University of California, Santa Barbara
Department of Electrical & Computer Engineering

ECE 147b: Digital Control

Lab 0: Introduction to Sampling

This lab is an introduction to the experimental equipment and a review of some of the principles of sampling. No writeup is due.

1 Hardware

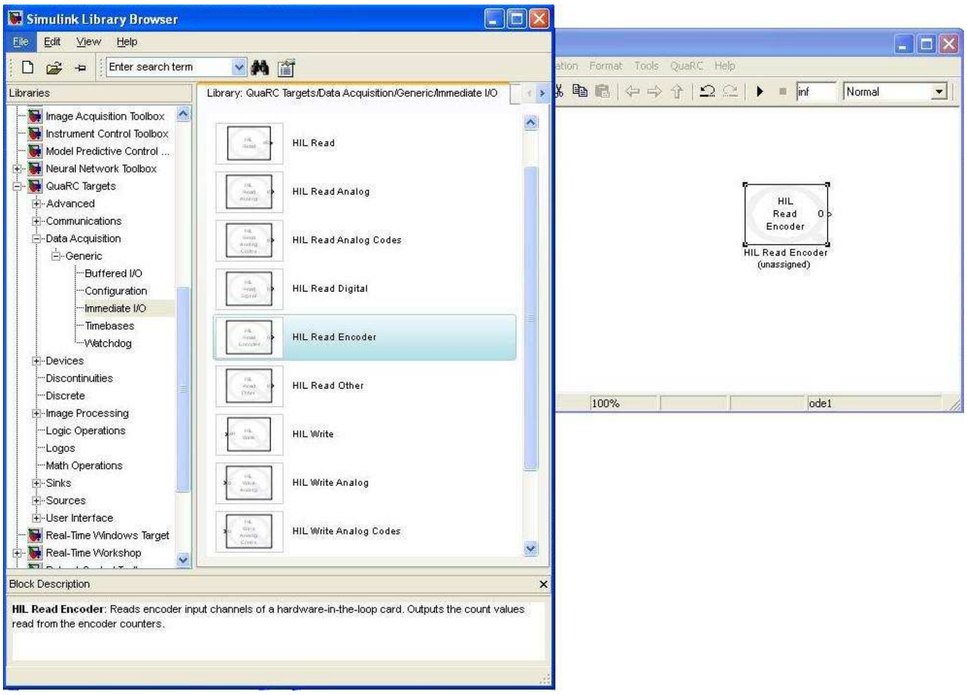
The control board used in this lab has 3 encoder inputs on the top row, and three PWM outputs on the bottom row. The back has digital inputs and outputs.

2 Software

The functionality of the control board can be completely controlled via SIMULINK and Real Time Workshop (RTW). RTW is a real-time Windows application that runs SIMULINK generated code on a PC.

To use the control board, go to the Simulink Library Browser and

1. Select the Quarc/Generic/Data Acquisition library.
2. Select HIL Write PWM, HIL Read Encoder, HIL Initialize blocks from the selections on the right.
3. Drag the selected block onto your Simulink Model.



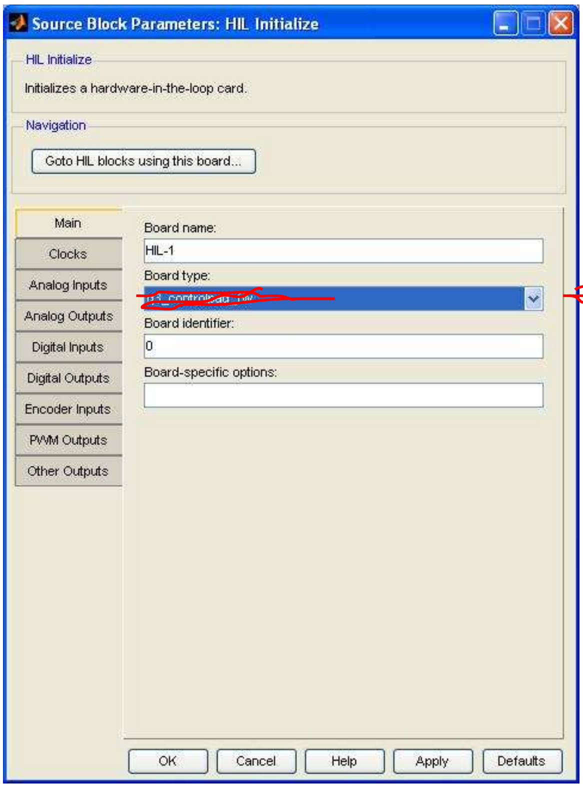
HIL Read Encoder: Reads encoder input channels of a hardware-in-the-loop card. Outputs the count values read from the encoder counters.

