**Testing Your Circuit**

Walter Heth

EE470

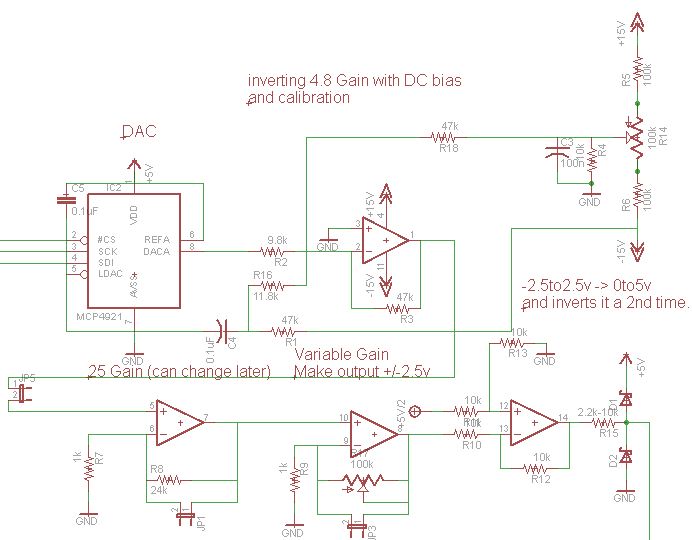


Table of Contents

[Soldering your circuit 3](#_Toc365533840)

[Bill of Materials (BOM) 3](#_Toc365533841)

[REV 1 4](#_Toc365533842)

[REV 1 fixes 5](#_Toc365533843)

[Rev2 6](#_Toc365533844)

[Testing your circuit 6](#_Toc365533845)

[Testing components 6](#_Toc365533846)

[Experiment 1: Testing the first stage. 7](#_Toc365533847)

[Experiment 2-4, Testing stages 2-4 9](#_Toc365533848)

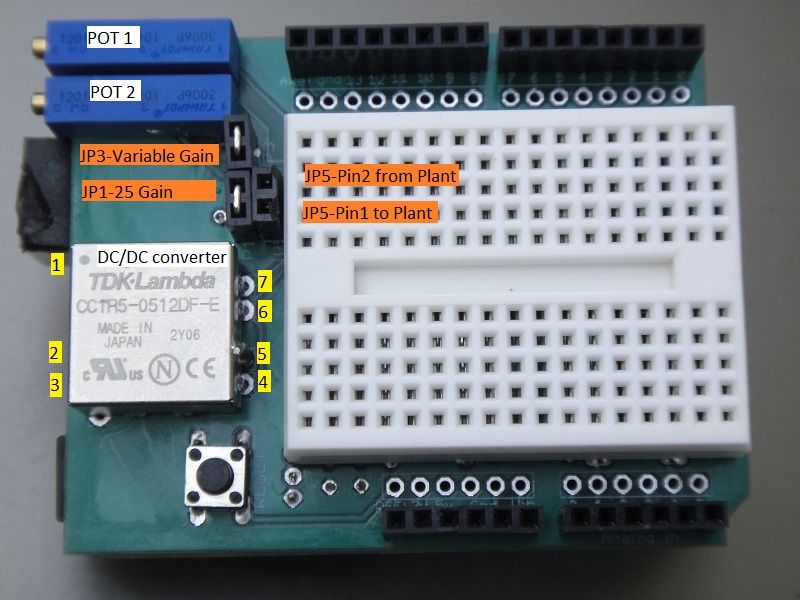
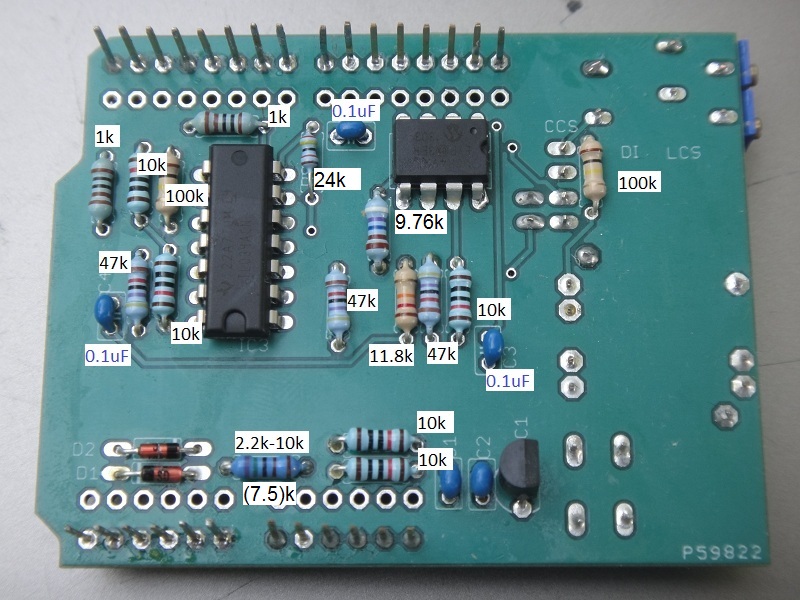
[SEND VOLTAGE CODE: 12](#_Toc365533849)

