Lab 0 - Introduction to Microcontrollers and Lab Tools

Part 1 - Hello World

Goals:

- Blink LED
    - Blink LED on board
    - Route to I/O look at with Analog Discover
- Output to Serial

Tools and Hardware:

- *Max32 or Arduino Board*
- *Arduino IDE*
- *Analog Discovery 1 or 2*

1. Getting Started

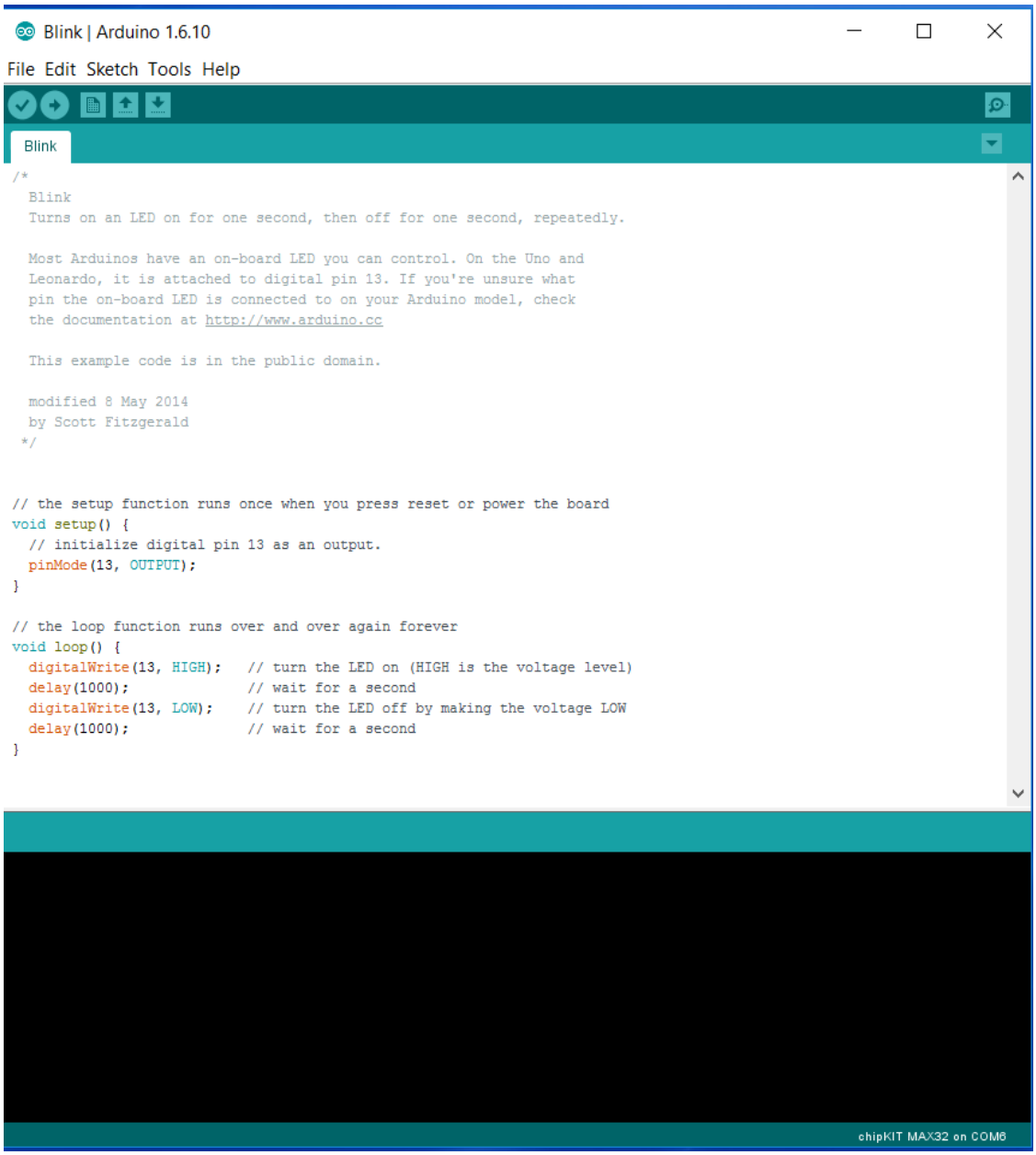
Download and Install the Arduino IDE software
https://www.arduino.cc/en/Main/Software

If you are using the MAX32 follow the additional steps in the **Setting Up Arduino IDE with chipKIT** document

2. Example Blink Code

Next we are going to set up a basic example code that is provided with the Arduino IDE software to ensure our software is working properly.

-In the Arduino IDE select **File>Examples>01.Basics>Blink**
-The following code should be displayed:



Infinite Loop

Examining the code from the image above it is important to notice some key parts when programming in microcontrollers. The first feature is the loop function, this can often be referred to as a **infinite loop**. Infinite loops are where the code inside the loop will run continuously until outward interference .

When using Arduino we simply call a loop function that doesn't return anything. Other common ways to write infinite loops are the following:

```
while(1)
{
    //Anything inside these brackets will run forever
}
```

```
for(;;)
{
  //Anything inside these brackets will run forever
}
```

While some methods of writing infinite loops are more commonplace there is a variety of different methods used.

For more information:
http://www.learningaboutelectronics.com/Articles/While-(1)-embedded-C.php

I/O - Input/Output

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

The next topic we are going to discuss is using the I/O of the Arduino board. Looking at the image above in the setup function we can see we are calling a function called pinMode. This function is used to determine what direction the I/O is set at that pin.

Above in the blink code we can see that pin 13 is assigned to an output. If we wanted to make pin 5 on our board a input it would look like something below

```
pinMode(5, INPUT);
```

As we can see the pinMode functions parameters are in the order of pin number and then direction, INPUT or OUTPUT.

Going back to the blink example code:

```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);              // wait for a second
}
```

We can see that inside the loop() function we assign pin 13 to a HIGH value and then we assign pin 13 to a LOW value using the digitalWrite() function, with a one second delay in between.

For Inputs we use the digitalRead() function to receive either a HIGH or LOW value. See the following example below.

```
int pin_5_value;

void setup() {
  // put your setup code here, to run once:
  pinMode(5, INPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  pin_5_value = digitalRead(5); //Returns a HIGH (1) or LOW (0)
}
```

For more information:
https://www.arduino.cc/en/Reference/PinMode
https://www.arduino.cc/en/Reference/DigitalWrite
https://www.arduino.cc/en/Reference/DigitalRead
https://www.arduino.cc/en/Reference/Constants

3. Program Device

Before programming the Arduino board we need to verify, compile, our code. As seen as the image below.

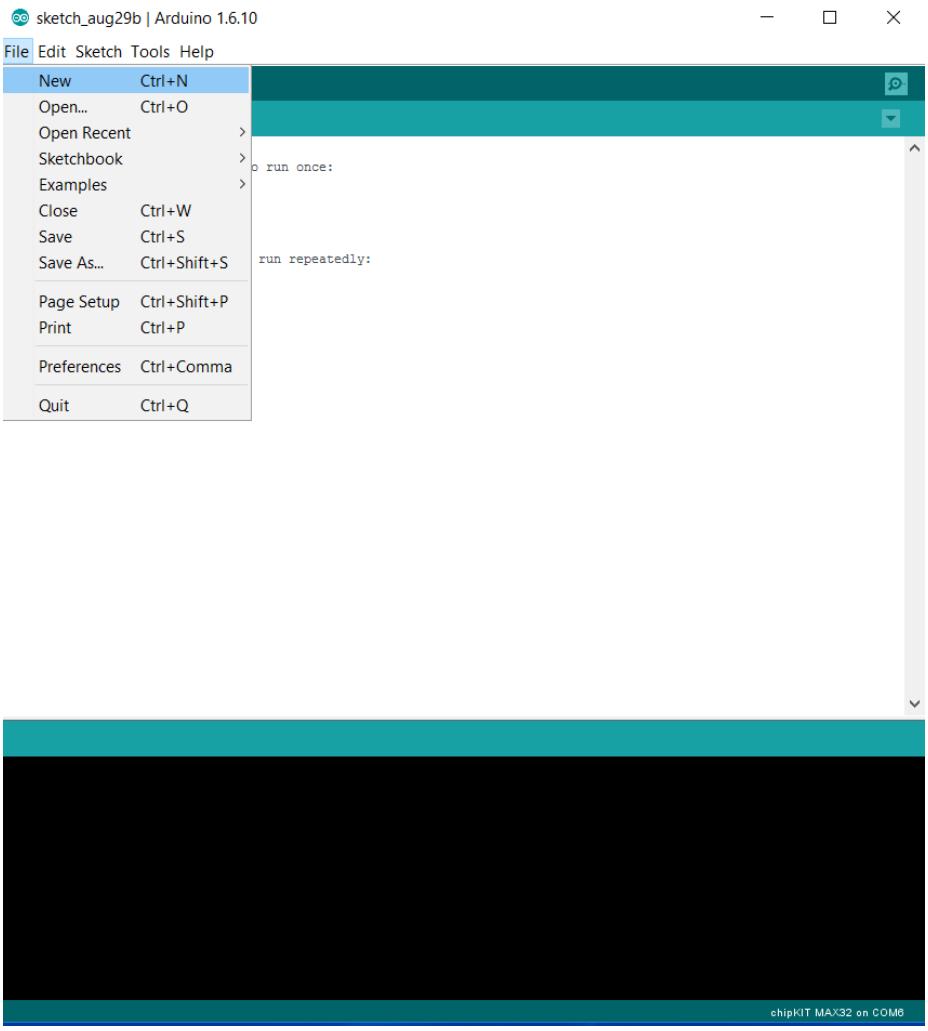
After verifying, compiling, the code we will upload by selecting the arrow button next to the verify button. After clicking that the following results should be displayed.

After successfully uploading, verify the that led on the board is blinking. As seen in the image below

Note: If device did not successfully program do not move forward.

4. Modify Code (Serial Monitor and add Output to pin)

Now that we have successfully programmed the device we are going to create a new file.



