ECE 118/218 Mechatronics

Tae Myung Huh

Lab 3 Report

Group 25:

Victor P.C. , Andrew Huang

May 3, 2025

Introduction

In this lab, we explored the relationship between different types of motors and a microcontroller. In this case, an Arduino Uno, the primary microcontroller, is interfacing with the software side, while the various driver modules and their associated motors act as the visual interface with its responses. For a DC motor, we are interfacing the DS3658 motor driver and the PWM (Pulse-Width Modulation) signals. For the stepper motor, we are interfacing the DRV8811 Stepper Motor module with the stepper motor. This lab explores how the duty cycle variation influences motor speed and it also allows us to analyze the performance of the outputs based on the oscilloscope readings. Additionally, modifying physical parameters such as adding a load to the motor shaft by gripping with one's fingers can impact the current draw, voltage behavior, and waveform shape. This lab provides experience in motor control, signal generation, and interpreting system responses through measurements.

Part 1 - Driving an RC Servo

Objective

In this part, we are driving a standard RC servo motor with software control. Specifically, we use functions from the following header files: AD.h, LED.h, and RC_Servo.h from the ece118_base directory.

Schematic

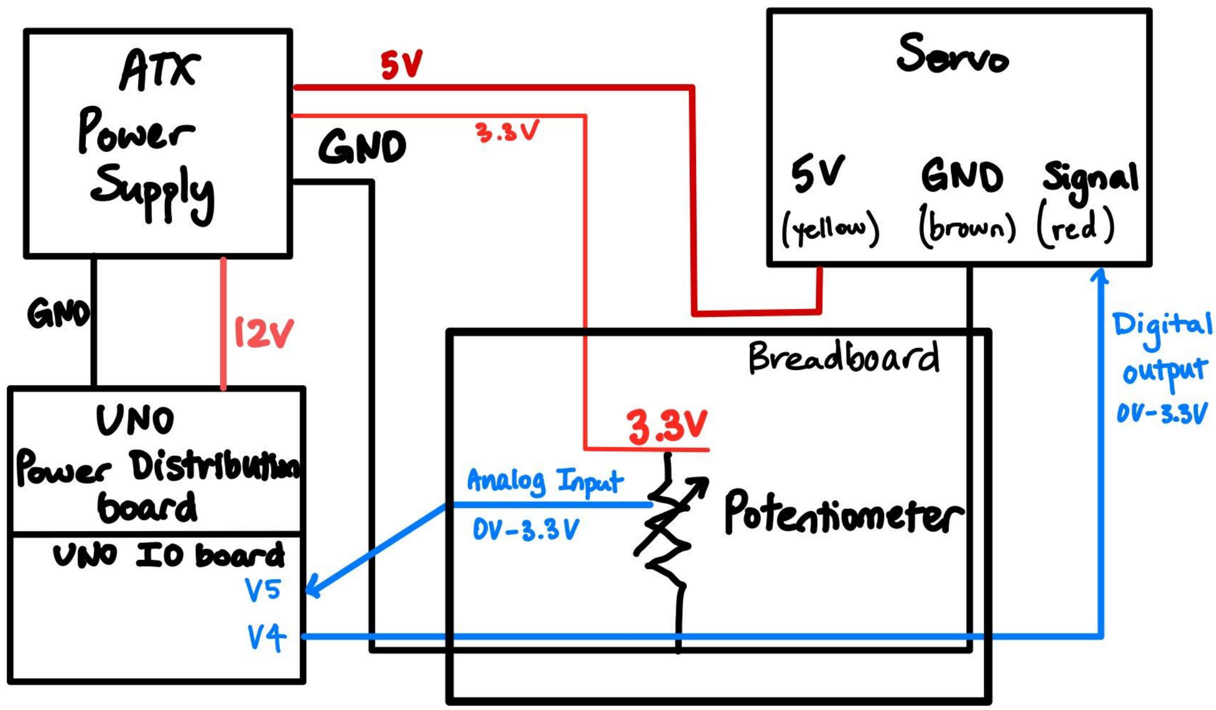

Figure 1. Block diagram of the complete circuit.

The figure above shows how each module connects to the other. When connecting the breadboard, there is an input voltage of 3.3V from the ATX power supply connecting to the positive terminal of the potentiometer. Please note that the potentiometer has three terminals: positive, negative (ground), and output terminal. There is a connection between the potentiometer and the UNO IO Board Port V Pin 5. This pin uses the output terminal, which feeds an analog input into that specific pin. Port V Pin 5 is a designated AD pin, which can act as an analog or digital input/output pin. The control signal pin takes a digital output reading from the Uno IO Board Port V Pin 4 on the servo motor module. Before testing, we must ensure the servo motor is powered with the correct input voltage and grounded so it does not go awry. We connect 5V from the power supply and ground the specified pin to a common ground set on the breadboard.