4. Double click the HIL Initialize block, set board type to q3-controlpaq-fw, set board number to HIL-1 and save settings.

5. Double click the HIL Encoder Input and HIL PWM Output blocks.

6. Set board number to HIL-1, set sample time, (ie. 0.001), set channel numbers (note: matlab channel numbers run from 1-3, board numbers run from 0-2, so channel 1 in simulink corresponds to channel 0 on the board).

...or, can optionally use
HIL Write Analog
(as in 147A).



Source Block Parameters: HIL Initialize

HIL Initialize

Initializes a hardware-in-the-loop card.

Navigation

Board name:

HIL-1

Board type:

~~q3_controlpad_fw~~

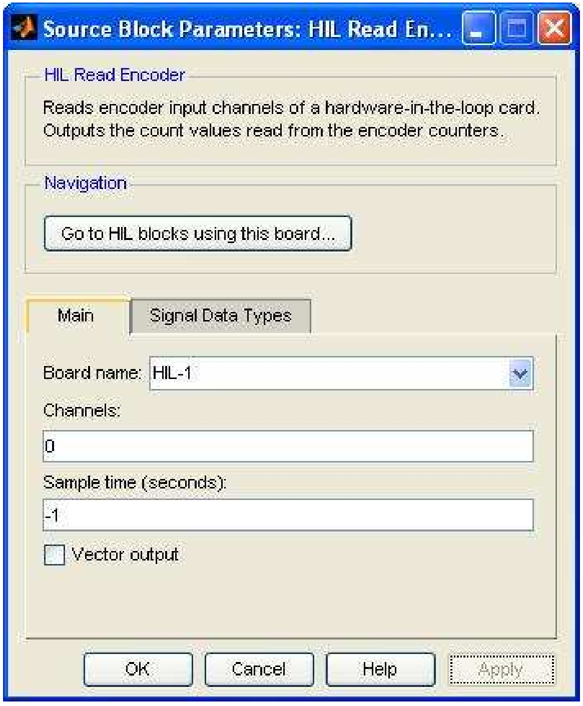
Board identifier:

0

Board-specific options:

board type:

q4



Source Block Parameters: HIL Read En...

HIL Read Encoder

Reads encoder input channels of a hardware-in-the-loop card. Outputs the count values read from the encoder counters.

Navigation

Board name: HIL-1

Channels:

0

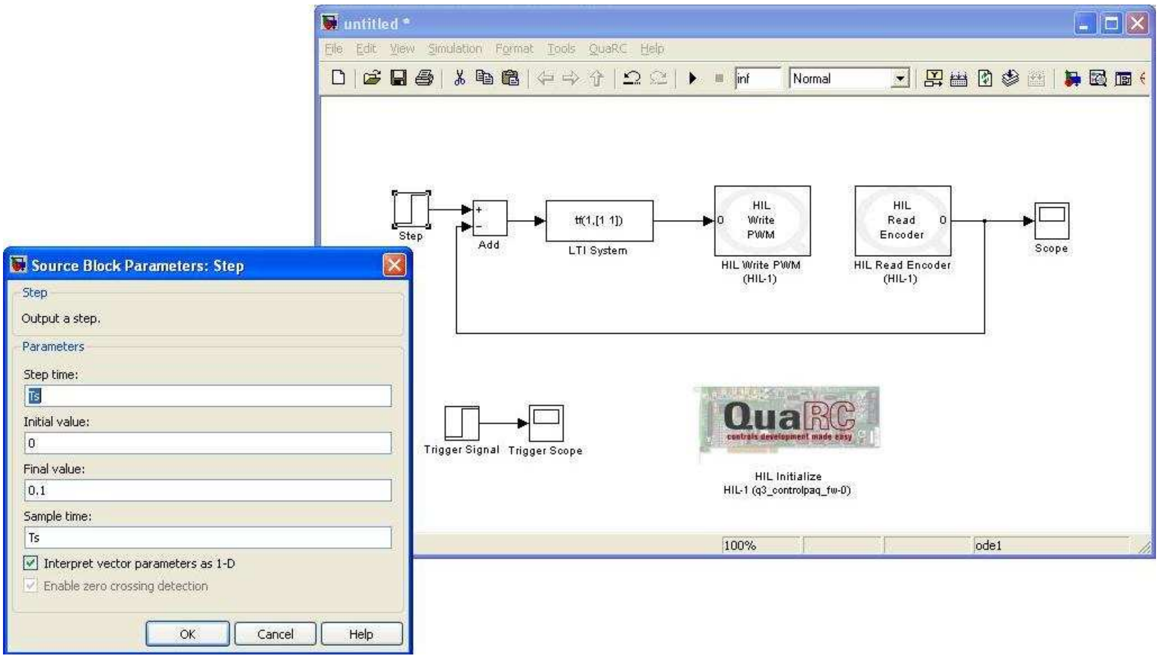
Sample time (seconds):

