**Memorandum**

**To:** Dr. Sys Tems, CEO of “Sys Tems LLC”

**From:** Steve Tamayo, “Professional Staff Engineer”, Raymond Yung, “Professional Staff Engineer”, Duy Le, “Professional Staff Engineer”, Angela Albrecht, “Professional Staff Engineer”,

**CC:** Dr. Thomas Chmielewski, “Engineering Advisor”

**Date:** November 8th, 2019

**Subject:** I2C Interface Testing with Mega2560 and Adafruit 12-bit DAC

Dr. Sys Tems,

The engineering team has decided to test the limits of the Mega2560 microcontroller I2C interface. The I2C protocol only requires two wires for communication with almost 1024 devices with 10 bit addressing. Each device has a unique device address so the master can choose with which device it will be communicating. The two wires are called the Serial Clock (SCL) and the Serial Data line (SDA). The SDA line carries the data and both lines are open drain. The Arduino Mega2560 supportsI2C communication using the Wire.h library from Arduino. It has 10 bits of resolution which equates to 1024 different values. These values measure from ground to 5V. The team tested how fast a signal could be reproduced on the DAC by using a 1 ms interrupt to start and slowly decreased the interrupt time to the smallest quantity which still retained a viable signal.

The team connected the Adafruit 12-bit DAC to the Mega2560 microcontroller. The 12-bit DAC maps the values 0 - 4095 to 0 to 5 volts. The DAC uses I2C which is a 2 pin interface where the SDA connects to I2C data and the SCL connects to the I2C clock. A0 allows the user to change the I2C address, which by default is the hex value 0x62. VOUT is the output voltage of the DAC which ranges from 0V to VDD. 0V implies that the DAC value is 0 while VDD is the maximum 12-bit number 0XFFF. The MC4725 library completes all the interfacing necessary to simply read from the DAC voltage output. Within the example library, the team used the Triangle Wave preliminary sketch on the Arduino IDE and connected the VOUT and GD of the DAC to the oscilloscope to view the signal. The triangle wave signal output is shown in Figure 1. By doing this sanity check, we verified that the circuit was wired properly and the Adafruit DAC functionality was preserved.

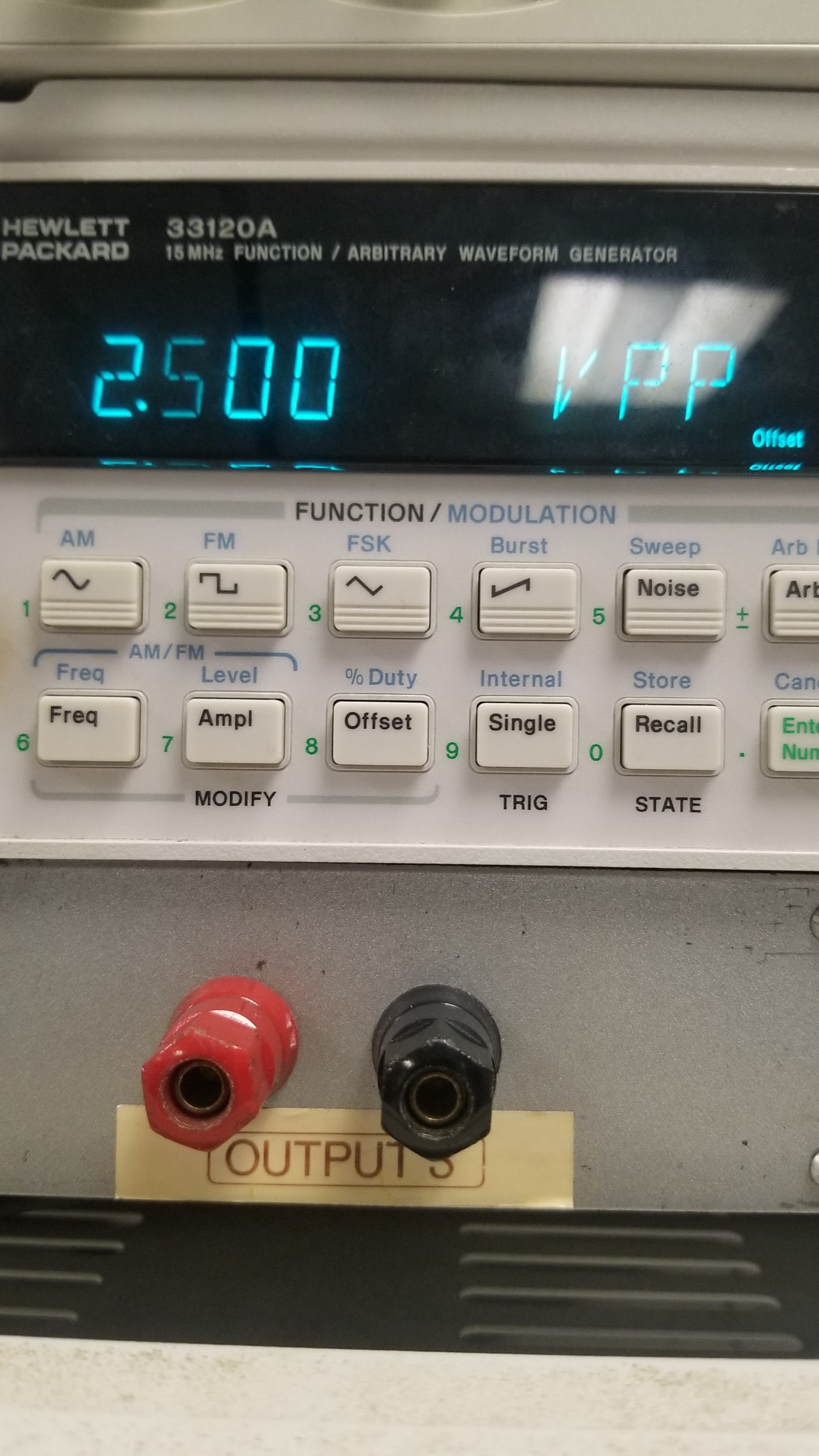
When we introduced a software interrupt into the code, the triangle wave was made with incremental steps. As seen on Figure 2. This is due to the fact that instead of sampling a continuous rate, we sampled the signal every 1 millisecond.

