MAE/ECE-5320 LAB 04: DC MOTOR FEEDFORWARD CONTROL AND ENCODER READING

Checklist for Lab04:

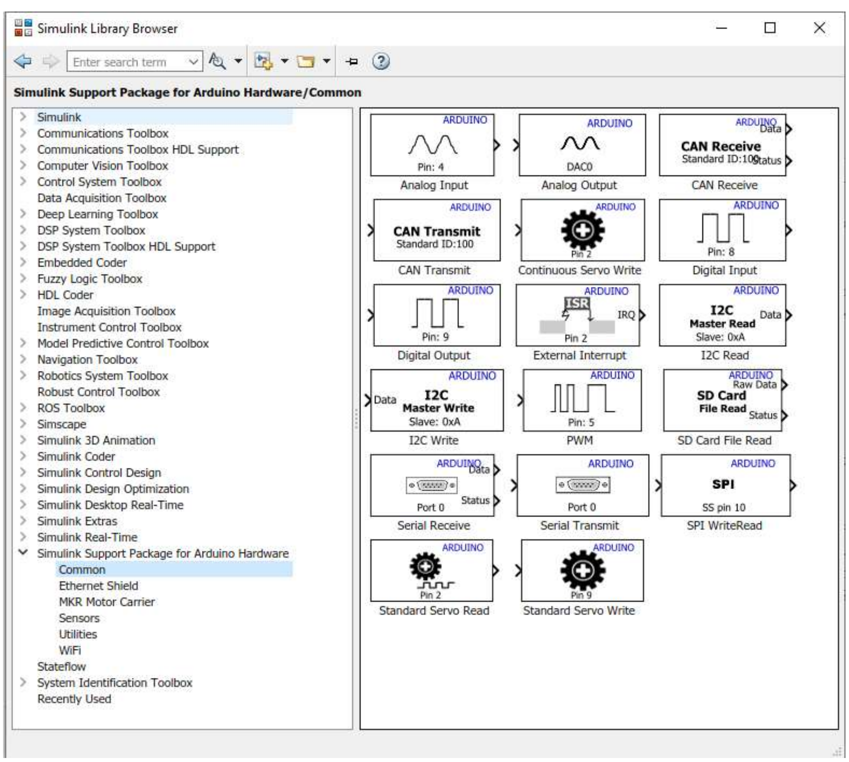
a) **A computer installed with MATLAB compatible with Arduino.**
b) **Uno or others**
c) **Battery compatible for DC motor (11 V provided in lab via portable batter or power supply box)**
d) **Breadboard and jumper wires**
e) **H-bridge motor drive L293**
f) **DC motor with encoder**
g) **2 people max per group, if desired**

Section 1. DC Motor open-loop Control

In the last lab section, the connection between Arduino and Simulink was built. During this lab section, we will realize the open-loop control of a DC motor via an H-bridge and PWM signal.

The H-bridge will be used to control the direction of the motor, and PWM signal is used to control speed of motor by changing the duty cycle.

Now let us have a look at the Arduino blocks from MATLAB support. In the Simulink Library Browser, navigate to Simulink Support for Arduino Hardware, and you will see a bunch of blocks.



MAE/ECE-5320 LAB 04: DC MOTOR FEEDFORWARD CONTROL AND ENCODER READING

Figure 1. Simulink Support for Arduino

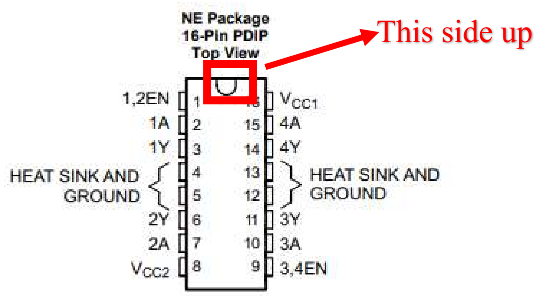
1.1 Control of rotation direction

In the lecture, we covered that the switches of H-bridge can change the directions of the supply current in the motor. In this way, the motor's rotation direction can be changed. In this section, we will use the H-bridge L293 to realize this functionality.