-1

□ Vector output

2.1 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 as illustrated below.



Simulation stop time: inf
Simulation mode: Normal

Source Block Parameters: Step

Step

Output a step.

Parameters

Step time: Ts

Initial value: 0

Final value: 0.1

Sample time: Ts

☑ Interpret vector parameters as 1-D

☑ Enable zero crossing detection

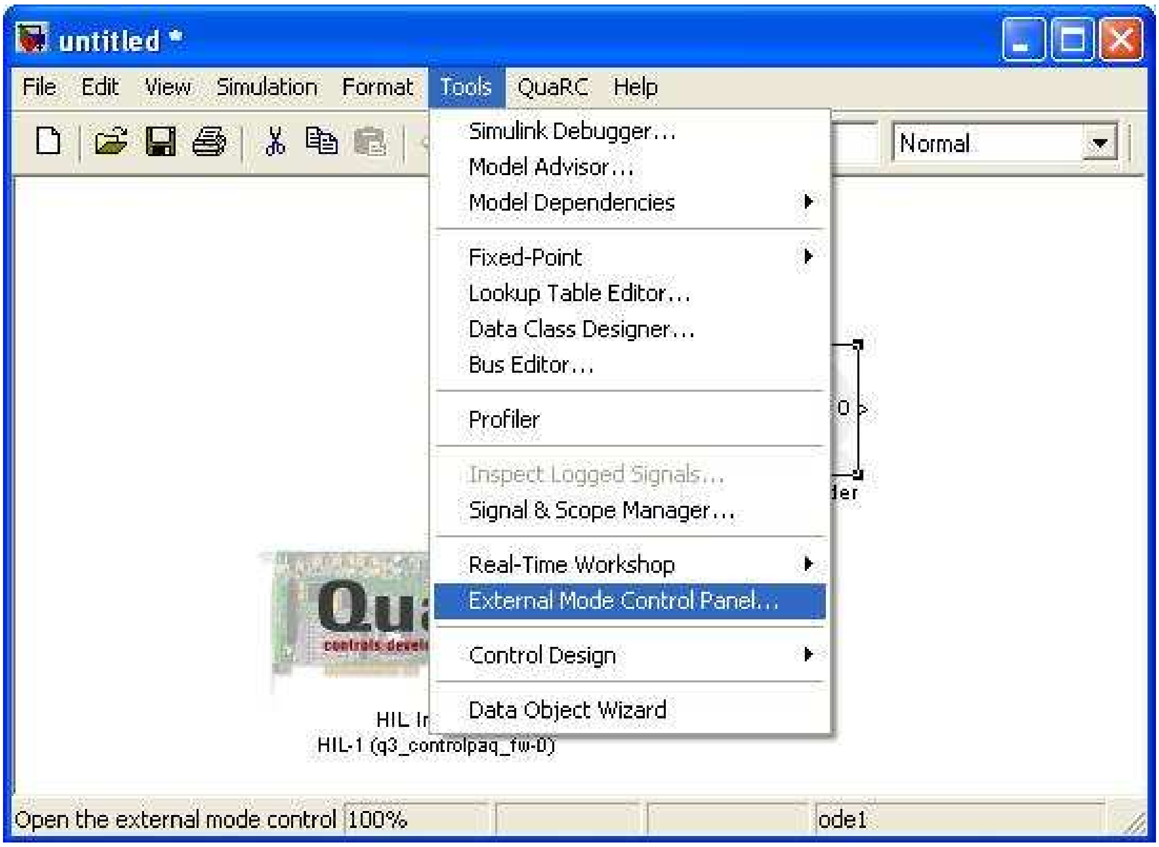
This shows a step function set with a step time of 0.001 seconds, which corresponds to one sample time. This set-up will save all but the very first data point. To record data, follow the procedure below.