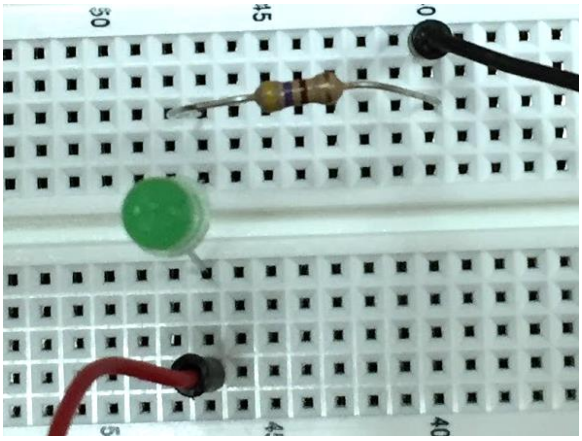
Add the following code:

```
void setup() {
  // put your setup code here, to run once:
  pinMode(11, OUTPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(11, HIGH);
  delay(1000);
  digitalWrite(11, LOW);
  delay(1000);
}
```

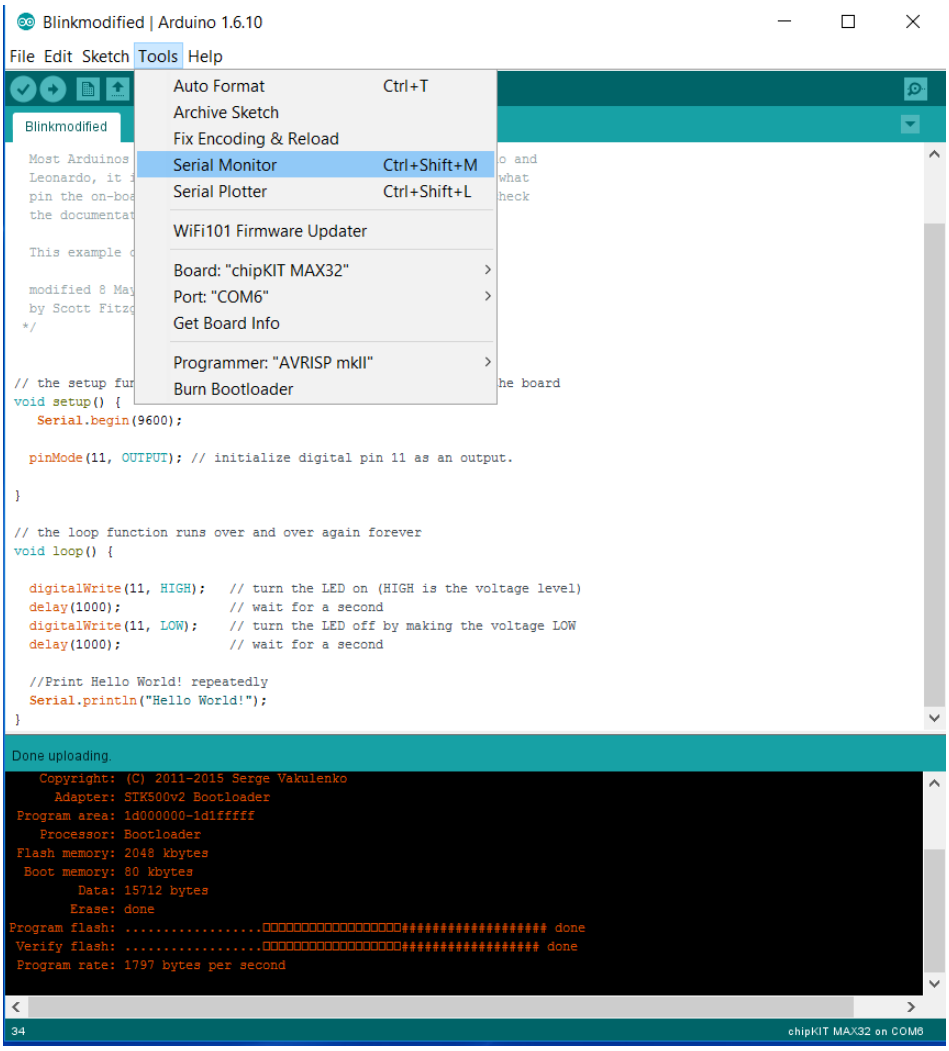
Next we are going to add the Serial output to the code. Like so

```
void setup() {
  // put your setup code here, to run once:
  pinMode(11, OUTPUT);

  Serial.begin(9600); //Initialize the Serial output to a baud rate of 9600
}

void loop() {
  // put your main code here, to run repeatedly:
  digitalWrite(11, HIGH);
  delay(1000);
  digitalWrite(11, LOW);
  delay(1000);

  //Prints Hello World! repeatedly
  Serial.println("Hello World!");
}
```

Now wire a led to a breadboard with a current limiting resistor, preferably a 330 ohm, in series with the led. Then connecting pin 11 of the Arduino board to the anode side of the led, and connecting the current limiting resistor to ground on the Arduino board. *See image below*



Verify and Upload the code to the Arduino board. Then open the serial monitor as seen in the image below.



Click Serial Monitor and the following results should be displayed.

We should also see the led blinking on the board.

For more information:
https://www.arduino.cc/en/Reference/Serial
https://www.arduino.cc/en/Serial/Println

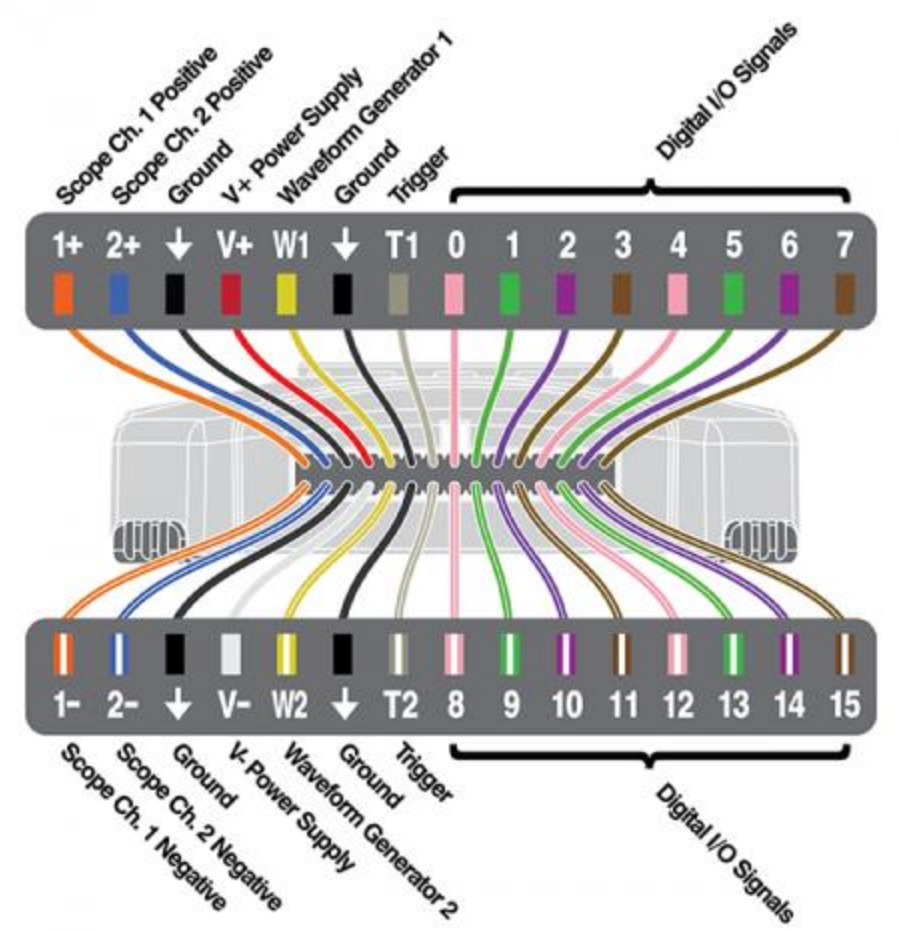
5. Connect Analog Discovery

The Analog Discovery can be a extremely useful debugging tool utilized in future labs and projects. This section we will examine one of its basic functions like checking voltage.

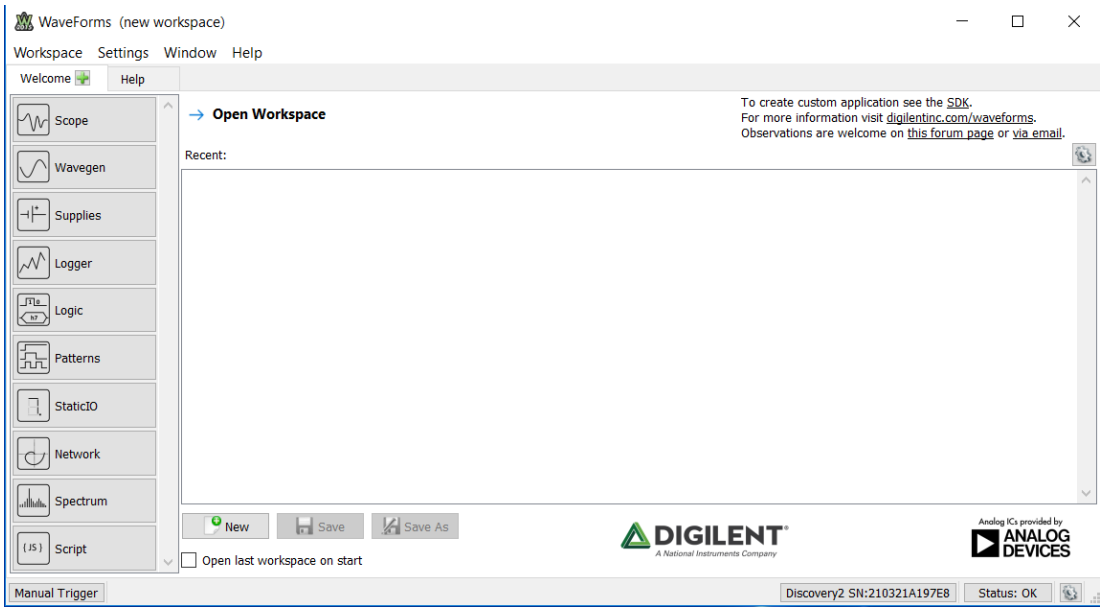
First download and install Waveforms 2015 from the link below:
http://store.digilentinc.com/waveforms-2015-download-only/

Once installed connect your Analog Discovery to the computer and open Waveforms 2015.

From there connect the positive and negative leads of channel one, orange and orange with white stripe, across the current limiting resistor on the circuit used in the previous step.

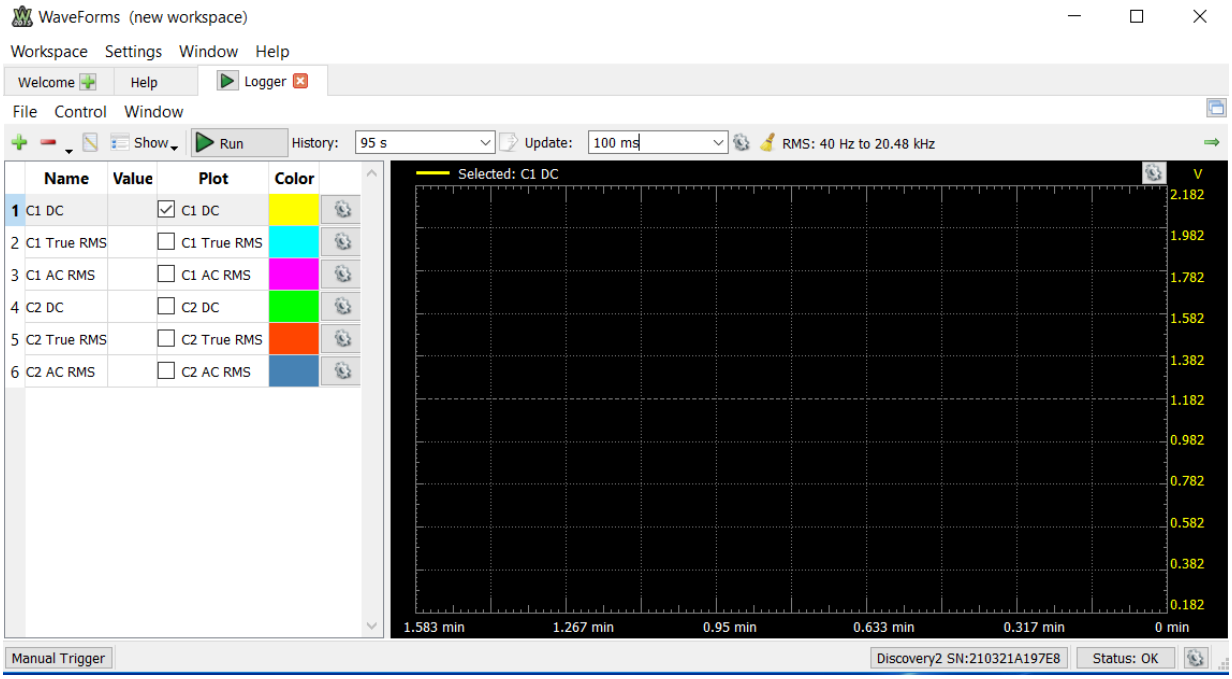


Going back to the Waveforms software the image below should be displayed on your screen.