Figure 2. motor driver L293

The technical document can be found here
https://www.ti.com/product/L293
We need to learn the pin maps of L293 in the Arduino kit.

5 Pin Configuration and Functions



Pin Functions

PIN		TYPE	DESCRIPTION
NAME	NO.		
1,2EN	1	I	Enable driver channels 1 and 2 (active high input)
<1:4>A	2, 7, 10, 15	I	Driver inputs, noninverting
<1:4>Y	3, 6, 11, 14	O	Driver outputs
3,4EN	9	I	Enable driver channels 3 and 4 (active high input)
GROUND	4, 5, 12, 13	—	Device ground and heat sink pin. Connect to printed-circuit-board ground plane with multiple solid vias
V_{CC1}	16	—	5-V supply for internal logic translation
V_{CC2}	8	—	Power VCC for drivers 4.5 V to 36 V

Figure 3. pin map and description of L293

MAE/ECE-5320 LAB 04: DC MOTOR FEEDFORWARD CONTROL AND ENCODER READING

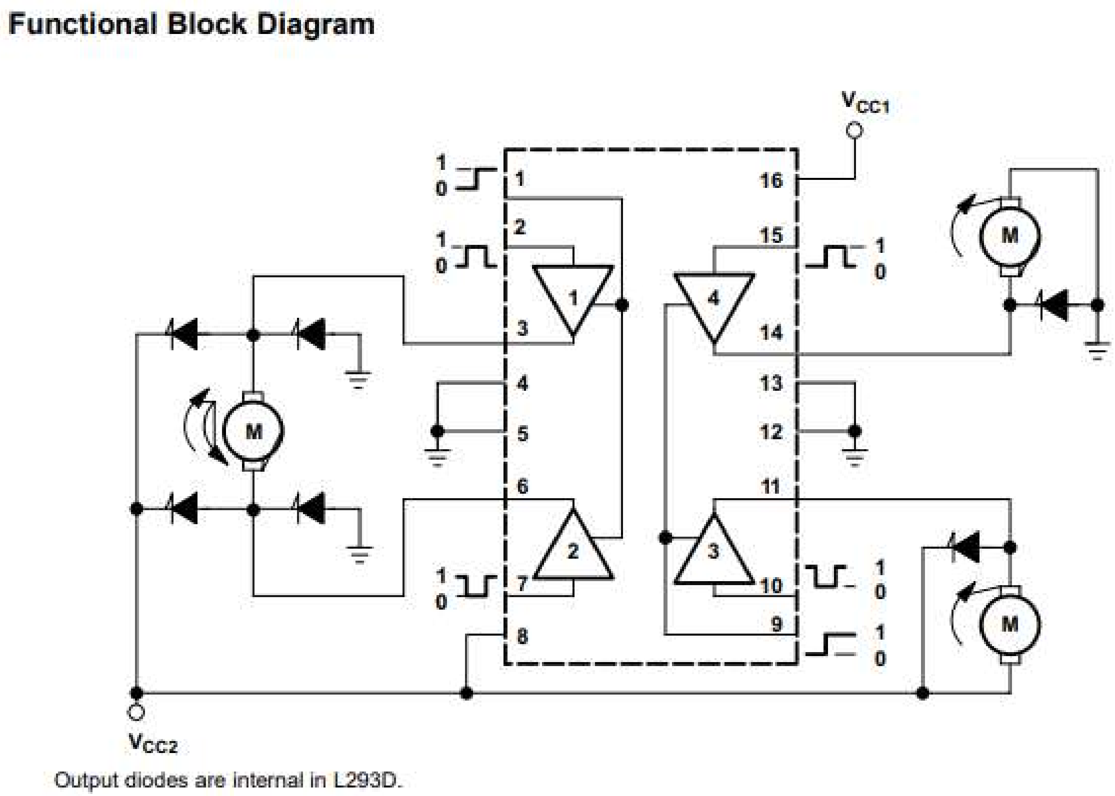


Figure 4. functional block diagram.