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.

4. Go to the TOOLS pull down menu and select EXTERNAL MODE CONTROL PANEL.



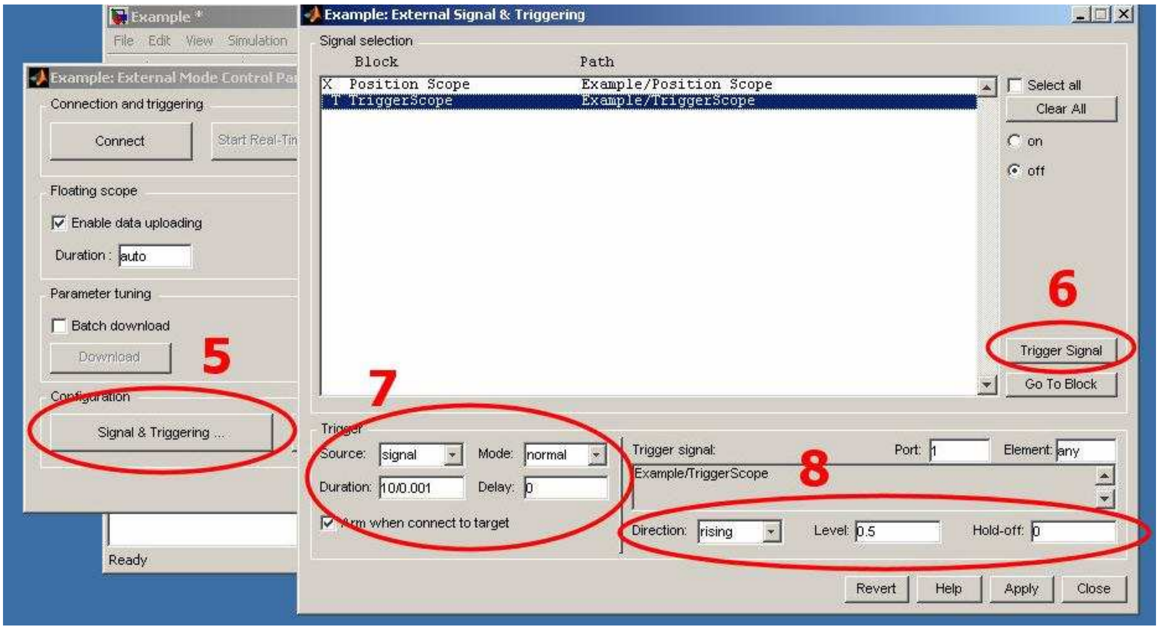
Open the external mode control

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 = Tf/Ts. The duration is the number of samples allotted in memory for each signal being logged. If we have a simulation time of Tf = 10 seconds and our sampling time Ts = 0.001, the 10/0.001 = 10,000 samples. If you actually run the model longer than Tf, then the data in memory will be overwritten when Tf/Ts samples have passed.

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.



