DC MOTOR SPEED MEASUREMENT USING ENCODER

1.0 Learning Outcomes

After completing this Exercise, you will be able to:

1. Assemble a DC motor control circuit using an Arduino Uno, Maker Drive motor driver, external power supply, and encoder feedback.
2. Explain how encoder counts can be used to calculate the rotational speed of a DC motor in RPM.
3. Configure Simulink to apply a PWM input signal and capture encoder-based speed data from the motor.
4. Use measured RPM data to develop and evaluate a mathematical model of the motor system.

After completing this Exercise, it is recommended that you refer to the Learning Outcomes.

2.0 Hardware Set-Up

This circuit builds on the previous motor-driver exercise by controlling a DC motor using an Arduino Uno and Maker Drive motor driver, but with additional control wiring. The Arduino sends signals to the Maker Drive to control the motor's speed, while the 9V battery provides the higher-power supply needed to operate the DC motor. As before, the Arduino and motor driver must share a common ground so that the control signals are correctly recognised.

As with the previous exercise, the required hardware for this exercise is as follows:

1. Arduino Uno
2. Maker Drive motor driver board
3. DC motor

4. 9V battery and clip
5. Breadboard
6. Breadboard wires (various)
7. USB cable

Begin by setting up the circuit in the same way as the previous exercise: connect the motor to the Maker Drive output terminals and connect the 9V battery to the motor power input. Then add the Arduino control wires to the Maker Drive input pins. The extra connections allow the Arduino to control the motor driver, for example by setting motor direction and using PWM to vary motor speed.

Step-by-step set-up instructions

1. Connect the **DC motor wires** to the Maker Drive motor output terminals. Use one motor channel, such as **M1A and M1B**.
2. Connect the **9V battery positive wire** to the Maker Drive motor power input, usually labelled **VIN**, **VM**, or **+**.
3. Connect the **9V battery negative wire** to the Maker Drive **GND** terminal.
4. Connect a **GND pin on the Arduino** to the Maker Drive **GND**. This is essential because both boards need a shared reference voltage.
5. Connect the Arduino **5V pin** to the Maker Drive logic supply pin if required by the board.
6. Use the breadboard to organise the shared **5V** and **GND** connections if needed.
7. Connect the Arduino control pins to the Maker Drive input pins. These wires are used to control the motor's direction and speed.
8. Connect the Maker Drive motor input pins for the selected motor channel to suitable Arduino digital or PWM pins.
9. Check that the motor is connected only to the Maker Drive output terminals and not directly to the Arduino.
10. Connect the Arduino to the computer using the USB cable.

