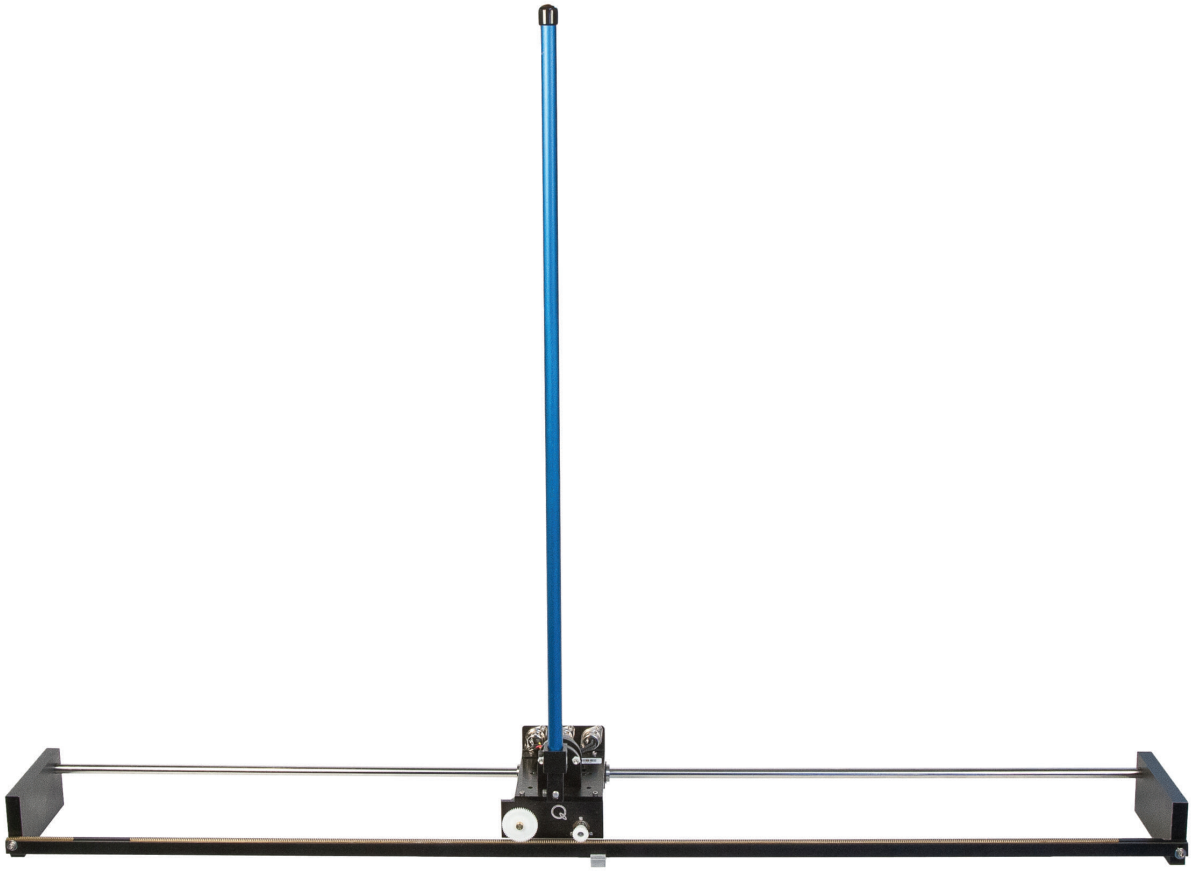
UCSB ECE Controls Lab TA Manual

By Max Crisafulli

September 2024



This document lives at
https://github.com/MaxCrisafulli/UCSB-ControlsLab-Manual-2024

TA Manual for the UCSB ECE Controls Lab

This document is to act as a guide for TAs in setting up the Quanser inverted pendulum cart hardware in the UCSB ECE Controls Lab (Harold Frank Hall 3120A). This guide is an update of the existing manual, located at:

`https://github.com/justinpearson/UCSB-Quanser-Inverted-Pendulum-Lab-Manual`

This updated guide will live (and hopefully be maintained) at

`https://github.com/MaxCrisafulli/UCSB-ControlsLab-Manual-2024`

Hardware In The Lab

The ECE147ABC labs make use of the Quanser 'Linear Servo Base Unit with Inverted Pendulum'. This manual will detail the setup of just the cart system, with motor voltage as input and linear position as output. There are 3 main components of the hardware setup, with each of which being explained below.

Q4 DAQ Board

The SIMULINK model that interfaces with the hardware interacts with the world via the Q4 HIL control card and DAQ (Data Acquisition Board). The control card is inside the lab PCs connected via PCIe and interfaces with the DAQ board via the two gray ribbon cables. Ensure that these are connected properly and that the red Status LED is lit, indicating that the board is powered.

