ME – 190 MECHATRONICS SYSTEM DESIGN

Lab 6*: DC Motor Speed and Direction – Step Response

Part 1: Motor Step Response

Objectives:

- Use the DRV8833 or SN754410 motor driver and obtain the motor step response
  - In External mode
  - Directly through the serial port
- Understand the logic needed to control the magnitude and direction of a motor
- Observe the different effects of different logic schemes to drive a motor

Background Information:

A motor driver chip converts a lower power signal (from the microcontroller: 5v, 40mA) to a high power signal (9v, 1.5A) to drive the motor. It can be thought of as a switch or relay to the driver supply voltage (9v or 5v in this case). Since the supply voltage is fixed this would result in a fixed motor speed. To regulate the speed a PWM signal is used to quickly switch the output on and off so that the average output voltage can be controlled.

The term motor "driver" is also commonly called "amplifier" or "chopper". The DRV8833 driver is capable of providing 2.7-10.8v at 1.5A RMS on each channel.

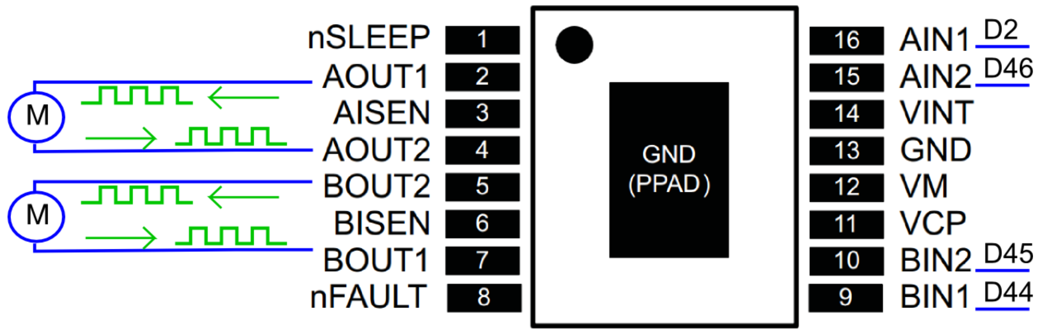


Figure 1: DRV8833pinout/wiring diagram

The SN754410 driver is capable of providing 3.5-36v at 1A on each channel.

*Courtesy of Dr. Joshua Hurst at RPI/MinSeg

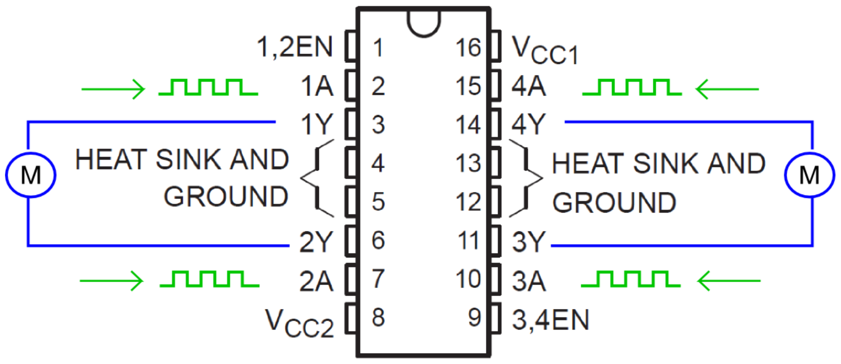


Figure 2: SN754410 pinout/wiring diagram

Simulink Model

Build and run the following Simulink diagram. Use the corresponding encoder/dual encoder block for your system, and the digital pins that correspond to your motor driver. Use the encoder blocks from RASPlib.