We use the Figure 4 as a guide to connect cables. On the left hand side, the connections can realize the bi-directions control, and connection on the right-hand side can only realize the uni-direction control. So, we use the left-hand side connection diagrams.
The functions of driver pins are summarized:

Pin 1: enable and PWM signal
Pin 2, 7: driver input (digital, ON/OFF) from Arduino
Pin 3, 6: driver output to motor to control direction
Pin 4, 5: ground
Pin 8: Vcc (11.1 V) for motor *(supply voltage varies for different motors)*
If a 11 V battery is not available, you may try use 5V output on Arduino, but motor torque and speed will be different with 11V.
Pin 16: Vcc (5V) for driver L293

Next, we connect Arduino and driver with jumper wires. **(unplug Arduino from computer !!!)**
Step 1: place H-bridge with notch facing up.
Step 2: connect Arduino 5V and ground to breadboard +/-
Step 3: connect driver pin 1 to Arduino pin 9. (enable and PWM)

MAE/ECE-5320 LAB 04: DC MOTOR FEEDFORWARD CONTROL AND ENCODER READING

When pin 1 receives 5V, then the driver is enabled. If pin 1 receives a PWM signal, then the signal will be used to control speed.

Step 4: connect driver Pin 2 to Arduino pin 3, driver pin 7 to Arduino pin 2.

If LOW (or 0) is set to Arduino pin 3 (equivalently driver pin 2), and HIGH (or 1) is set to Arduino pin 2 (equivalently driver pin 7), the motor will rotate in one direction. If set oppositely, then the motor will rotate in opposite direction.

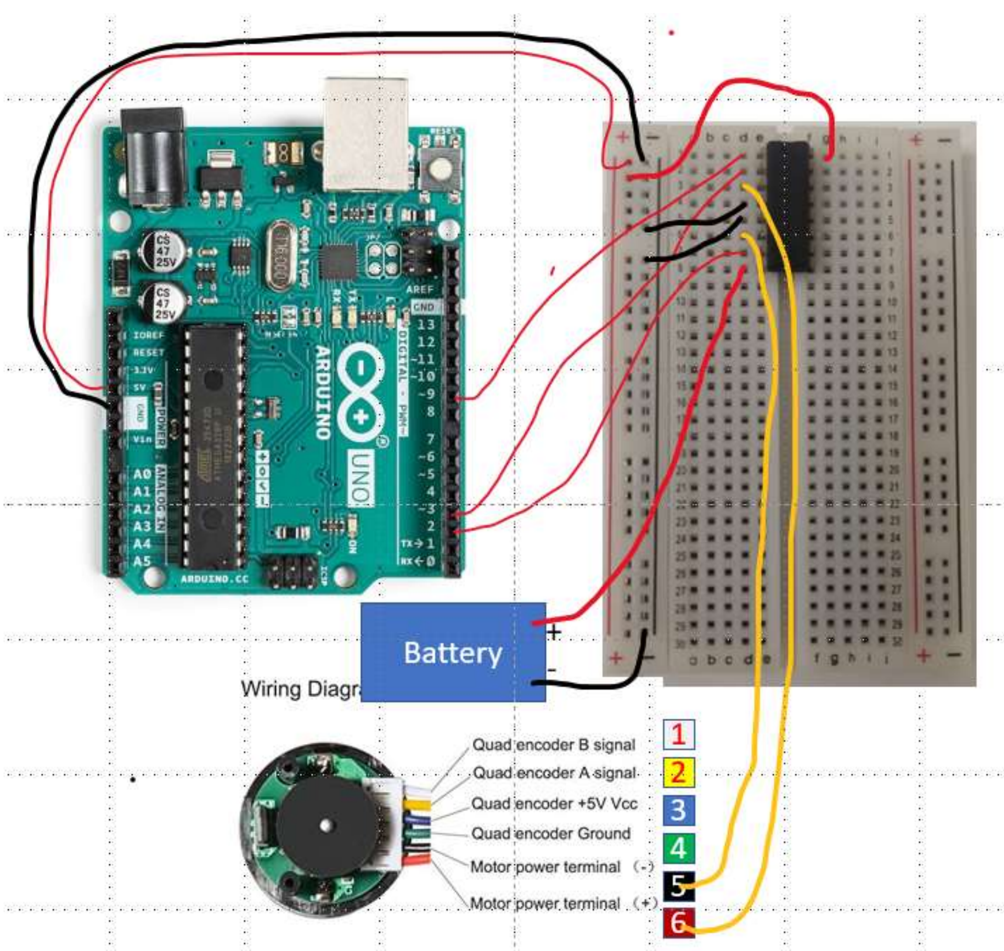
Step 5: connect 5V to supply voltage driver pin 16, driver pin 4 and pin 5 connect to ground.