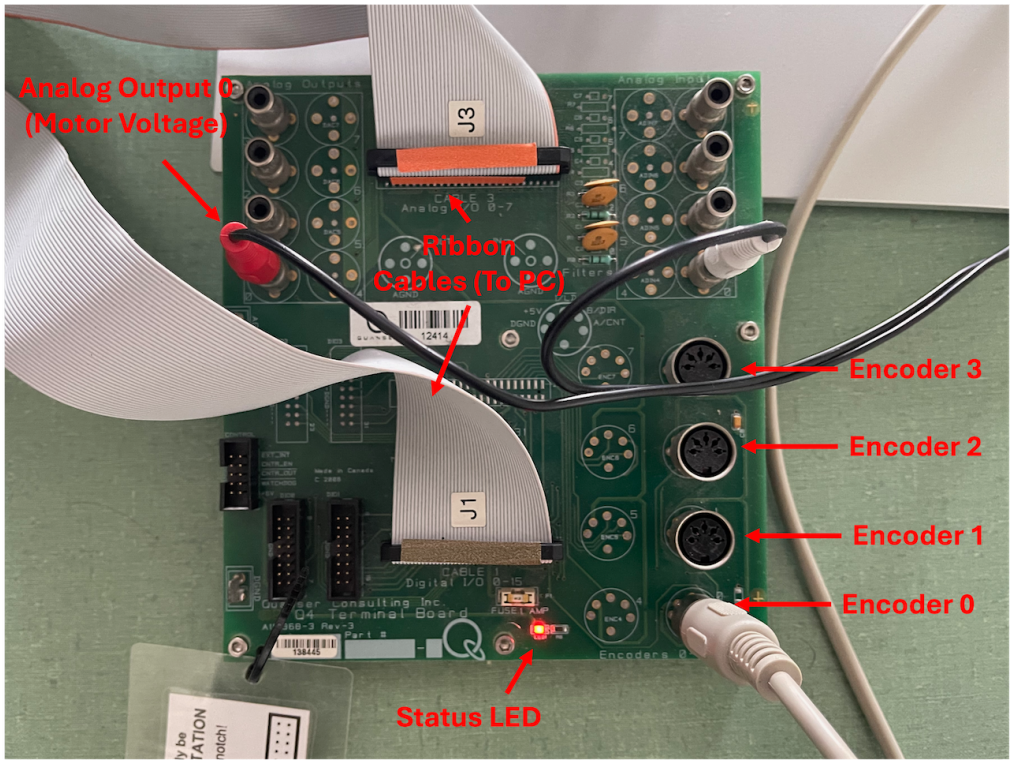


Figure 1: Quanser Q4 Terminal DAQ Board

The DAQ board uses the four encoder ports to read signals from the hardware. Generally, Encoder 0 is used for the linear position encoder. In the top left corner of the board are the Analog Outputs, Analog Output 0 is used to drive the amplifier box which ultimately drives the cart motor.

VoltPAQ-X1 Amplifier

The Quanser VoltPAQ-X1 Amplifier is powered from a standard wall socket, and turned on via a switch at the back of the box. Make sure that the green status LED in the center of the amplifier diagram is lit, indicating that the amplifier is powered on. Also check that the Amplifier Gain switch is set to 1x.

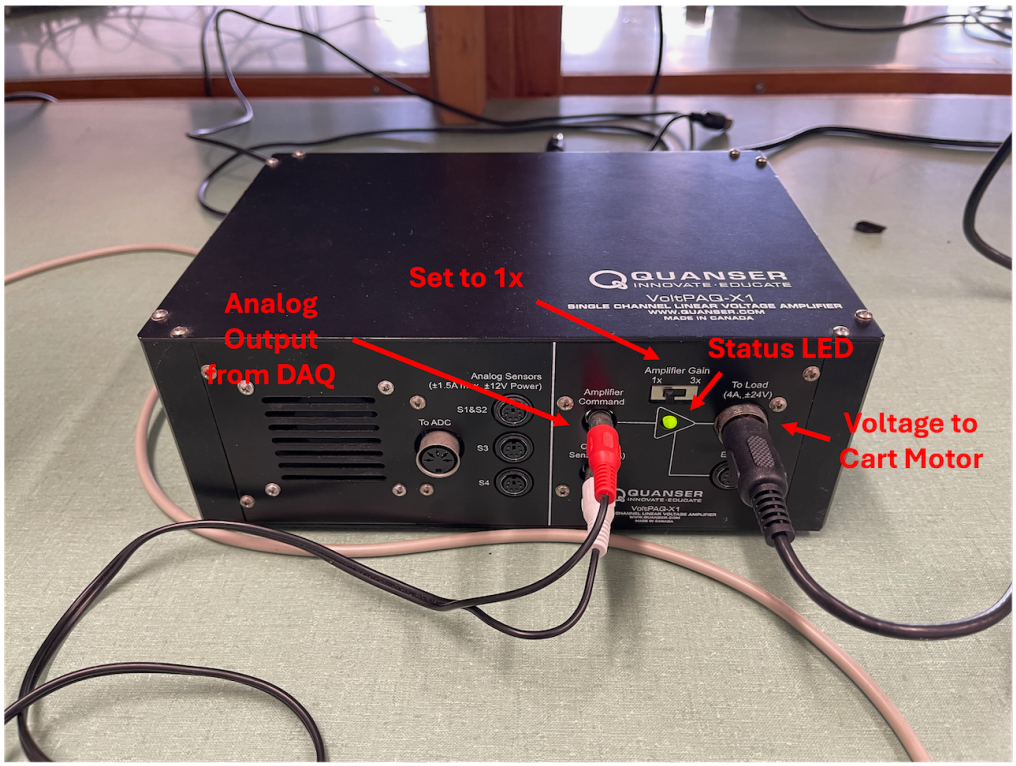


Figure 2: Quanser VoltPAQ-X1 Linear Voltage Amplifier

The red AV cable coming from Analog Output 0 on the DAQ board should be connected to the Amplifier Command (input) port. The white AV cable should be connected to Current Sense on the amplifier and Analog Input 0 on the DAQ board, although this is not necessary.

The output of the amplifier (To Load) is what drives the motor onboard the cart. Check that all the cable connections are correct and that the cable going to the motor has enough slack to allow the cart to move along the track.

