Contents

DC Motor with an Encoder (Hall Effect Sensor)

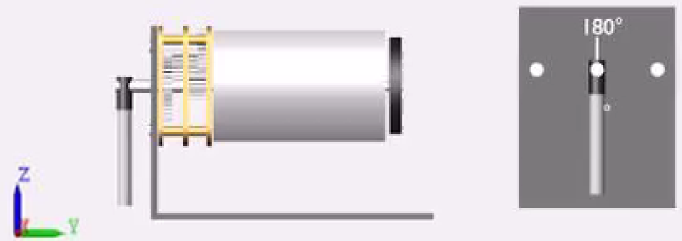
Key Learning Points

After this Lecture, you will be able to understand the following:

1. How to capture input and output data from an DC motor with an encoder using an Arduino Uno
2. Simulink has been configured to capture/measure the DC motor input and output data

3.1 Introduction

- The DC motor will be subject to a voltage step input, with the revolutions per minute (RPM) output measured using a hall effect sensor
- Based on the known/measured input-output data, a black-box model is to be developed (i.e., not considering the underlying physics of the motor) – with the model then used to develop the controller



$G_a(s)$

input $V_{in}(s)$ → $\dfrac{113.7}{s + 3.398}$ → output $\omega(s)$

Encoder A phase
Encoder B phase
Encoder ground
Encoder power
Motor power
Motor ground

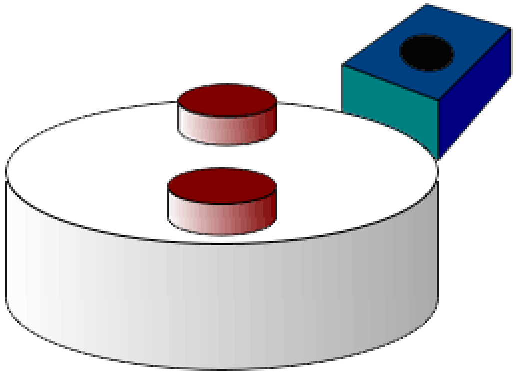
$G_a(s)$

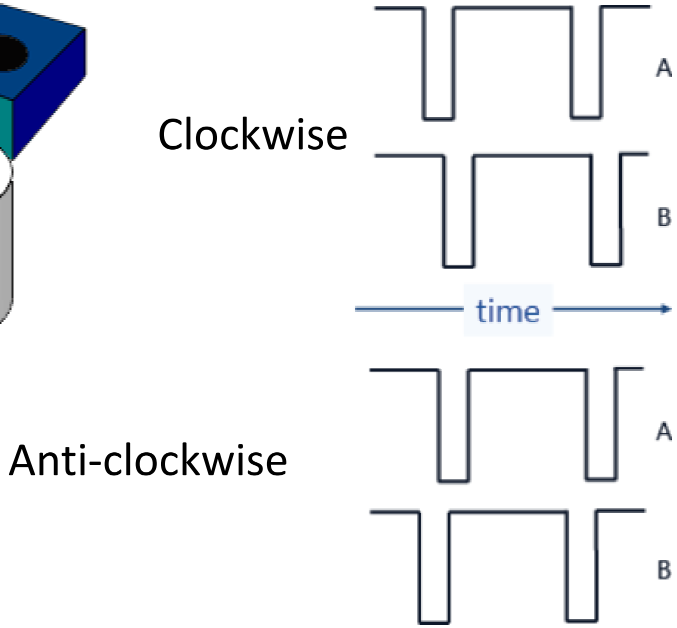
input $V(s)$ → $\dfrac{K}{(Ls + R)(Js + b) + K^2}$ → output $\omega(s)$