Example *

Example: External Mode Control Pa

Connection and triggering

Floating scope

☑ Enable data uploading

Duration : auto

Parameter tuning

□ Batch download

Configuration

Ready

Example: External Signal & Triggering

Signal selection

	Block	Path
X	Position Scope	Example/Position Scope
T	TriggerScope	Example/TriggerScope

□ Select all

○ on

● off

Trigger

Source: signal

Mode: normal

Duration: 10/0.001

Delay: 0

☑ Arm when connect to target

Trigger signal: Example/TriggerScope

Port: 1

Element: any

Direction: rising

Level: 0.5

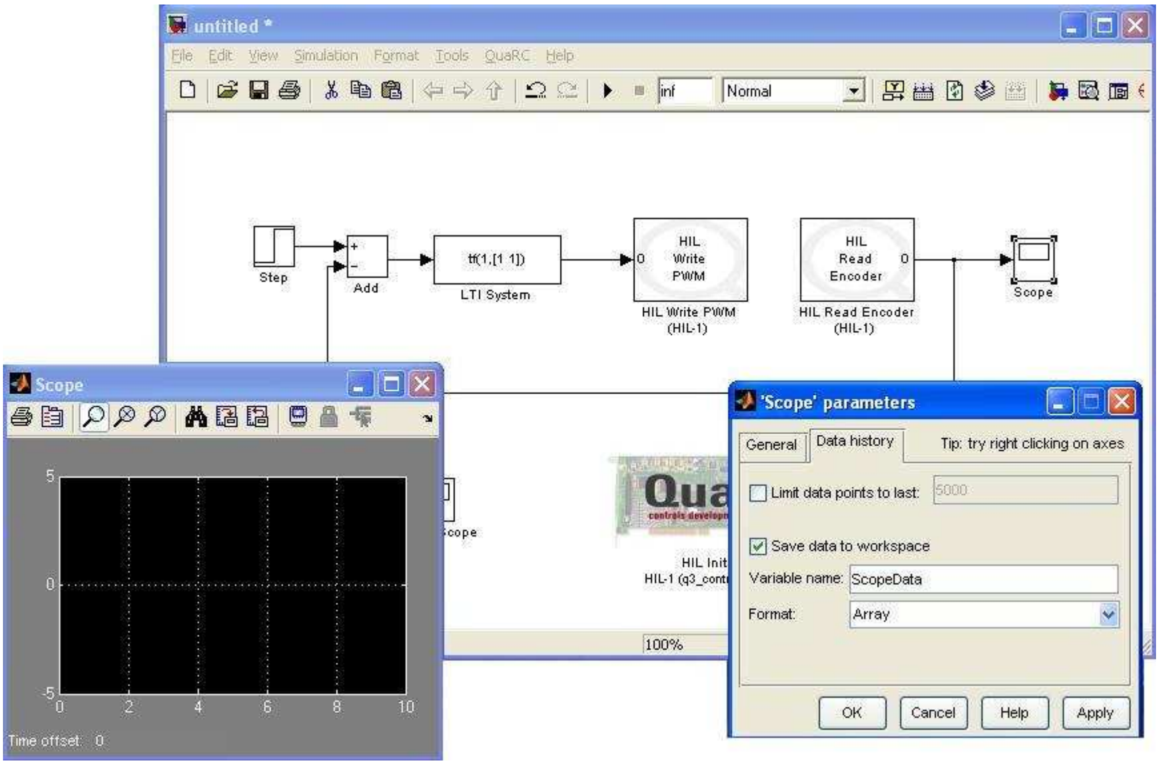
Hold-off: 0

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

10. Click the SCOPE PROPERTIES Icon, and select the DATA HISTORY 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 Save data to workspace box, and give the Variable name a unique and meaningful name for easy reference later. In the FORMAT menu choose the ARRAY option.