Output Traces

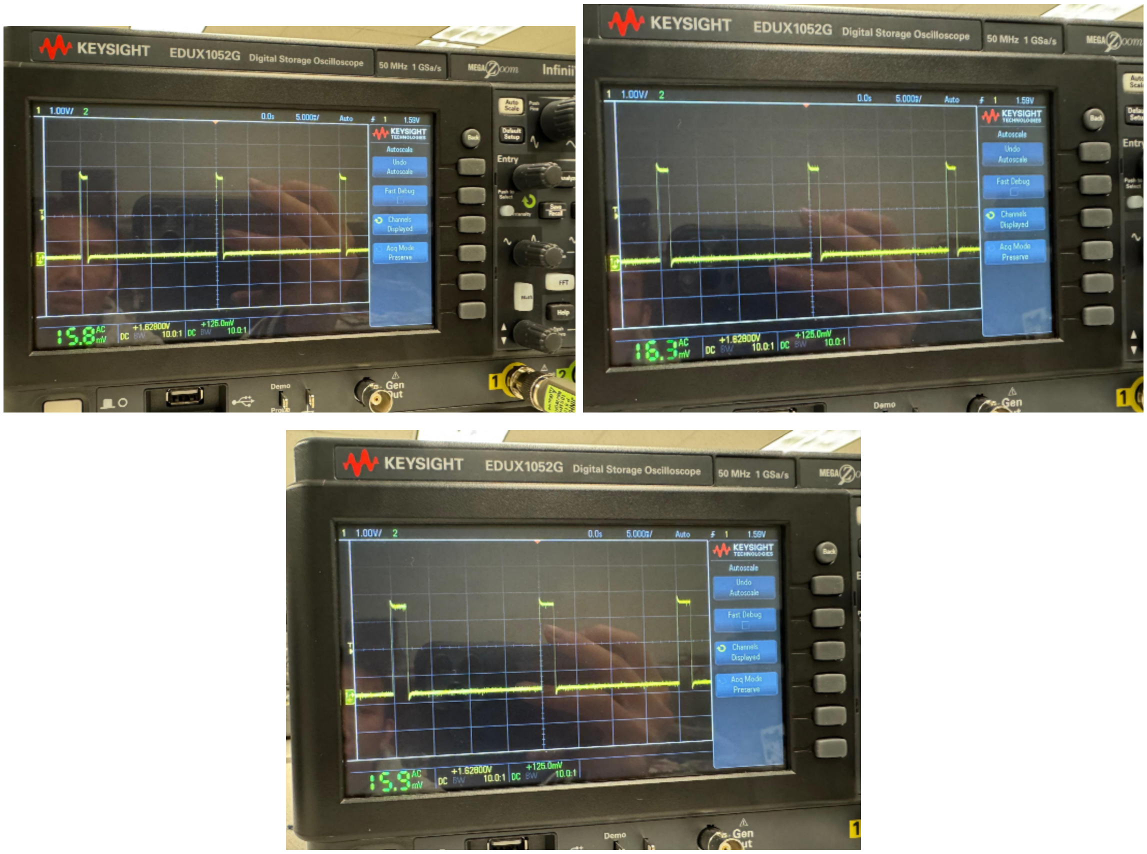

Figure 2. Waveform Measured at the Control Signal pin of the Uno IO Board starting from low to high duty cycle (top is low duty cycle, middle is medium, bottom is high).

Code Snippet

Observations

After building the circuit, the next step is testing. When testing, the first step is to check whether the potentiometer shows a varying voltage reading depending on how much the knob is twisted; this corresponds to the resistance. Next, using the oscilloscope, we ensure a square wave output when measuring the control signal pin. Once the two previous steps are correct, then we can turn off the power supply and connect the servo motor. After connecting them and turning on the power supply, the measurement is taken from the control signal pin, which also connects to the servo signal pin. The three waveform outputs shown in Figure 2 correspond to how much the knob of the potentiometer is turned. As it is turned, the resistance increases; thus, the time the signal is high also increases. In addition to the difference in signal and resistance, the servo has limited range but it rotates to its limits. There are LEDs on the Uno32 board which are programmed to correspond to the value of the duty cycle. If the duty cycle is set to 100%, the expected result is having all 3 LED banks light up and vice versa for a duty cycle of 0%.

Part 2 - Unidirectional Drive of a DC Motor

Objective

In this part, we are driving a DC motor in one direction with software control. The software is programmed into the microcontroller using the DS3658 high current peripheral driver. The PWM software drives the motor from the input reading of the potentiometer.

Schematic

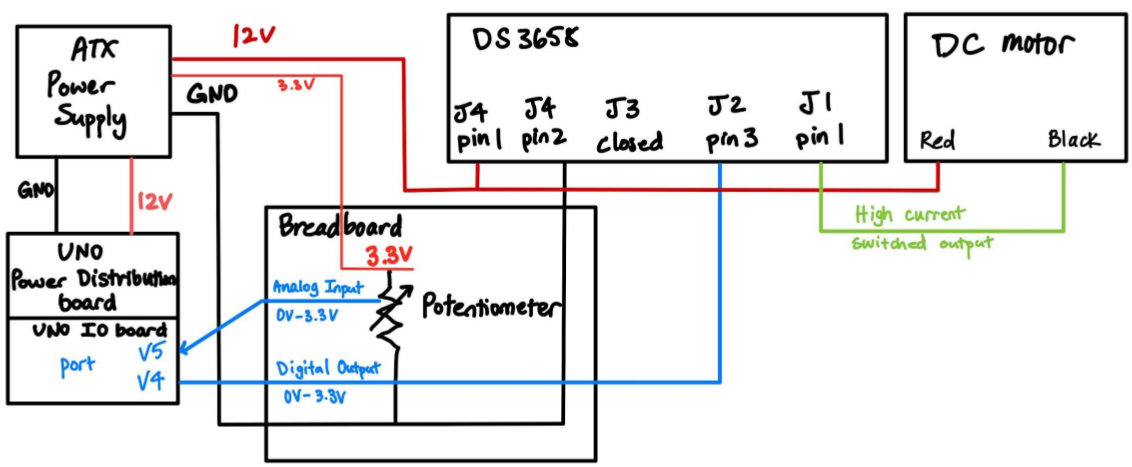

Figure 3. Schematic of the complete circuit, connecting the power supply, DS3658 DC Motor Driver, DC Motor, and Breadboard module with Uno IO Board.

