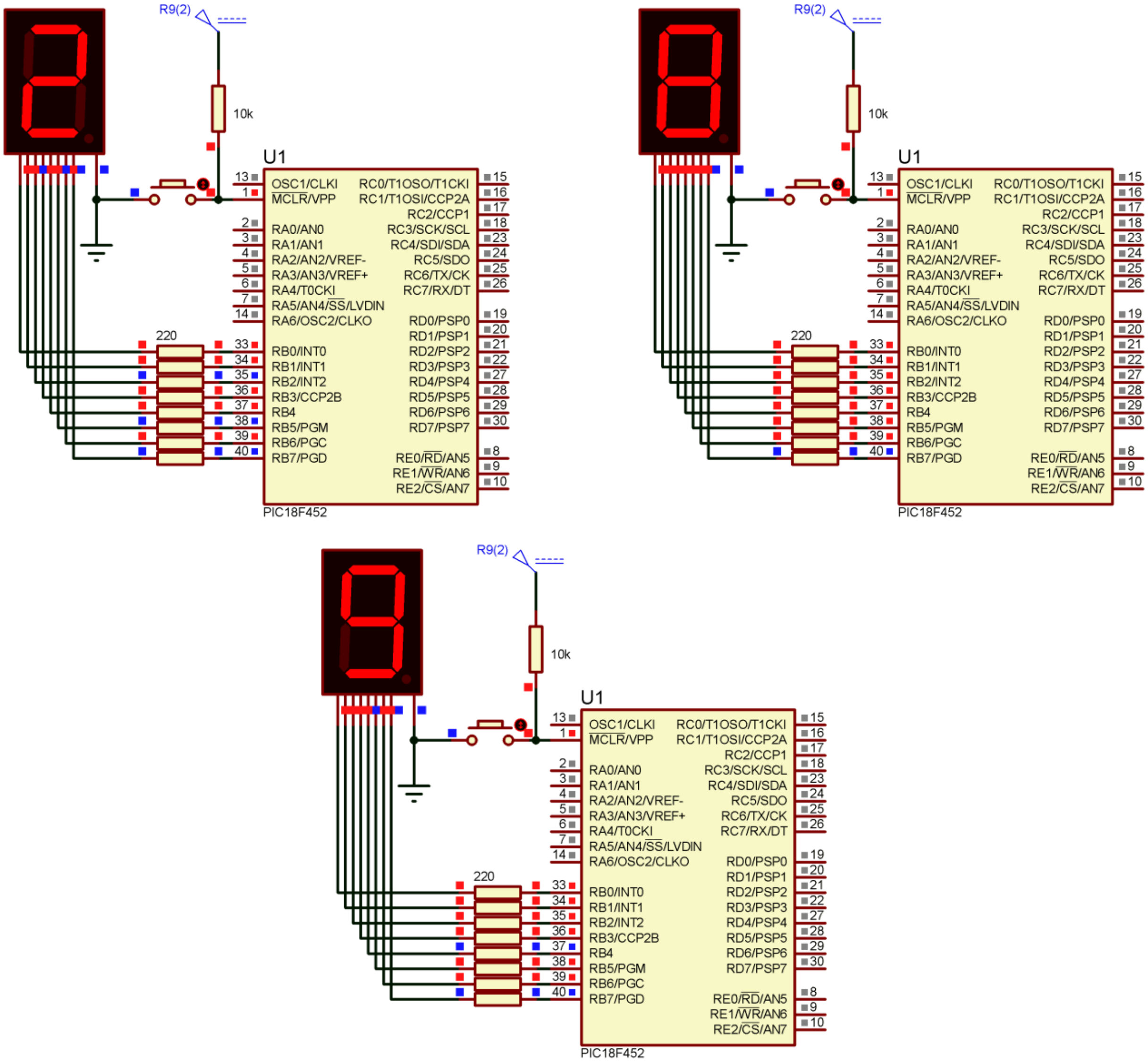


Figure 2.3: Some Digits being Displayed on the Display

CONCLUSION: -

- In this I have learnt what is 7-segment display and how to use them.
- I also learnt how a 7-esgment is configured with the PIC using assembly language.
- It was learnt that how to use the RETLW instruction in PIC assembly programming.
- Look-Up-Table was also introduced in the lab and learnt that how to use Look-Up-Table in the assembly and how data is saved in Program ROM permanently.
- Look-Up-Table read instructions and TBLPRT registers are being introduces in the code and learnt how to use them.