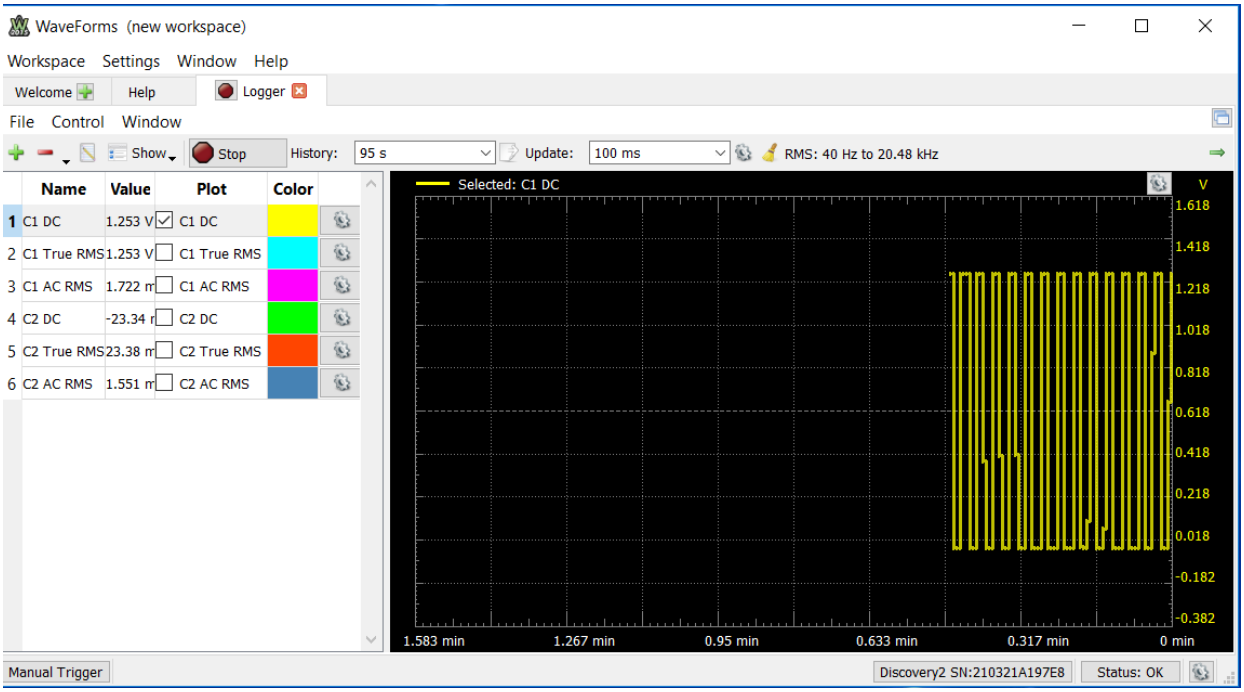


Logger (Voltmeter)

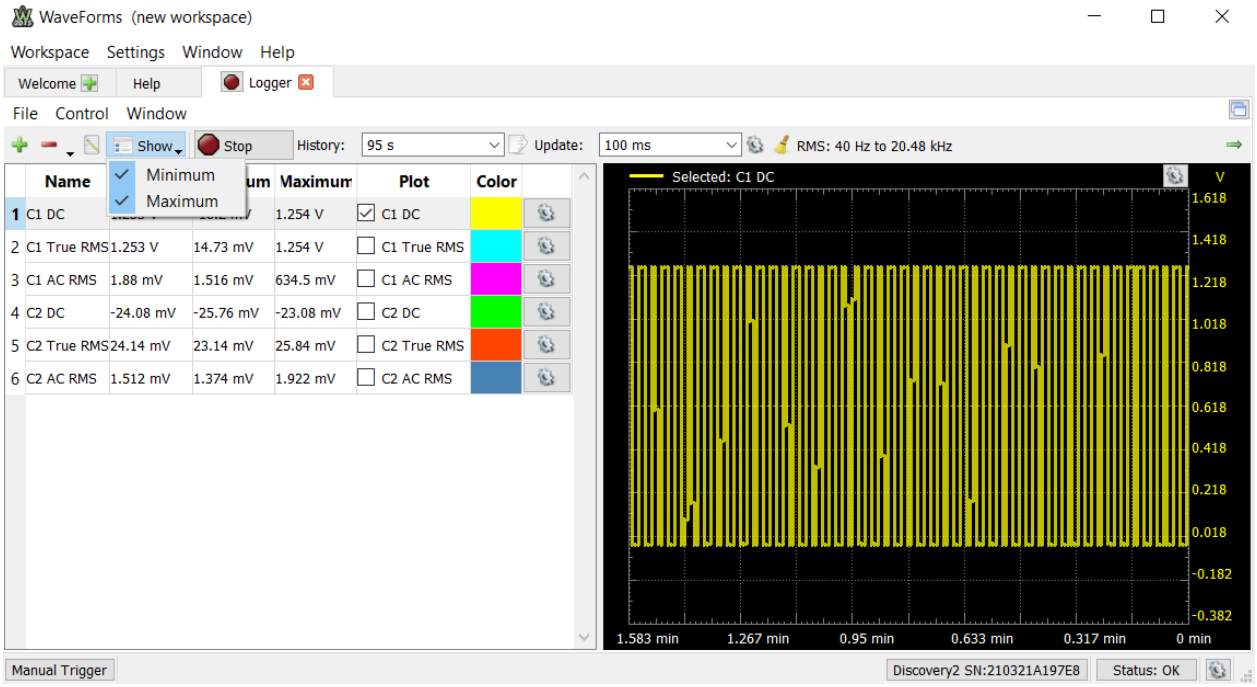
Next to select the Logger tab on the side. A new space should appeared. *See image below.*



Change the settings to match the image above. Then select the Run button on the screen. A graph of the voltages should display. *See image below.*



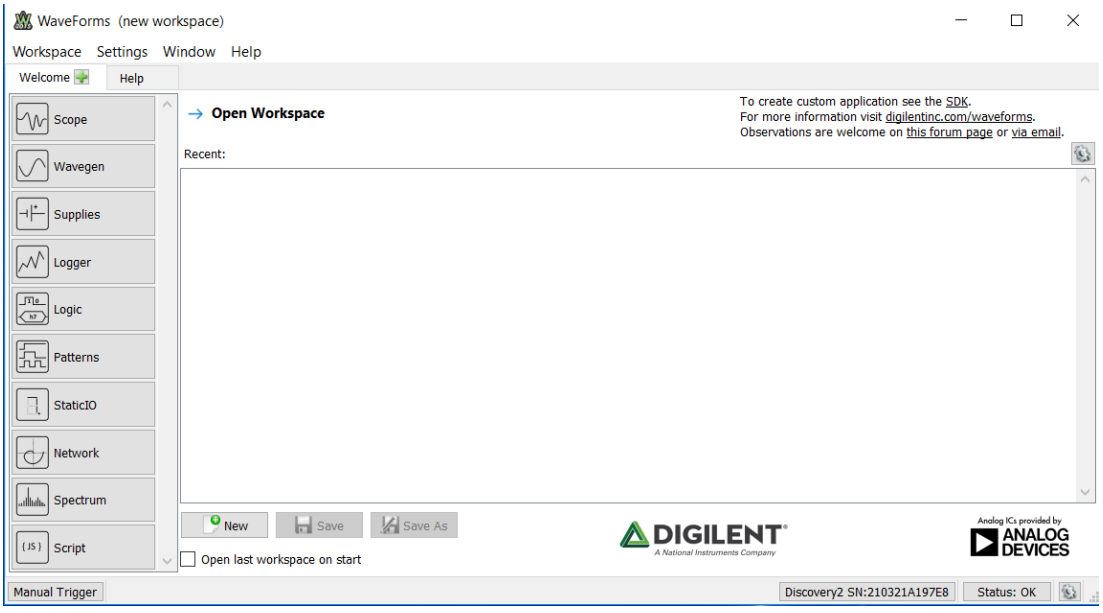
Next select the show tab that is located next to the stop button. Select the Minimum and Maximum tabs to display the Maximum and Minimum voltage. *See image below.*



Record the values, and then move the positive channel one lead, solid orange, to anode side of the led. Then record the maximum and minimum voltages.

Oscilloscope

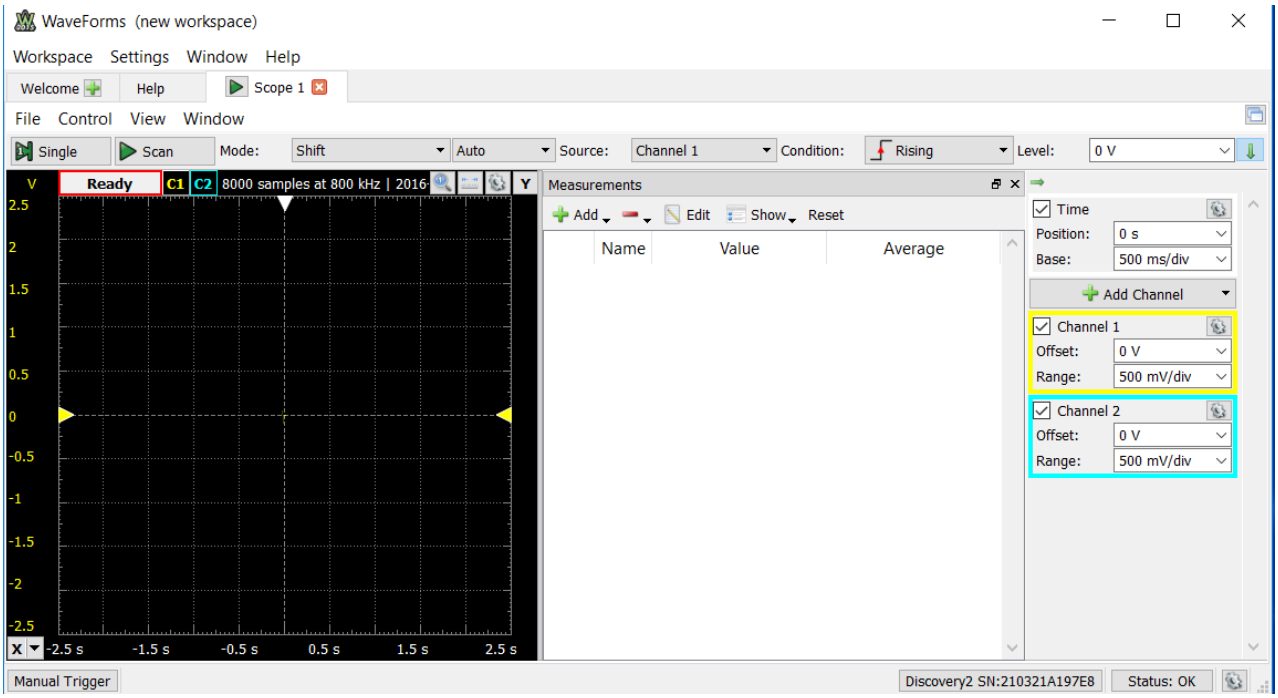
Go back to the home screen for Waveforms, and place the channel one probe, orange lead, across the resistor. *See image below.*



