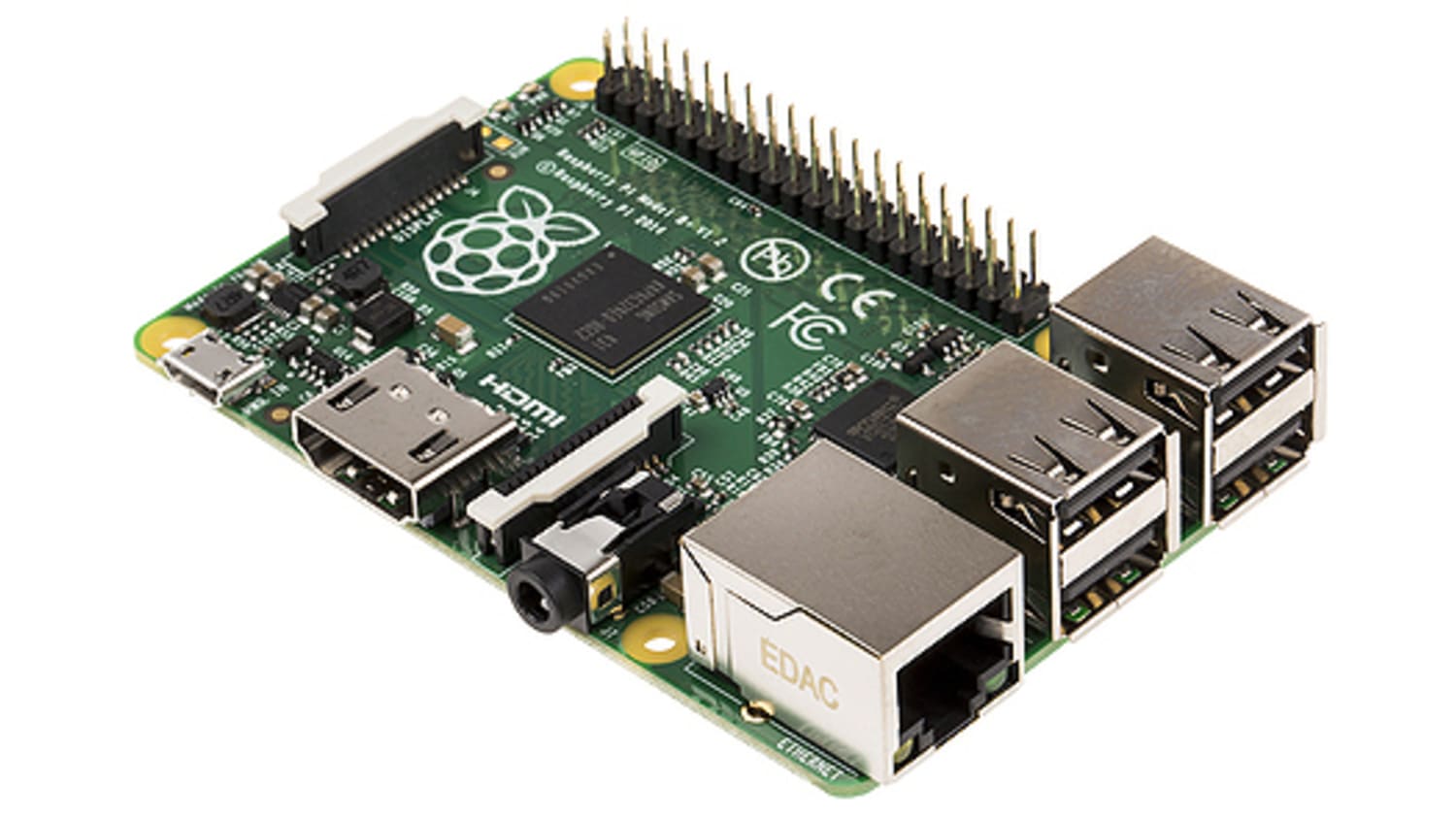
Motor control with Simulink on Raspberry Pi





Author: Kristian Snel

Tutor: Jeedella, Jeedella

Date: 02-11-2020

**Goal**

The goal of this project is to setup a motor controller with a Maxon brushless DC motor using a Raspberry Pi and Simulink. This system will then be used by students to design an elevator control system.

Requirements:

* Motor control in both directions.
* Reading out the ENX encoder to determine motor position.
* Simulink model to control the motor on Raspberry Pi hardware.