'Scope' parameters

General | Data history | Tip: try right clicking on axes

□ Limit data points to last: 5000

☑ Save data to workspace

Variable name: ScopeData

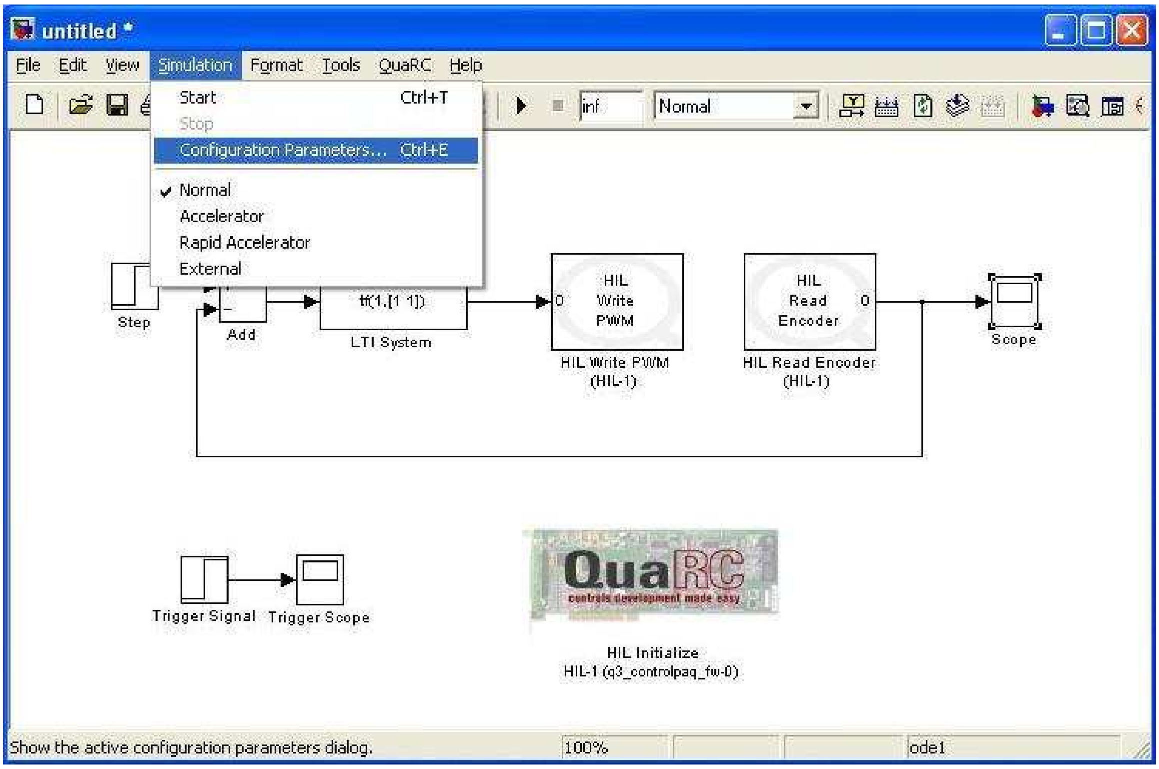
Format: Array

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.

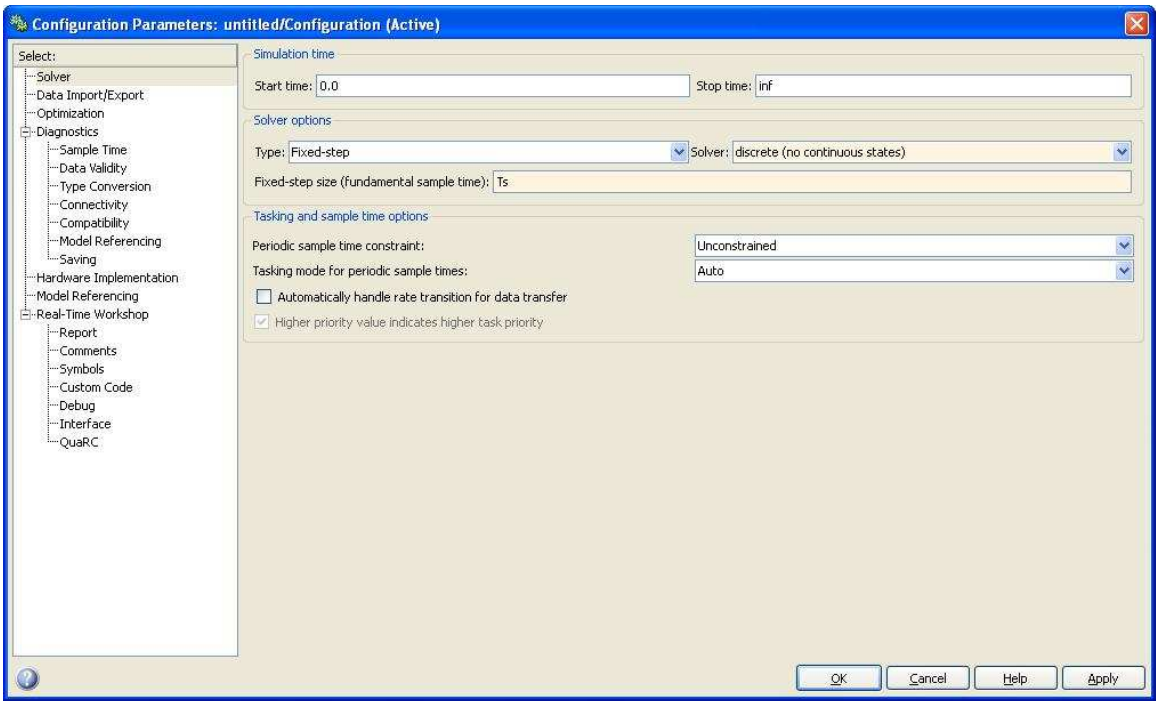
2.2 Real Time Windows Target Simulation Parameters