Shield or Board	Motor Number	Driver Pins	Encoder Pins
M1V4	1	5, 4	2, 3
	2	9, 11	6* or 0*, 10*
M2V3	1	6, 8	15, A8
	2	2, 5	18, 19
MinSegMega	1	2, 46	PE6, PE7
	2	44, 45	18,19

* Requires Jumpers on J4, J5, J6

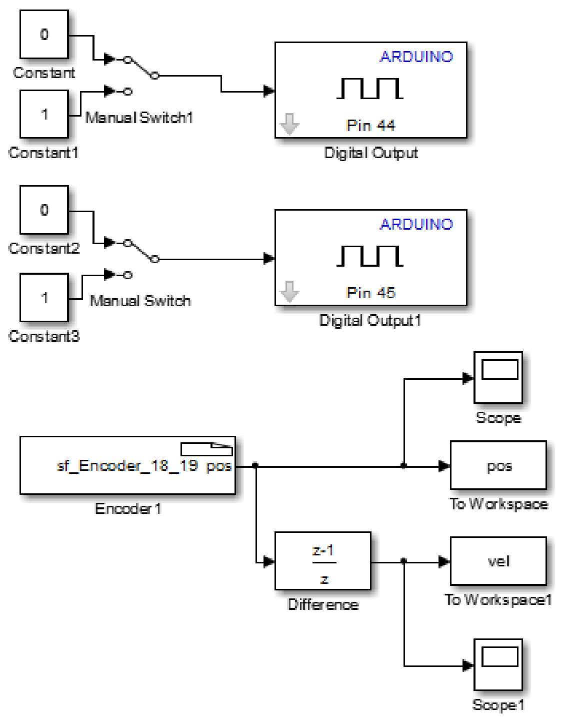
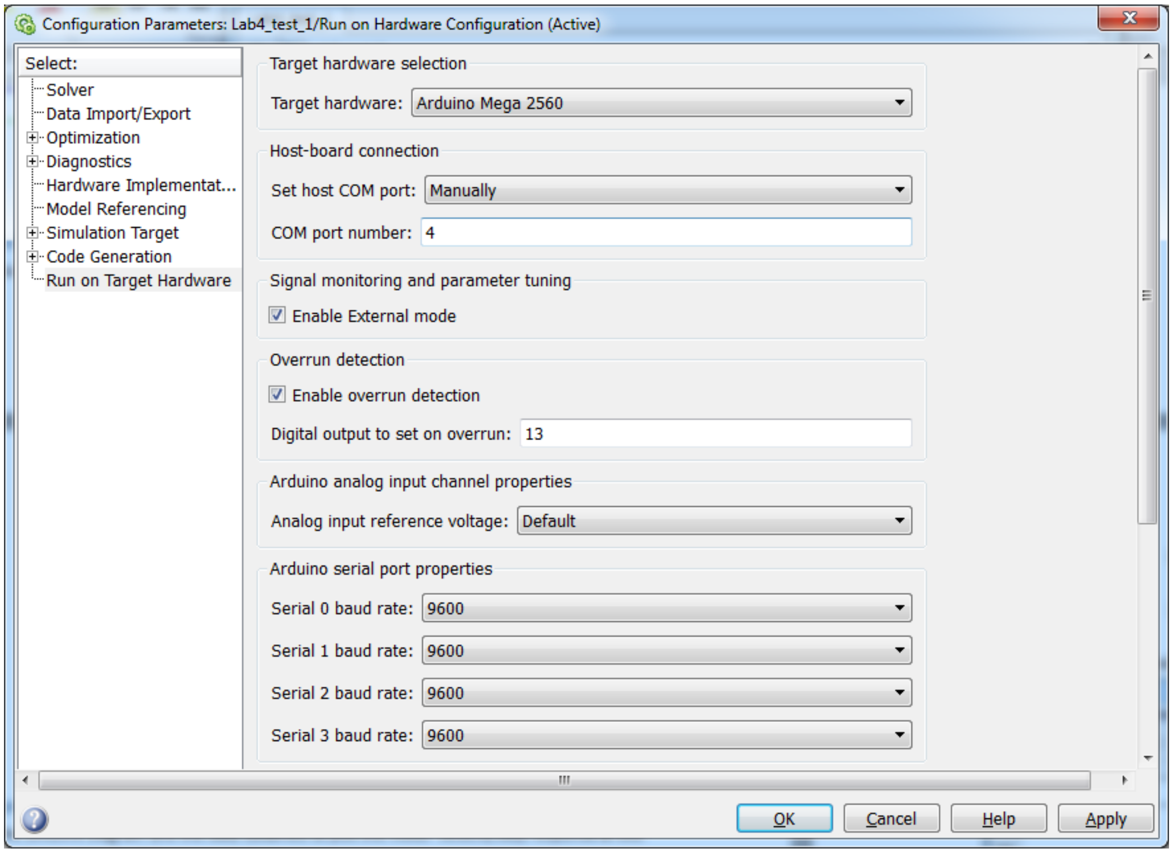


