UNIVERSITY OF ALASKA FAIRBANKS

COLLEGE OF ENGINEERING AND MINES
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

COURSE CODE	EE F102 F01 (CRN: 34544)		
COURSE NAME	INTRODUCTION TO ELECTRICAL AND COMPUTER ENGINEERING		
SEMESTER	SPRING	**YEAR**	2022
LABORATORY LOCATION	ELIF 331 (ELECTRONICS LAB)		
LAB SESSION DATE AND TIME	MONDAY 07 FEB 2022		
TYPE OF SUBMISSION	LABORATORY REPORT	**NUMBER OF SUBMISSION**	3
TITLE OF SUBMISSION	PROGRAMMING THE ARDUINO NANO		
METHOD OF SUBMISSION	ONLINE TO: maher.albadri@alaska.edu		
DUE DATE OF SUBMISSION	FRIDAY 18 FEB 2022	**DUE TIME OF SUBMISSION**	23:59

STUDENT NAME	Jacob Guenther

MAKE THIS FORM A "COVER PAGE" FOR YOUR REPORT SUBMISSION.

FOR THE TA USE ONLY

REMARKS:

1 Objective

In this lab we learn how to compile and upload a program to an Arduino Nano using the Arduino IDE. We also gain experience using an oscilloscope by using it to measure frequency.

2 Equipment

- Ocilloscope
- Arduino Nano
- Resistors
- Potentiometer
- LEDs
- Breadboard
- Jumpers

3 Setup

For the first experiment we use the schematic shown if figure (1). We probe the circuit with the occiliscope at digital pin 9 and ground.

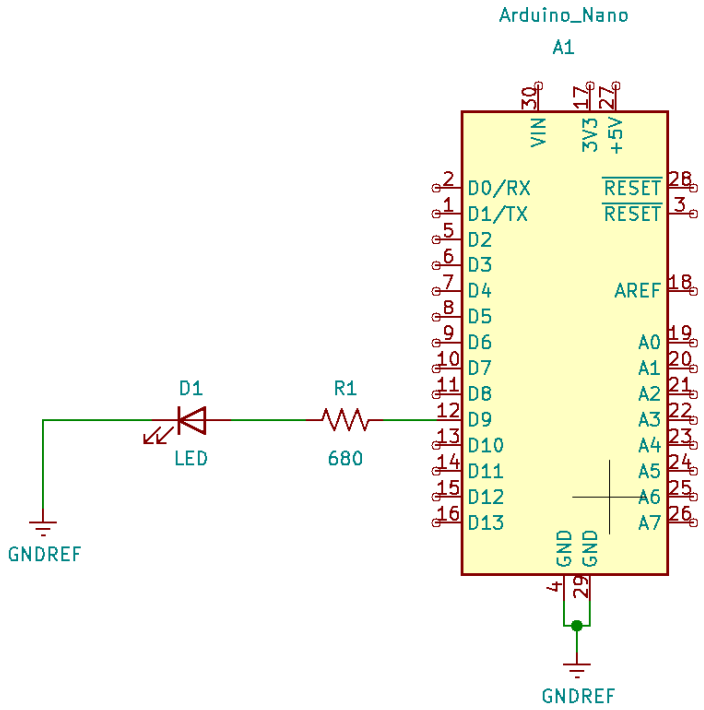

Figure 1: Schematic used to create a 25kHz frequency with a 30% duty cycle.

In the second experiment we use the circuit shown if figure (2).

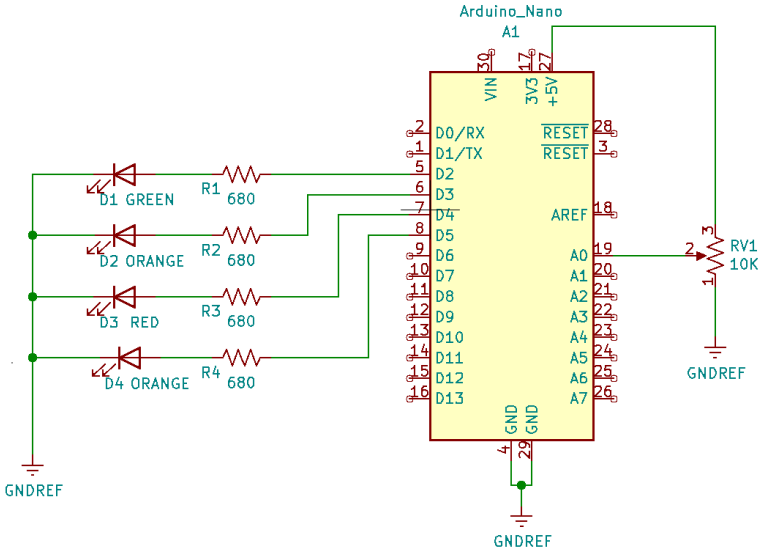

Figure 2: Schematic used to measure voltage on an analog input and display the value of that voltage using LEDs.

4 Observations and Results

We use the circuit shown in figure (1) along with the code shown in figure (3) to produce the plot shown in figure (4). The code was moddified so that the duty is 30% instead of 50%. To do this we multiply 1023 by 0.3. 1023 is the max value for the duty cycle.