Code Snippet

Output Traces

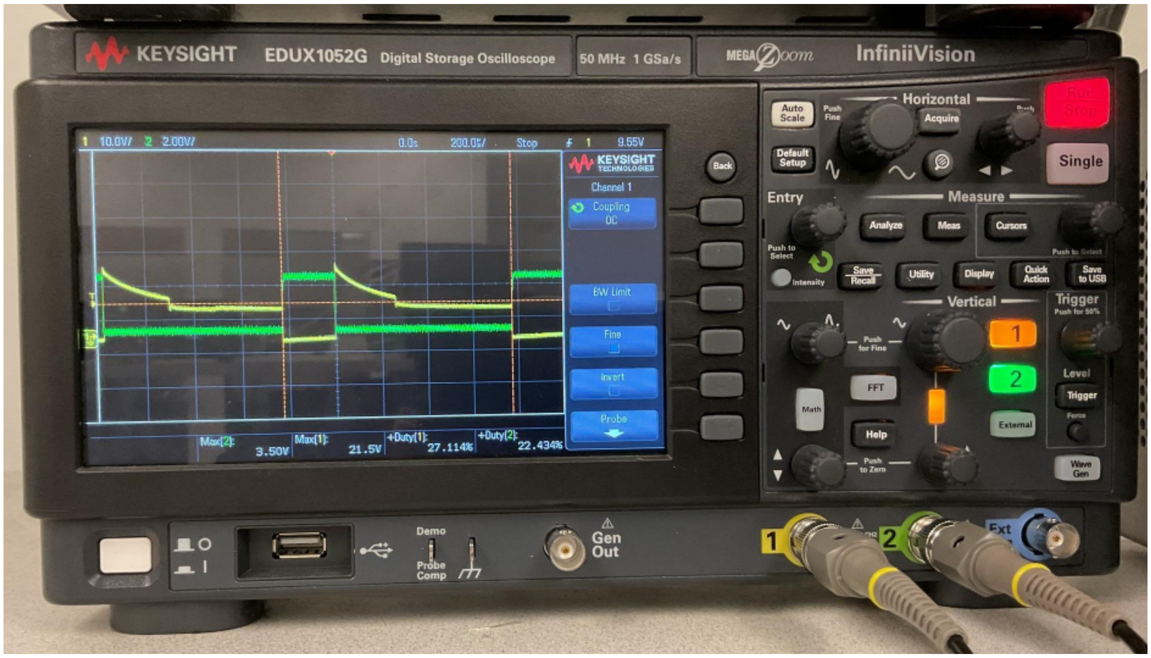

Figure X. Waveform output measurement at 20% duty cycle.

Figure X. Waveform output measurement at 80% duty cycle.

Observations

When testing the circuit, there are some key points to note. In Figure X, where the duty cycle is set to 20%, the yellow waveform measures the output at the motor terminal. Specifically, inductive kickback happens as the PWM signal goes low. This is due to the motor's inductive nature. As the duty cycle is increased, there is less inductive kickback happening. When comparing the two outputs, the waveform when the duty cycle is 20% is inverted, whereas the output of the motor terminal is set to high when there is a LOW reading from the PWM signal. As the duty cycle increases, the output motor terminal follows the PWM signal more closely. Similarly to the previous part, the LEDs light up incrementally in response to the potentiometer value reading.

Part 3 - Snubbing the Inductive Kickback

Objective

In this part, we are exploring the use of diodes and Zener diodes in different configurations to snub the inductive kickback on the DC motor. The DC motor will still be driven using the software from the previous part.

Schematic

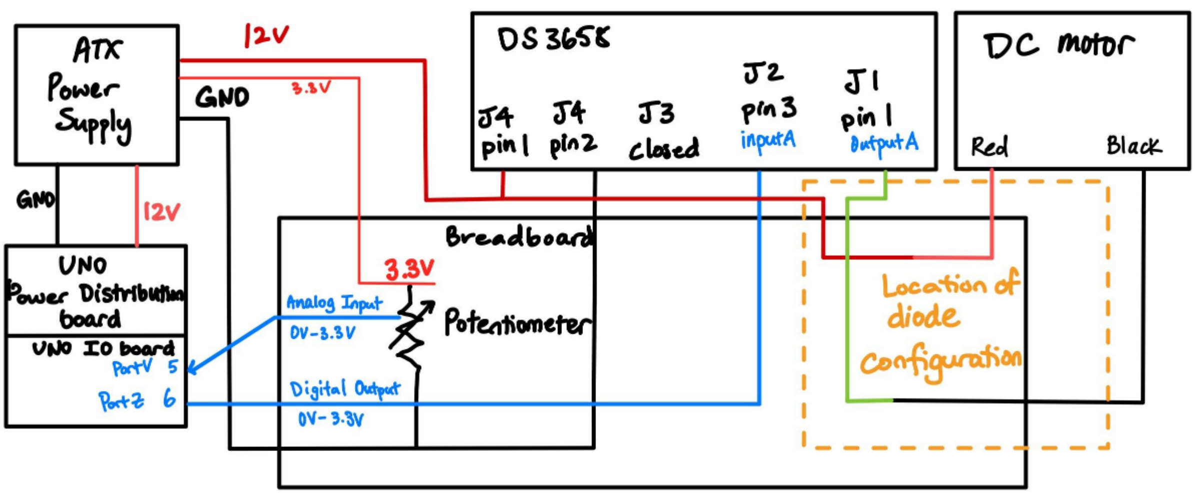

Figure X. Schematic of the complete circuit, connecting the power supply, DS3658 DC Motor Driver, DC Motor, and Breadboard module with Uno IO Board.

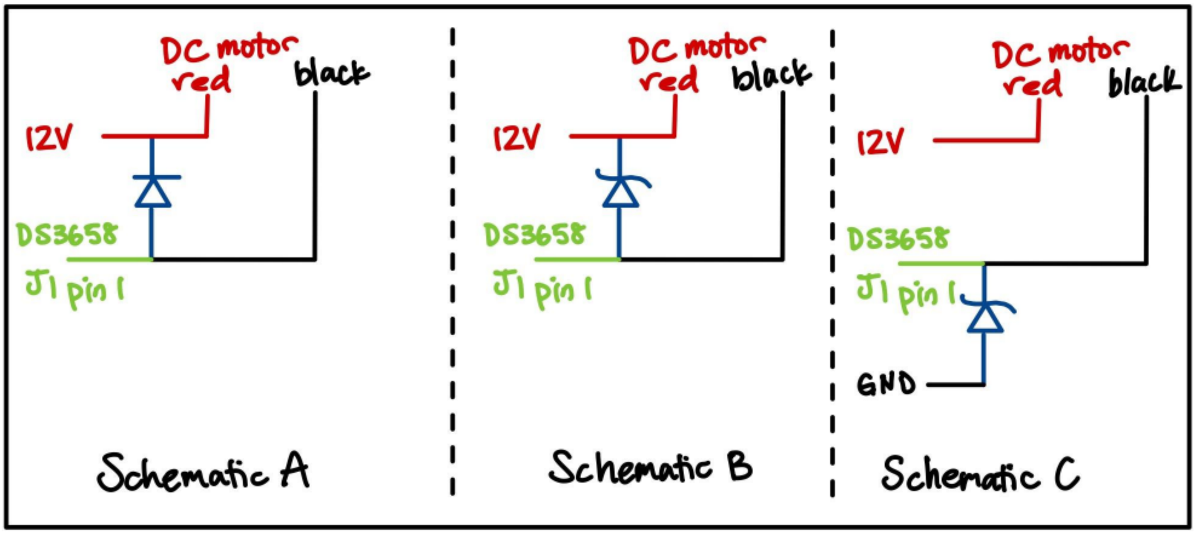

Figure X. 3 different diode configurations required for testing.