Figure 3: MinSegMega with generic encoder block

- Run the model in external mode to log the data
  - Sampling rate of .03 seconds
- Observe the response of the system
  - toggle the manual switches
  - toggle the On/Off switch on your board – this will determine the voltage source of the driver chip (VM):
    - OFF - USB voltage (~4.5 volts)
    - ON – Battery voltage (~9v fully charged)
- Stop/disconnect from the system, save the data then plot the results
  - The commented line below stores the variables 'pos' and 'vel' into the data file ext_dat.

```
%save ext_dat pos vel    % save the data
load ext_dat             % load the data
figure, plot(pos)
figure, plot(vel)
```

The velocity data obtained in external mode will be noisy. The primary reason seems to be that the time between samples is not actually fixed/constant. In the configuration parameters if you check the box "Enable overrun detection" then the digital pin you specify will be set high if the controller cannot execute your code in the sample time you specify. Check this box, select pin 13, and connect to the device. You will notice that the LED on pin 13 is always on. This indicates that in external mode the microprocessor has trouble meeting the sample time. This implies the time between each sample is not exactly 30 milliseconds. Any calculation assuming a fixed sample time, such as velocity, will then contain some error (noise) due to the sample time not being constant. Note that this noise is from the measurement system - not actual noise present in the signal or system.



Target hardware selection

Target hardware: Arduino Mega 2560

Host-board connection

Set host COM port: Manually

COM port number: 4

Signal monitoring and parameter tuning

☑ Enable External mode

Overrun detection

☑ Enable overrun detection

Digital output to set on overrun: 13

Arduino analog input channel properties

Analog input reference voltage: Default

Arduino serial port properties

Serial 0 baud rate: 9600

Serial 1 baud rate: 9600

Serial 2 baud rate: 9600

Serial 3 baud rate: 9600

- The actual motor velocity will not be fluctuating like the "noisy" data indicates. To make the data more useful use the "`smooth`" command in Matlab (if available). Type "`help smooth`" to obtain details on the smoothing method
  - `vel_smooth=smooth(double(Vel), 10)`
    - This function will use a moving average over 10 samples to smooth the velocity data
    - `double` – this converts the int16 values in Vel to doubles. If this is not done the function will complain

Questions:

- Provide a plot that contains the velocity step response. Provide the raw data and a smoothed response. Plot the velocity in RPM and time in seconds. An example plot (for two different voltage steps) is shown below for an unknown motor with unknown units:

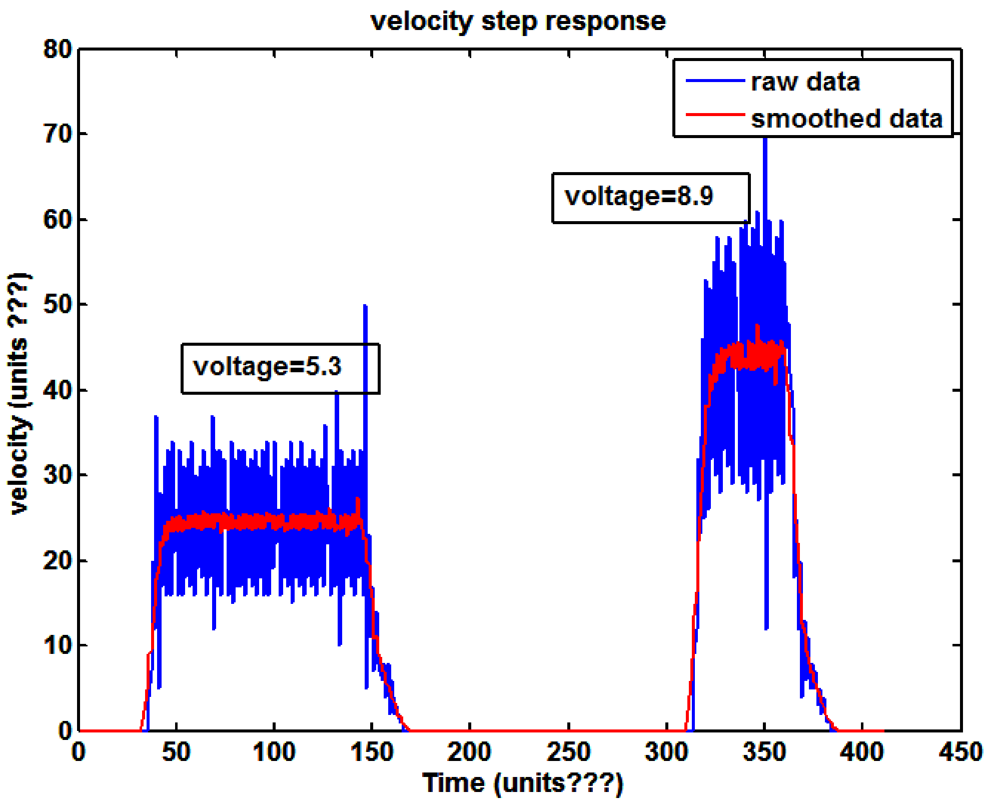


Figure 3: Step Response Graph

Steady State Data for Parameter Identification (Optional)

The DC motor parameters can be determined from steady-state voltage and current measurements.

- From the pinout diagram for your system find the jumper that connects the driver to one of the motor terminals
  - Leave the jumper in so the motor is spinning at a constant velocity – use a multimeter to measure from the ground on the batter terminal block to the pins for the motor jumper. NOTE: the voltage measured at this motor lead will only be a meaningful value when the motor is spinning in one direction, in the other direction this motor lead will be ground and will have a voltage around zero. If your voltage is not near the expected

value reverse the direction of the motor. This is because the jumper is connect to only one of the motor leads.

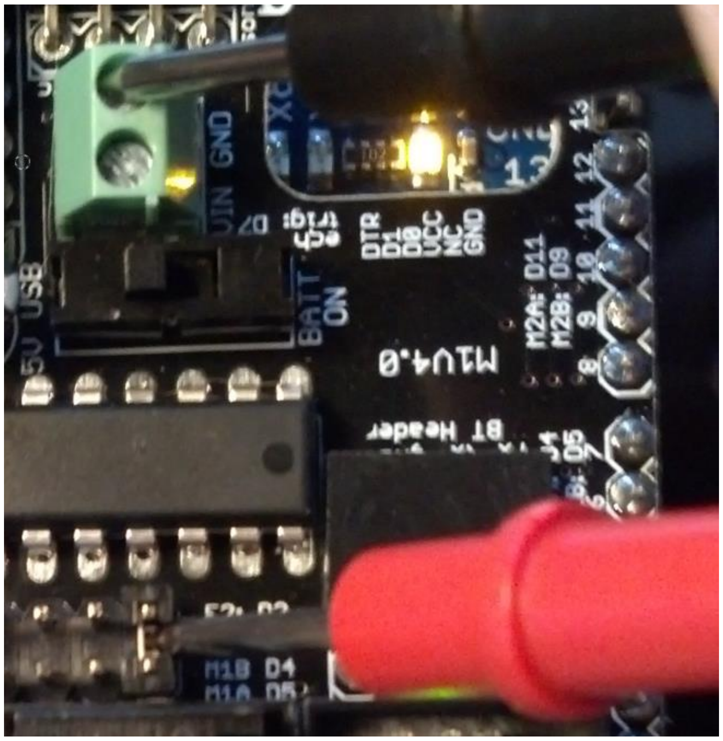


Figure 4: M1V4 Voltage measurement jumper

- Before you can measure the current determine the maximum current your multimeter can read. At steady-state most small DC motors will be less than 200ma.
- Remove the jumper and use a multimeter in current mode to complete the circuit and allow the motor to spin at a steady state speed and measure the current. Be careful not to short the multimeter probes when doing this. It is recommended you use jumper wires to reduce the risk of shorting!

- When the motor is running at a steady-state constant velocity record the steady state:
  - velocity average (with correct units)
  - voltage with a multimeter
  - current with a multimeter

Questions:

- What is the maximum current your multimeter can read?
- When powered from USB what is the steady state speed, current, and voltage (include units)?

Velocity: _______ Voltage: _______ Current: _______

- Annotate the plot with the steady state current, and voltage on the step response graph like the graph in Figure 3.