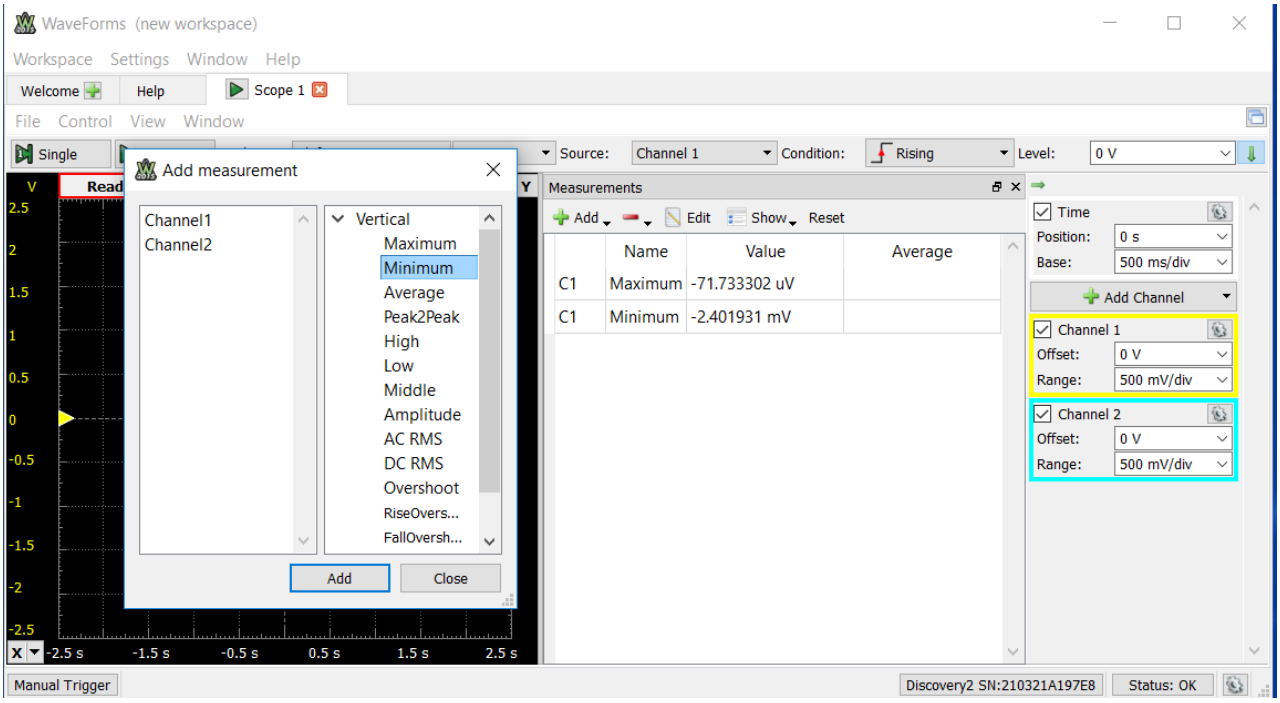
Select the Scope tab on the side. The following screen should appear. *See image below.*

Adjust the settings to match the image above. Next select the View > Measurements

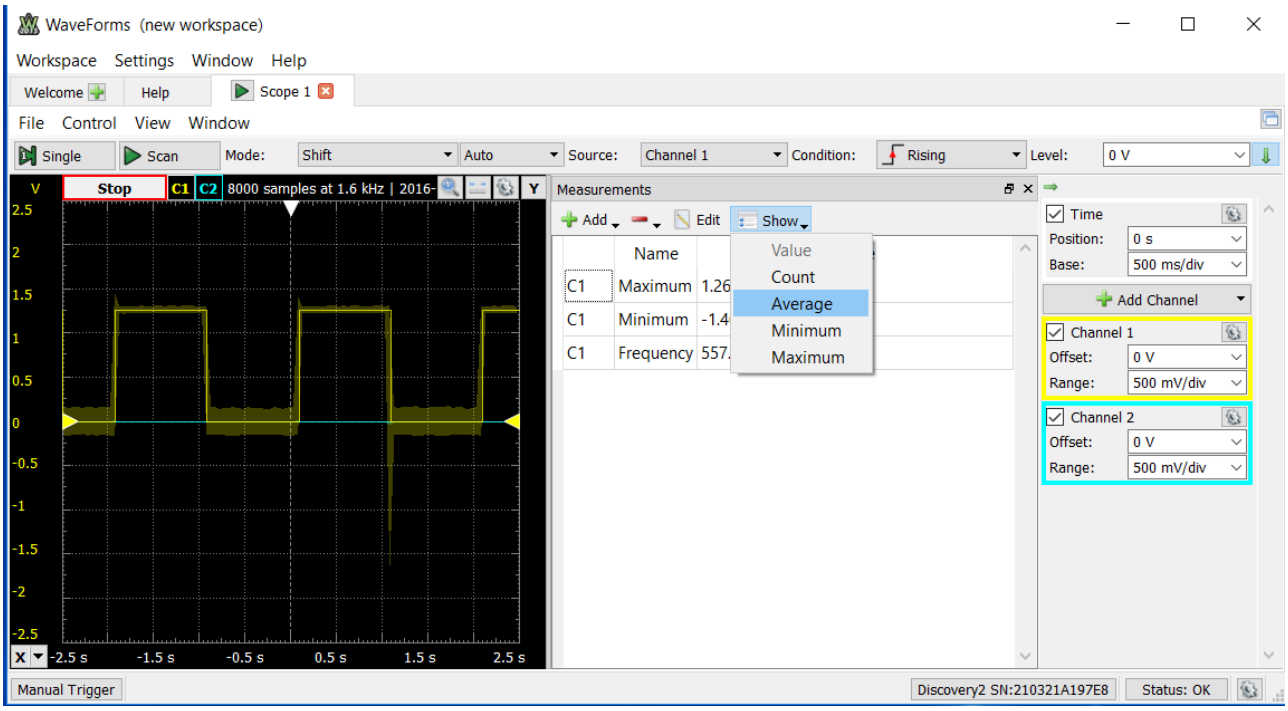
A new window should appear. *See image below*



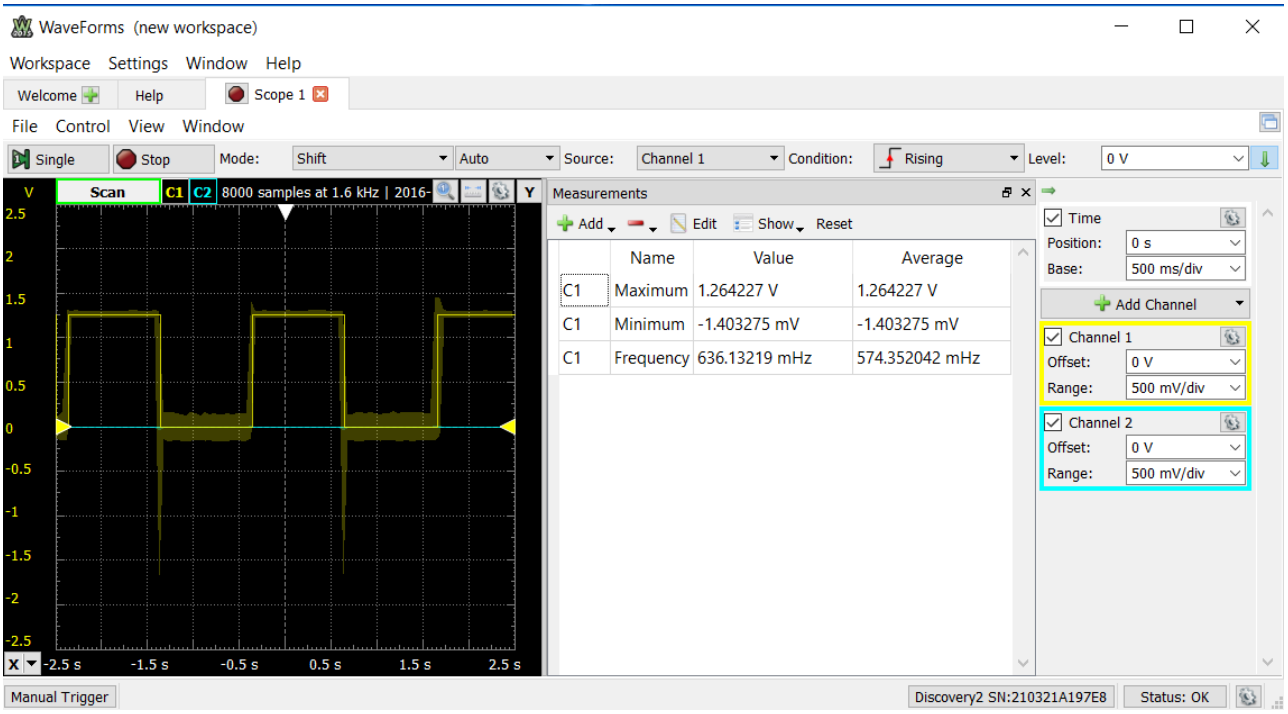
Next select the Scan button, then select the Add button. A new window should appear and a square wave should appear across the screen. *See image below*



Then select Maximum and Minimum in the Vertical section, as well as Frequency in the Horizontal section. Then select the Show tab and Average. *See image below*