Optional:

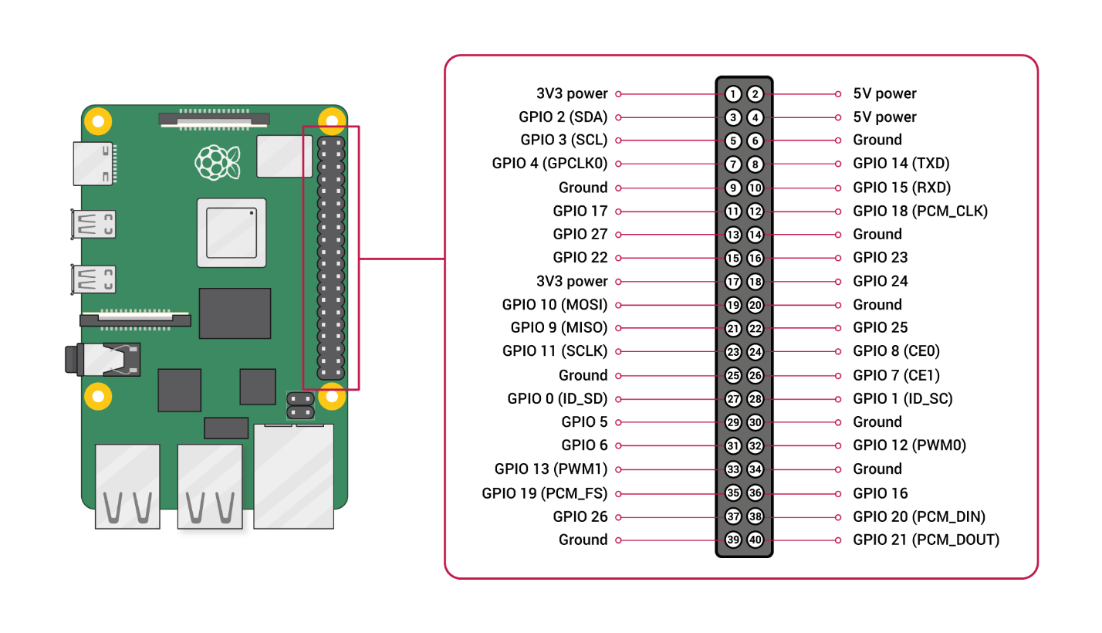
* Serial communication with the motor control board instead of analog input.
* Raspberry Pi shield with connectors for the motor controller and motor encoder.

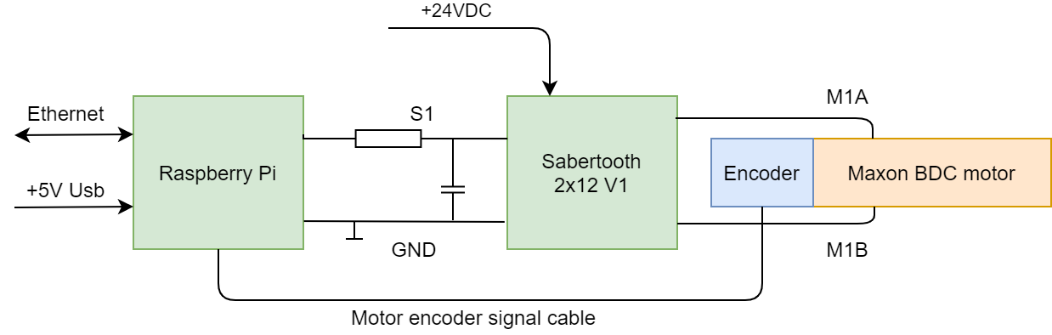
Equipment used:

* Laptop with Matlab
* USB type A to micro usb cable
* Raspberry Pi 3B+
* 24VDC power supply
* Multimeter
* Breadboard and components for signal filtering
* Ethernet cable
* Sabertooth2x12 V 1.00 motor control board

**Suggested improvements**

* To change from analog control to serial control a level converter is required to turn the 3.3V raspberry to 5V logic for the sabertooth (<https://www.tinytronics.nl/shop/en/dc-dc-converters/level-converters/i2c-uart-bi-directional-logic-level-converter-5v-3.3v-2-channel-with-supply>)

**Wiring diagram**



Pins:

* GPIO 13 (PWM output for analog motor control
* GPIO 17 (Encoder in 1)
* GPIO 18 (Encoder in 2) only one that is currently working

Components:

* Voltage dividers (5V logic from encoder to 3.2V with 4k7 and 2k7 in serial)

**Mode 1: Analog Input**

Analog input mode takes one or two analog inputs and uses those to set the speed and direction

of the motor. The valid input range is 0v to 5v. This makes the Sabertooth easy control using a