# Soldering your circuit

## Bill of Materials (BOM)

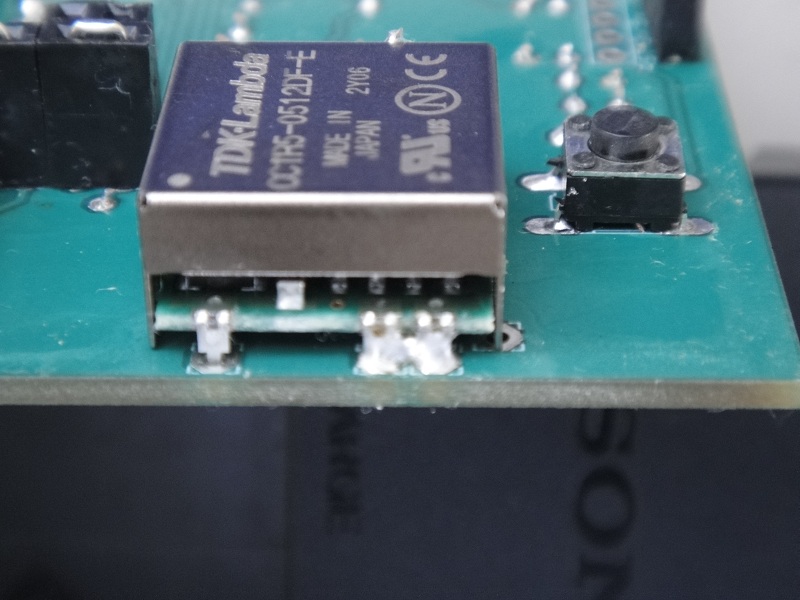
|  |  |  |
| --- | --- | --- |
| Description | Quantity | Part# |
| DC/DC converter | 1 | CC1R5-0512DF-E |
| Quad Op-amp | 1 | TL034ACN |
| DAC | 1 |  |
| 2.5Vref | 1 |  |
| Reset button | 1 |  |
| 100k Potentiometers | 2 |  |
| 1x2 header | 3 |  |
| 1x6 | 2 |  |
| 1x8 | 2 |  |
| 0.1uF | 5 |  |
| 10k | 5 | R4, R10-13 |
| 100k | 2 | R5, R6 |
| 47k | 3 | R1, R3, R18 |
| 1k | 2 | R7, R9 |
| 24k | 1 | R8 |
| 9.8k | 1 | R2 |
| 11.8k | 1 | R16 |
| 2.2k-10k (7.5k) | 1 | R15 |
| Small R around 300 Ohms | 4 | R19-22 |
| LEDs | 4 | LED1-4 |
| Diodes | 2 |  |
| Mini-breadboard | 1 |  |