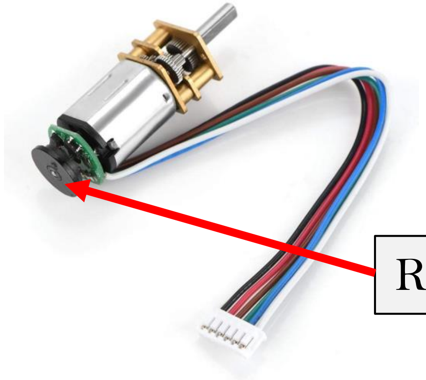
- Hall effects sensors are commonly used to time the speed of wheels and shafts, such as for internal combustion engine ignition timing, tachometers and anti-lock braking systems
- Hall effect sensors are used in brushless DC electric motors to detect the position
- Employing an offset Hall effect sensor can generate a quadrature output for determining the direction of the motion

3.1 Introduction

- A Hall effect sensor detects a magnetic field
- The output voltage of a Hall sensor is directly proportional to the strength of the field. It is named for the American physicist Edwin Hall.
- By detecting the passing of a magnet, the Hall effect sensor generates a pulse-train output



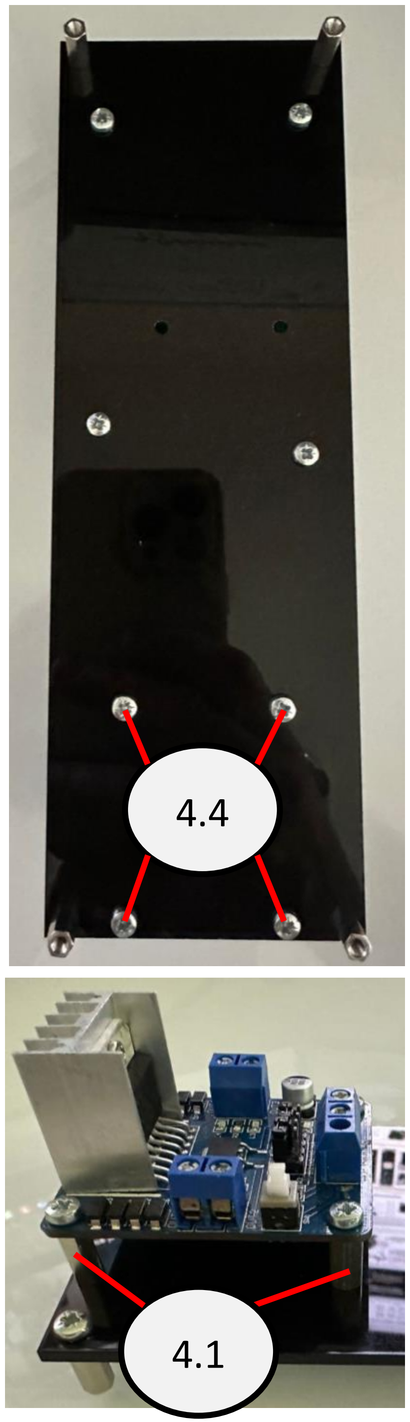


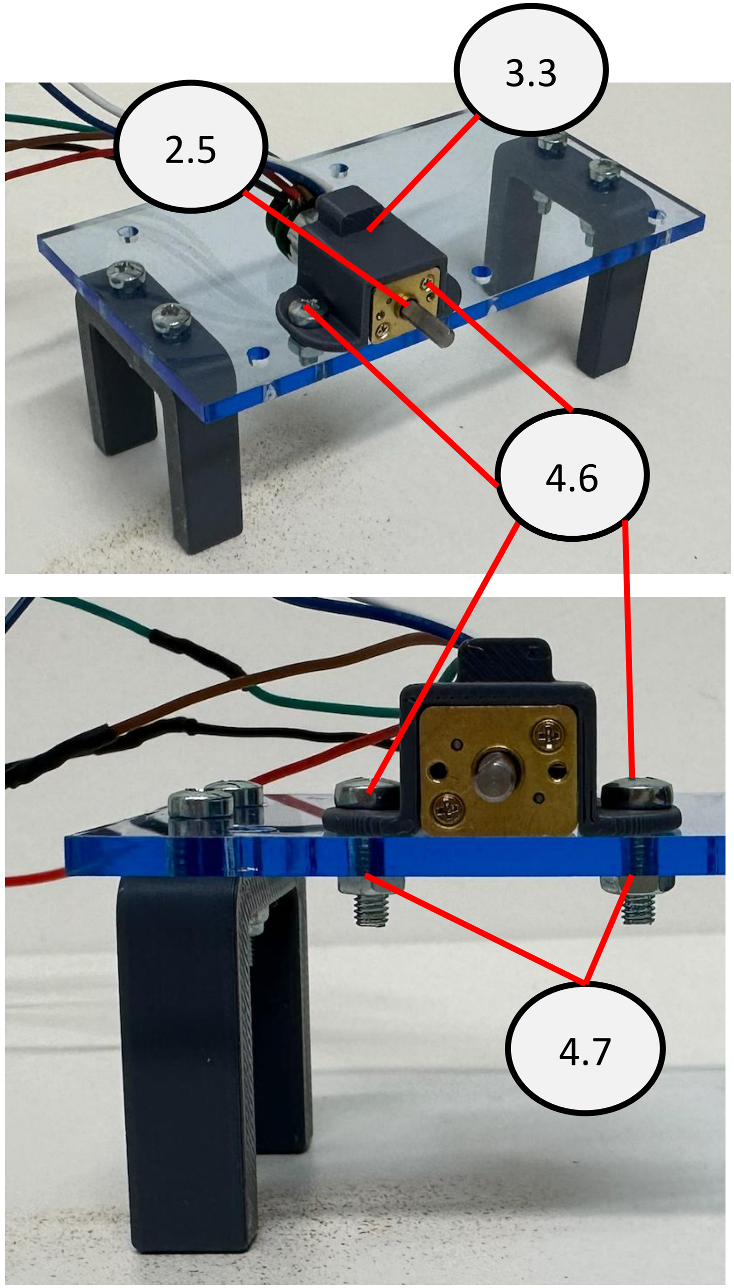


Resolution of 12 impulses per revolution

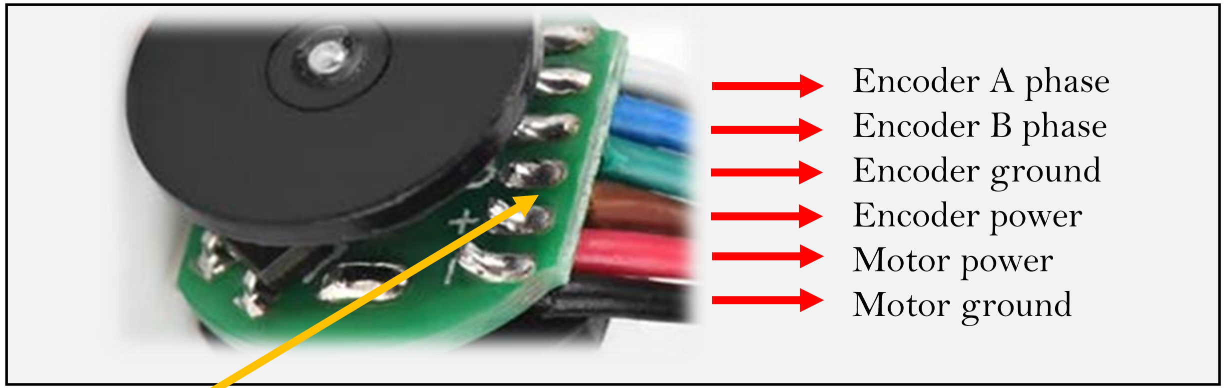
3.1 Introduction

- Set-up the CLB rig as illustrated to the right, using the following components:
  - 2.3: L298N dual H-bridge motor driver
  - 2.5: Brushed geared DC motor with encoder
  - 3.3: DC motor mount
  - 4.1: 4 x hex threaded spacer, 12mm, M3
  - 4.4: 8 x bolt, 6mm, M3
  - 4.6: 2 x bolt, 16mm, M3
  - 4.7: 2 x nuts, M3