Output Traces

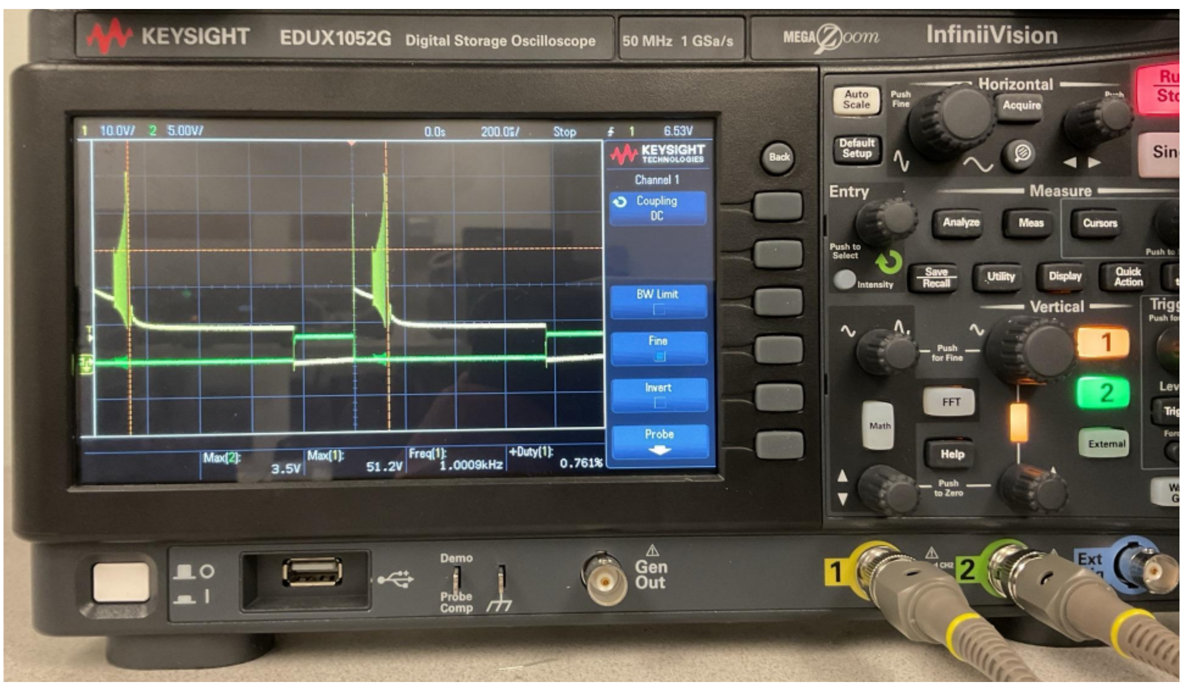

Figure X. Waveform measurement of output motor terminal using Schematic A without normal diode.

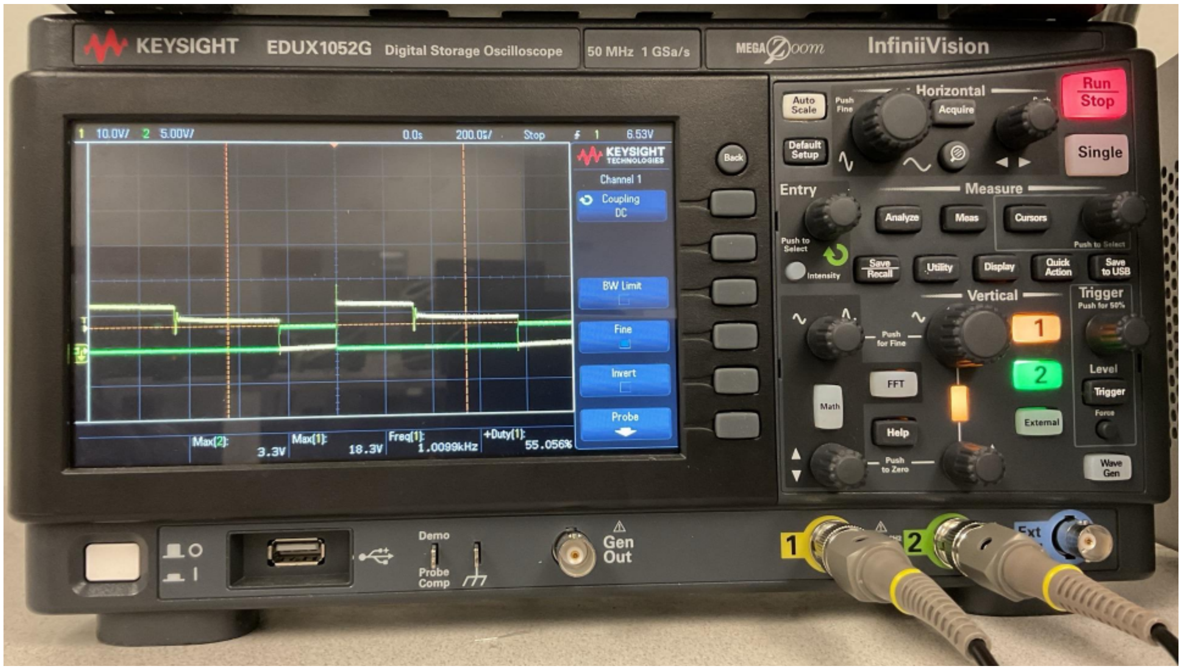

Figure X. Waveform measurement of output motor terminal using Schematic A with normal diode.