In the second half of our experiment the team implemented a 3-point nulling filter to null out a frequency from the signal. To accomplish this, we placed inside the interrupt arduino code several calculations based on the following equation:

OUTPUT = X1 - 2\*cos(nullingFV)\*X2\*X3 [Eq: 1]

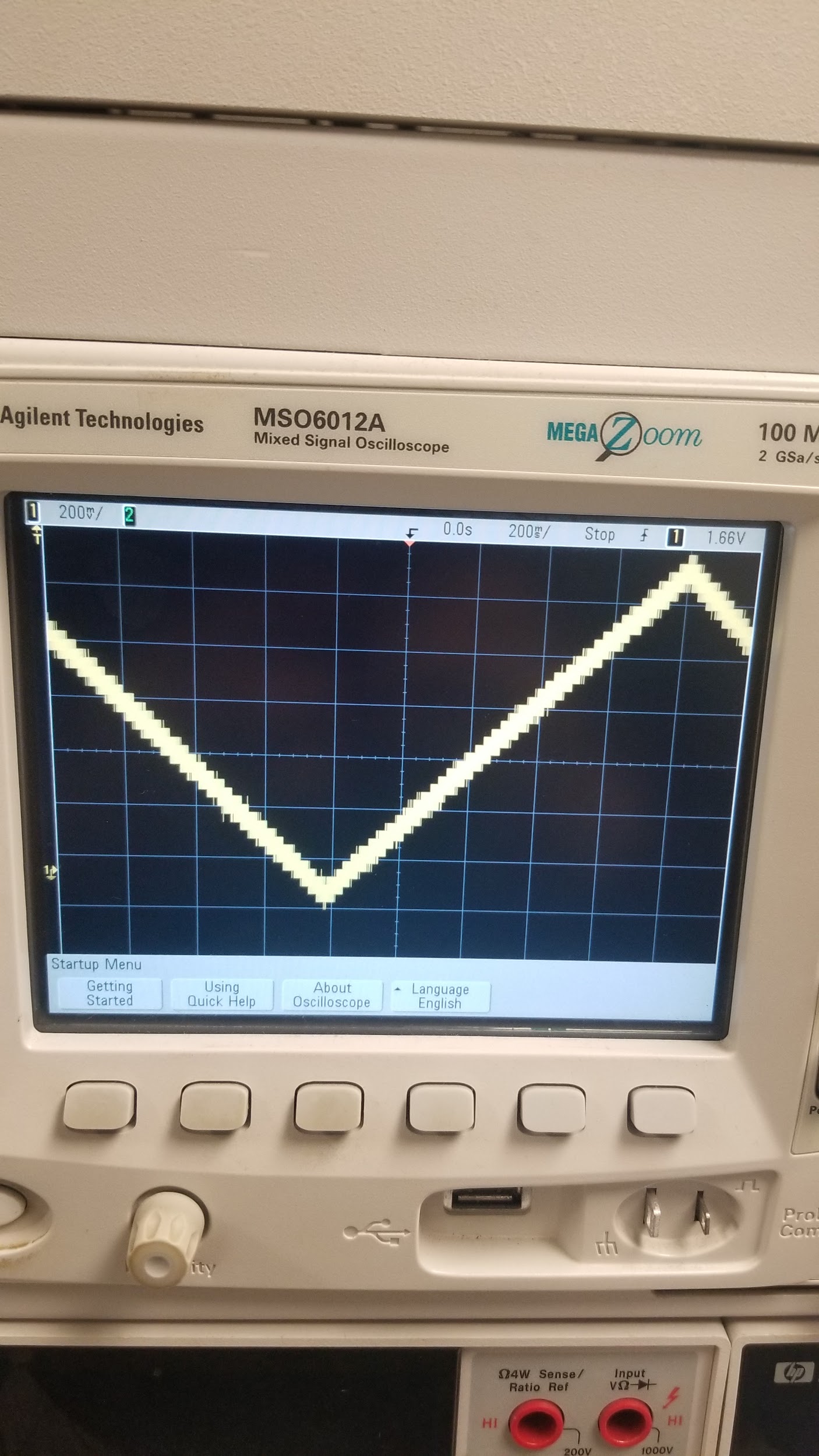
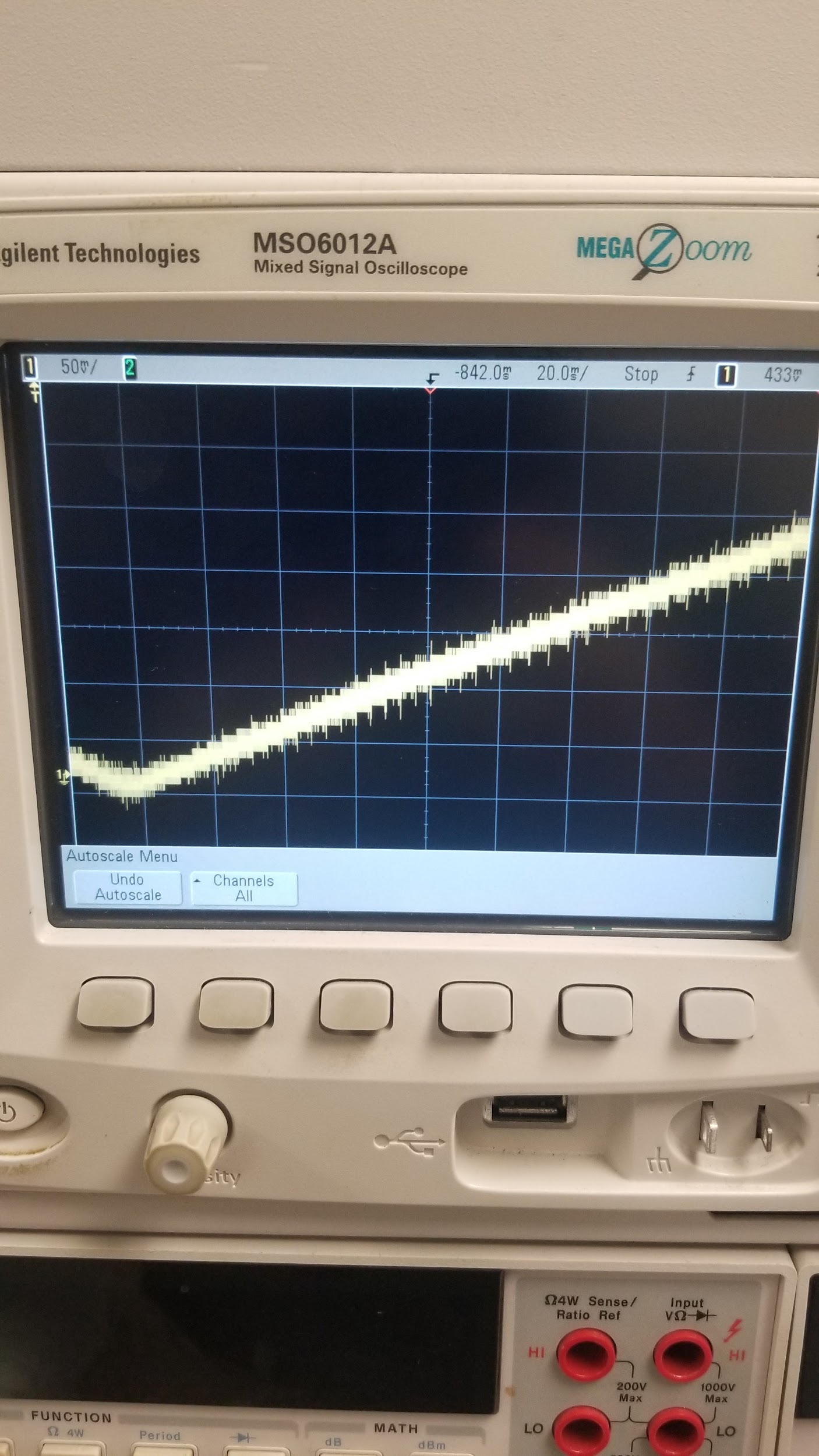
The X values are the delay inputs that are stored as variables. X3 is the oldest value, followed by X2, and X1 which is always the current input value. The value of the nulling frequency is determined by the following equation:

*nullingFV = 2\*PI\*F \* (delay T in s)*  [Eq: 2]

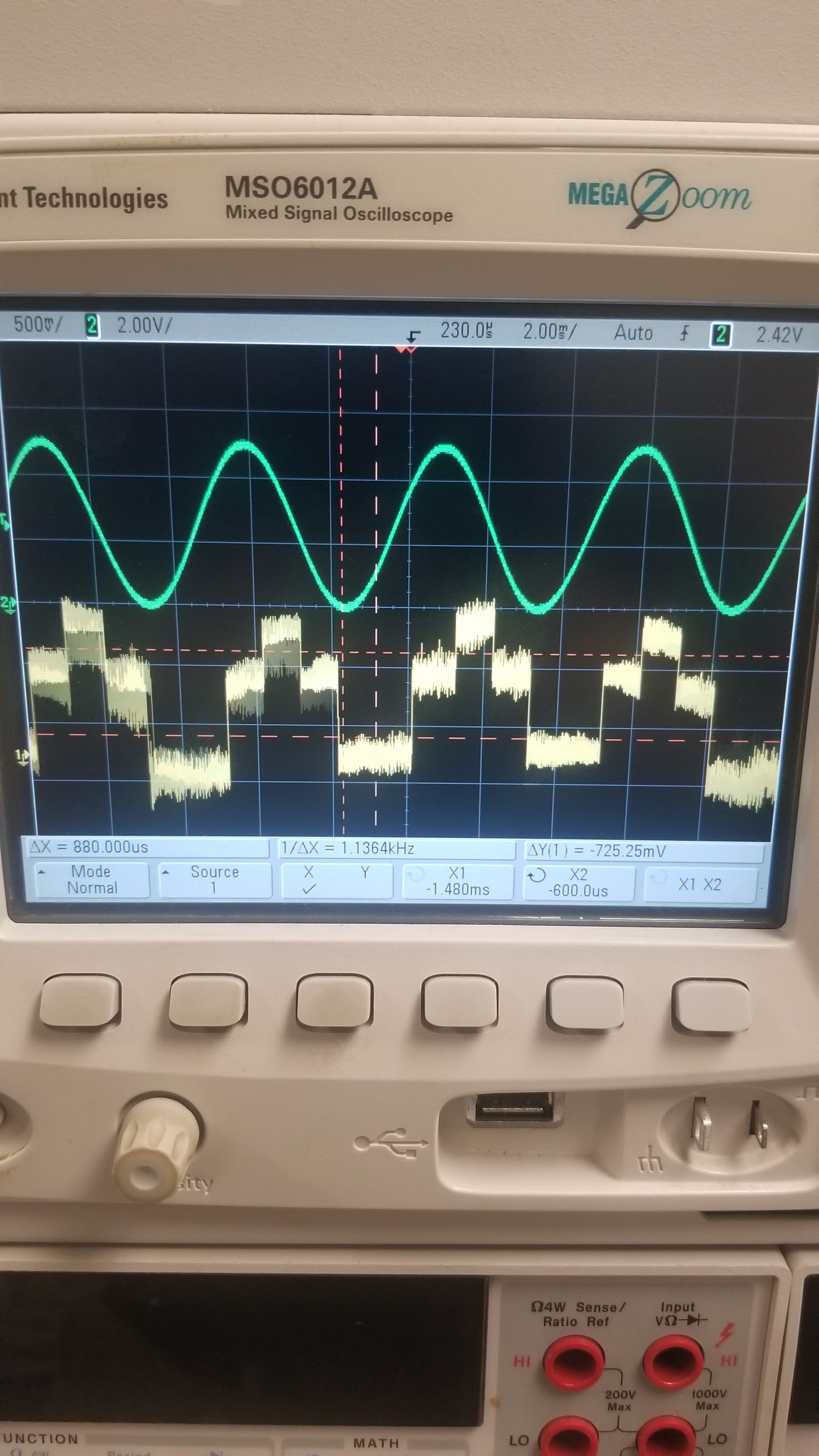
The team used a sine wave which went from 0 to 5V using the arbitrary waveform generator as the input. We ran the input into the arduino through an analog pin A14. This is the analog to digital convertor in the circuit. Using a 1 ms interrupt time and not yet implementing the nulling filter, we determined the actual delay between the input and output as shown in Figure X. This value was 1.136 ms, which we then input as the delay time shown in Equation 2. After implementing the nulling filter using these values, the resulting output is shown in the video #1 attached to this memo. 