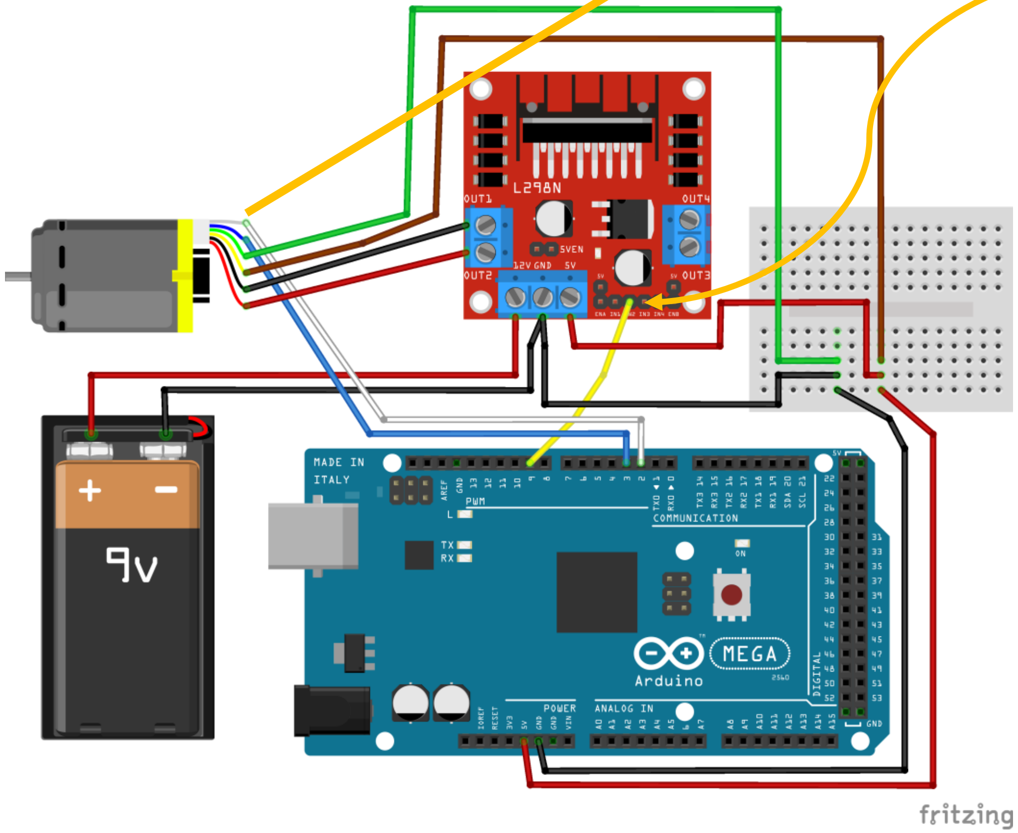
Circuit diagram given here varies ever so slightly to the motor we are using



Note that 'In1' appears in a different position on the H-bridge we are using in class

