

ME-190 MECHATRONICS SYSTEMS DESIGN

Lab 3*: Getting Started with MinSeg

Lab Objectives

- Connecting the MinSeg robot to Simulink and blinking its onboard LED
- Communicate with the target board (Arduino) using external mode by changing the brightness of an LED with PWM
- Control the brightness of a LED using pulse width modulation (PWM) based on the position of encoder

Support Packages:

To connect the MinSeg robot to Simulink, we have installed the “Simulink Support Package for Arduino Hardware”, as well as the “Rensselaer Arduino Support Package Library (RASPLib)” on the lab computers. If you would like to connect the robot to your personal computer you need to install these two libraries (Please see the appendix file for details).

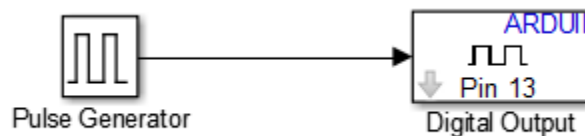
***Important:** Before opening Simulink, add the following directory to Matlab’s main path through Matlab’s “Set Path” tool:

C:\ME-190\RASPLib

This will make Simulink have access to the Rensselaer Arduino Support Package Library.

Simulink Setup:

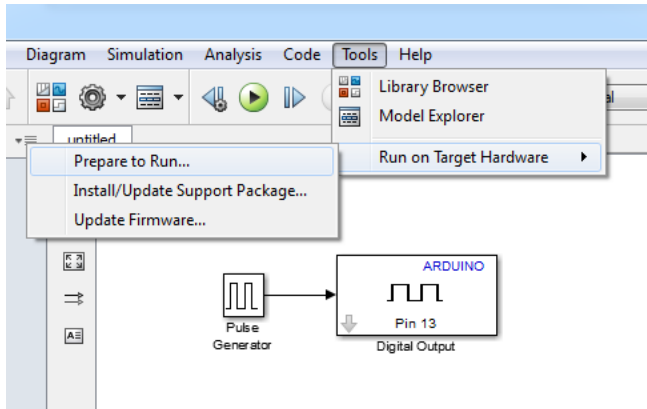
In order to ensure that your computer can properly communicate with the Arduino board, build and run the following Simulink diagram using the steps below:



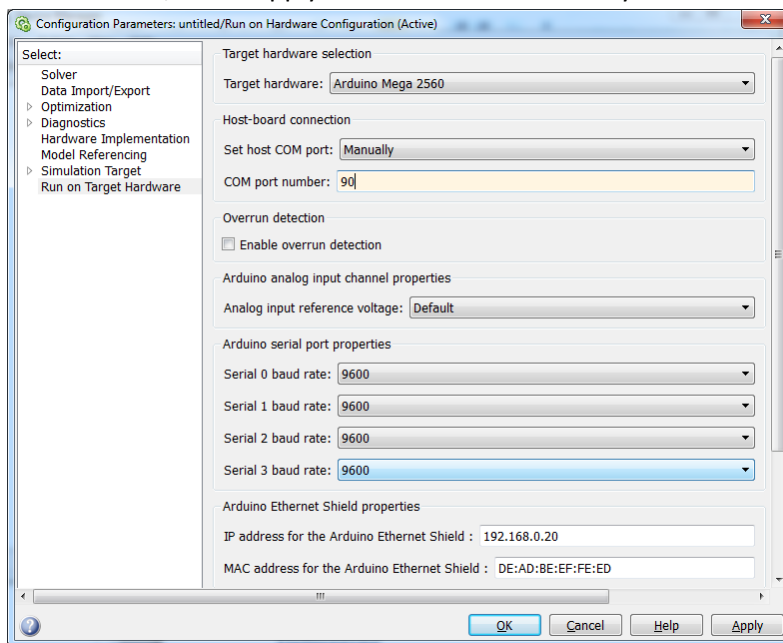
- 1) Open MATLAB 2015a then open a new Simulink model.
- 2) Find and assemble the required blocks.
 - Pulse Generator is under View->Library Browser->Simulink->Sources
 - Digital Output is under View->Library Browser->Simulink Support Package for Arduino Hardware->Common
- 3) Change the output pin to 13, which is connected to the onboard LED. This is done by double clicking on the Digital Output block.


* Courtesy of Dr. Joshua Hurst at RPI/MinSeg.