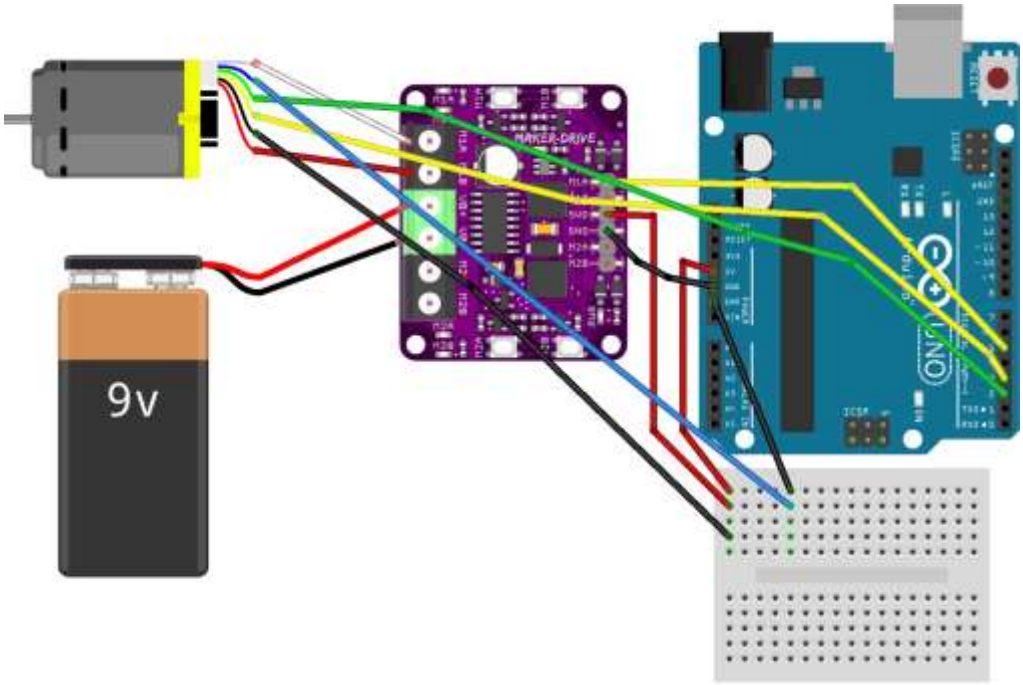

Figure 1: *Hardware Set Up for DC Motor Speed Measurement using Encoder*

3.0 Simulink Set Up and Results

A 5 Volt step input (i.e., value of 255) is applied to pin 9 of the Arduino, see Figure 2, with this connected to the DC motor, see Figure 1.

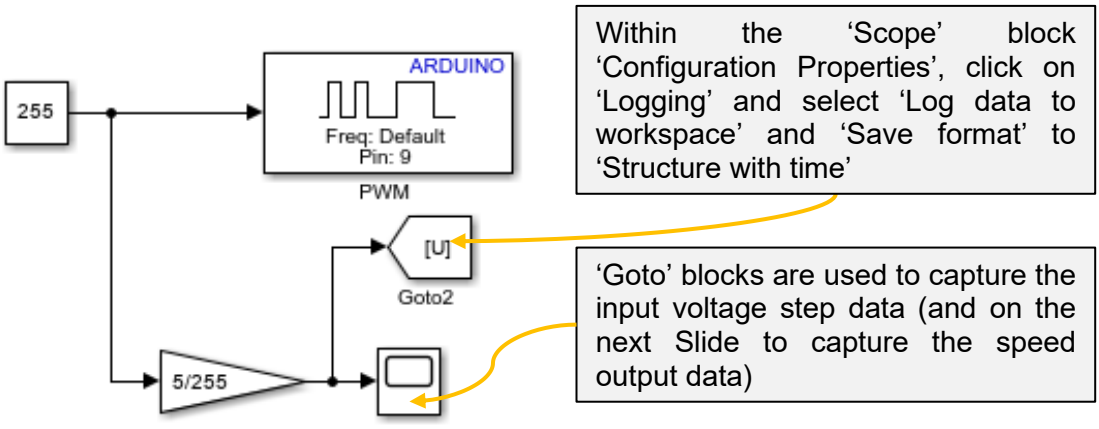

Figure 2: *Input Voltage Applied to the DC Motor with an Encoder*

In Simulink, the encoder on the DC motor is configured to determine the revolutions per minute (RPM) of the motor. This is achieved by using the encoder counts, which indicate the motor's position at any given time. By analysing the change in position over a specific sampling interval, the motor's speed can be calculated. Essentially, the encoder acts as a feedback device,

measuring the motor's position in discrete steps. By calculating the rate of change of these steps, the RPM is estimated, providing a crucial metric for controlling the motor's speed in various applications. The details of this setup are illustrated in Figure 3.

In Figure 3, the four gain blocks are combined into a single gain block to represent the equivalent mathematical relationship. This is necessary as Simulink requires this format. Additionally, the input voltage Simulink diagram from Figure 2 is included because these components need to be 'Run' together. A scope has also been added to display the counts per second from the encoder, with the scope data provided in Figure 4. 'Goto' blocks are used in Simulink to capture both the input (voltage) and output (RPM) signals. These signals are then sent to a scope using 'From' blocks, allowing the input-output data to be viewed on the scope.