You must configure the way that the RTW builds its real time executable C-code. Real time Simulink models can run only with a Fixed Step Size. The Target Language Compiler must also be set to Real Time Windows Target so that the generated code is made for running on a PC which is running Widows and Simulink on the same machine. To do this,

1. Click on the SIMULATION pull down menu, and select CONFIGURATION PARAMETERS.



2. On the left side panel select SOLVER, and in the SOLVER OPTIONS area, select FIXED STEP for the solver TYPE. Choose and appropriate SOLVER for your model. If all blocks use discrete time blocks, then the discrete(no continuous states) is a good choice. If you are implementing a continuous time controller for example with a Laplace transform C(s), then you need to choose a differential equation solver like Ode4.



Configuration Parameters: untitled/Configuration (Active)

Simulation time

Start time: 0.0 Stop time: inf

Solver options

Type: Fixed-step Solver: discrete (no continuous states)

Fixed-step size (fundamental sample time): Ts

Tasking and sample time options

Periodic sample time constraint: Unconstrained

Tasking mode for periodic sample times: Auto

□ Automatically handle rate transition for data transfer

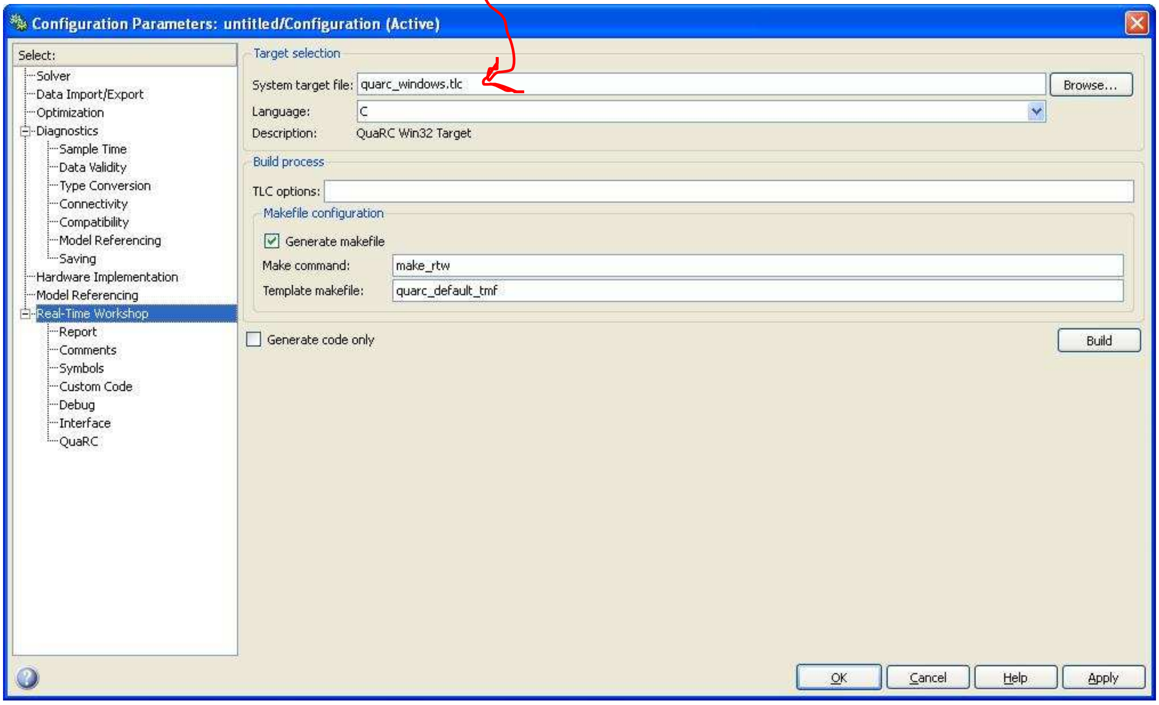
☑ Higher priority value indicates higher task priority

3. Set your fixed step size for the model, we usually use 0.001 if we are simulating a continuous time system. For a discrete-time system, use the sample period used for the design steps.