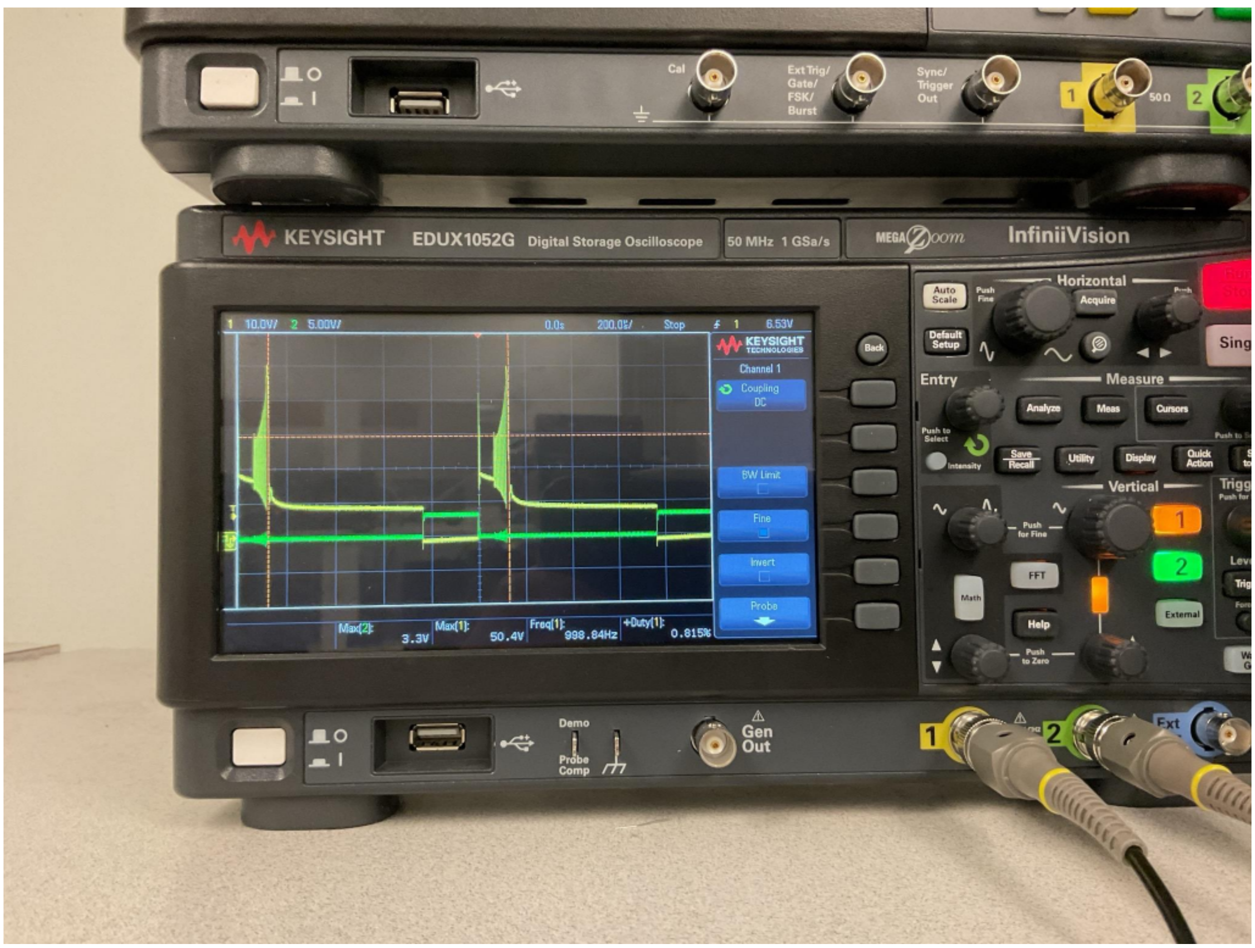

Figure X. Waveform measurement of output motor terminal using Schematic B without Zener diode.

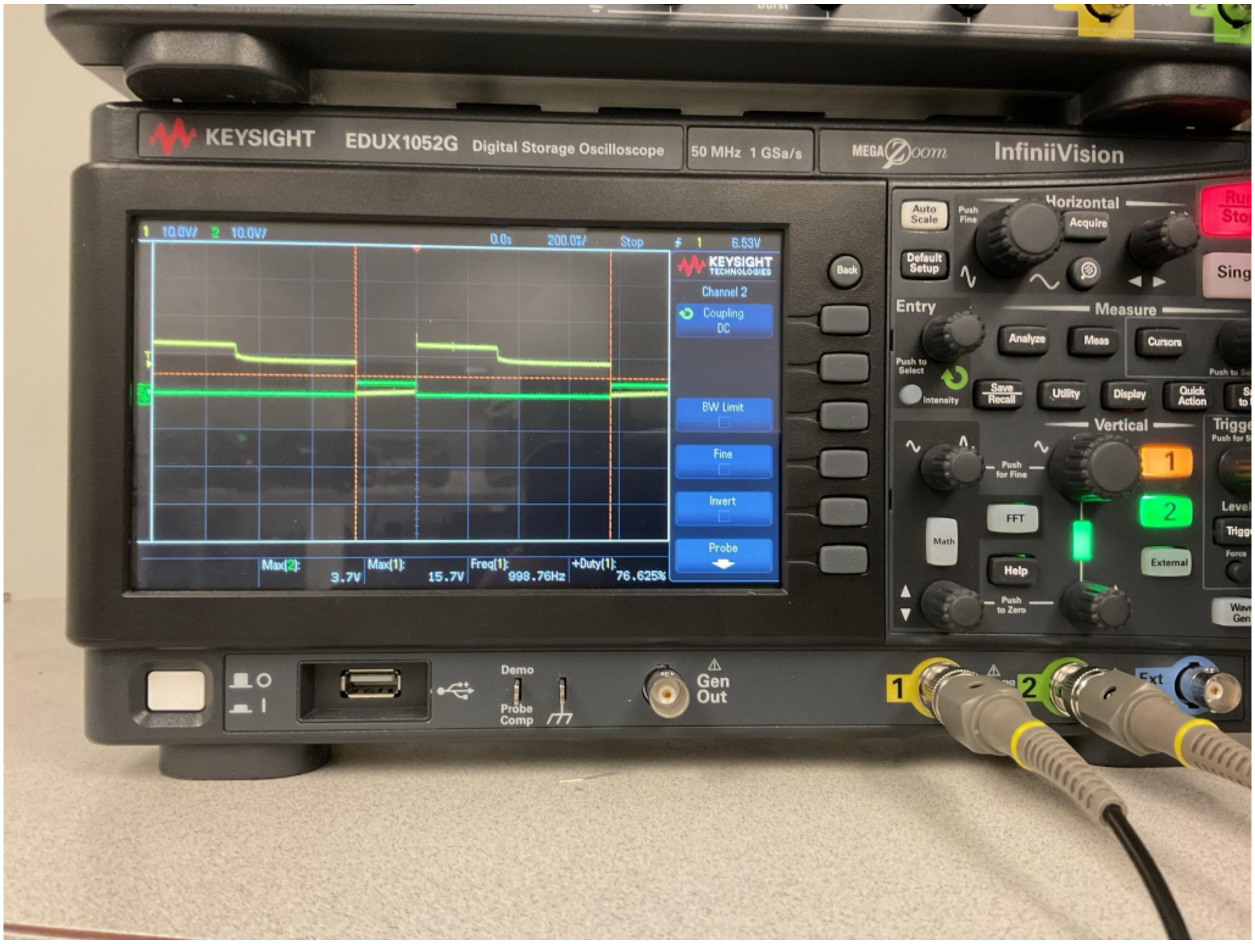

Figure X. Waveform measurement of output motor terminal using Schematic B with Zener diode.