From the plot we can see that the max voltage is 4.65 volts. The period between peaks is 39.8 us, and the frequency of the square wave is 25.126 kHz. The positioning of the reference lines may be slightly off.

```
#include <TimerOne.h>

const int LED_PIN = 9;

int hz_to_microseconds(int hz) {
  // float is used because seconds can be less than 1.
  // Also to prevent integer division errors.
  float seconds = 1/(float)hz;
  int us = seconds * 1000000;
  return us;
}
void setup() {
  pinMode(LED_PIN, OUTPUT);
  Timer1.initialize();

  // pwm(pin, duty, period) = generates waveform on specified pin.
  // Duty cycle is specified between 0 and 1023.
  // Period specified in microseconds.

  int duty = 1023 * 0.3; // 30%
  int period_hz = 25000;
  int period = hz_to_microseconds(period_hz);
  Timer1.pwm(LED_PIN, duty, period);
}

void loop() {}
```

Figure 3: Code for 25kHz frequency with 30% duty cycle.

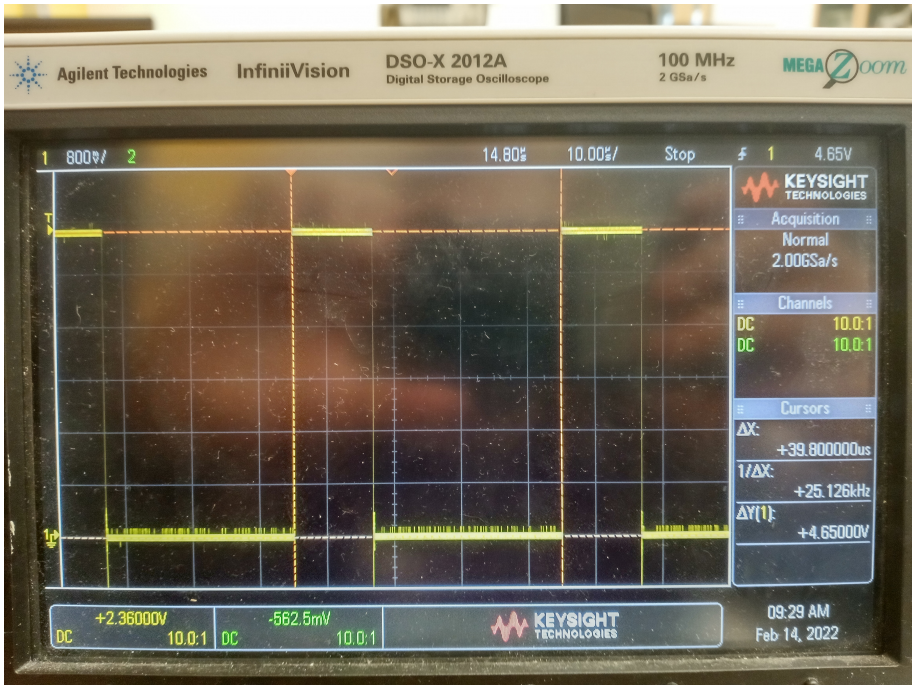

Figure 4: The plot of the occiliscope showing a 25 kHz frequency with a 30% duty cycle.

In the next part of the lab we use the circuit shown in figure (2) along with the code shown in figure (5) to indicate the voltage at anolog pin 0. The code was modified from the provided code by using else if statements to turn on only 1 LED.

```
int led0 = 2;
int led1 = 3;
int led2 = 4;
int led3 = 5;

int sensorValue;
float sensorVoltage;

void setup() {
  pinMode(led0, OUTPUT);
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(led3, OUTPUT);

  Serial.begin(9600);
  while (!Serial) { };
  Serial.println("ready\n");
}

void loop() {
  sensorValue = analogRead(A0); // reads count (0 to 1023) on analog input A0
  sensorVoltage = sensorValue*(5.0/1023); //converts sensorValue count value to voltage value

  Serial.print(sensorValue);     //prints count value to Serial Monitor
  Serial.print("\t Voltage ");   //prints tab then Voltage to serial Monitor
  Serial.println(sensorVoltage); //prints voltage value to Serial Monitor

  // Turn off all LEDs
  digitalWrite(led0, LOW);
  digitalWrite(led1, LOW);
  digitalWrite(led2, LOW);
  digitalWrite(led3, LOW);

  // Then turn on 1 LED indicating the voltage
  if(sensorVoltage > 4) {        // Set led 3 on if sensorVoltage greater than 4 V.
    digitalWrite(led3, HIGH);
  } else if(sensorVoltage > 3) { // Set led 2 on if sensorVoltage greater than 3 V.
    digitalWrite(led2, HIGH);
  } else if(sensorVoltage > 2) { // Set led 1 on if sensorVoltage greater than 2 V.
    digitalWrite(led1, HIGH);
  } else if(sensorVoltage > 1) { // Set led 0 on if sensorVoltage greater than 1 V.
    digitalWrite(led0, HIGH);
  }

  // sleep for 100 ms
  delay(100);
}
```

Figure 5: Code used to turn on an LED depending on the measured voltage at anolog pin 0.

5 Conclusion

In this lab we learned how to program an Arduino using the Arduino IDE. We used the PWM functionality of some of the digital pins, to create a specific square wave. Finally we programmed the Arduino to read an analog input and display the voltage using a digital output pin.

To read a temperature using a thermistor we can use a voltage divider with the thermistor, resistor, and an anolog input pin. Then we can convert the analog value to voltage, and then convert to voltage to temperature.

6 References

[1] Denise Thorsen, Maher Al-Badri, INTRODUCTION TO ELECTRICAL AND COMPUTER ENGINEERING, University of Alaska Fairbanks, 2022.