If the motor spins in the opposite direction, swap the motor wires on the Maker Drive output terminals or change the direction logic in the code.

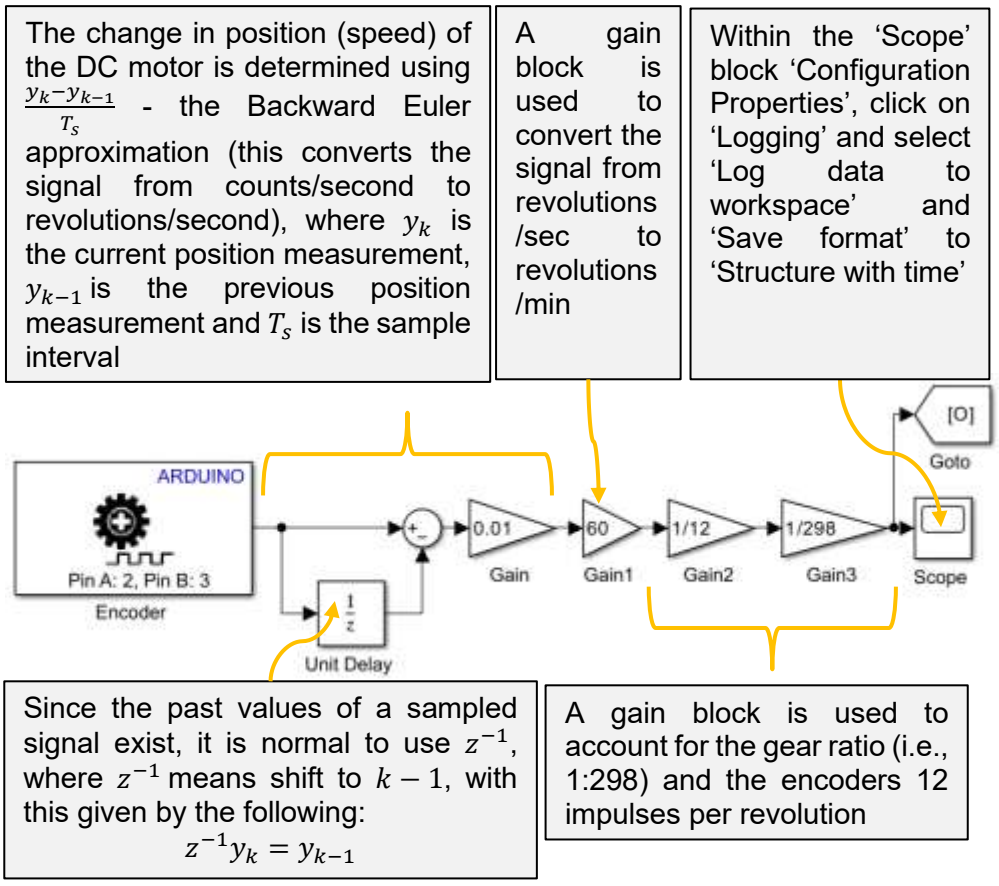

Figure 3: *Encoder Data Capture and Configuration using Simulink*

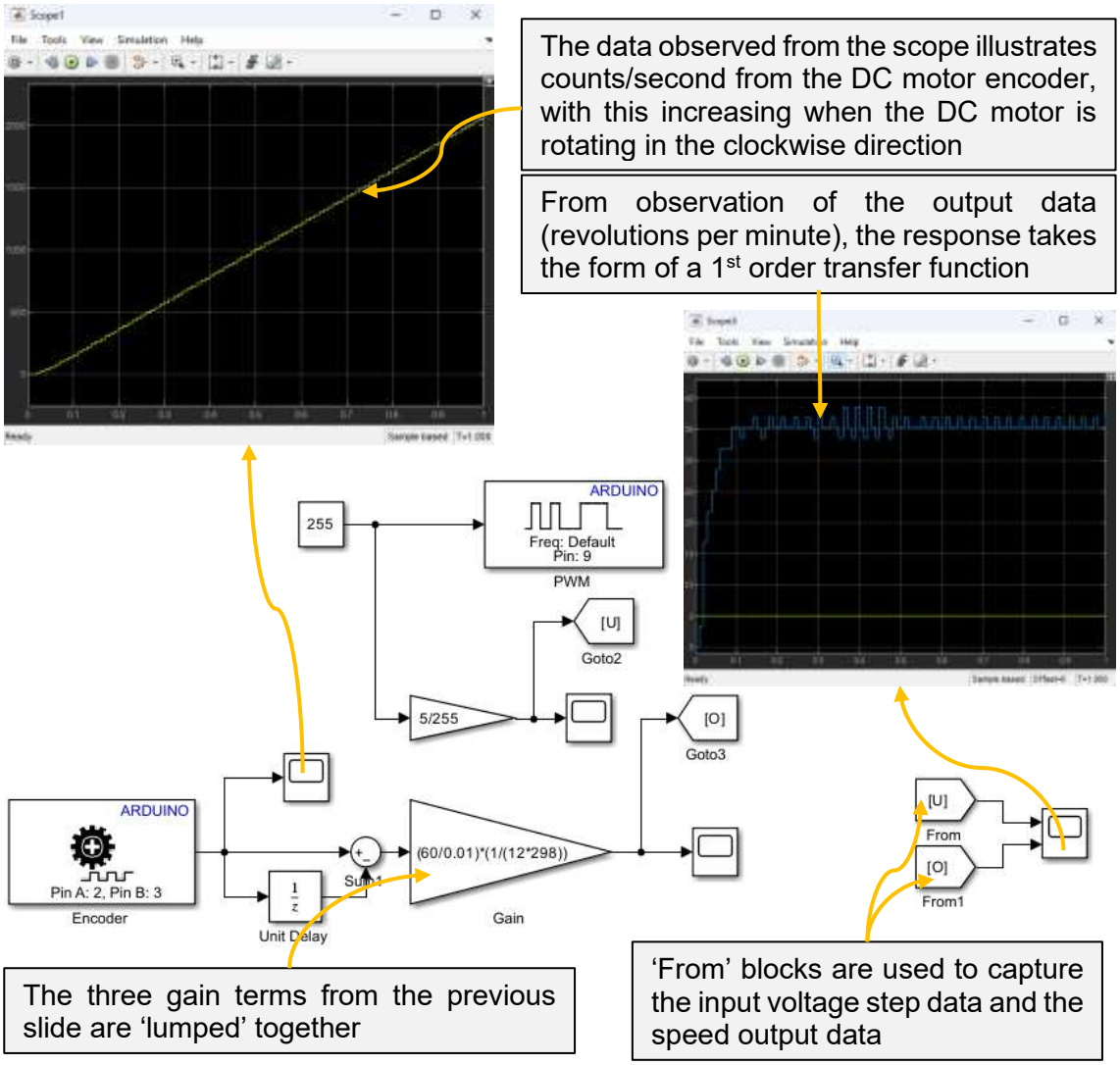

Figure 4: *Encoder Data Capture and Configuration using Simulink* (With One Gain Block used for the Main Components)

4.0 Additional Exercise

Using the revolutions per minute (RPM) output data obtained from the scope in the above Simulink diagram, develop a mathematical model that accurately captures the key dynamics of the system. Plot the RPM output data alongside the model output data and compare the model fit by calculating the mean squared error (MSE).

5.0 Concluding Remarks

This exercise introduced the use of encoder feedback to measure the speed of a DC motor controlled by an Arduino Uno and Maker Drive motor driver. By applying a PWM input and processing the encoder counts in Simulink, the motor speed can be estimated in RPM and displayed for analysis. The additional modelling task extends the practical work by using the measured data to develop a mathematical model of the motor response and assess its accuracy using the mean squared error.

Recommended citation:
Pickering, J.E. (2026). DC Motor Speed Measurement using Encoder. ACE-Lab Teaching Notes. Available at: www.ace-lab.co.uk.