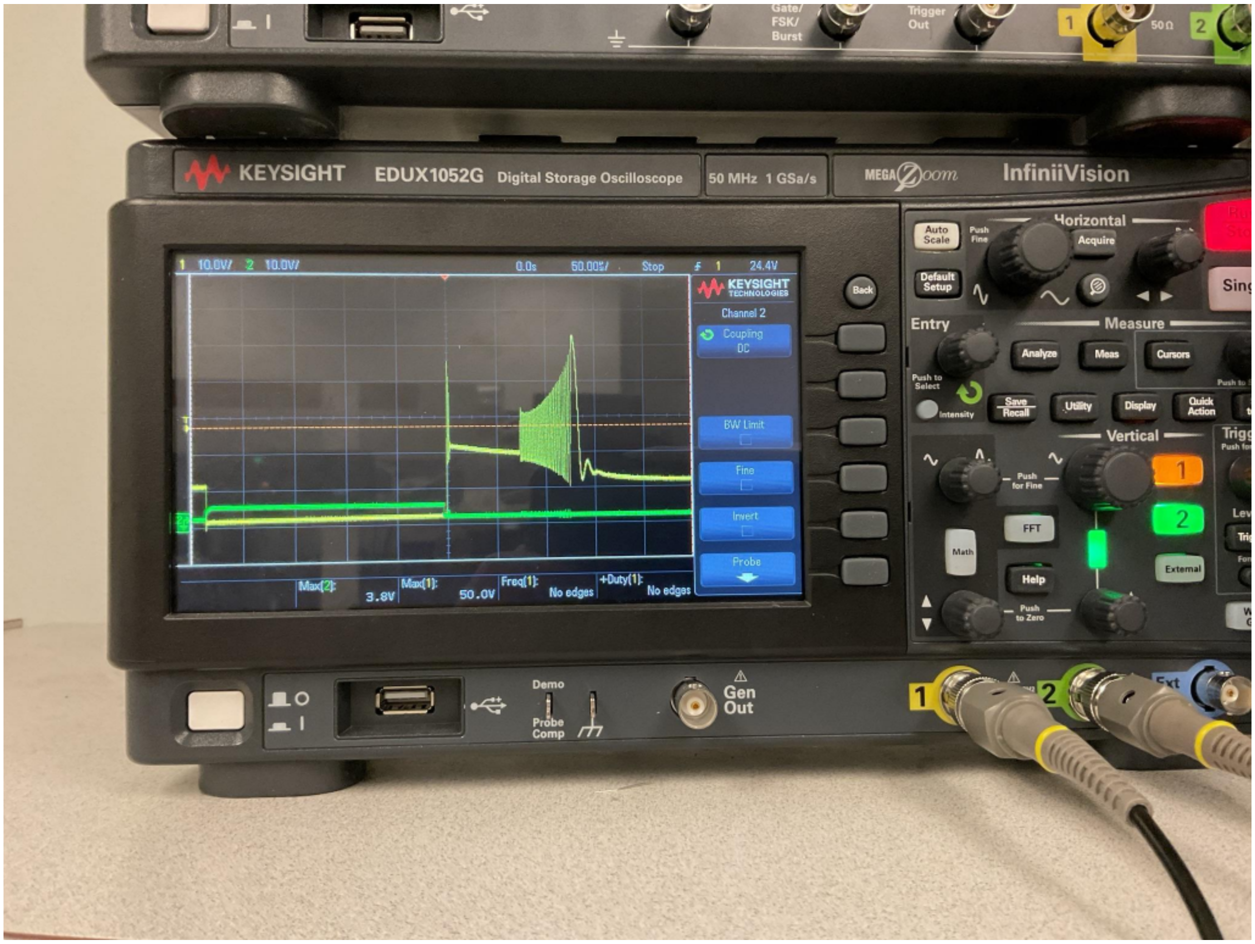

Figure X. Waveform measurement of output motor terminal using Schematic C without Zener diode.