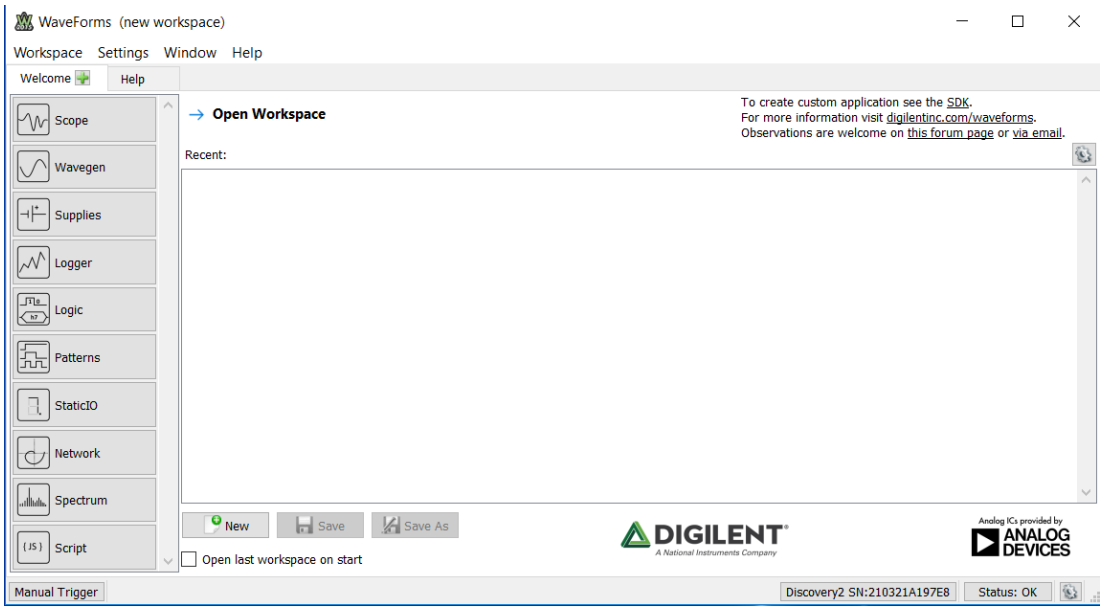
A Average column will appear then select the Scan button. Record the values displayed. *See image below*



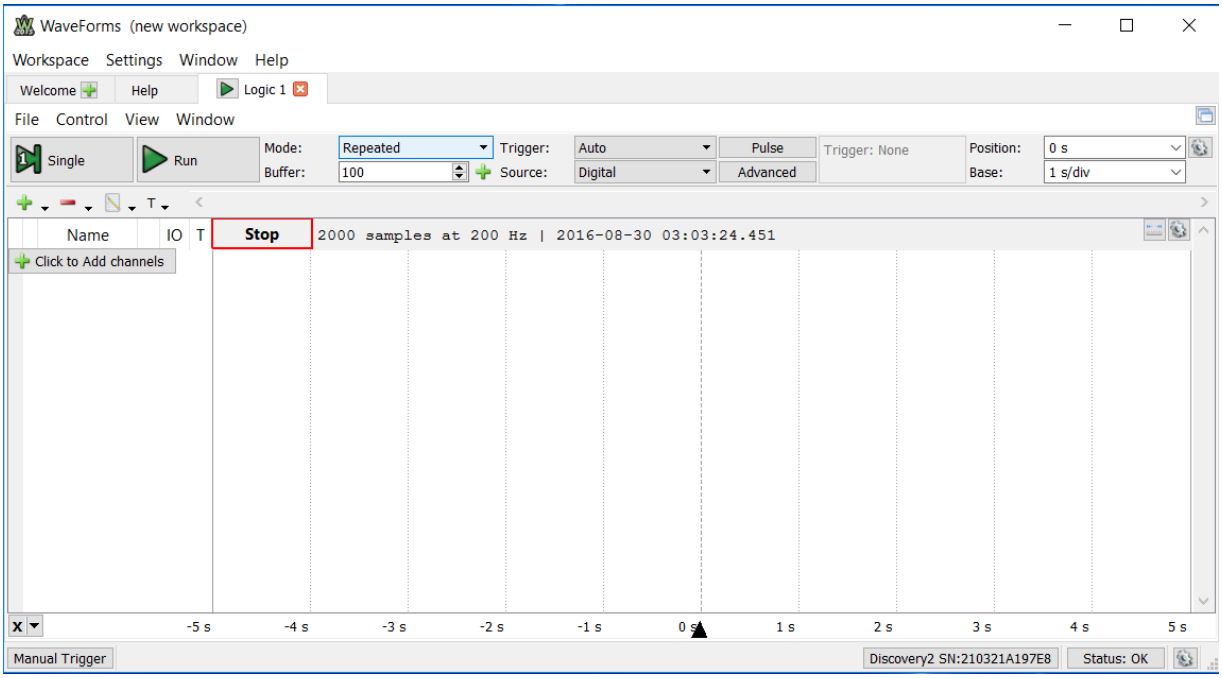
Adjust the channel one probe, orange lead at the anode side of the led and record the data.

Logic Analyzer

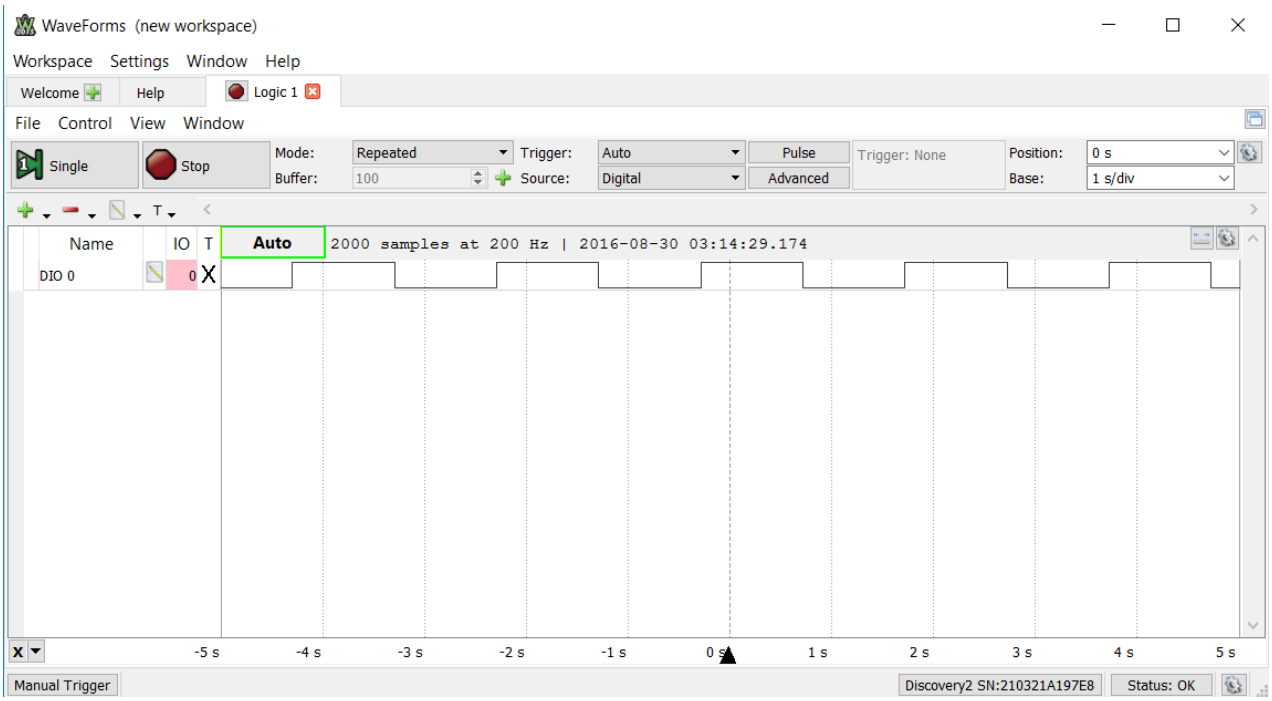
Go back to the home screen for Waveforms, and place the digital io signal 0, pink lead, to the anode side of the LED. *See image below.*



Select the Logic tab on the side. The following window should appear. *See image below*

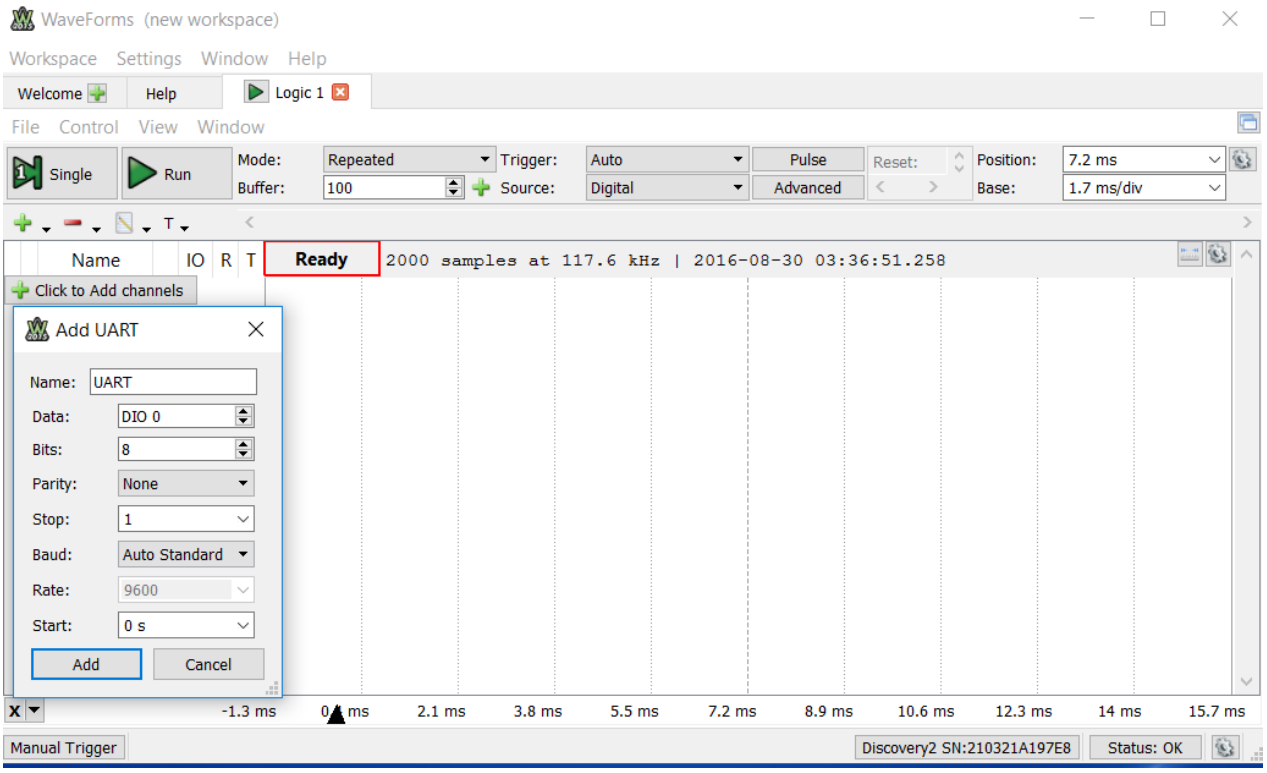


Adjust the settings so they match the image above. Then select “Click to Add channels” > Signal > DIO 0.Then select Run and the following image should appear in the window.