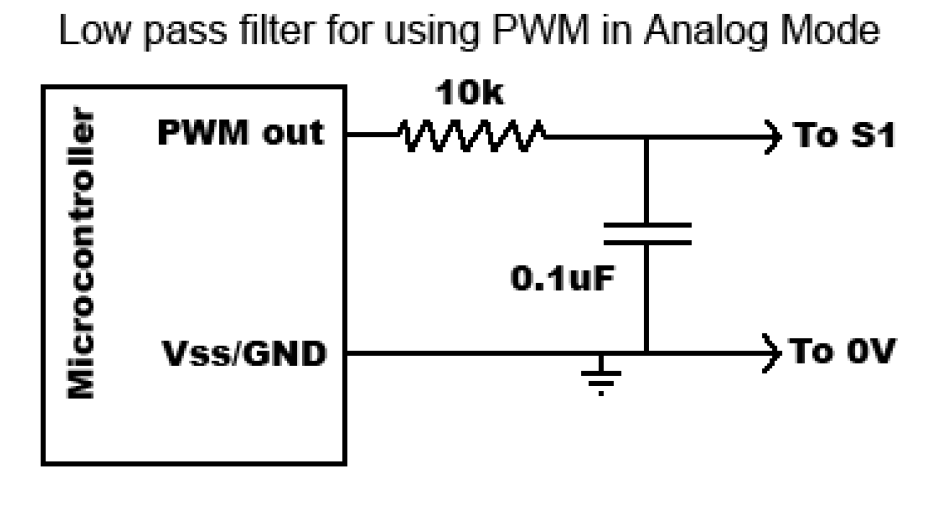
potentiometer, the PWM output of a microcontroller (with an RC filter) or an analog circuit.

Major uses include joystick or foot-pedal controlled vehicles, speed and direction control for

pumps and machines, and analog feedback loops.

Note on using filtered PWM in Analog

Mode



If you are using a filtered PWM signal from a

microcontroller to generate the analog voltage, an R/C filter

with component values 10k ohms and at least .1uf is

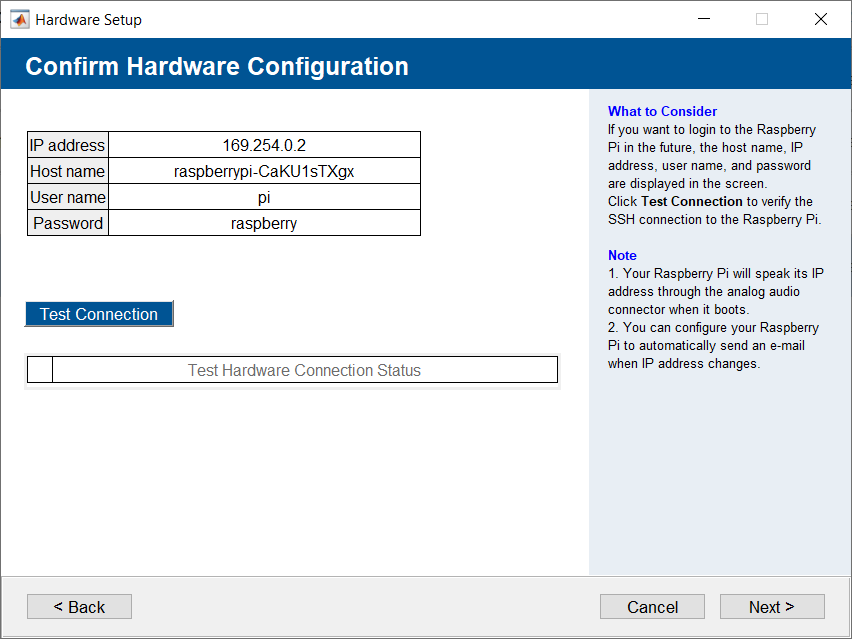
recommended as shown in **Figure 4.1.** Using a larger value

filter capacitor such as 1uf or 10uf will result in smoother

motor operation, at a cost of slower transient response. A PWM frequency higher than 1000Hz is

recommended.