Inverted Pendulum Cart System

The cart system has 2 encoder outputs and 1 analog voltage input. The positions of these ports may vary between the carts but in general they are as depicted in Figure 3. You should check what each of the ports connect to before plugging anything in.

The cart controls its position along the track with the smaller motor drive wheel, which is driven by the motor voltage coming from the amplifier. The larger position encoder wheel measures the relative position along the track from the time that the program was initialized in 'ticks'. The conversion ratio from ticks to inches is approximately 5000 ticks per 6 inches or 35000 ticks per 1m.

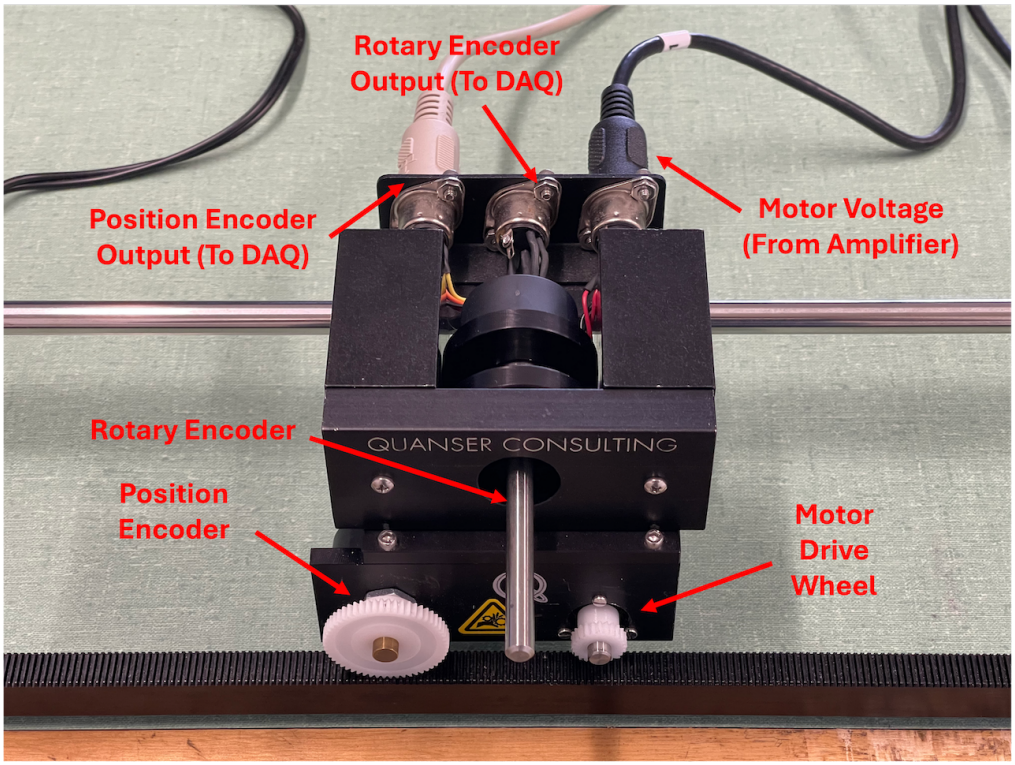


Figure 3: Quanser Linear Servo Base Unit (The 'Cart')

General Info

- If the DAQ board light is off, check that the ribbon cables going to/from the PC are connected properly. If there is still an issue then the (minimum 1 Amp) fuse above the LED may need replacing. Make everything is completely unplugged from the DAQ board before doing this. Spare fuses can be found in a toolbox in the gray cabinet at the back right of the lab.
- If the DAQ board is not connecting to the PC and the light is on try restarting the computer.
- Contact the IT people for computer problems, MATLAB issues, and getting TA/student login access:

  `support@ece.ucsb.edu`

- Contact the ECE shop people for issues with the lab hardware:

  `shop@ece.ucsb.edu`

Setup Instructions (ECE147A Lab 2)

Hardware

The *MultiQ-4 Terminal Board* (Figure 4) is a general purpose data acquisition and control board which has eight single ended analog inputs, eight analog outputs, sixteen bits of digital input, sixteen bits of digital output, three programmable timers, and eight encoder inputs decoded in quadrature. For the purposes of this lab, we will be primarily concerned with the analog outputs, the encoder inputs, and, to a lesser extent, the analog inputs.