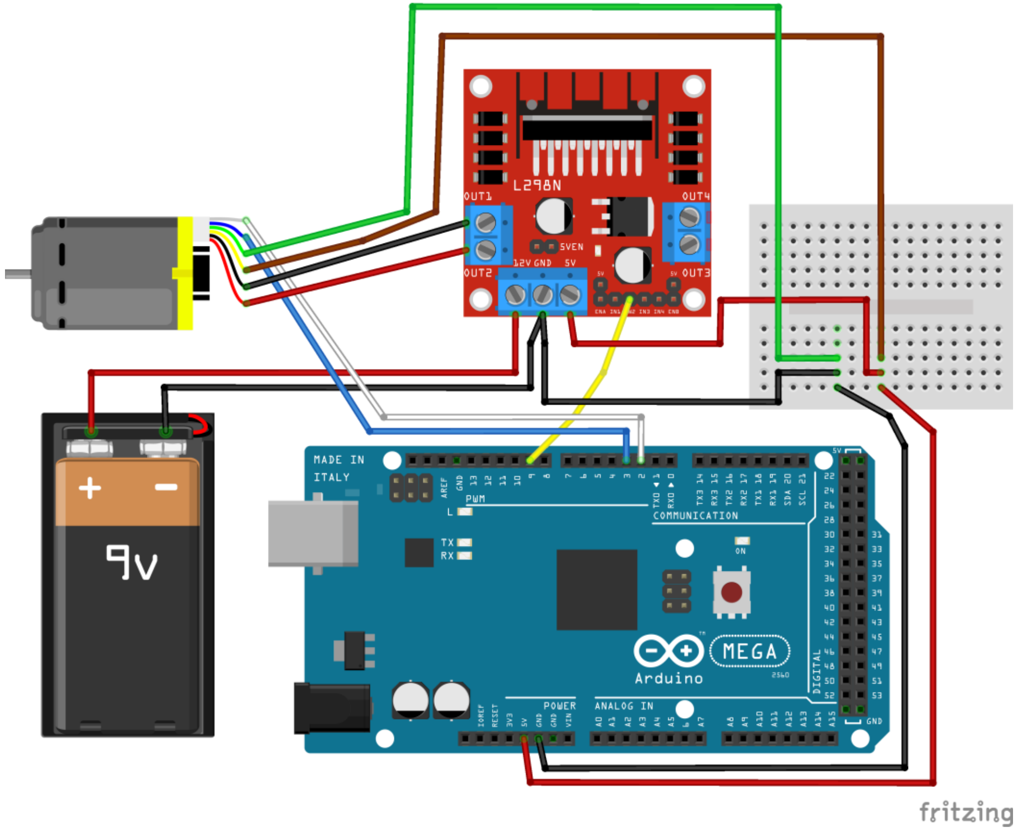
3.2 Hardware

- The task involves connecting a H-bridge and DC motor with an encoder to an Arduino
- Required hardware for the exercise:
  - i. Supported Arduino Mega 2560 board
  - ii. USB cable
  - iii. H-bridge
  - iv. 6V DC motor with encoder
  - v. 9V battery
  - vi. 9V power jack
  - vii. Various wires

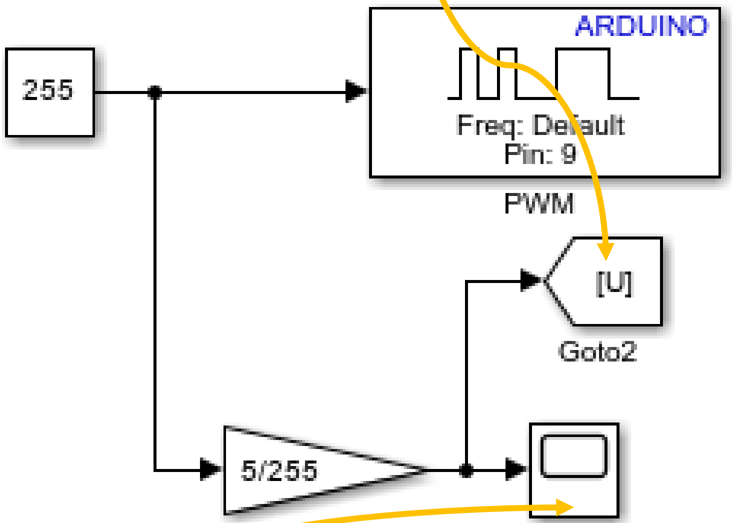


3.3 Algorithm Design

- A 5 Volt step input is applied to the DC motor



‘Goto’ blocks are used to capture the input voltage step data (and on the next Slide to capture the speed output data)



Within the ‘Scope’ block ‘Configuration Properties’, click on ‘Logging’ and select ‘Log data to workspace’ and ‘Save format’ to ‘Structure with time’