Part 2: Step Response Data from Serial Port

The max velocity of the NXT motor is about 170 RPM at 9volts or about 18 radians/second. Since there are 720 pulses per revolution from the encoder (quadrature decoding), this is about 2040 counts/second. If we sample the system every .03 seconds this would give a maximum of 2040*.03 =61.2 counts every sample time. Since this the maximum value for the velocity we would expect, and this number is below 255, we can send the velocity data as uint8 data type.

Obtain same velocity information directly from the serial port:

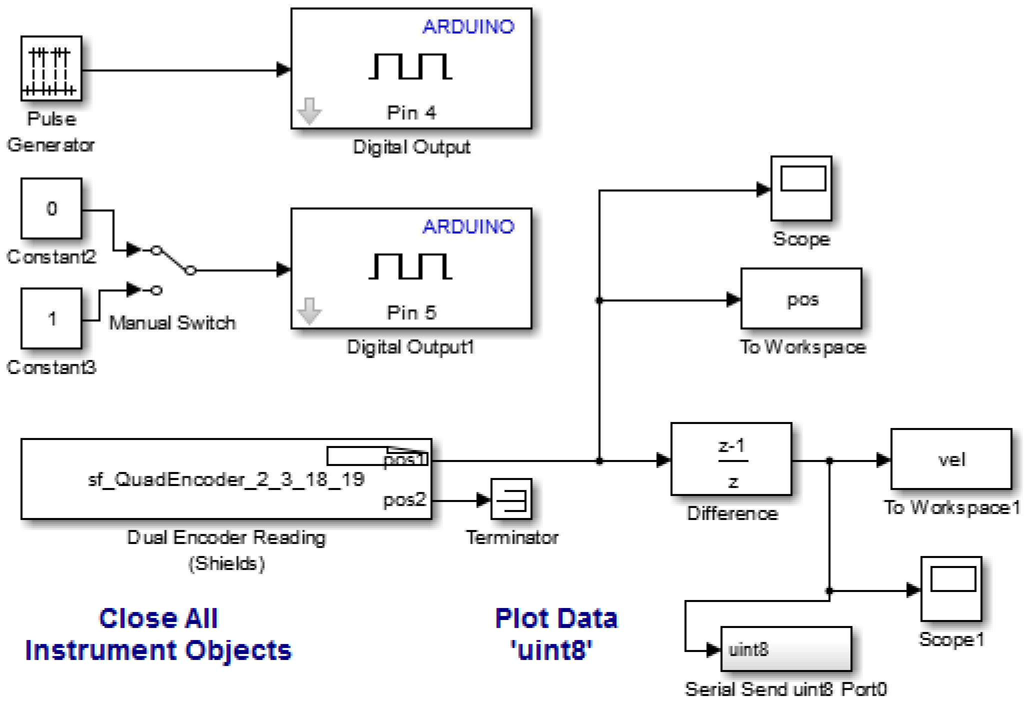
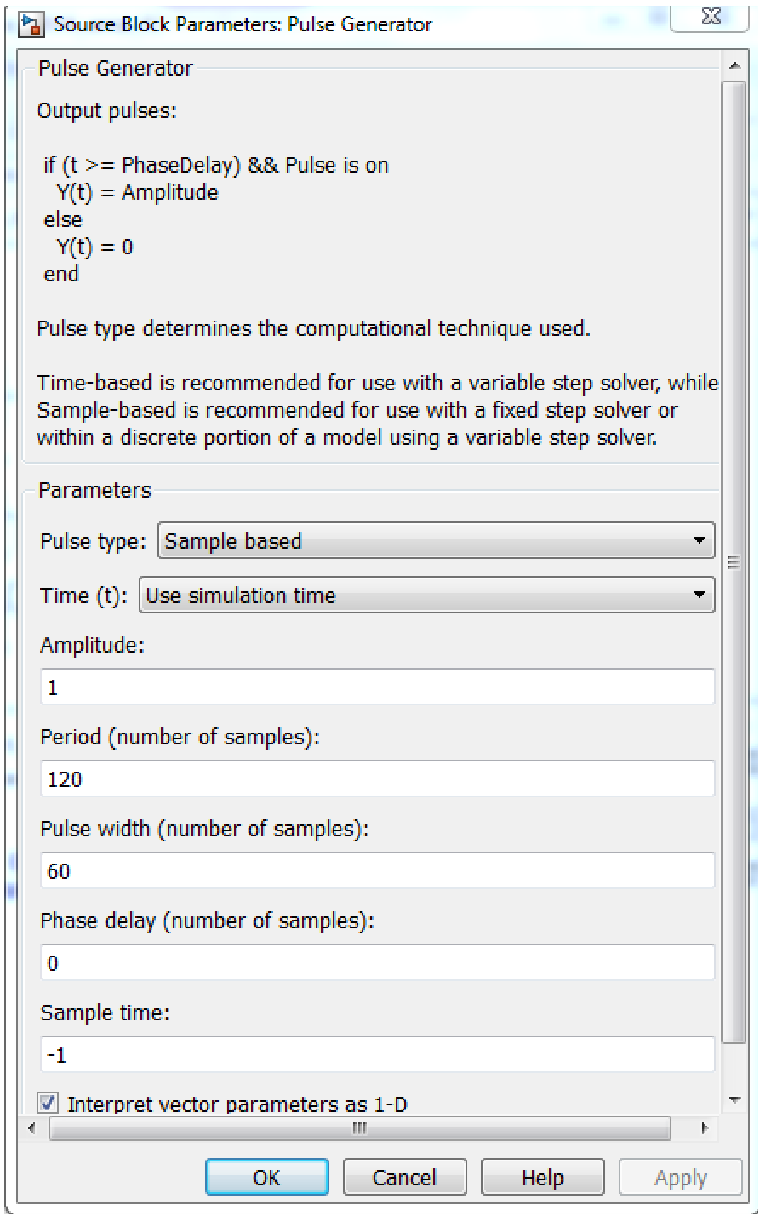