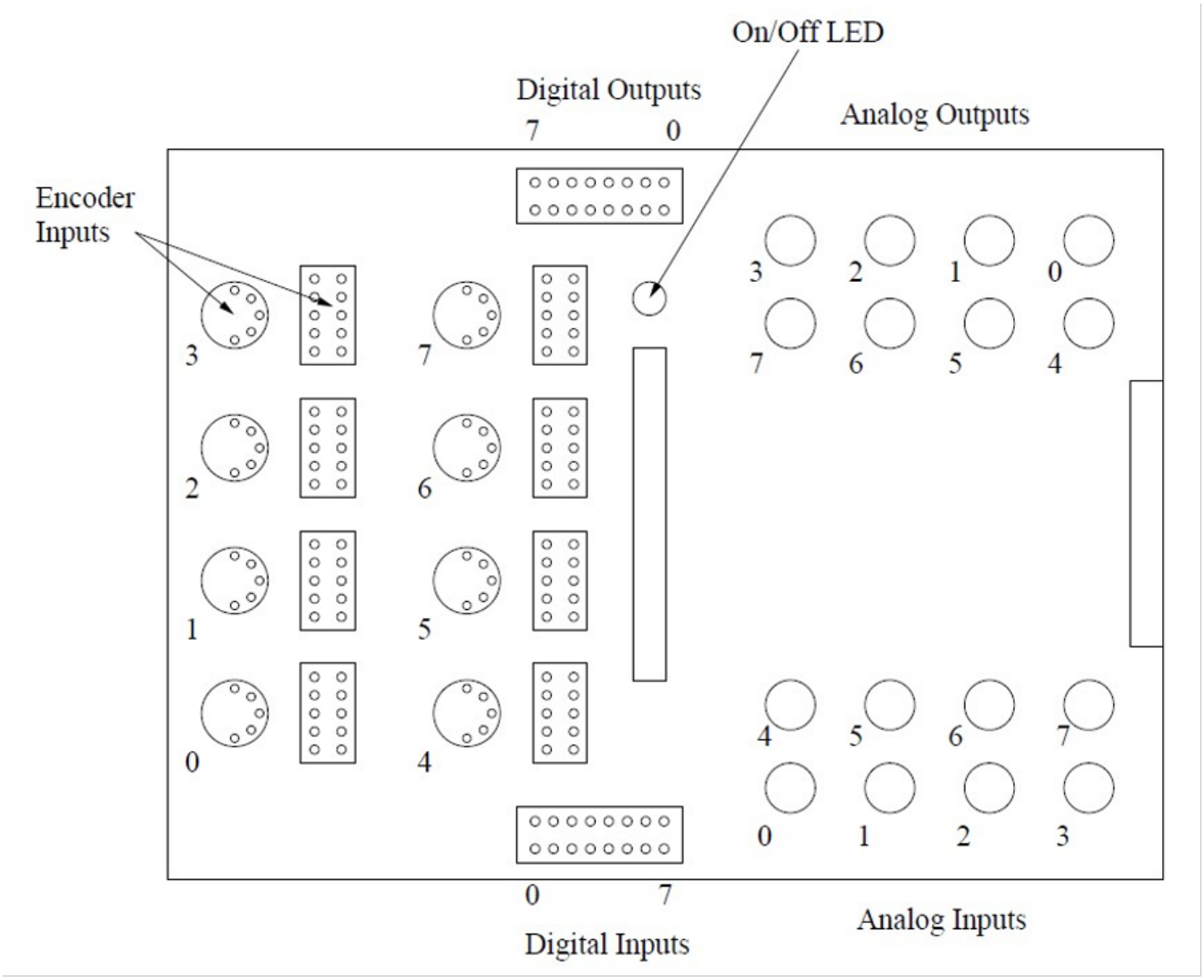


Figure 4: MultiQ-4 Terminal Board Schematic

Both the analog inputs and outputs are accessed via clearly labeled RCA jacks on the MultiQ-4 board and have a range of ±5 volts. The encoder inputs are accessed via a 5 pin DIN socket and connect directly to the experimental equipment.

Hardware Setup

As a basic example, we will set up the system to read in the encoder from the motor cart and output a constant voltage to the motor.

1. You must use **MATLAB R2023a** *only*. The other versions will not work. Start SIMULINK and create a new model.
2. Choose **Modeling → Model Settings**
3. Set the following parameters
   (a) Select **Solver** from the tree at the left
   (b) Stop time: **inf** - or whatever T_f is desired (in seconds)
   (c) Solver options, solver type: **Fixed-Step**
   (d) Solver options, solver: **discrete** if there are no continuous states. Select a solver like **ode4** if you are using continuous states (like a continuous transfer function)
   (e) Solver options, Fixed-step size (under Solver details): **0.001**
   (f) The settings are illustrated:

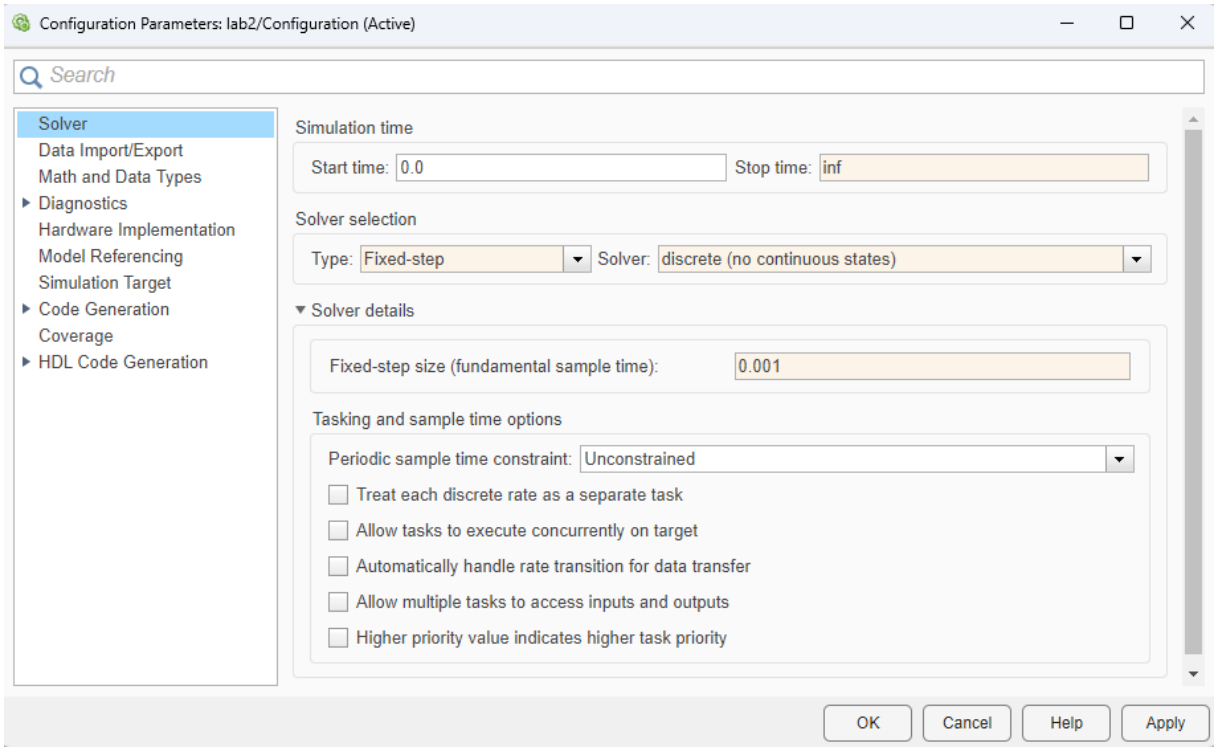


Figure 5: Configuration Parameters

   (g) Choose **Code Generation** in the tree on the left.
   (h) Ensure the System target file is **quarc_win64.tlc**. If not, select the **Browse** button and select it. The settings should look as follows.

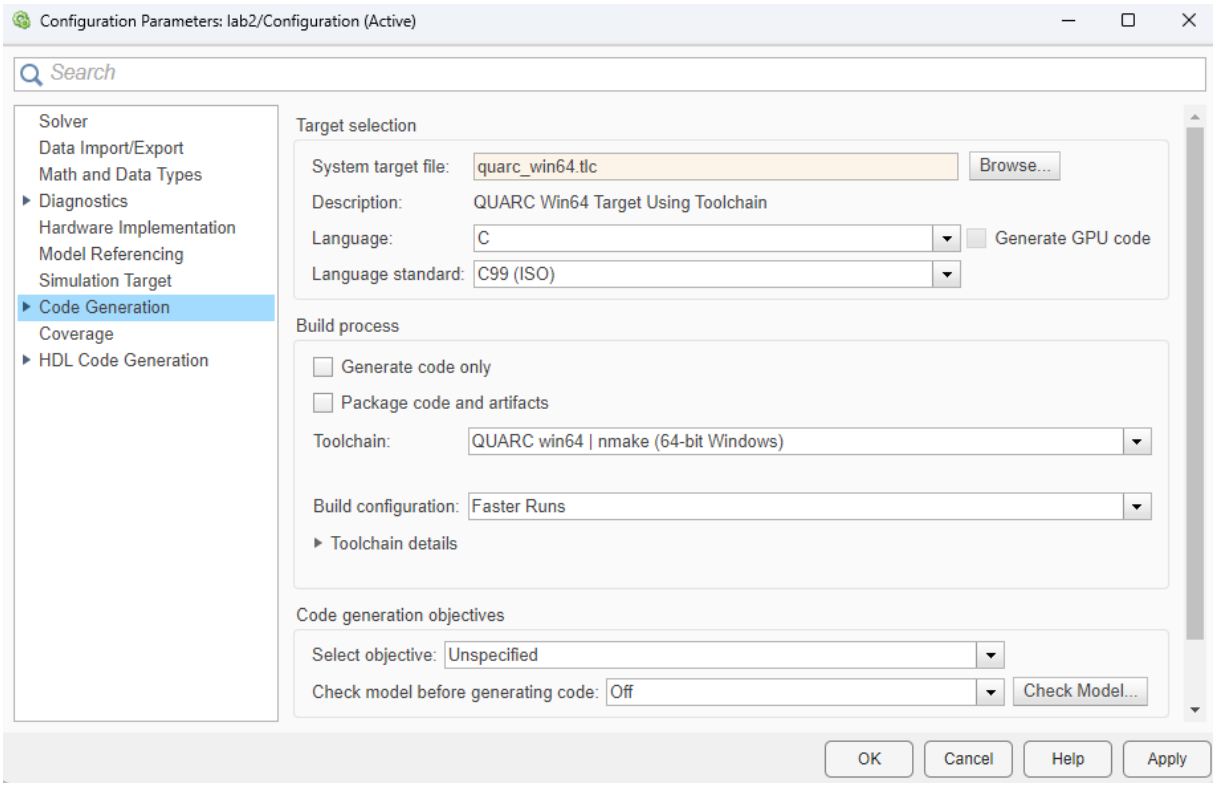


Figure 6: RTW Parameters

4. Select OK and return to the SIMULINK model.

5. In the SIMULINK Library Browser, find the **HIL Initialize** block. It is located in **QuaRC Targets → Data Acquisition → Generic → Configuration**. You can also search by double clicking the model background. Add the block into your model.

   (a) *Aside*: HIL stands for *Hardware In the Loop*. It is a design technique in which the controller hardware is tested with a simulated plant. A flight simulator is an example; the pilot (controller) is connected to simulated inputs and outputs. The technique we are using in the lab is often called *Rapid Control Prototyping*, where a PC controller is connected to actual plant hardware. If you are simulating everything on a computer, like in the prelab, you are doing *offline simulation*.

6. Double click on the **HIL Initialize** block in your model. Change the Board type to **q4**. Make sure the Board Identifier is **0**. Click Apply and Ok out of the block.