**Figure 2.** A) The voltage A) and offset B) required to generate a sine wave



**Figure 3.** Triangle wave through the 1 ms interrupt



**Figure 4.** The green line represents the input and the yellow is the output. Displayed between the two red vertical bars is the time delay, 1.136 ms.

**Appendix:**

The nulling filter equation: nullingFV = 2\*PI\*F \* (delay T in s)

The equation = X1 - 2\*cos(nullingFV)\*X2\*X3

F = 200 Hz

X3 = X2

X2 = X1

X1 = analog pin input

For the first interrupt, X1=X2=X3=0V, but each of the X values are subsequently delayed by 1 interrupt instance which is 1 ms.

Arduino Code:

Part 1

#include "TimerOne.h"

#include <Wire.h>

#include <Adafruit\_MCP4725.h>

Adafruit\_MCP4725 dac;

//storage variables

boolean toggle0 = 0;

uint32\_t counter;

const int COUNTEND = 1023;

const int COUNTBEG = 0;

void setup(void) {

noInterrupts();//stop interrupts

//pinMode(13,OUTPUT);

interrupts();//allow interrupts

TWBR = 12;// 400khz

Timer1.initialize(1000);

Timer1.attachInterrupt(Blink);

counter = 0;

Serial.begin(9600);

Serial.println("Hello!");

// For Adafruit MCP4725A1 the address is 0x62 (default) or 0x63 (ADDR pin tied to VCC)

// For MCP4725A0 the address is 0x60 or 0x61

// For MCP4725A2 the address is 0x64 or 0x65

dac.begin(0x62);

}

void Blink(){

digitalWrite(13, digitalRead(13) ^ 1);//Toggle pin 10 for blink

if (!toggle0 && counter < COUNTEND) {

counter++;

}

else if (toggle0 && counter > COUNTBEG) {

counter--;

}

if (!toggle0 && counter >= COUNTEND) {

toggle0 = 1;

} else if (toggle0 && counter <= COUNTBEG) {

toggle0 = 0;

}

}

void loop(void) {

dac.setVoltage(counter, false);

}

Part 2

#include "TimerOne.h"

#include <Wire.h>

#include <Adafruit\_MCP4725.h>

/\*

\* AVG Settings

\* Freq = 200 Hz

\* Amp = 2.5

\* Offset +1.25

\*/

Adafruit\_MCP4725 dac;

//storage variables

const int F = 200;//Hz

const int interuptTime = 1000;//us

const int HzDiff = 1136;

//const int nullingFV = 2\*PI\*F\*(interuptTime\*10^-6);

const int nullingFV = 2\*PI\*F\*(HzDiff\*10^-6);

long x1;

long x2;

long x3;

long y;

int analogPin = A14; // potentiometer wiper (middle terminal) connected to analog pin 3

// outside leads to ground and +5V

int val = 0; // variable to store the value read

void setup(void) {

noInterrupts();//stop interrupts

//pinMode(13,OUTPUT);

interrupts();//allow interrupts

TWBR = 12;// 400khz

Timer1.initialize(interuptTime);

Timer1.attachInterrupt(Blink);

Serial.begin(9600);

Serial.println("Hello!");

// For Adafruit MCP4725A1 the address is 0x62 (default) or 0x63 (ADDR pin tied to VCC)

// For MCP4725A0 the address is 0x60 or 0x61

// For MCP4725A2 the address is 0x64 or 0x65

dac.begin(0x62);

}

void Blink(){

SetTheVoltage();

}

void SetTheVoltage(){

val = analogRead(analogPin); // read the input pin

//TODO: Convert Offset to Digital 10 Bit, Remove Offset, Convert to 12 Bit, add at the end

x3=x2;

x2=x1;

x1=val;

y=x1-2\*cos(nullingFV)\*x2+x3;

//dac.setVoltage(val, false);

}

void loop(void) {

//sSerial.println(y);

//dac.setVoltage(val, false);//withoutFilter

dac.setVoltage(y, false);//withFilter

}