## REV 1

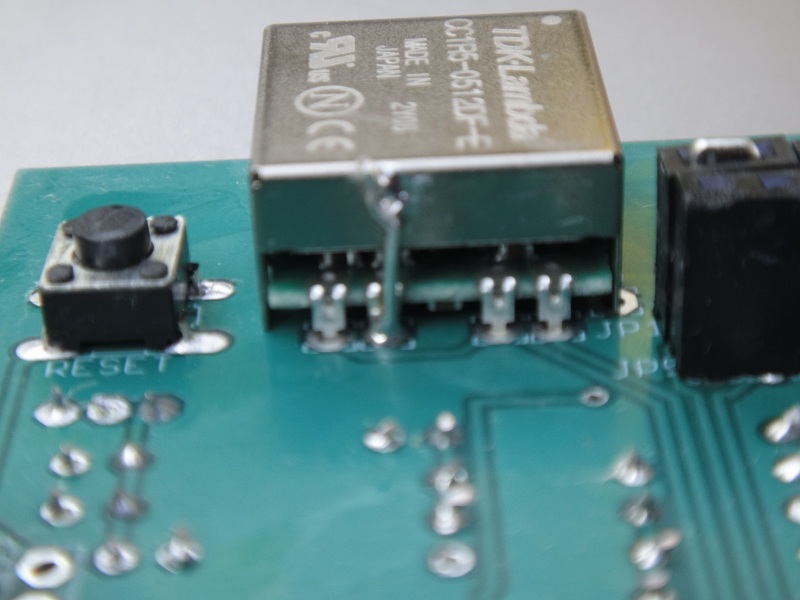


## REV 1 fixes

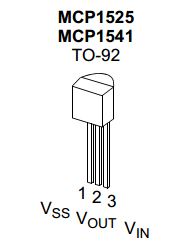
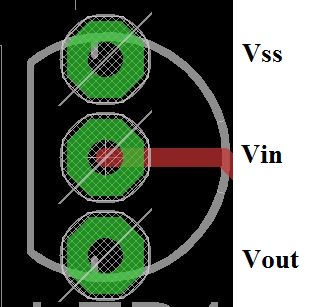
Soulder pins 2 and 3 together to have them both grounded



Solder a short from pin 5 to the chassis (ground).



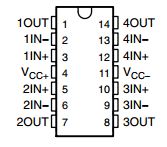
Pins 2 (Vout) and 3 (Vin) of the Vref need to be swapped.

# Rev2

Identical to Rev 1 except you do not need to make the fixes listed above. Just solder them straight in.

Also added LEDs 1-4 and R19-R22. These are not necessary but if you wish you can solder them in.



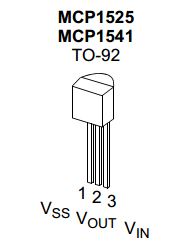
# Testing your circuit

## Testing components

First test the DAC by plugging in 0-5v into A0 and using the code from LAB2 to read A0 and send the digital value to the DAC. Check pin8 to make sure you are getting the identical voltage.

Check the Vref

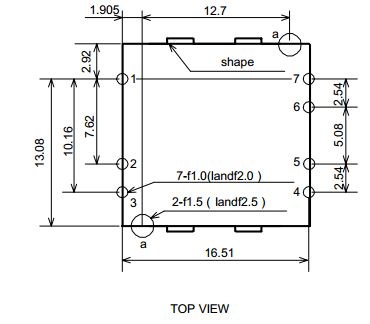
If the flat surface is facing towards you, count the pins 1-3 from left to right.

Pin# Description Expectred voltage

1 Vss Gnd

2 Vout 2.5v

3 Vin 5v



Check the DC/DC converter

Pin# Description Expected voltage

1 +Vin 5v

2 RC (enable) Gnd

3 -Vin Gnd

4 -Vout -15v

5 COM Gnd

6 TRM -15v

7 +Vout +15v

## Experiment 1: Testing the first stage.

This stage shifts the 0-5v to +/-2.5v, amplifies it by 4.8 to get +/-12v and then inverts it to -/+12v.

Therefore if we plug in 0v we should get +12v and 5v should give us -12v. This can be measure either on pin1 of the quad opamp IC or on the jumper that leads to the breadboard. To calibrate the shield use the Send Voltage code provided at the end of this document to set Vout = 0; This will send 0v to the **output of the first stage**. If you’re not getting exactly 0v then try adjusting the 1st potentiometer, the one in line with the headers until it is zeroed. Now try putting in Vout = +/-12v to make sure it hits these limits dead on.