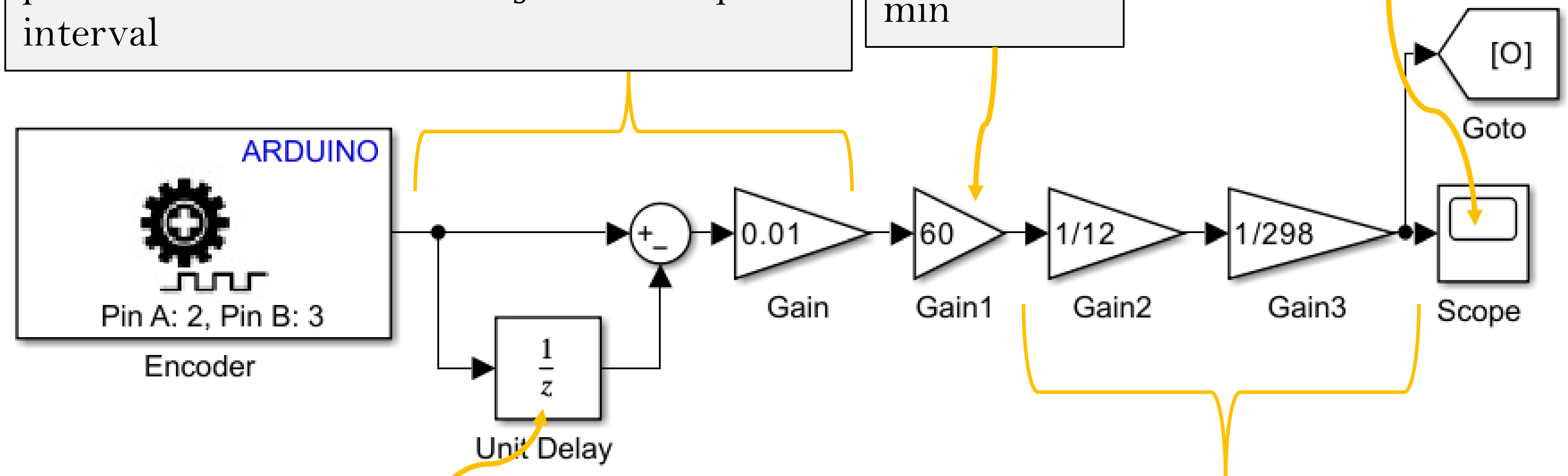
4.3 Algorithm Design

- The encoder on the DC motor is now used to estimate the motor speed based on the encoder counts
- The encoder counts indicate the motors position, hence the motors speed (rate of change of the position) over a specified sample interval can be determined

The change in position (speed) of the DC motor is determined using $\frac{y_k - y_{k-1}}{T_s}$ - the Backward Euler approximation (this converts the signal from counts/second to revolutions/second), where y_k is the current position measurement, y_{k-1} is the previous position measurement and T_s is the sample interval

A gain block is used to convert the signal from revolutions/sec to revolutions/min

Within the 'Scope' block 'Configuration Properties', click on 'Logging' and select 'Log data to workspace' and 'Save format' to 'Structure with time'