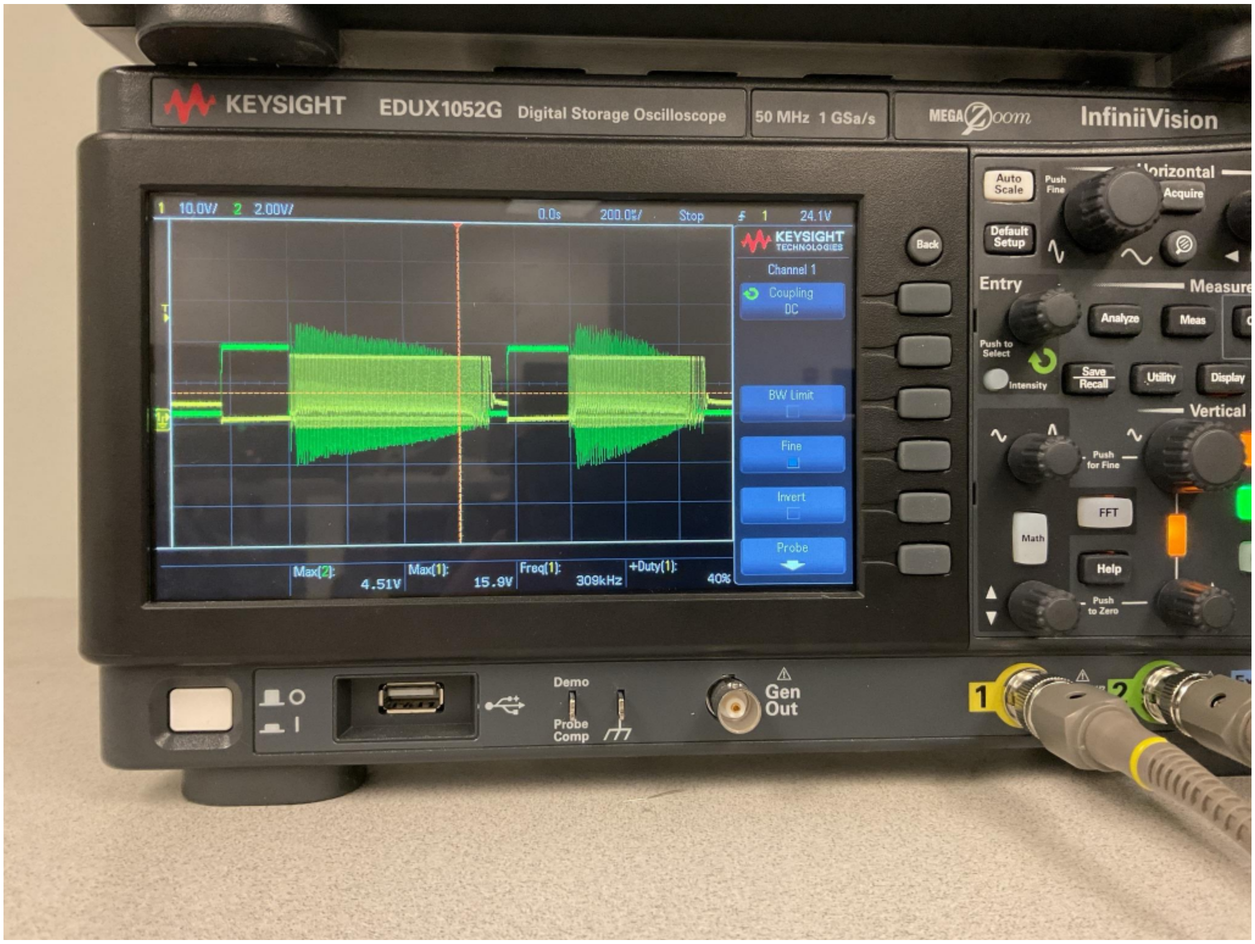

Figure X. Waveform measurement of output motor terminal using Schematic C with Zener diode.

Observations

In this part, a flyback diode was added in parallel with the motor terminals to suppress voltage spikes caused by the motor's inductive nature. When observing the motor terminal waveforms on the oscilloscope, the voltage spikes rise up to 50V, resulting in the motor not operating as smoothly as expected. Using the diodes significantly reduces the sharp inductive kickback. Inductive kickback occurs when there is a sudden change in current flow. If the current flow is disrupted, a voltage difference occurs inside the inductor, where the voltage keeps increasing. This indicates that the flyback diodes are effective at clamping the reverse EMF. When looking at the figures corresponding to the use of the flyback diodes, the voltage clamps to 15-17V, which is slightly higher than the input voltage. This is called snubbing.

When the motor shaft is manually loaded by gripping with fingers, the waveform's high interval tends to extend closer to the falling edge of the PWM signal. This indicates that the motor's back-EMF is resisting the change in current flow. The waveform overall is more stable, and voltage transitions are less abrupt.

The PWM waveform remains the same, but the motor terminal voltage is cleaner as the flyback diode provides a path for the inductive current. The diode protects driving circuitry and also improves waveform consistency.

Part 4 - Bidirectional Control of a DC Motor

Objective

In this part, we are controlling a DC motor in both directions using an H-bridge module (L298N). The input reading of the potentiometer to the Uno32 acts as the speed control, but direction is controlled by a switch.

Schematic

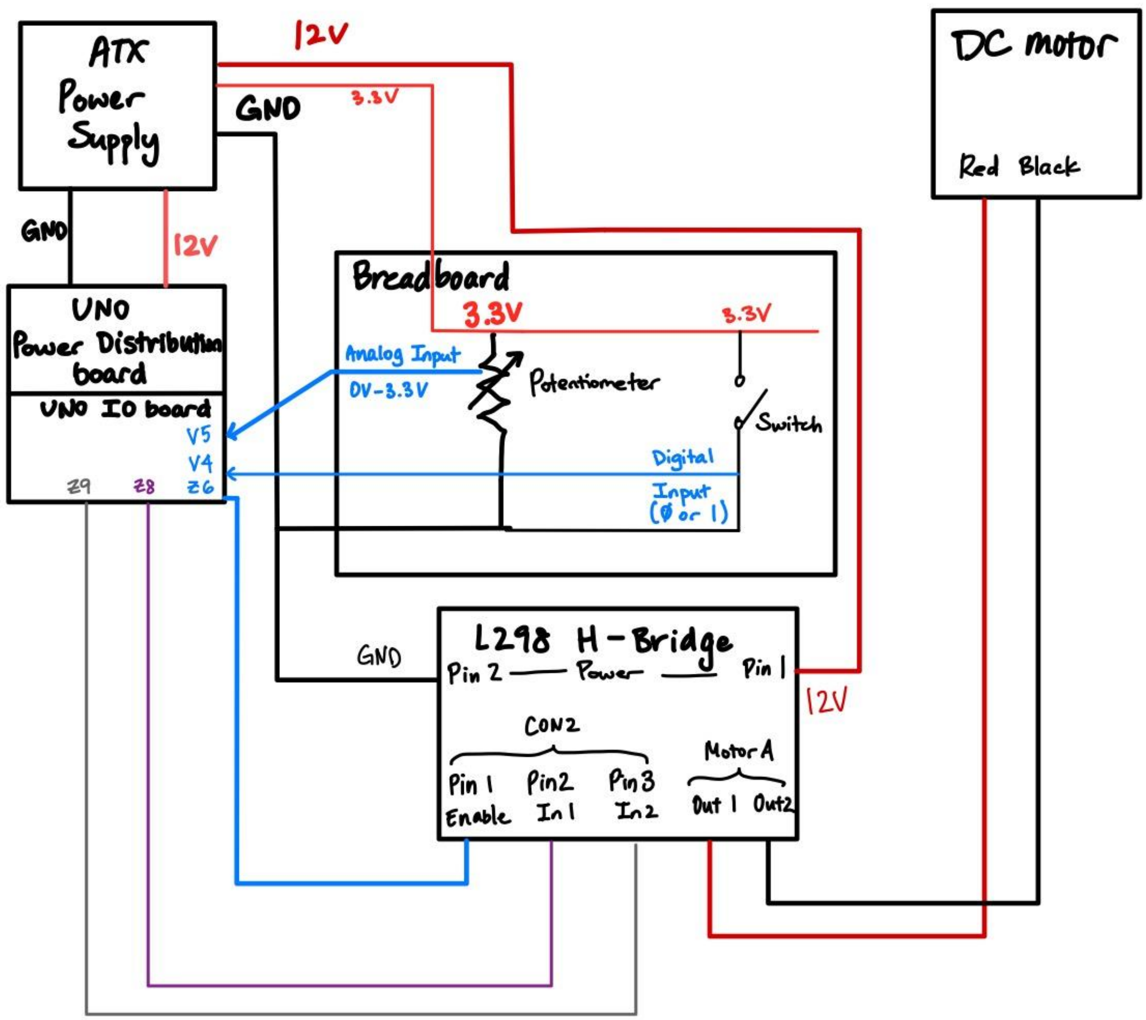

Figure X. Schematic of the complete circuit, connecting the power supply, L298N H-Bridge module, DC Motor, and Breadboard module with Uno IO Board.

