ECE 3720

Microcomputer Interfacing Laboratory

Section 005

Aaron Bruner

Date Performed: 02 / 04 / 2021

Lab 3

ABSTRACT:

A lab designed to demonstrate how to program the PIC32 microcontroller and utilize it’s built in comparators. The built-in comparators will be comparing different voltages coming in from a potentiometer. The purpose of the lab is to control three LEDs using the potentiometer. Similar to the previous labs, this will also enhance skills in reading documentations for new and existing parts.

**INTRODUCTION:**

The goal of lab 3 is to program our microcontroller to compare voltage readings that are coming from a potentiometer. We will be sending our voltage into three comparators and based on the value and where it lies in the ranges the LEDs will respond accordingly. Thus, when the potentiometer is turned to position 1 only the far-left LED will be on. When the potentiometer is turned slightly past halfway LED 2 will come on. And finally, once it’s at position 3 it will light up the far-right LED resulting in all three LEDs being on. To perform our lab, we required a variety of materials including:

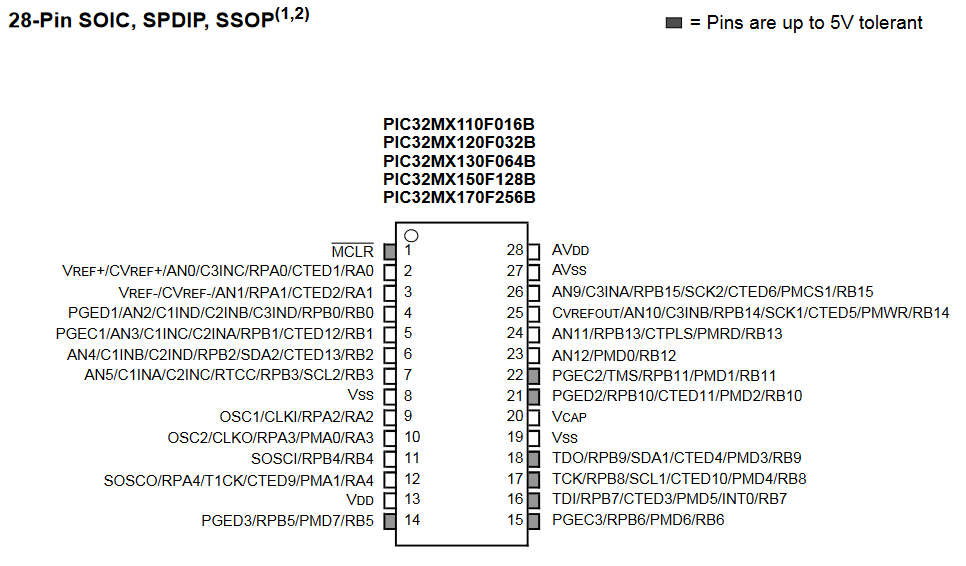
* A breadboard
* Wires
* Potentiometer ( x1 10kΩ )
* PIC32MX150F128D
* Analog Discovery 2 (AD2)
* Digilent WaveForms
* MPLAB Snap Debugger
* MPLAB X IDE (Programming Software)

The AD2 is going to be a partial power source and our output LEDs. Instead of using actual LEDs we will use the WaveForms software and simulate the LED output. The partial power source will power the PIC32 chip and be a 3V reference source. However, it will not power the MPLAB Snap Debugger. It will be powered off of a Micro USB cord.

**EXPERIMENTAL PROCEDURES:**

We must assume that the individual reproducing this lab has already setup their breadboard in a manner that their PIC32 chip can be programmed using the MPLAB X IDE software.

To begin we need to setup our circuit by referencing the diagram below for each element. We need to utilize the following pins for our lab: V\_REF+ pin (2), V\_REF- (3), C3INA (26), C1INA (7), C2IND (6), C3IND (4), B15 (Potentiometer), and B12-B14 for our LEDs.



The first portion of our circuit is going to be the potentiometer. To setup our potentiometer we first need to set the left pin (1) to V\_REF-, the middle pin (2) to B15 which is our output, and finally the right pin (3) to V\_REF+. The voltage output of our potentiometer will be compared to IV\_REF, CV\_REF and C3INA. As the potentiometer is turned the LED outputs will light up in order from left to right as the voltage into the microcontroller increases.

Normally we use our 3.3V positive line (pin 2) for everything but since we’re wanting to compare from V\_REF- (3) to C3INA which needs to be 3V we need use the square wave generator function in WaveForms and apply that to our circuit to pin 26.

The output LEDs will be tied to B12, B13 and B14 which correspond to pins 23, 24, and 25.

