Due on: December 23, 2021

Introduction to Microcomputers

Lab6: Using the Seven Segment Displays

The goal of this lab is to make use of the Seven Segment Displays (SSD) connected to PORTD of PIC16F877A on PICSIM. Recall that the SSD to be lit up is first selected by making its selection bit A2-A5 HIGH. Thus, to display a value on the first SSD (DIS4 on PICSIM), we first set A5 to logical 1 (HIGH), and then write the appropriate value corresponding to the number to be displayed to PORTD. The bit sequence necessary to display each number was discussed in class.

Assignment

You are asked to implement a simple counter that starts at 0 and counts up to decimal 20. When your counter reaches 20, it goes back 0 and counts up again. You counter must increment every second.

The most important thing in implementing this project is to avoid flicker on the SSDs. Notice that you will be displaying a two digit number, i.e., you will be displaying a value on the first and second SSDs (DIS3 & DIS4 on PICSIM). To display the counter, make sure that you select and display the first digit on the first SSD (DIS4). Then wait for 5ms so that the LEDs on the first SSD light up completely. Then select the second SSD (DIS3) and display the second digit on the second SSD. Again wait for 5ms to make the LEDs on the second SSD to light up completely.

So far you displayed the two digit number on both SSDs and spent about 10ms. Before you increment your counter, you need to wait for another 980ms. If you simply wait for 980ms in an idle loop now, your first SSD will turn off and you will not be able to see the first digit anymore! To avoid this problem, write a display loop that iterates for some number of iterations (NO_ITERATIONS). Within the loop, display the first digit and wait for 5ms. Then display the second digit and wait for 5ms. Thus, the two digit number will be visible on the SSDs all the time. When the loop terminates, you have spent about 1000ms, i.e., 1 second. You can increment your counter and go back to the beginning of the main loop to display this number. Make sure that after your counter reaches 20, it rolls back to 0 and counts up from there.

Here is the pseudocode for this project in C:

```
// Counter is represented as <digit1><digit0>
#define NO_ITERATIONS 90   // Since the computation in the loop also take some time, we iterate less than 100 times
digit0 = 0;
digit1 = 0;
while (1){
   for (int i=0; i<NO_ITERATIONS; i++){
        PORTA5 = 1;                    // Select the first SSD
        PORTA4 = 0;                    // Second SSD is not selected
        PORTD = SSDCodes [digit0];    // The code to display the first digit
        DelayMs(5);                   // 5 millisecond delay

        PORTA5 = 0;                   // First SSD is deselected
        PORTA4 = 1;                   // Select the second SSD
        PORTD = SSDCodes[digit1];    // The code to display the second digit
        DelayMs (5);                  // 5 millisecond delay
   } //end-for
```

```
    digit0++;
    if (digit0 == 10){
       digit0 = 0;
       digit1++;
    } //end-if

    if (digit1 == 2 && digit0 == 1){
        digit1 = digit0 = 0;
    } //end-if
 } //end-while
```

Notice that in the code above, the display loop iterates 90 times rather than 100 times, which is what you may expect theoretically. This is due to the fact that although we wait 5+5=10ms after displaying the digits on the SSDs, the other computations in the loop also take some time. To compensate for that time, we need to iterate less than 100 times. My experiments have shown that iterating 90 times gives you about 1 second delay.

A more accurate way of counting up to 1 second would have been to start a timer (Timer1 for example), and let the counter count up to 1 second. We will then continuously check if the timer has expired and then exit the loop. To do this, we have to cover timers, which we will do later in the course.