Code Snippet

Observations

When testing, the H-bridge controlled the current direction through the DC motor, enabling bidirectional rotation. By toggling the control signals, the motor could rotate clockwise or counterclockwise.

On the oscilloscope, the PWM signal remained similar to the previous parts, where it controlled the speed of the motor. When observing the motor terminals, the voltages showed polarity reversals corresponding to directional changes. This reversal was evident when observing the waveform on the oscilloscope.

When observing the motor under varying duty cycles, the speed changes as expected, but the motor's ability to change direction is fascinating. At lower duty cycles, the voltage is lower, resulting in reduced torque. As the duty cycle increased, the motor responded with higher speeds and more substantial torque in the specified direction.

Part 5 - Control of a Stepper Motor

Objective

In this part, the goal is to write software that controls the stepper motor using the H-Bridge module. The important part is to explore the step rate at which it is possible to drive the stepper motor.

Schematic

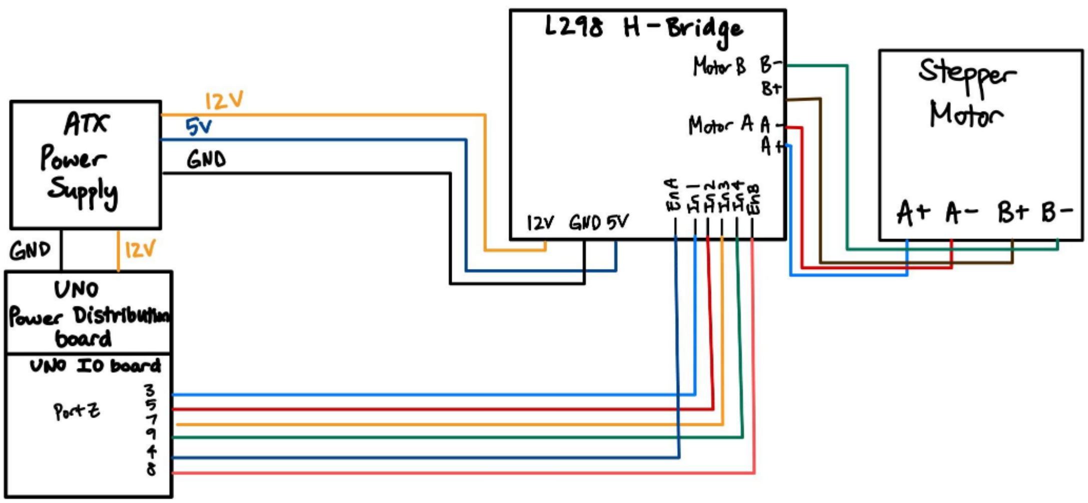

Figure X. Schematic of the complete circuit, connecting the power supply, L298N H-Bridge module, DC Motor, and Breadboard module with Uno IO Board.

Code Snippet

Observations

Part 6 - Stepper Motor Using Dedicated Board

Objective

This part is similar to the previous part but now, we are using a dedicated stepper driver board (DRV8811 Stepper Motor module).

Schematic

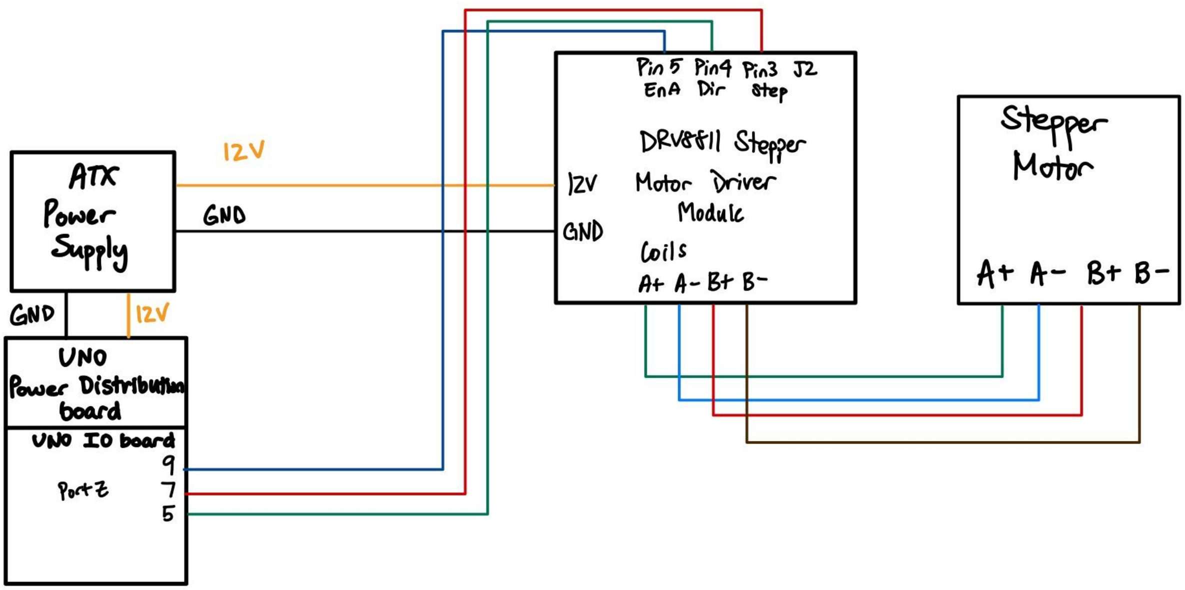

Figure X. Schematic of the complete circuit, connecting the power supply, DRV8811 Stepper Motor Driver module and stepper motor with the Uno IO Board.

Code Snippet

Observations

Conclusion