- 4) Double click on the Pulse Generator to change the amplitude to 1, the Period to 10, and the pulse width to 50%.
- 5) Under the tools menu go to 'Run on Target Hardware' and click 'Prepare to Run'. In the window that pops up you will need to select Arduino Mega 2560 in the drop down for Target Hardware.

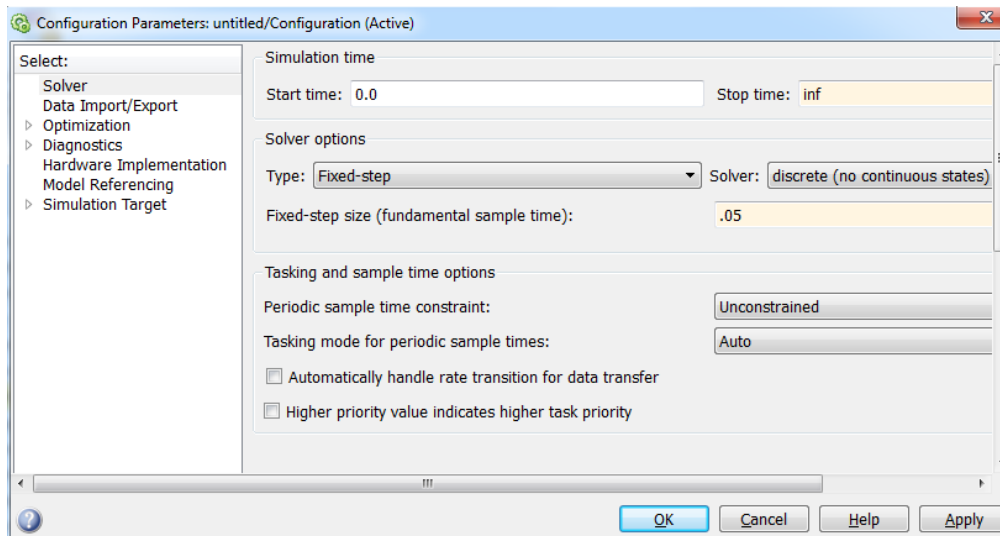


- “Set host COM port: “ should be set to “Manually” and specify the COM port your board is connected to, click “Apply” and “Ok”. Automatically does not work reliably on some computers.



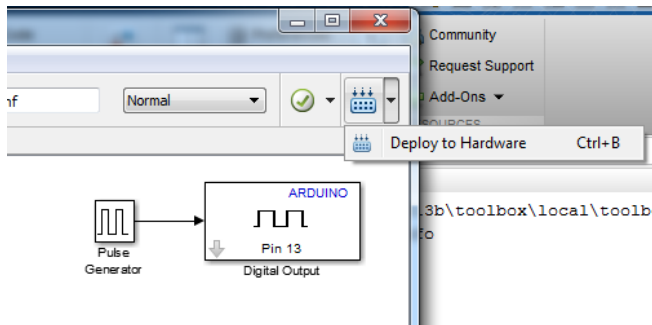
- Right click on the background and select 'Model Configuration Parameters', or .
- go to 'Solver' on the right hand menu and type 'inf' into the 'Stop time' spot so that the model will keep running. Make sure the solver options are set to “Fixed-step” and “discrete (no continuous states)”. The “Fixed-step size” should be 0.05. This is how fast in seconds the

control loop will execute. In this case every 50 milliseconds it will execute the Simulink code.



6) Compile and download the code to the board:

- Click “Deploy to Hardware Button” or the keystroke Ctrl+B:



At this point the on board LED connected to pin 13 should start blinking on and off. If not, check the debugging section below.

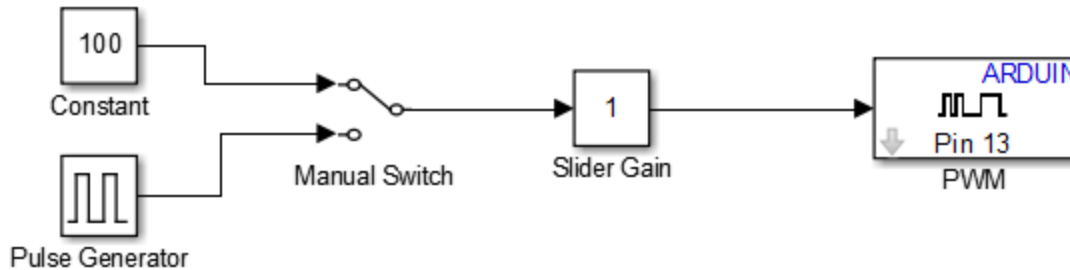
Part 2: Communicating with External Mode