Explaination for Send Voltage code:

We know from Lab 2 that

Again stage 1 shifts 0-5v down -2.5v, amplifies it by 4.8 and inverts it. The equation for stage 1 is

Plugging Vin for Voltage desired we get

Example Values

STAGE1 OUT DAC Value DAC Output

+12v 0 0v

+2.5 489.2 2.39v

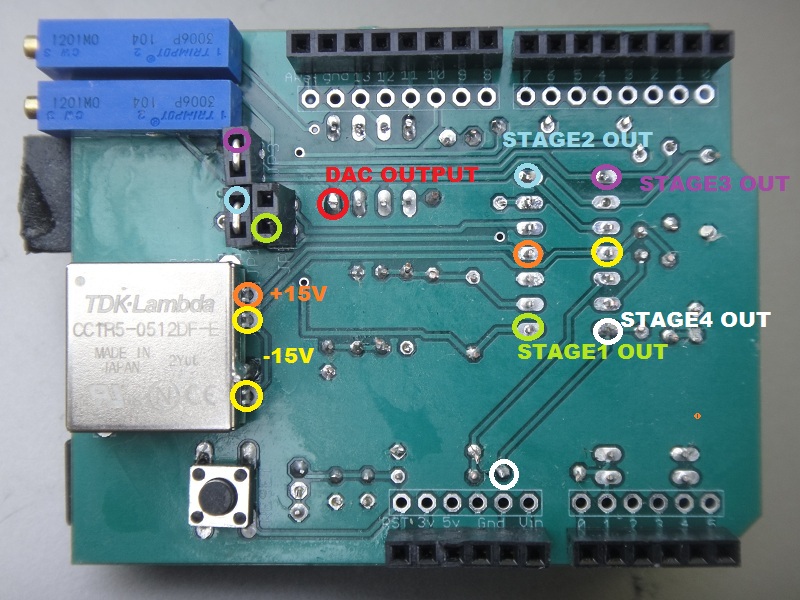
+0.1 507.2 2.4789v

0 511.5 2.5v

-0.1 515.7 2.52v

-2.5 618.1 3.021v

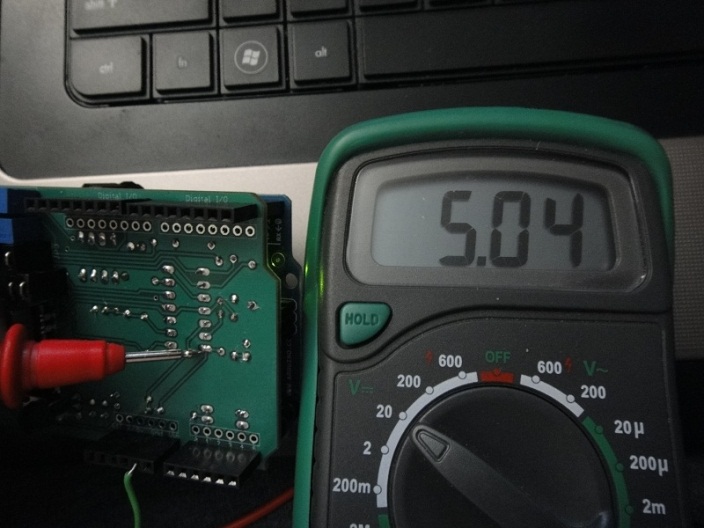
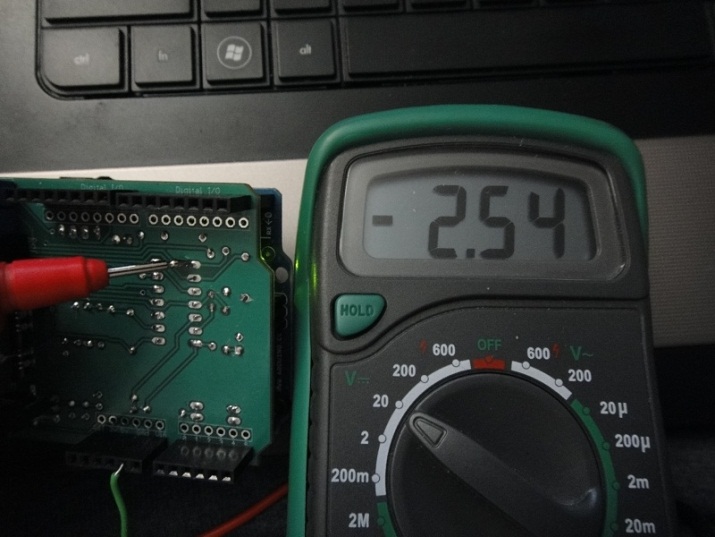
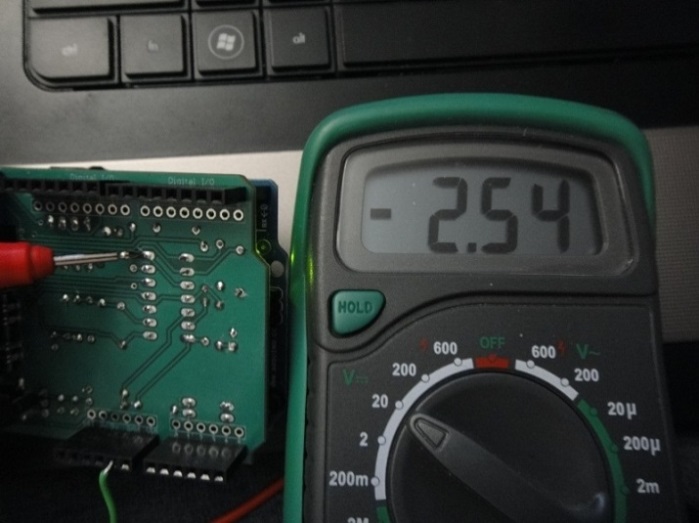
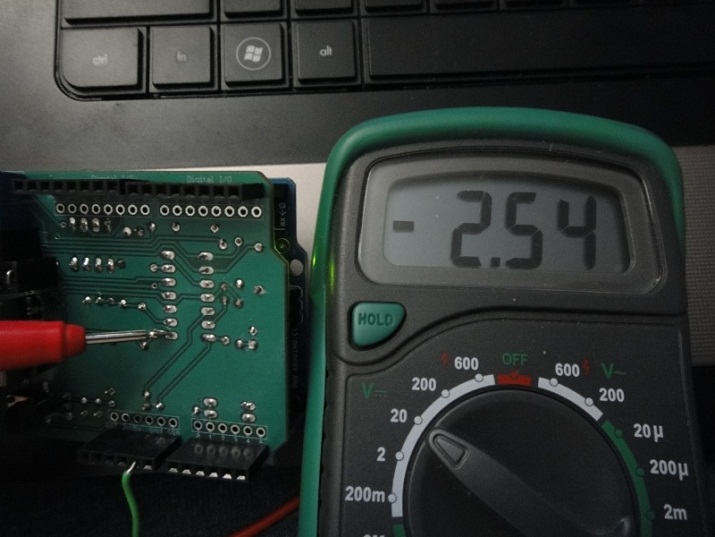
-12v 1023 5v