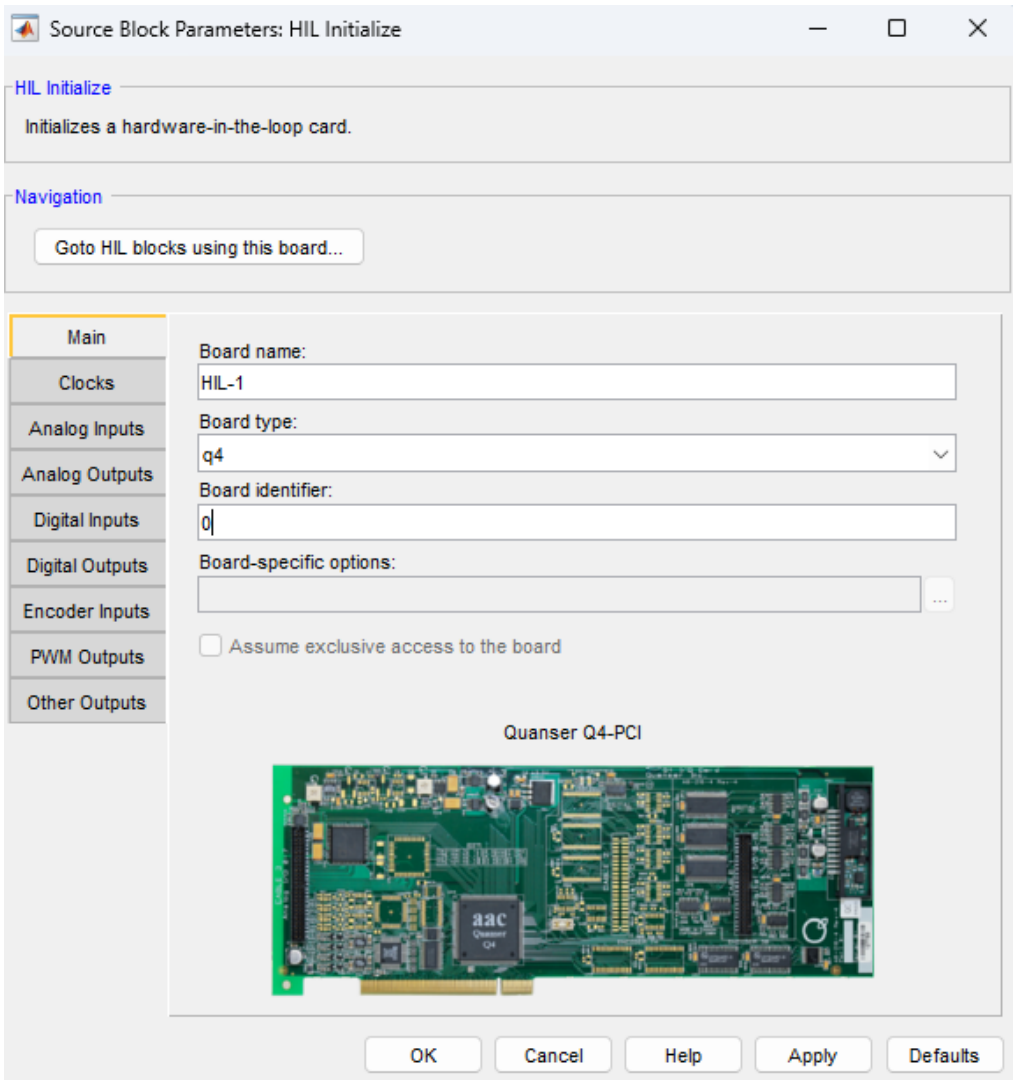


Figure 7: HIL Initialize Parameters

7. Add a **HIL Write Analog** block and a **HIL Read Encoder** block to your model. They are in **QuaRC → Data Acquisition → Generic → Immediate I/O**.

8. Connect these blocks as depicted in Figure 8. In this example, we place a simple constant output and a display for the encoder. *For the purposes of this example, do not output more than 1 volt to the cart to prevent uncontrolled motion.*

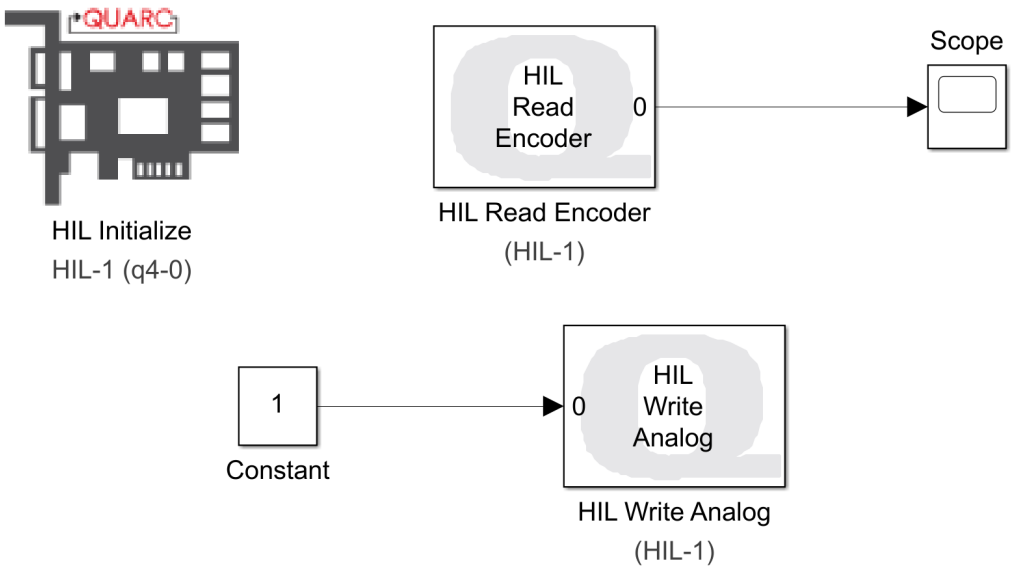


Figure 8: Simple Model

9. When your model is ready, save it. From the **QuaRC** menu, select **Build**. MATLAB will compile your SIMULINK model into native machine code. You can check the status of the build by clicking **View diagnostics** at the bottom of the SIMULINK window. *Every time you structurally change your model (add new blocks, rewire existing blocks) you will need to rebuild it or an error will occur.*

   (a) *Tip*: You do not need to rebuild the model if you are changing constants, or blocks that take a constant, like the gain of the gain block. You can even change these constants while the model is running and the change will take effect immediately.