Objective

- Communicate with the target board (Arduino) using external mode by changing the brightness of an LED with PWM

Simulink

Modify your Simulink diagram to the following



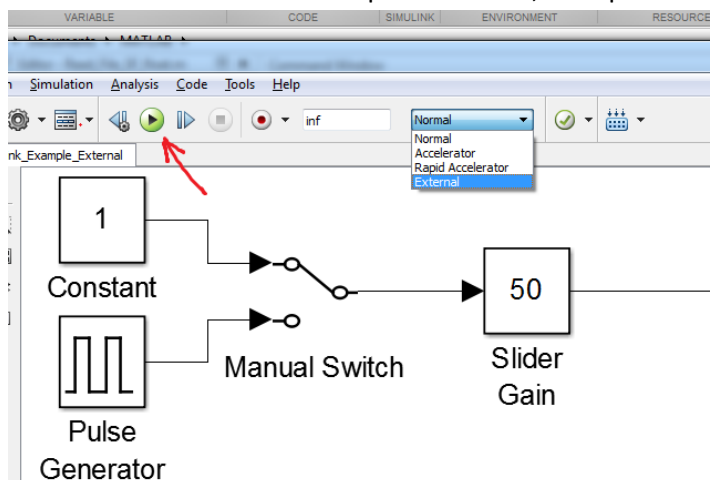
- Pin 13 can also be set as a PWM instead of just an on/off digital output – replace the Digital Output block with the PWM block
- The “Slider Gain” can be found in View->Library Browser->Simulink-> Math Operations. Change the high value to 255 (the max value of the PWM)
- The “Manual Switch” can be found in View->Library Browser->Simulink ->Signal Routing
- The “Constant” can be found in View->Library Browser->Simulink ->“Commonly Used Blocks”

External Mode

- External mode allows bi-directional communication to/from the application board to the PC.
- When external mode is selected it downloads additional code to the board to allow this communication so this does increase the program size.
- When you use external mode you can change parameters while the system is running
 - However this has an impact on the performance: Due to the communication bandwidth, the fastest you can run in this mode and still meet your control loop time is approximately 30 milliseconds
- The values change on the Slider Gain and the Manual Switch can be changed while the program is running.
- The code can be run much faster (in the orders of 1 or 2 milliseconds) if external mode is not used.

Enabling External Mode:

- Select “External” from the drop down menu, then press the “Play” button



Run the code and experiment with the setup. Observe the effects on the Arduino LED when changing the value of the Slider Gain, Manual Switch, and the Pulse Generator.

- While running the simulation, you can change the position of the switch by double clicking on the switch
- You can change the gain of the Slider Gain while running the simulation by double clicking it.
- Adjusting the Pulse Generator, experiment with all three parameters. Which increases brightness? What is the approximate range over which you can observe a difference?

Stopping the Simulation:

When you are running the simulation in external mode you are connected to the device and are sending data and receiving data. When this is occurring do NOT use the stop button to stop the simulation:

- Use the Play button to download and run the code in external mode
- Use the square Stop button to stop

Checkpoint 1:

To complete this section of the lab you will need to show the ability to control the blink of the LED in one program with both the 'manual switch' and the 'slider gain'.

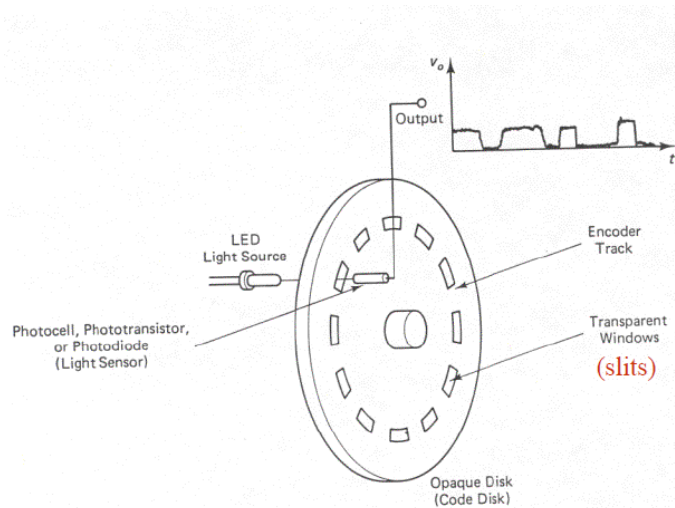
Part 3: Reading Encoders

Objective:

For Part 3, an encoder will be used to determine the position of the output motor shaft. The brightness of the LED will be controlled based on the position.