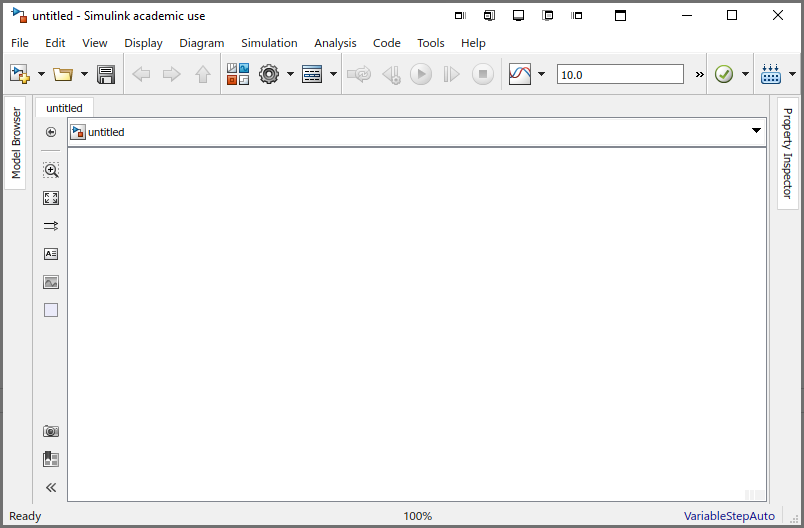
Hardware setup in Simulink:



Debugging the raspberry pi or clearing Matlab files:

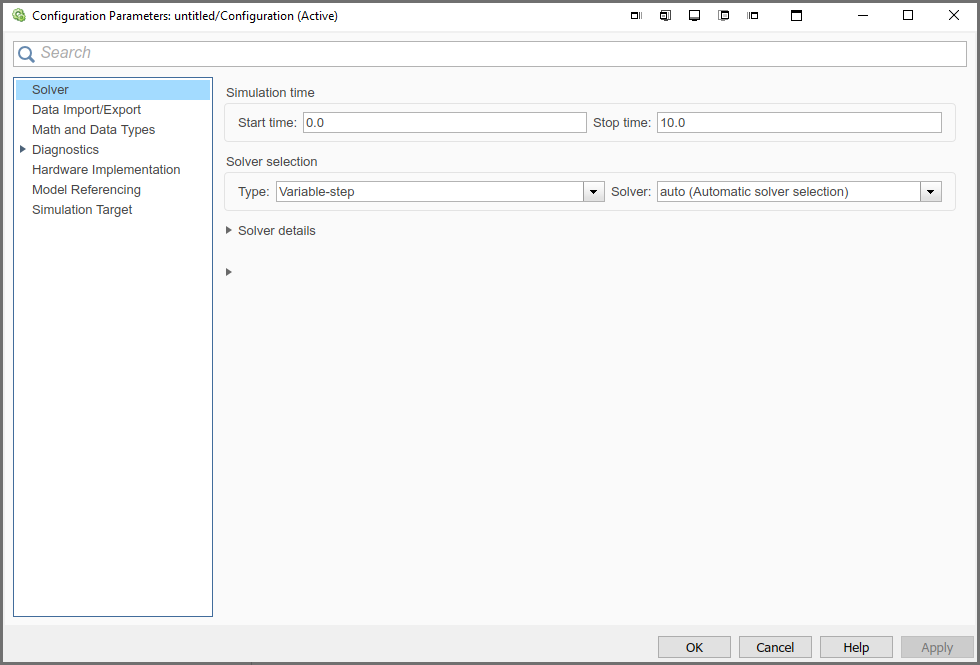
* Launch Putty and connect with ssh to the raspberry entering the Raspberry Pi address

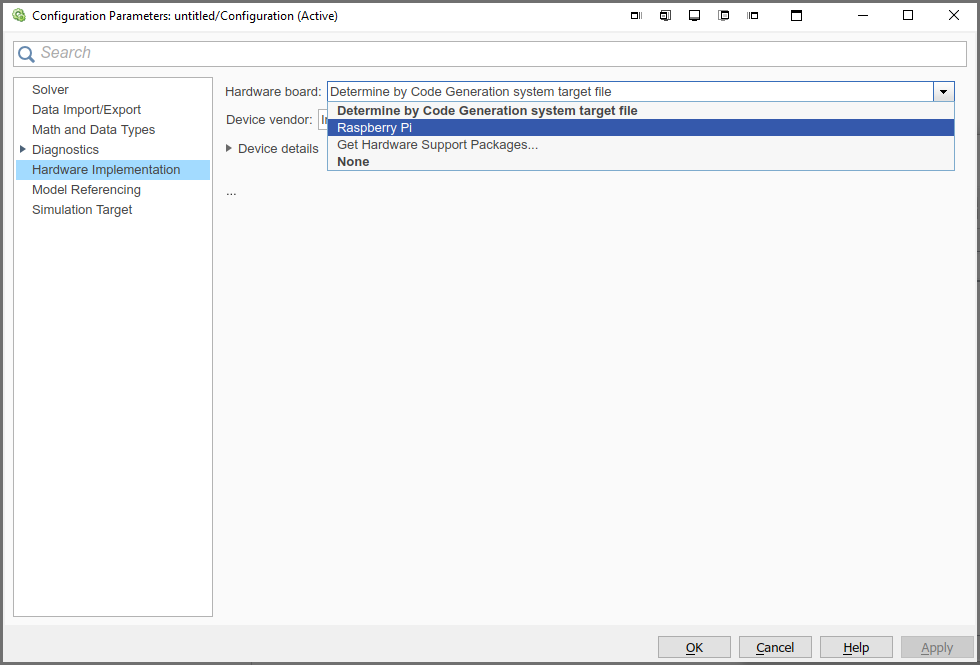
How to use Simulink Support Package for Raspberry Pi Hardware