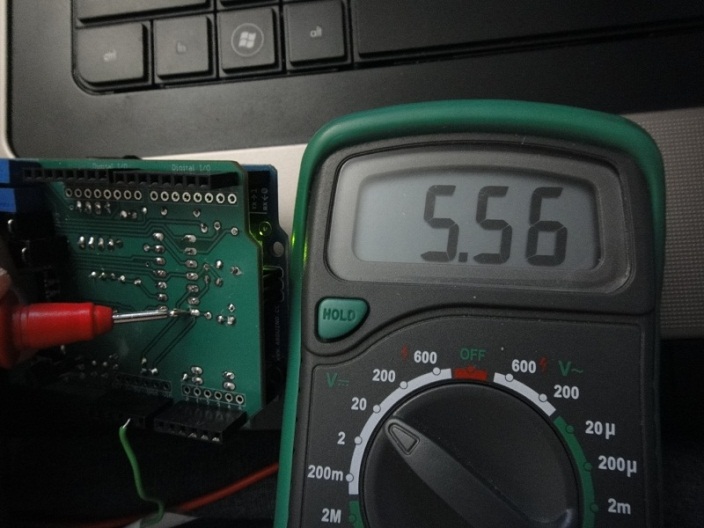
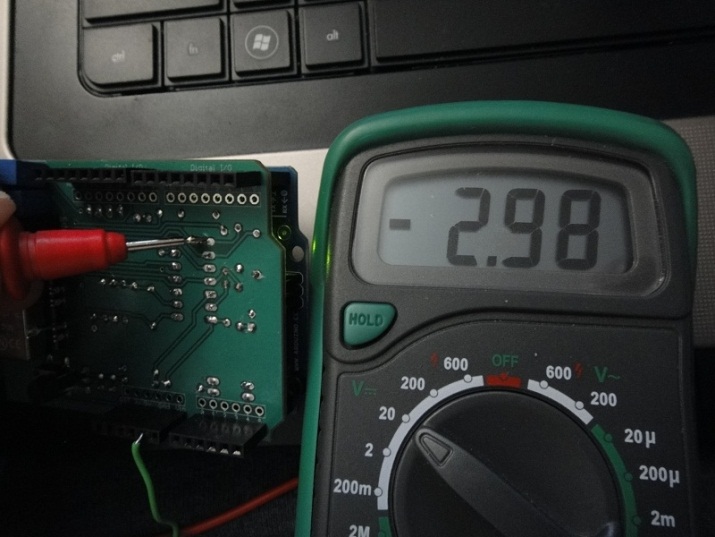
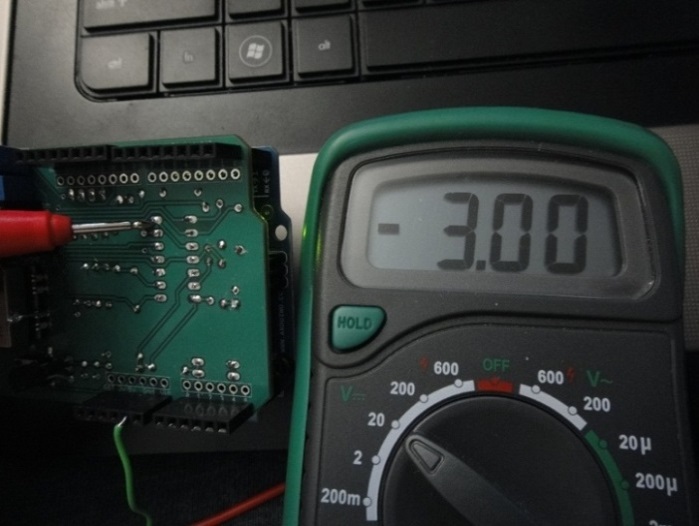
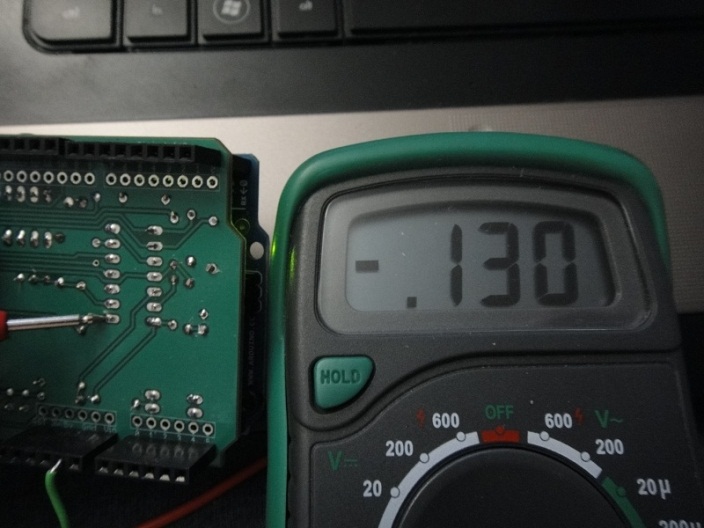
## Experiment 2-4, Testing stages 2-4