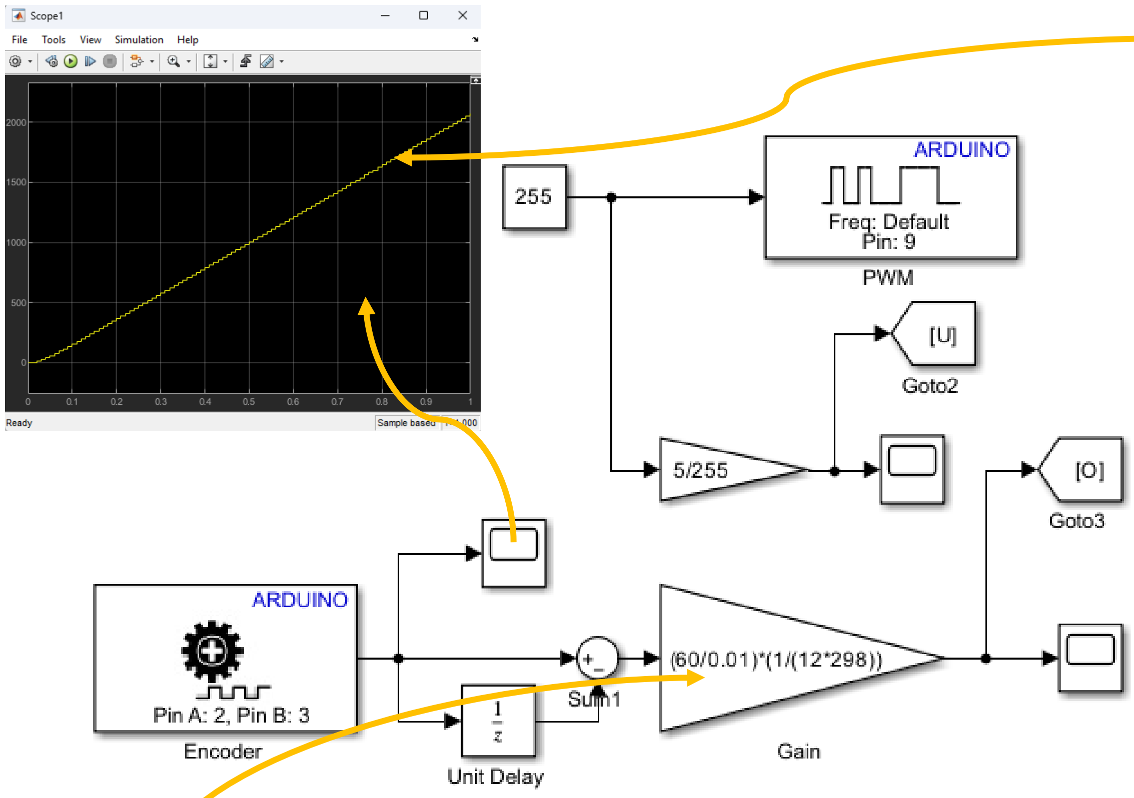
Since the past values of a sampled signal exist, it is normal to use z^{-1}, where z^{-1} means shift to $k-1$, with this given by the following:

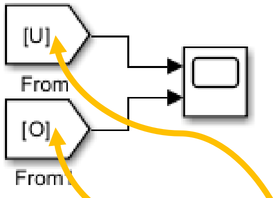
$$z^{-1} y_k = y_{k-1}$$

A gain block is used to account for the gear ratio (i.e., 1:298) and the encoders 12 impulses per revolution

4.3 Algorithm Design

- Before running the algorithm design on the Arduino, ensure the following are set:
  i. Run time of 1 second
  ii. Sample interval of 0.01 second

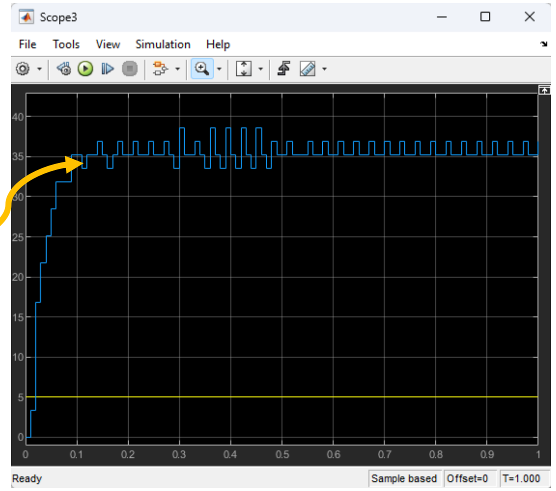




The data observed from the scope illustrates counts/second from the DC motor encoder, with this increasing when the DC motor is rotating in the clockwise direction

The three gain terms from the previous slide are ‘lumped’ together

From observation of the output data (revolutions per minute), the response takes the form of a 1st order transfer function



‘From’ blocks are used to capture the input voltage step data and the speed output data

Contents

DC Motor with an Encoder (Hall Effect Sensor)

3.4 Summary

- The hardware required for capturing the input and output data from an DC motor with an encoder has been presented
- Simulink has been configured to capture/measure the DC motor input and output data