   (b) *Tip*: You can change the constants in a transfer function while stopped or running such that they do not affect the dimensions of the state. That is, the number of poles and zeros must remain the same. Any other change requires a rebuild.

   (c) *Use care when changing a running model! Be careful not to leave the model running by accident.*

10. Ask your TA to set up the cart for motion. It should be placed in the center of the track and the Quanser Amplifier switched off.

11. When the build is completed, go back to the model and choose **QuaRC → Monitor & Tune**. This downloads the compiled model to the QuaRC execution program and then model will run. The **Stop** button in the Quarc menu will stop the model as normal.

   (a) *Troubleshooting Tip*: If the SIMULINK time is not running (at the bottom of the model window), the QuaRC execution environment may have crashed. Try logging out and logging back in or rebooting the computer.

   (b) *Tip*: Suppose your model is running and MATLAB crashes. The model will continue to run. First, ensure the hardware is not damaged by lifting it off the track and then switching off the Amplifier.

12. As the model runs, observe that moving the cart will change the encoder output, as shown on the display. Lift the cart up from the track and turn on the Quanser Amplifier. Observe the motor motion.

13. Save your basic model to your network drive to use as a starting point for future designs.

Data Logging

In the Real Time Windows Target, data is logged by scope blocks. Like an oscilloscope, to start the data logging procedure, a trigger is needed to initiate the process. A simple approach is to define a dedicated scope block, with one trigger signal entering it. Put a step function with a step time of 0.001 seconds, which corresponds to one sample time. This set-up will save all but the very first data point. To record data, follow the procedure below.

1. Place a STEP function block from the SOURCES library and a SCOPE block from the SINKS library in your model.

2. Name the SCOPE block TRIGGER SCOPE so that it is easy to identify in a list.

3. Set the step time of the STEP block to the time you wish to begin saving data. Set the Initial Value and Final Value to 0 and 1 respectively.