Go to Hardware Implementation

Open the configuration parameters





Select Raspberry Pi

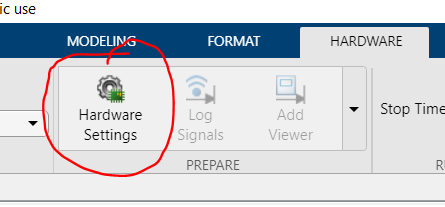
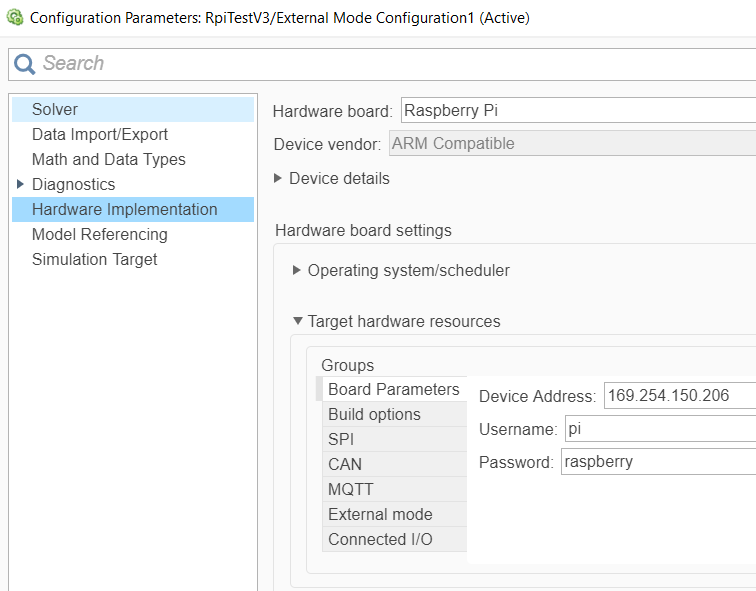
F.A.Q.

**Q: Your current working folder is: C:\Program Files\MATLAB\R2018b\bin Simulink does not permit you to modify the MATLAB installation area. Please change to a working folder that is not in the MATLAB installation area and re-try the previous command.**

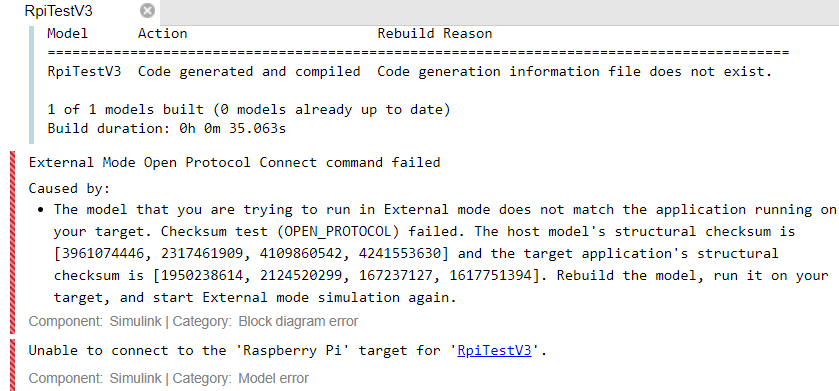
S: <https://nl.mathworks.com/matlabcentral/answers/443075-your-current-working-folder-is-c-program-files-matlab-r2018b-bin-simulink-does-not-permit-you-to>

**Q: Where can I find the Raspberry Pi hardware configuration settings?**

S: after you have set up the hardware as described in this document you may click the Hardware settings button under the hardware tab:

And then go to Hardware implementation 🡪 Target hardware resources.

Q: Error 

S: Reinstall Image on Raspberry Pi. (Seek better solution to solve this regularly occurring error.)

Sources:

* <https://www.instructables.com/DIY-Standalone-Arduino-Uno/>