Background Information:

An encoder is a device that can be used to determine position. Most encoders use optical sensors to output a pulse train which can be decoded to determine position and direction. In this exercise the NXT's internal encoder will be used to track angular position.

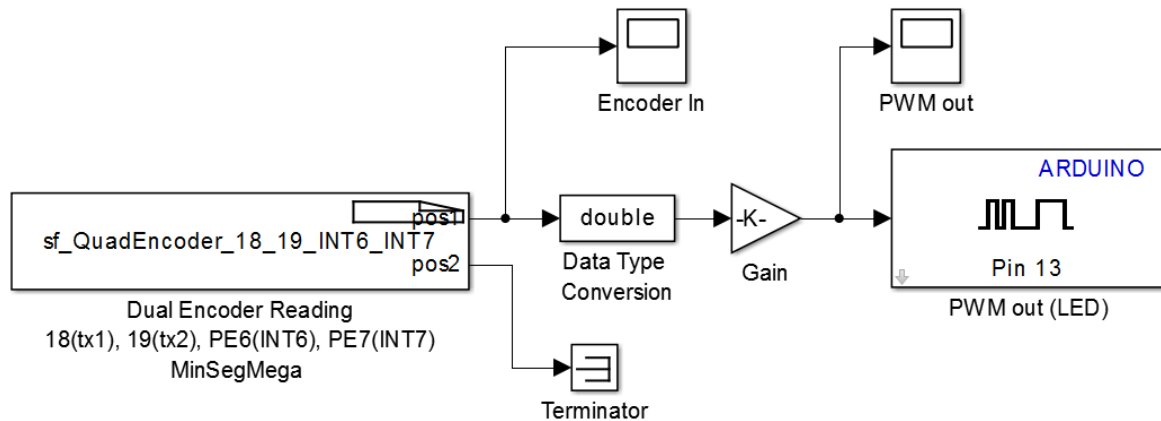


- The NXT encoder has a raw resolution of 12 counts. The figure on the left below show the actual encoder wheel in the NXT motor. It has 12 holes. The optical sensor will output a high or low (1 or 0) depending on the position of this encoder. Each revolution will produce 12 pulses.
- If the output motor shaft moves 1 revolution the encoder wheel turns 15 revolutions – this means for one revolution of the motor output shaft there will be $15 \times 12 = 180$ pulses from the encoder.
- The encoder device provides two channels A & B. These channels are the encoder pulses with one channel shifted by 90 degrees. By monitoring these two channels both the position and direction of the motor shaft can be determined. If quadrature decoding is used the resolution can be increased by 4 times.
- The Encoder block included with this lab performs quadrature decoding so the total resolution is:

$$\text{Angular Resolution} = 15 * 12 * 4 = 720 \text{ pulses per revolution of output shaft}$$

Simulink Model

Right click on the Rensselaer Arduino Support Package to open the library, open MinSegMega. Now you can drag the encoder block to your Simulink Model. Build the following Simulink Model.



For Data Type Conversion: Go to Simulink-> Signal Attributes. Double click this block and set output data type to double. This is done so that floating point math is performed to avoid truncation and division errors associated with fixed point numbers.

- **Configuration Parameters:** The “Fixed-step size” should be 0.03 (the fastest recommended size for external mode).
- **PWM:** Set the Pin to 13 (the onboard LED).

Note: *Pulse width modulation (PWM)* is used to create an *average* voltage by varying the duty cycle, or the amount of time the signal is on/off. A 100% duty cycle corresponds to fully powering the LED with 5v and a 0% duty cycle means the LED will receive 0v. A 50% duty cycle means *on average* the voltage seen by the LED is 2.5v. The value assigned to the PWM block can be from 0 to 255: 100% duty cycle corresponding to 255 and 0% duty cycle corresponding to 0.

Question: What value does the gain need to be so that 1 revolution of the encoder will obtain the maximum value of the PWM? What happens if the motor is turned more than 720? What happens when the encoder position is negative?

Checkpoint 2:

Run the simulation and observe the scope and the LED when you rotate the encoder’s wheel clockwise and counter-clockwise.

Verify that one revolution of the output motor shaft results in a position of 720. You can also add a ‘Sinks -> Display’ block to see exact values.