Stage 2 by default is set for a gain of 25 non inverting. Stage 3 by default has a variable gain using the 2nd potentiometer. Stage 4 re-inverts the signal so that it is back in phase with the signal sent to stage 1 from the DAC and then shifts the signal back to the 0-5v range. To test this we will need a signal going from +/-2.5v into Stage 4. This can be done by setting both opamps to unity gain by shorting their jumpers and sending in +/-2.5v from stage 1. In other words, short all the jumpers. Or we can leave Stage2 at its default gain of 25, short stage 3 for unity gain and have stage 1 send in +/-0.1v.

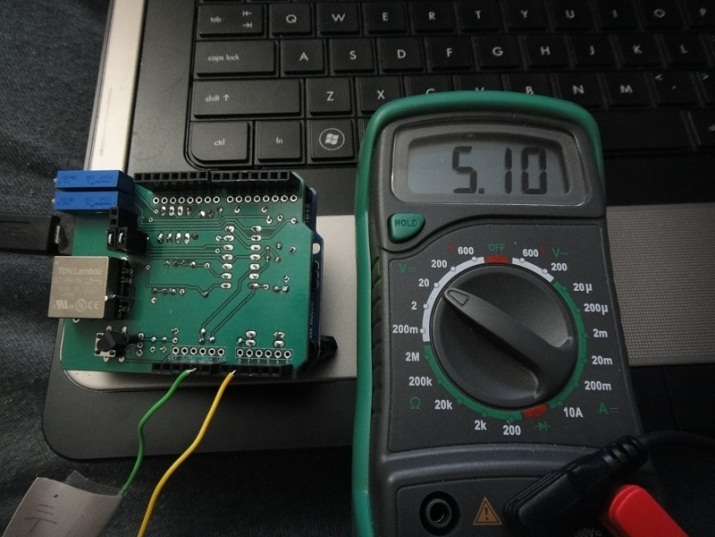
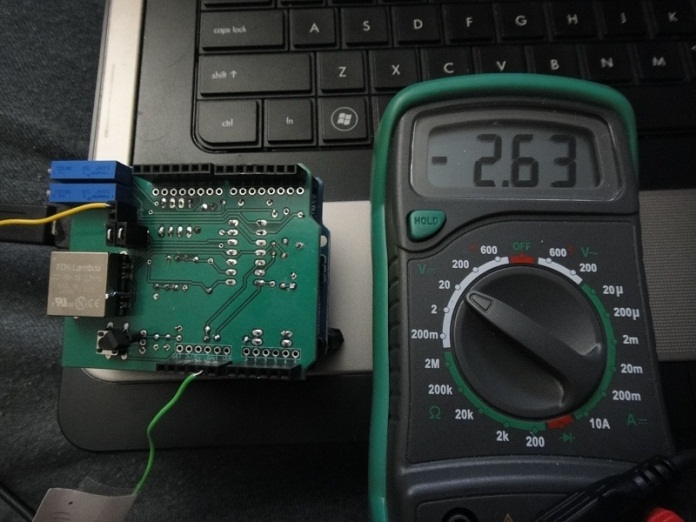
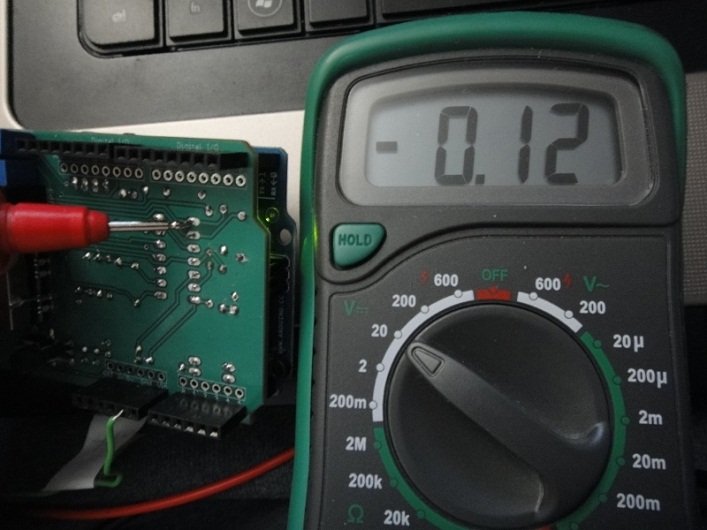
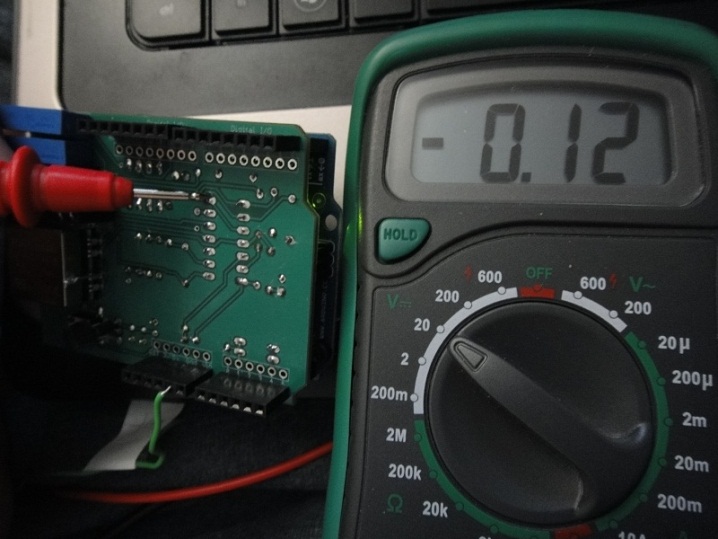
In this example we send -2.5v and set stages 2 and 3 to unity gain. When you do the actual lab you will probably use this configuration as the gain stages are not necessary for the double integrator plant. Also notice how the final stage exceeds 5v! This is because the quad op amp on the PCB is powered by +/-15v. When we connect our plant we will power it with +/-2.5v making it impossible to exceed 5v.