The code we wrote was rather lengthy since we had to do many registry operations to ensure that our compare registers were correct. However, it’s possible to shrink our code by a large portion by using hexadecimal values instead of setting every register. At the beginning of our code we first set every pin we need to analog instead of digital so that we can measure the voltage reading on each pin. Next we need to determine which pins are going to be inputs and which will be outputs. The pins for our potentiometer (C1INA, C2IND, C3IND) and C3INA. We will have output pins for our three LEDs. Next we need to enable our three compare registers using CMxCONbits.ON = 1;. Now that the compare registers are on we need to configure each one individually. We will be configuring CPOL, CREF, and CCH for the registers. The CPOL option is compare output inversion, it was used to set the polarity of the comparator. The CREF option was used to choose CV\_REF or the specific comparator C3INA as comparators positive terminal input. Lastly, the CCH option was the negative input select, also know as the negative terminal select. For the first comparator we set the negative terminal to 0x3 which implies the IV\_REF value as shown in the PIC32 datasheet. Similarly, for comparator 2 and 3 we set it to 0x2 which is the CxIND value. Lastly, we need to set up our 0.75\*V\_REF which is CV\_REF using CVRCONbits.ON, CVRCONbits.CVRSS, CVRCONbits.CVRR and CVRCONbits.CVR. CVRCONbits.ON enables the module in our PIC32. CVRCONbits.CVRR is set to 0 because we want out CV\_REF to be between 0.25V\_REF and 0.75V\_REF. CVRCONbits.CVRSS is set to 1 so that CV\_RSRC is equal to the difference between V\_REF+ and V\_REF-. CVRCONbits.CVR was set to 0xF so that we’d have which simplifies to which is close to our 0.75V\_REF that we wanted. Lastly, we developed a while loop to constantly takes the values from CMSTAT for each register.

**RESULTS and DISCUSSION:**

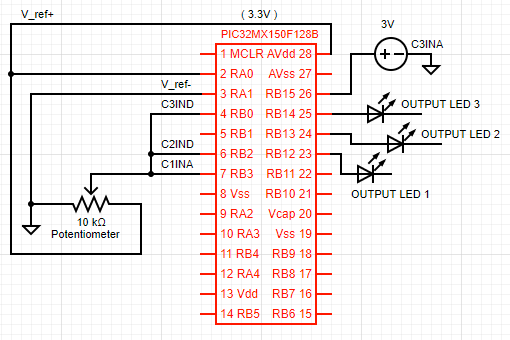
Once the microcontroller was programmed it was clear that things were working when we began to turn the potentiometer and the LEDs started lighting up in order. And then, when we turned the dial back the LEDs turned off in reverse order back to completely off.

One problem during this experiment was that an LED was being always driven high and thus the output was constantly incorrect no matter where the potentiometer was set to. For the longest time I could no determine why the pin was instructing the LED to light up until I measured the pin’s voltage using a voltmeter and noticed that it’s on at all times, even when the circuit is now powered. Thus, once the LED output pins were changed everything began to work correctly again.

This week we demonstrated how to utilize a potentiometer and compare registers. This lab is a good learning experience because potentiometers are used in almost every application that changes values and compare registers are used in many simple calculations.

**CONCLUSION:**

To conclude, lab 3 taught us many useful techniques for using compare registers on the PIC32 and demonstrating these measurements using digital LED outputs. This lab also taught us more complex ways to analyze problems and how to solve them.

**FIGURES AND TABLES:**

**Figure 1: Wiring for lab 3 (Pin connections described in experimental procedures)**

**CODE:**

#include <xc.h>

main(void){

ANSELAbits.ANSA0 = 1;

ANSELAbits.ANSA1 = 1;

ANSELBbits.ANSB0 = 1;

ANSELBbits.ANSB2 = 1;

ANSELBbits.ANSB3 = 1;

ANSELBbits.ANSB15 = 1;

// Inputs for the potentiometer

TRISBbits.TRISB15 = 1;

TRISAbits.TRISA0 = 1;

TRISAbits.TRISA1 = 1;

// Set C1INA, C2IND, C3IND to input

TRISBbits.TRISB3 = 1;

TRISBbits.TRISB2 = 1;

TRISBbits.TRISB0 = 1;

TRISBbits.TRISB15 = 1;

// Digital Outputs for LEDs

ANSELBbits.ANSB12 = 0;

ANSELBbits.ANSB13 = 0;

ANSELBbits.ANSB14 = 0;

// Set output LED pins to output

TRISBbits.TRISB12 = 0;

TRISBbits.TRISB13 = 0;

TRISBbits.TRISB14 = 0;

// Enable compare register 1, 2 and 3

CM1CONbits.ON = 1;

CM2CONbits.ON = 1;

CM3CONbits.ON = 1;

// Compare output inversion bit

CM1CONbits.CPOL = 0;

CM2CONbits.CPOL = 1;

CM3CONbits.CPOL = 1;

// Comparator Positive Input Configure bit

CM1CONbits.CREF = 0;

CM2CONbits.CREF = 1;

CM3CONbits.CREF = 0;

// Comparator Negative Input Select bits for Comparator

CM1CONbits.CCH = 0x3;

CM2CONbits.CCH = 0x2;

CM3CONbits.CCH = 0x2;

// Set the comparator reference voltage (CV\_REF)

CVRCONbits.ON = 1;

CVRCONbits.CVRSS = 1;

CVRCONbits.CVRR = 0;

CVRCONbits.CVR = 0xF;

while(1) {

LATBbits.LATB12 = CMSTATbits.C1OUT;

LATBbits.LATB13 = CMSTATbits.C2OUT;

LATBbits.LATB14 = CMSTATbits.C3OUT;

}

}