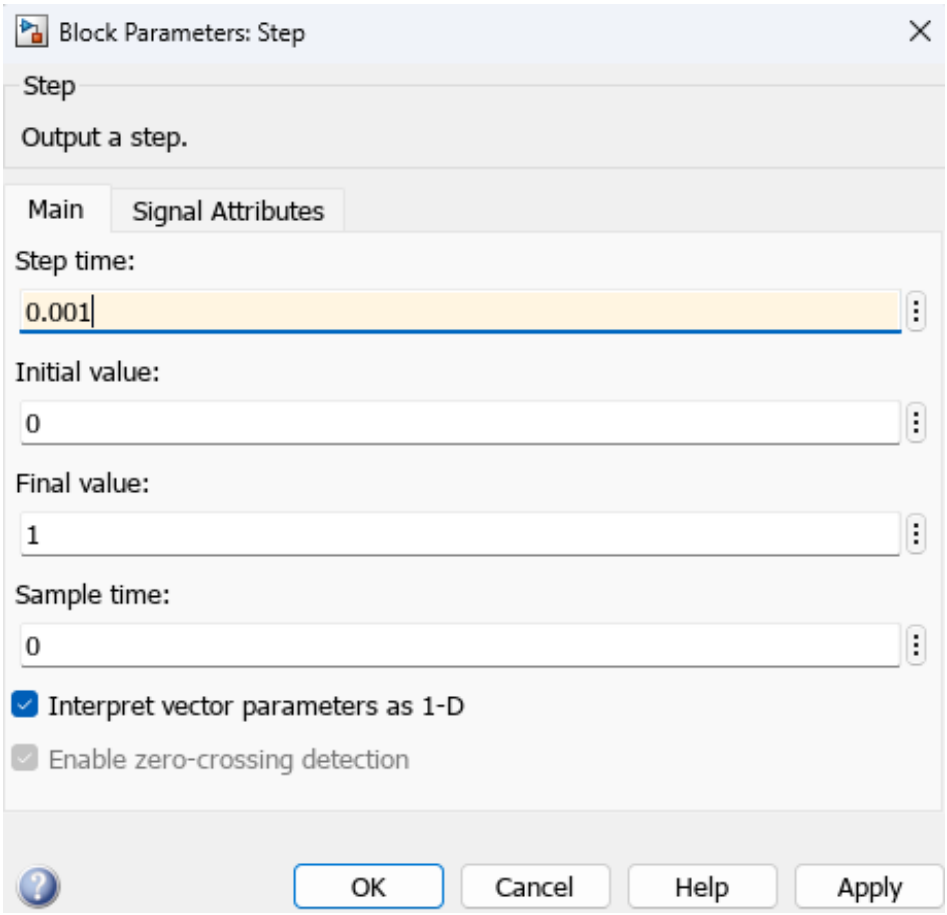


Figure 9: Step Function Block as a Trigger Signal

4. Go to the HARDWARE menu and select CONTROL PANEL.

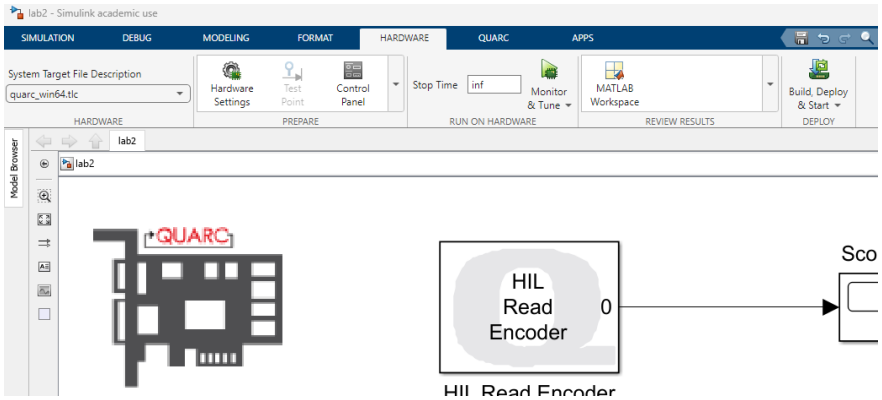


Figure 10: Hardware Control Panel

5. Click on the SIGNALS AND TRIGGERING button.

6. Select the TRIGGER SCOPE block from the list and click on the TRIGGER SIGNAL button. This selects the input to the scope block as the trigger signal.

7. In the TRIGGER area make sure that SOURCE = SIGNAL, MODE = NORMAL, DELAY = 0, and set DURATION = T_f/T_s. The duration is the number of samples allotted in memory for each signal being logged. If we have a simulation time of $T_f = 10$ seconds and our sampling time $T_s = 0.001$, then $10/0.001 = 10{,}000$ samples. If you actually run the model longer than T_f, then the data in memory will be overwritten when Tf/Ts samples have passed. Check Figure 11.

8. In the TRIGGER SIGNAL area make sure that DIRECTION = EITHER or RISING, and LEVEL = 0.5. Our STEP signal will transition from 0 to 1 at 0.001 seconds, and is a rising edge. When it crosses the value 0.5 this will initiate the signal logging.

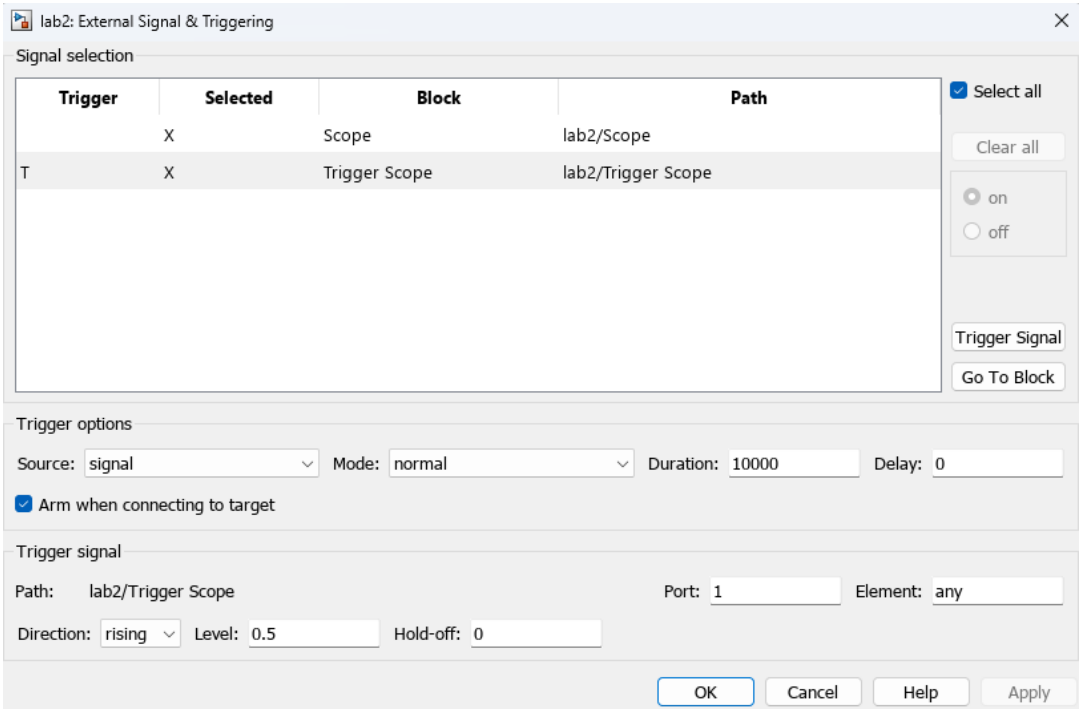


Figure 11: Configuring the Trigger Signal

9. Open all SCOPE blocks and configure them in the following way.

10. Click the SCOPE PROPERTIES Icon (gear icon), and select the LOGGING tab.

11. Make sure Limit data points to last is UNCHECKED.

12. If you wish to save the data gathered by the scope, CHECK the Log data to workspace box, and give the Variable name a unique and meaningful name for easy reference later. In the SAVE FORMAT menu choose the ARRAY option. Click Apply and Ok out of the menu.

13. *Hint*: Once you have set these parameters as you like, you can RIGHT CLICK on a scope block, drag and drop, and a copy of that block will be placed where you let go of the mouse. The new block will have all of the parameters of the previous, except the Variable name, that will be different and should be changed to something meaningful.