Step 6: connect driver pin 3 to motor pin 6 (red), connect driver pin 6 to motor pin 5 (black).

Step 7: connect driver pin 8 to battery +, and ground to battery -.

★If using the power supply box (Triple output DC power supply) as your battery, use either the A supply or B supply + and – binding posts to connect to (if we don't have many alligator clips, you can directly screw the Arduino wires into the binding posts on the power box or, to give more wiggle room, you can use banana plugs to screw the wires into to Dr. Cripp's U-channel binding posts). When turned on, you can use the course and fine tuning nobs to adjust your voltage to 11.1V.

★ If using the Arduino 5V power, just connect driver pin 8 to breadboard +.



MAE/ECE-5320 LAB 04: DC MOTOR FEEDFORWARD CONTROL AND ENCODER READING

Next, we build the Simulink model below. Choose ***digital output*** and ***PWM*** blocks under the folder Simulink Support for Arduino Hardware => Common. Then, construct the model below by using ***constant*** blocks, ***display***, and ***manual switch*** block.

Double click to open digital output 1 and change the pin number to 2. Change digital output 2 block pin number to 3. Change PWM block pin number to 9. Set the PWM's constant input value to 100.

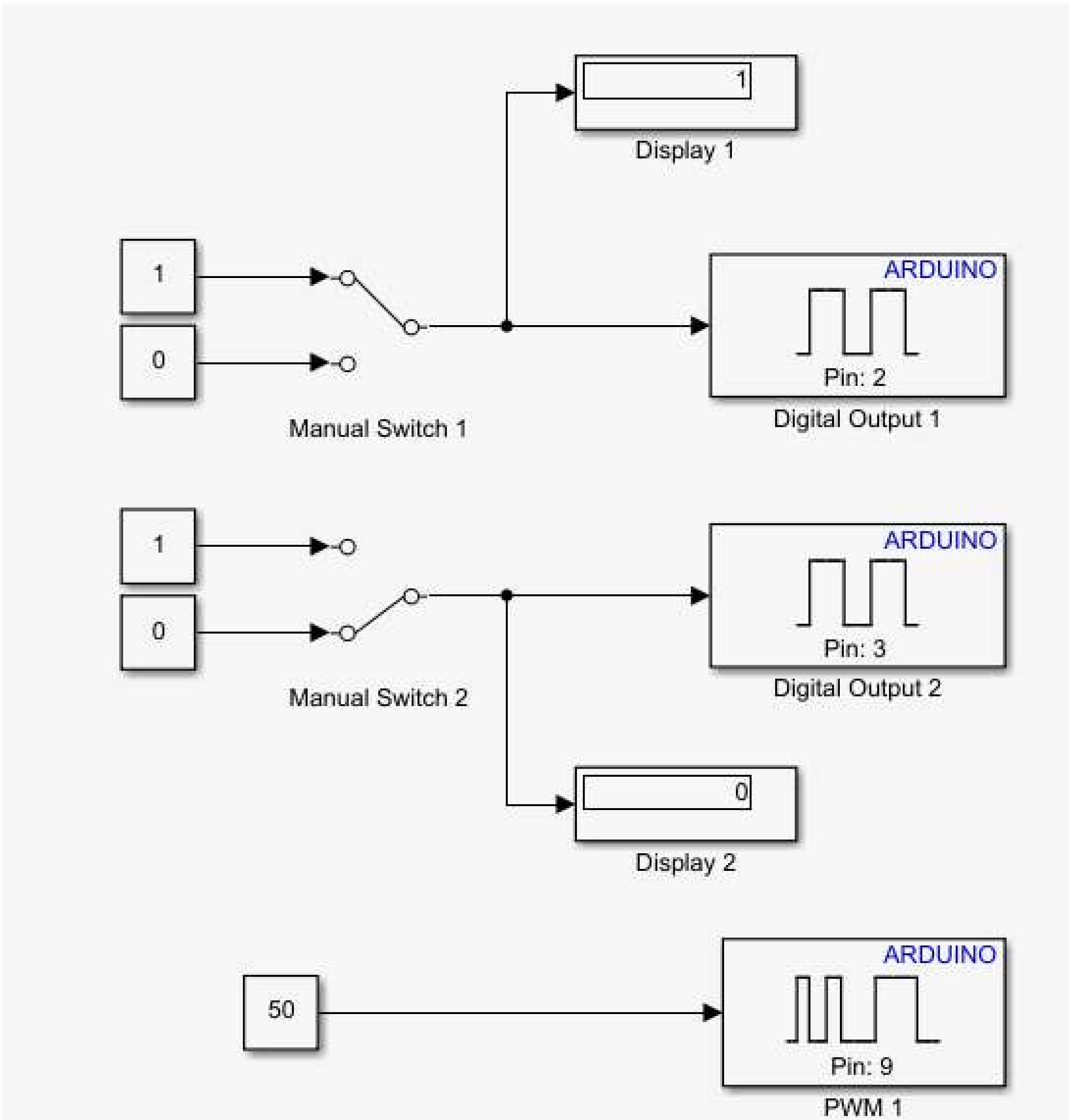
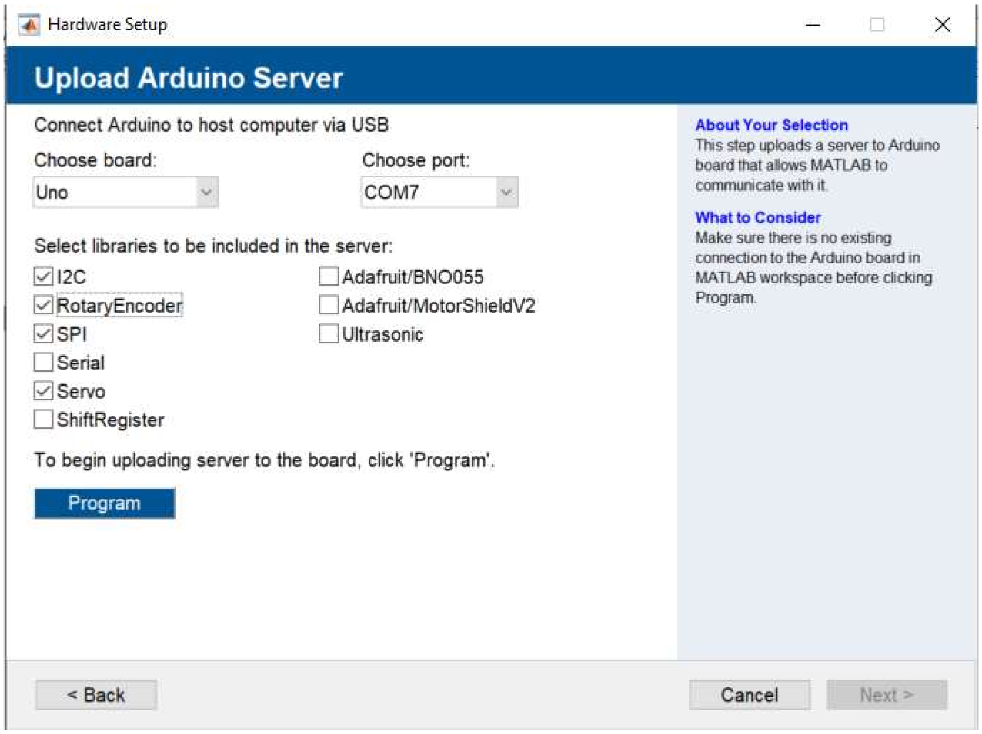
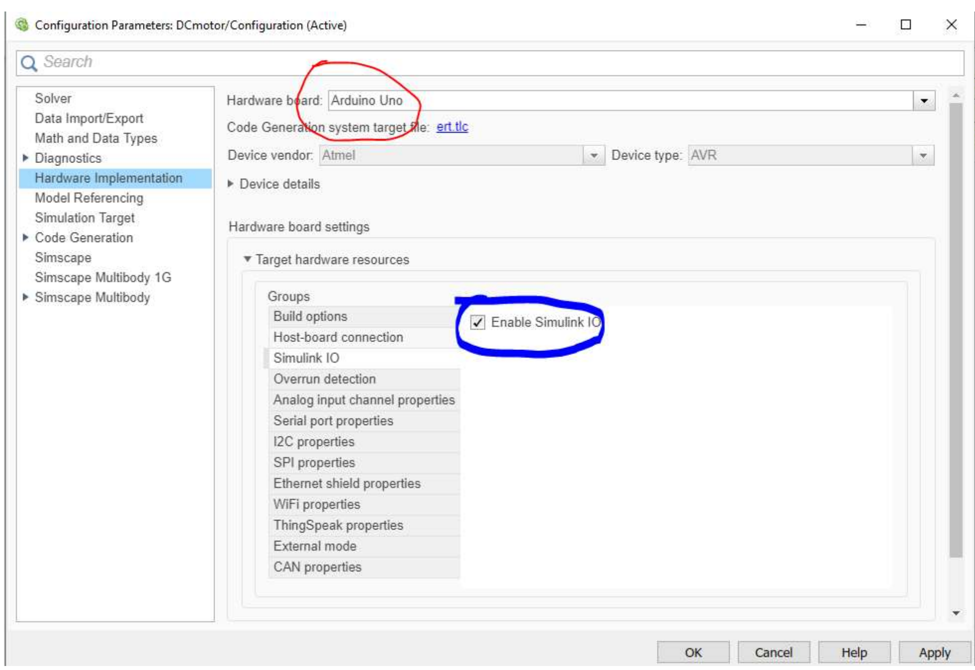