Figure 5: Serial Step Response

- Use the "Serial Send uint8 Port0" block and the corresponding "Plot Data 'uint8'" block to send and plot the data. Don't forget to right click the plot data block and change the COM port and number of samples to the right values if you want to change the default values.

- The “Close All Instruments Objects” is used to ensure all the COM ports Matlab is trying to access are closed. Click this if for some reason you cannot download code to your board (because Matlab has it open).
- Be sure external mode is **not** checked
    - 2013a – checkbox in the configuration parameters
    - 2013b-2015a– drop down menu on the toolbar ribbon
- Toggle the ON/OFF switch on the board to obtain the step response at USB voltage and at battery voltage if desired
- The switched shown in Figure 5 cannot be toggled when external mode is not used so use a pulse generator block to create the step input:



Source Block Parameters: Pulse Generator

Pulse Generator

Output pulses:

```
if (t >= PhaseDelay) && Pulse is on
  Y(t) = Amplitude
else
  Y(t) = 0
end
```

Pulse type determines the computational technique used.

Time-based is recommended for use with a variable step solver, while Sample-based is recommended for use with a fixed step solver or within a discrete portion of a model using a variable step solver.

Parameters

Pulse type: Sample based

Time (t): Use simulation time

Amplitude: 1

Period (number of samples): 120

Pulse width (number of samples): 60

Phase delay (number of samples): 0

Sample time: -1

☑ Interpret vector parameters as 1-D

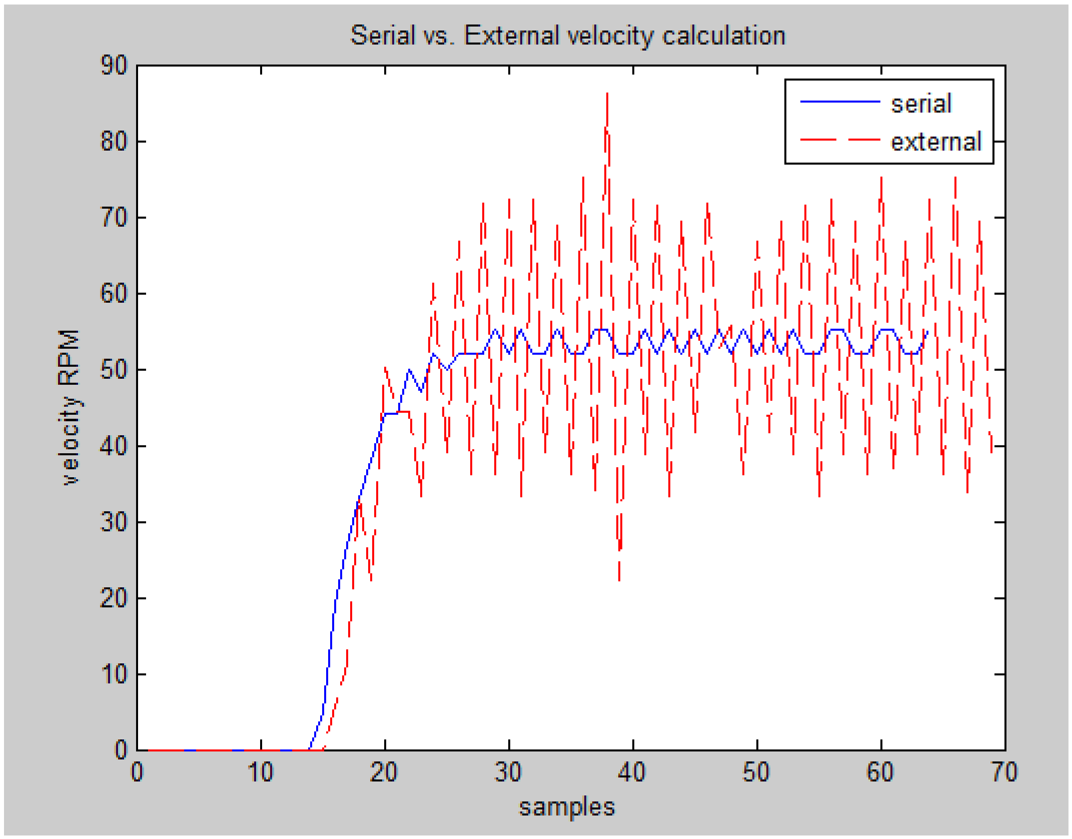
- Also use this model in external mode to obtain the data in the pos and vel variables
    - When the system is running toggle the ON/OFF button to get the response at ~9V in addition to the USB voltage

Use the following snippet to help plot this data and the previous data to observe the differences. You might have to modify and adjust your data appropriately.