Example 2: Send -0.1v, Stage 2 default, Stage 3 unity gain.



Example 3: Send -0.1v, Stage 2 unity gain, Stage 3 slowly increase resistance.



# SEND VOLTAGE CODE:

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*//

// Name : Send\_Voltage //

// Author : Walter Heth //

// Date : April 8, 2013 //

// Version : 1.0 //

// Function : Sends a specific voltage to Stage1 output //

// Notes 1. Enter a specific value for Vout1 //

// 2. Upload program and measure Stage1 output //

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// SPI Interface SS\_PIN(PB2), SCK\_PIN(PB5), MOSI\_PIN(PB3), MISO\_PIN

// Arduino Pin 10 13 11 12

// MCP4921 DAC SS SCK MOSI n/a

#include <SPI.h>

#include <SoftwareSerial.h>

int analogPin = 0;

const int slaveSelectPin = 10;

double Vout1 = 0;

word sensorValue = 0;

int DACval;

byte data = 0;

void setup()

{

// set pin(s) to input and output

pinMode(analogPin + A0, INPUT);

// set the slaveSelectPin as an output:

pinMode (slaveSelectPin, OUTPUT);

// Initializes the SPI bus by setting SCK, MOSI, and SS to outputs,

// pulling SCK and MOSI low, and SS high.

SPI.begin();

}

void loop()

{

DACval=(((-Vout1/4.8)+2.5)\*1023)/5;

//Serial.println(DACval);

DACval = DACval << 2 ;

digitalWrite(slaveSelectPin, LOW);

data = highByte(DACval);

data = 0b00001111 & data;

data = 0b00110000 | data;

SPI.transfer(data);

data = lowByte(DACval);

SPI.transfer(data);

digitalWrite(slaveSelectPin, HIGH);

delay(1000);

}