Figure 5. Simulink model for H-bridge direction control.

Plug in the Arduino to computer via USB, and repeat the Arduino setup (>> arduinosetup) in Lab 03 (remember that the COM# & rotary encoder library was selected installed).

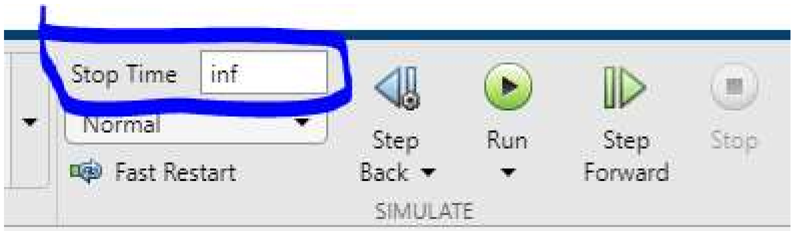


After setup, go to ***'Model Settings'***, change fixed-step size to 0.001. Go to ***'Hardware implementation'*** and change Hardware board to ***'Arduino Uno'.*** Go to ***Hardware board settings***, Check the external mode's Communication interface is serial (XCP on Serial). If using an older version of MATLAB, check the ***Enable Simulink IO***.



MAE/ECE-5320 LAB 04: DC MOTOR FEEDFORWARD CONTROL AND ENCODER READING

Change stop time to inf, then we are ready to click "Monitor & Tune" in the HARDWARE tab and take off.



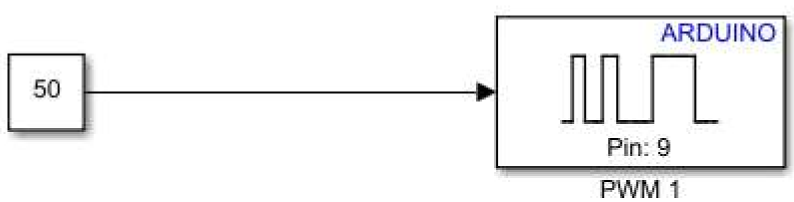
<u>Excise 1: control motor direction</u>

A. As the program runs, double click on the switch pins to switch pin 2 to ON/OFF, and pin 3 to ON/OFF. Look down at the motor with the motor shaft pointing towards you and observe the rotation direction and complete the following table. (put ***clockwise***, ***counterclockwise*** and ***stationary***)

Switch 1	Switch 2	Direction
1	**0**	
0	**1**	
1	**1**	
0	**0**	

1.2 Control of rotation speed

In the lecture, we discussed that the rotation speed is controlled by the PWM duty cycle. So, let us play with changing the duty cycle applied to PWM block.



Double click on the constant block, and gradually change value to 0, 50, 150, 255.
Can you observe the speed difference?

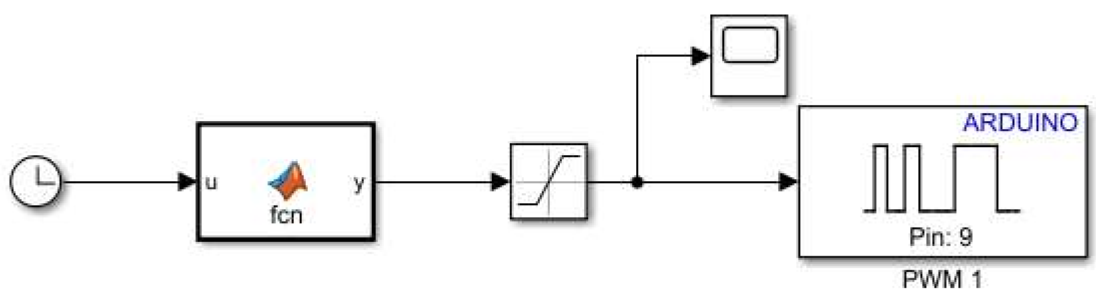
B. Using switch 1 = 1 and switch 2 = 0, List the power voltage you used and describe the speed at each duty cycle value

Power Voltage	

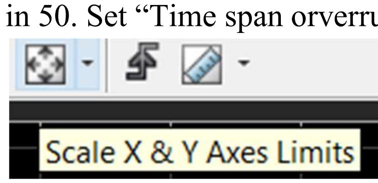
Duty Cycle Value	Description
0	
50	
150	
255	

<u>Excise 2: Open-loop speed control</u>

Replace your PWM block with the Simulink model below using the MATLAB Function block, saturation, and scope.
Set the upper limit of saturation to 255, and lower limit to 0.



Try to write a <u>MATLAB function script</u> to generate the duty cycle function so that open-loop motor speed following $\dot{\theta} = \omega_{max} sin(0.2 * t)$. (Hint: duty cycle 255 is corresponding to maximum rotation speed.)
<u>Attach your MATLAB Function script, block diagram, and scope response. Describe in 2-3 sentences how your motor is acting with respect to the scope in terms of speed amount, timing, and rotation direction,</u>

★Set scope settings: View=> Configuration Properties => Time. In time span select “<user-defined>” then click on the “<user-defined>” words and type in 50. Set “Time span orverrun action to “Scroll”. Apply the changes. Use the scaling button 

on the main scope page to see the full function.
★Clean up: Unplug all wires and components from breadboard and put items back in the small cardboard tray.