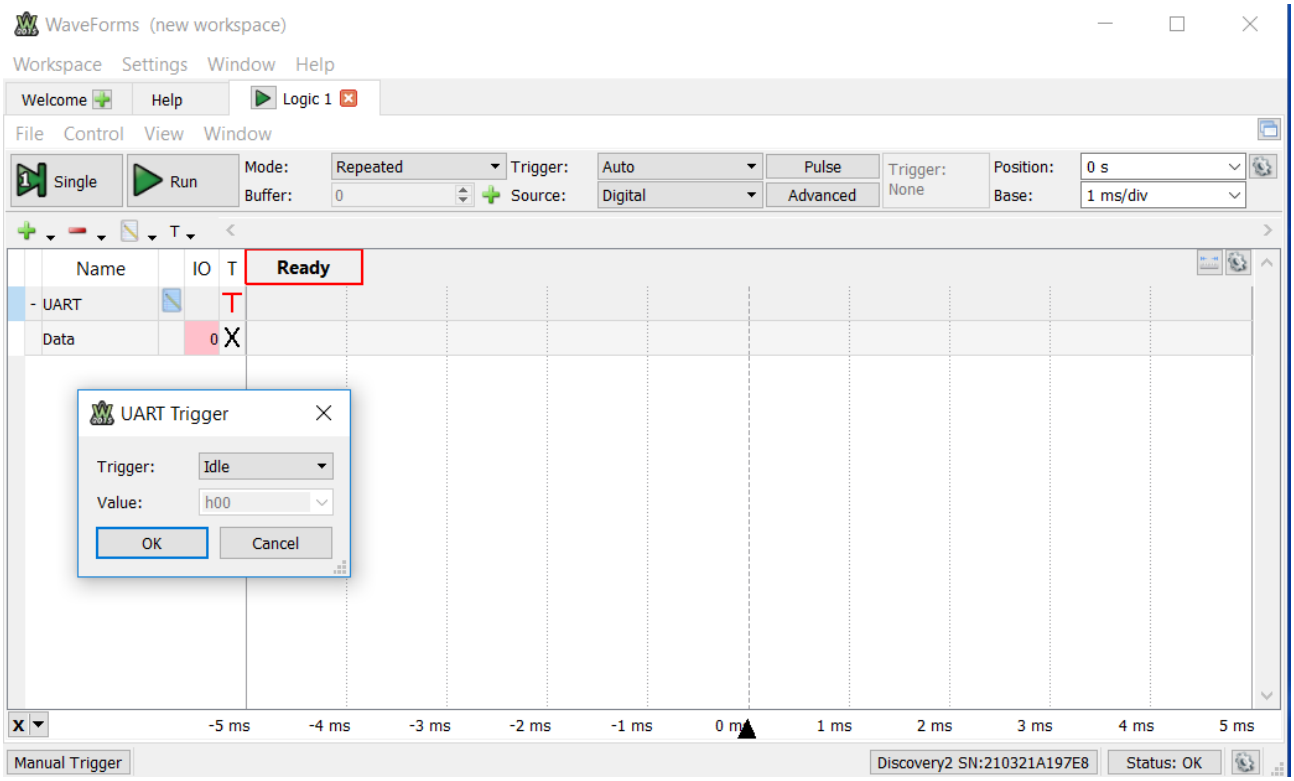


Bus Analyzer

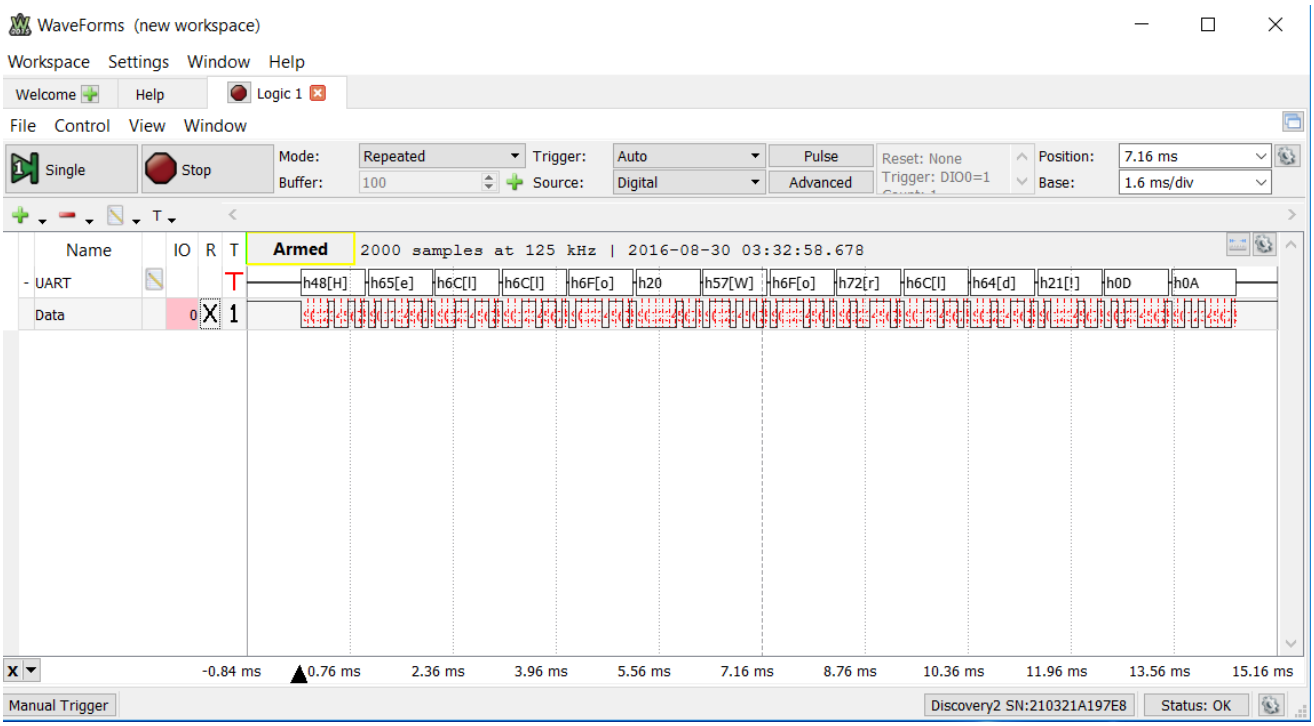
Stay in the Logic window. Move the digital io signal 0 to the TX0 pin on the Arduino board. Next select the add "Click to Add channels" > UART. Set the settings to the same as the image below.



Click Add. Then Select the red T. This will open the UART Trigger. Set it to the image seen below.



Select OK. Then select the RUN button. Drag the black triangle, located near the bottom of the window, drag left till all the data is displayed, and scroll till full message fits on the screen. Final output should match the image seen below.



Change the message to display your name and record the information for lab report.

For more information:
https://reference.digilentinc.com/reference/instrumentation/analog-discovery-2/start
http://store.digilentinc.com/waveforms-2015-download-only/

6. Practical Exercise

Connect a push button to your Arduino board that when pressed lights a led using the Arduino. Also output the status of the push button using the Serial Monitor. Measure the voltages at the led and push button. Capture the data for lab report and show live demo to lab professor.

CONCEPT REVIEW:

- *Infinite loop*
- *I/O*
- *Basic Programming Concept*

Part 2 - ADC

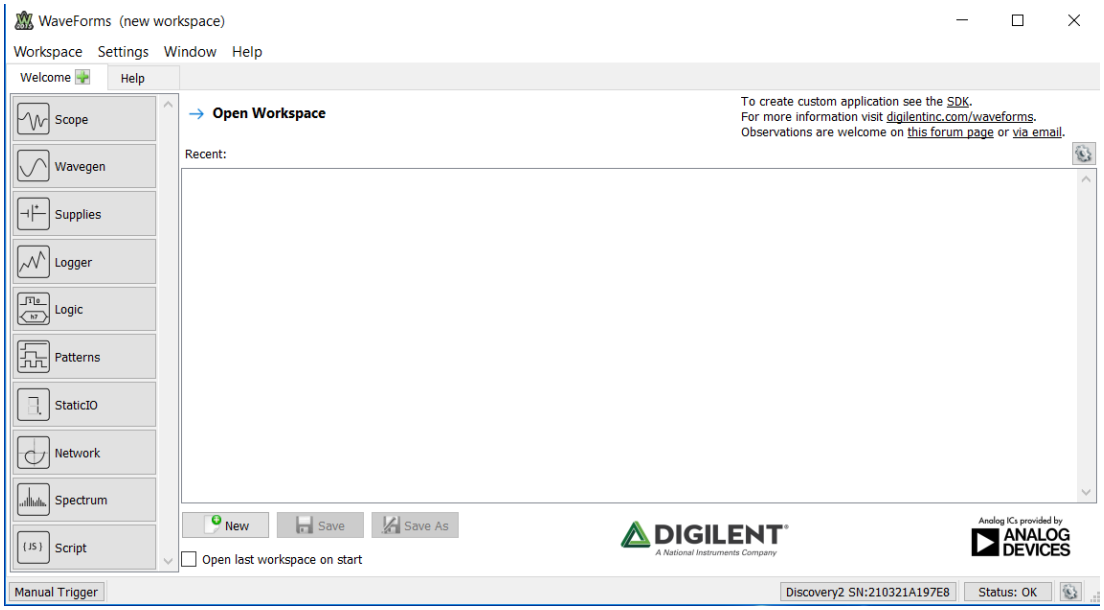
Many microcontrollers contain a built in Analog to Digital Converter (ADC) that are used to convert analog signals that can be used for digital system.

First we are going to examine a additional function of the Analog Discovery. The Arbitrary Wave Generator (AWG), AWGs allow us to generate a number of different types of signals and with these signals we can create test cases for our code to run on.

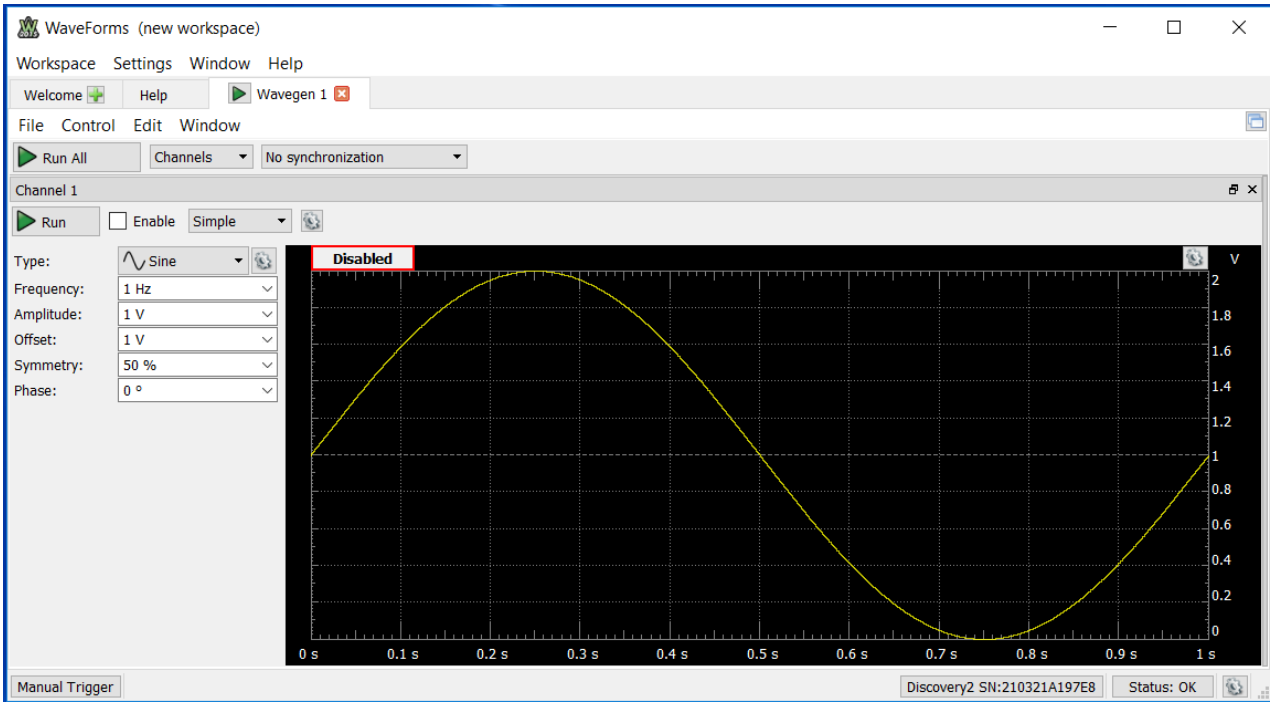
Arbitrary Wave Generator (AWG)

Connect positive channel 1, orange lead, to the wave 1, yellow wire, the negative channel 1 to ground lead on the Arduino board. Next connect a wire for positive channel 1 and wave 1 signal to the analog in 0, A0, on the arduino board.