```
%save ext_dat pos vel    % save the data
load ext_dat             % load the data
figure, plot(pos)
figure, plot(vel), hold on

% plot serial data:
% save ser_dat FullDat
load ser_dat
plot(FullDat, 'r')
legend('external mode', 'direct serial data')

% shift data so the the start of the step lines up
% only plot 100 data points
figure, hold on
plot(vel(118:118+100))            % external data
plot(FullDat(107:107+100), 'r') % serial data
legend('external mode', 'direct serial data')
```



Notice how the velocity obtained in serial mode is better, this affirms the hypothesis that the calculation in external mode may be affected by a varying sampling time due to the processing overhead associated with external mode.

Questions:

- Obtain the step response of motor in serial mode at a time step of .03 seconds – be sure to keep this data to plot later
- Obtain the step response of the motor in serial mode at a time step of .003 seconds – be sure to keep this data to plot later. You will need to modify the pulse generator parameters so the motor stays on for a longer time.
- Plot these two data sets (serial mode .03 and serial mode .003) on the same axis – you will have to modify the data so that they are both plotted on the same scale (you should use seconds instead of samples, and use the same velocity units). Explain why these look so different!
- KEEP This data - you will need this in future labs!

Part 3: Motor Logic: Direction and Magnitude

To regulate the speed a PWM signal is used to quickly switch the output on and off so that the average output voltage can be controlled. In addition the correct switching logic needs to be implemented so that the motor can change direction based on the sign of the input.

The subsystem block below is used to correctly output the correct magnitude and logic of the PWM if the only input is a scalar input voltage (which could be negative). You do not have to make this Simulink diagram – it is shown for reference only.