Go back to the home screen for Waveforms. *See image below*

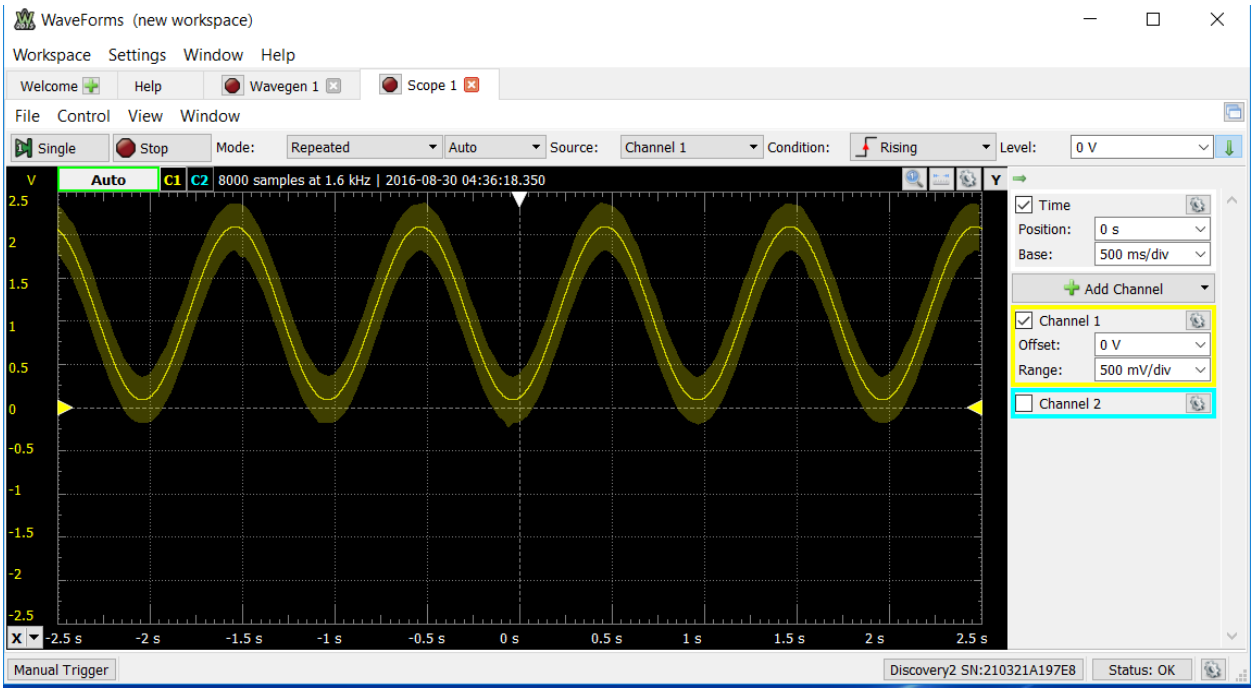


Next select the Wavegen tab. Adjust the settings so they match the image below.

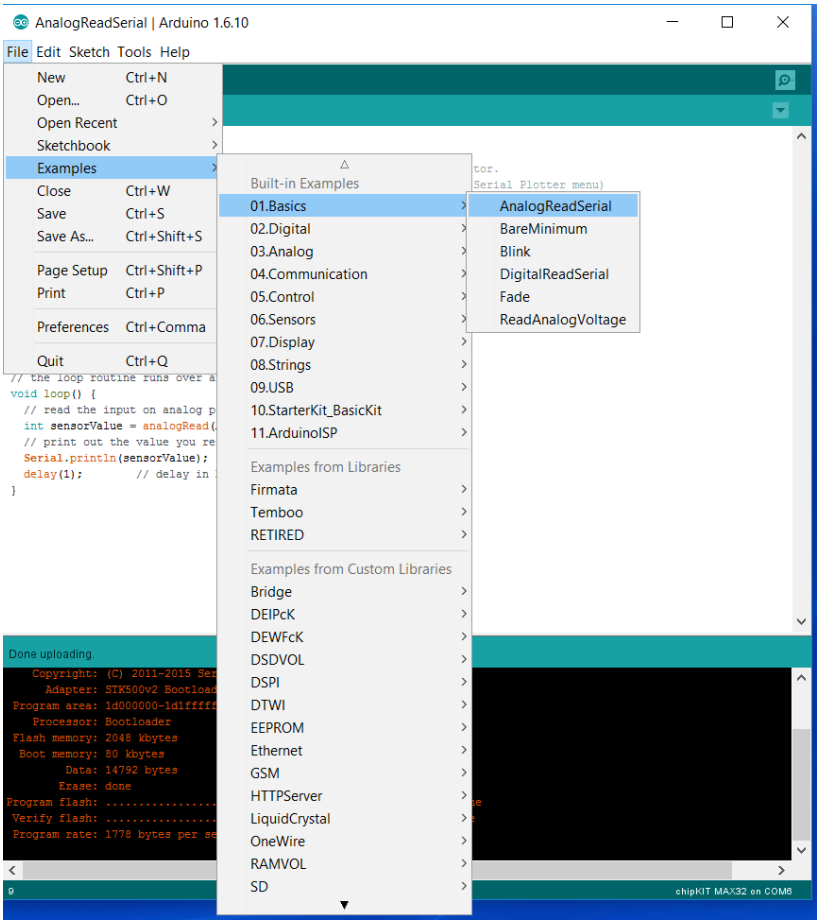


Next select the Welcome tab at the top right of the waveform window and then select the oscilloscope tab. Adjust the settings of Channel on so that they match the image below.

Now go back to the Wavegen 1 tab and RUN the AWG, then head back to the Scope 1 tab and select RUN. The following image should be displayed on the oscilloscope. *See image below*



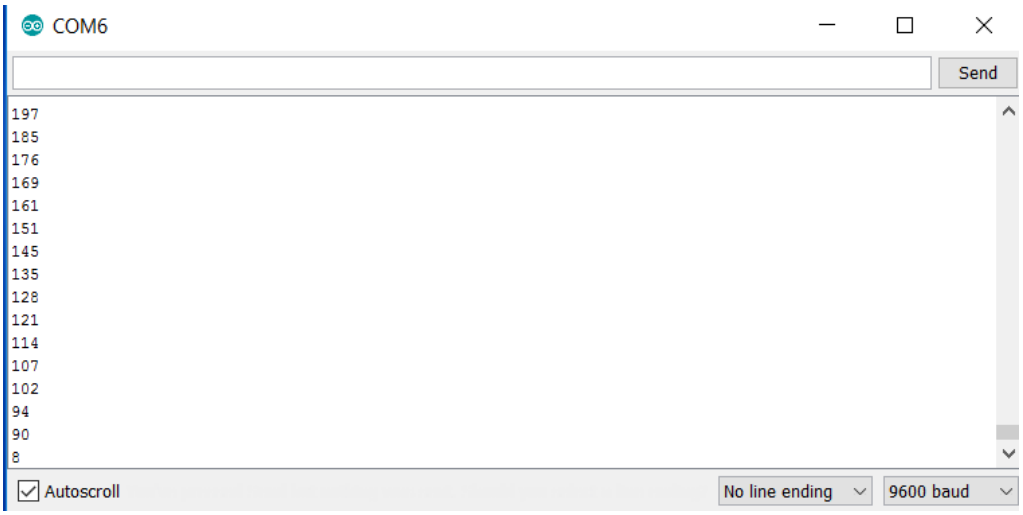
Now that we have confirmed the AWG is working properly. We are going to load the one of the default sketches to the Arduino code. *See the image below*



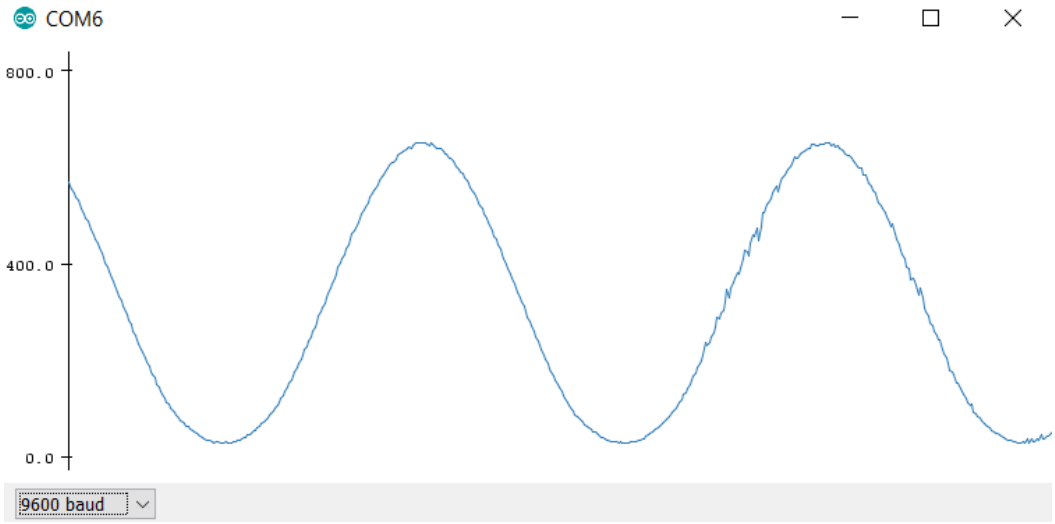
Examining the code below we can see that the analogRead function returns a integer value from 0 to 1023. That integer is then displayed through the serial monitor.

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);        // delay in between reads for stability
}
```

Next verify and upload the code then open up a serial monitor.



As we can see the value display will increase than decrease. Now close the serial monitor and open the serial plotter under the Tools tab > Serial Plotter. Something similar to the image below should display.



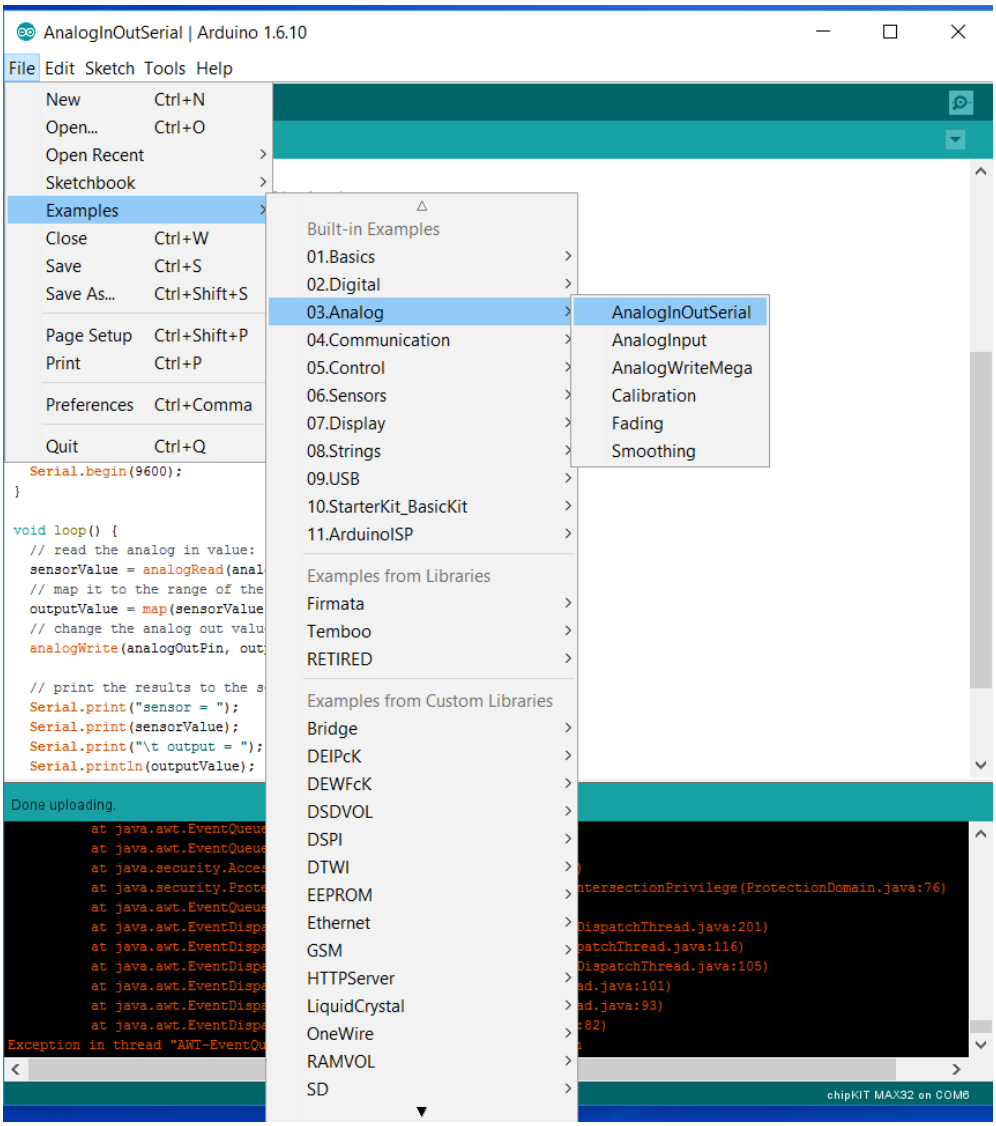
Play with another wave types in the AWG and record results for lab report.

For more information:
https://www.arduino.cc/en/Reference/AnalogRead

Part 3 - PWM

Pulse width modulation (PWM) is a techniques to generate a analog signal with a digital system. Building on the same circuit that was used in the previous part, we are going to add a LED to our breadboard with a current limiting resistor, around 330 ohm. Add positive channel 2, blue lead, to the anode side of the LED and negative channel 2, blue with white stripe, to ground. Next connect a wire from the anode side of the LED to pin 9 on the arduino board.

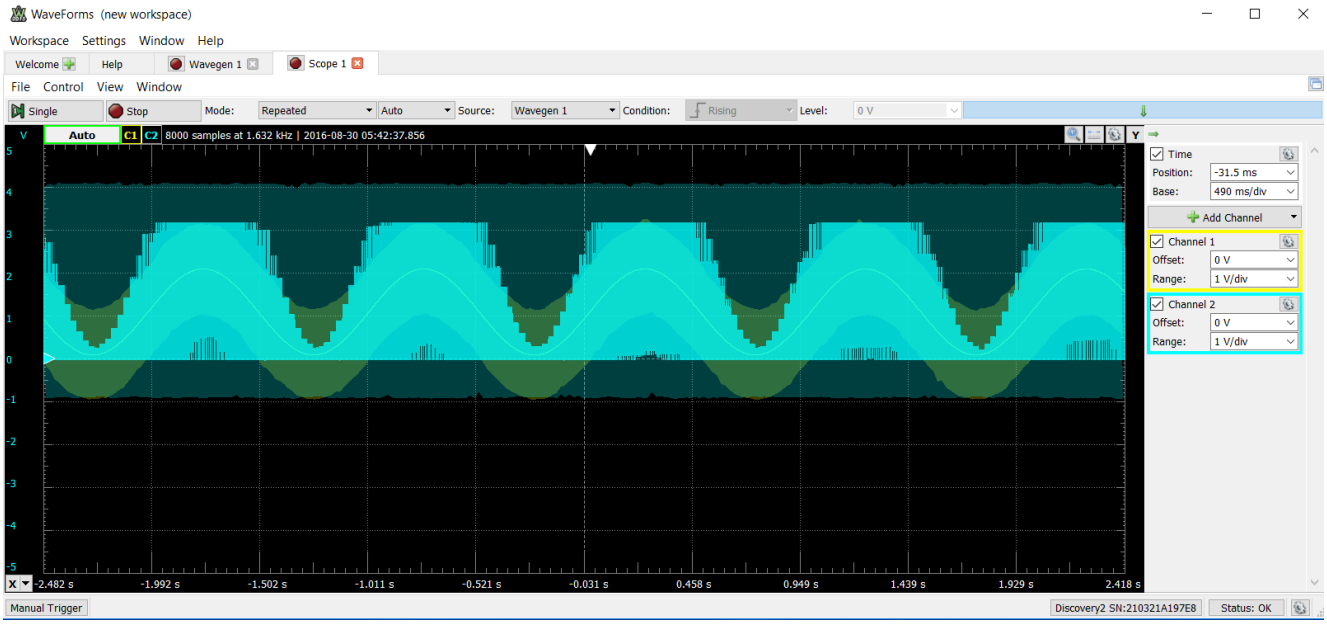
Next in the Arduino IDE open up the and open the AnalogInOutSerial file as seen in the image below.



Examining the code in the image below we can see a few new functions. One of the most notable ones for PWM is the analogWrite() function. The analogWrite() takes the PWM pin and then a value between 0 to 255. Unfortunately the analogRead() function returns a value between 0 to 1023. In order to make the value from the analogRead() be transferred to the analogWrite() we use the map() function. The map function allows the user to map from one number range to another. *See image below.*

```
const int analogInPin = A0;  // Analog input pin that the potentiometer is attached to
const int analogOutPin = 9; // Analog output pin that the LED is attached to

int sensorValue = 0;        // value read from the pot
int outputValue = 0;        // value output to the PWM (analog out)

void setup() {
  // initialize serial communications at 9600 bps:
  Serial.begin(9600);
}

void loop() {
  // read the analog in value:
  sensorValue = analogRead(analogInPin);
  // map it to the range of the analog out:
  outputValue = map(sensorValue, 0, 1023, 0, 255);
  // change the analog out value:
  analogWrite(analogOutPin, outputValue);

  // print the results to the serial monitor:
  Serial.print("sensor = ");
  Serial.print(sensorValue);
  Serial.print("\t output = ");
  Serial.println(outputValue);

  // wait 2 milliseconds before the next loop
  // for the analog-to-digital converter to settle
  // after the last reading:
  delay(2);
}
```

Upload and verify the code to the Arduino board. You should be able to see the light brighten and dim slowly. Now pull up the oscilloscope on the Analog Discovery and compare the signals in channel 1 and channel 2. They should look similar to the image below.



For more information:

https://www.arduino.cc/en/Tutorial/PWM
https://www.arduino.cc/en/Reference/AnalogWrite
https://www.arduino.cc/en/Reference/Map

Practical Exercise

Connect the photoresistor to a analog pin of the Arduino then connect a led to pwm pin on the Arduino board. Output the value from the photoresistor to the serial monitor as well as use pwm to change the brightness of the led. Record the input and outputs with the analog discover using the Oscilloscope function. Capture the data for lab report and show live demo to lab professor.

Part 4 - Applications

Choose one of the optional parts and demonstrate it in lab.

Part 5 - I^2C Communication

I^2C is a common communication protocol that is found in embedded systems. Used in sensors and communication between devices I^2C allows for many devices to share one communication bus and reduces the number of wires on a board.

The Arduino already contains the libraries and examples necessary for I^2C, The wire library and examples explain the use of I^2C in Arduino.

For this part in the lab you will work with a partner. One will select “master_writer” the other “slave_reciever” from the wire example. You and your partner will determine what data should be sent and display it on the Serial Monitor. Then connect the bus analyzer on the Analog Discovery and display the information that was sent on the bus analyzer between the 2 devices. Record relevant information and demonstrate for lab instructor.

For more information:
https://learn.sparkfun.com/tutorials/i2c
https://www.arduino.cc/en/Reference/Wire
http://www.allaboutcircuits.com/technical-articles/introduction-to-the-i2c-bus/

Part 7 - Project

Design a simple project that takes some analog input and provides an output. Consider the optional parts or different analog sensors use in the project.

For example:

- Light sensor that will display the amount of light in a room on a LCD and adjust the brightness the LCD.
- Take input from IR remote and move motor or servo.
- Take temperature from temperature sensor and display and update on LCD

Record information and demonstrate to lab instructor.

Part 8 - Wiping chipKIT MAX32 Bootloader

Note: This step should be done last. This step will lead into the Microchip lab.

Optional Parts:

Part 9 - Motor Controls

Complete circuits 8 and 12 of the SIK experiment guide using the chipKIT Max32

Links:

Circuit 8 Driving a Servo Motor
Circuit 12 Driving a Motor

Part 10 - Temperature Sensor

Complete circuit 7 of the SIK experiment guide using the chipKIT Max32

Links:

Circuit 7 Temperature Sensor

Part 11 - LCD

Complete circuit 15 of the SIK experiment guide using the chipKIT Max32

Links:

Circuit 15 Using an LCD

Part 12 - Relays

Complete circuit 13 of the SIK experiment guide using the chipKIT Max32

Links:

Circuit 13 Relays