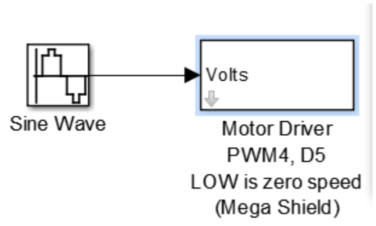


Figure 6: Motor driver block to control magnitude and direction

The contents of this subsystem are shown below:

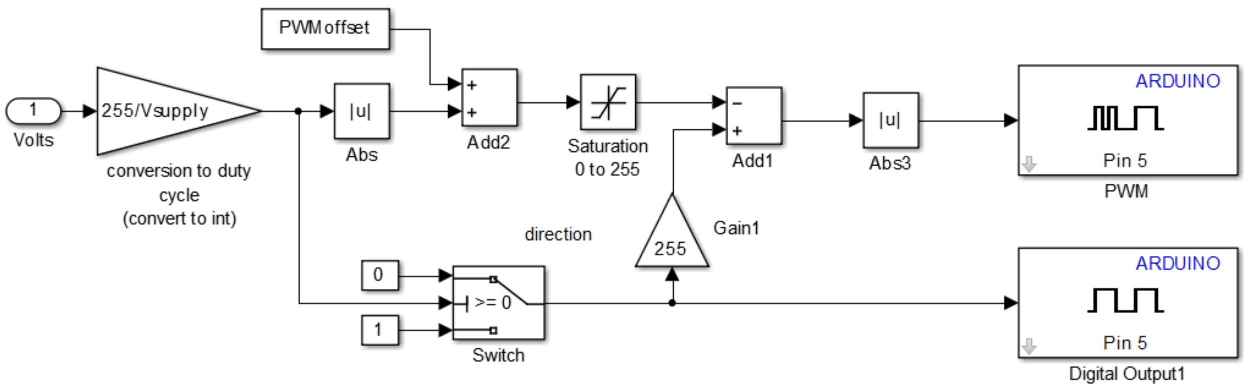
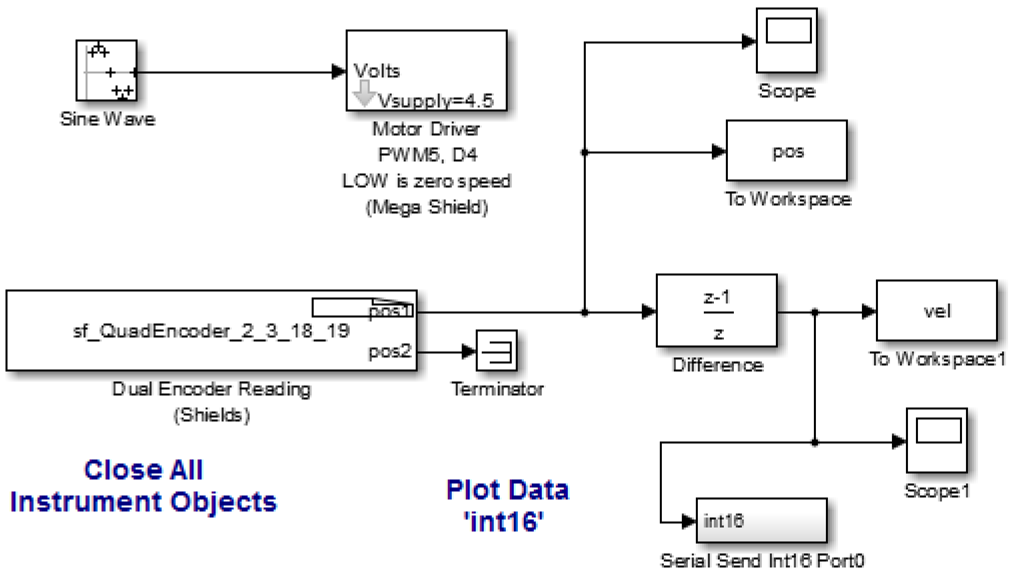


Figure 7: Logic for controlling magnitude and direction of motor voltage (for reference only - do not build this diagram)

- Use this subsystem block in Figure 6 to generate a sinusoidal motor response:



- Edit the Subsystem parameters to correctly specify the driver supply voltage (4.5 if power is from a USB) and a PWM offset – normally zero.
- Set the parameters of the sine wave so it has a magnitude of 4.5 volts and a period of around 3 seconds.

Questions:

- Provide a plot of the sinusoidal velocity in external mode at .03 seconds and by obtaining data directly with the serial port at .03 seconds.