4. Next, select the REAL-TIME WORKSHOP on the left.

5. In the TARGET SELECTION area, click BROWSE, and scroll down to the bottom, and select ~~rtwin.tlc~~ from the menu. Verify that the other settings in the window match the screen shot below.

...as in picture below, select: quarc_window.tlc



Configuration Parameters: untitled/Configuration (Active)

Target selection

System target file: quarc_windows.tlc

Language: C

Description: QuaRC Win32 Target

Build process

TLC options:

Makefile configuration

☑ Generate makefile

Make command: make_rtw

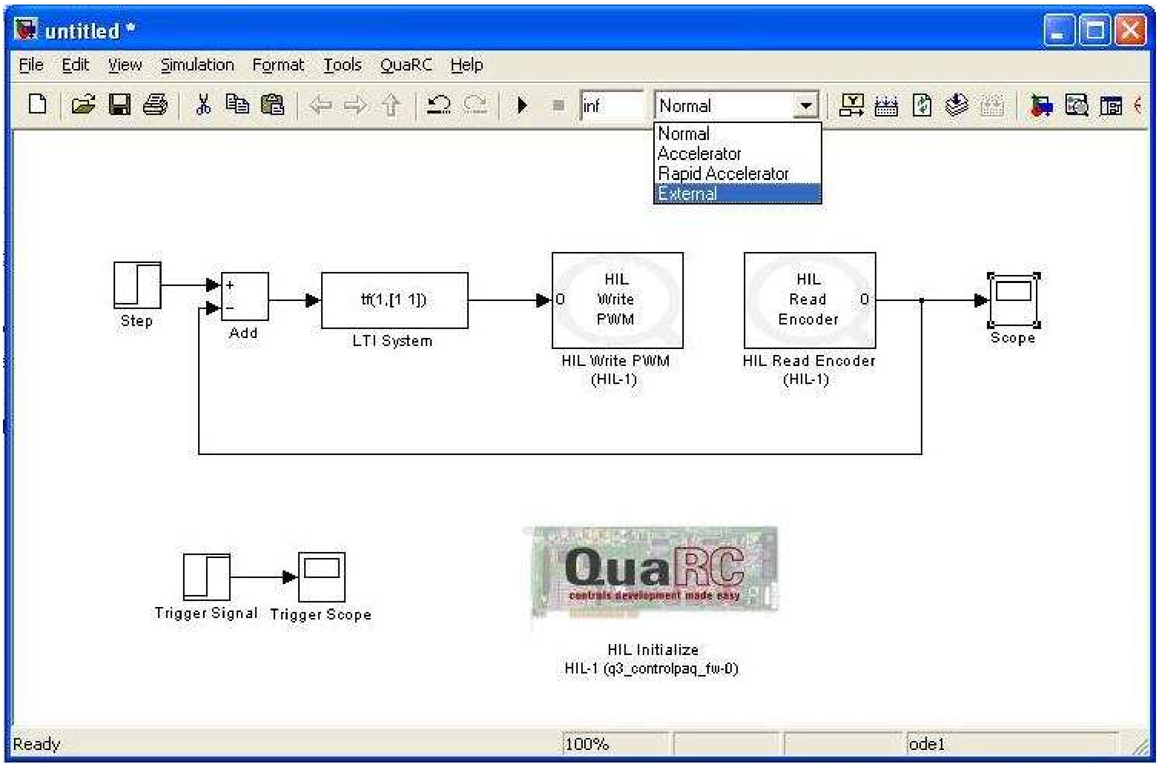
Template makefile: quarc_default_tmf

□ Generate code only

6. This sets the windows machine as the target for the real time code.

7. Click APPLY and OK

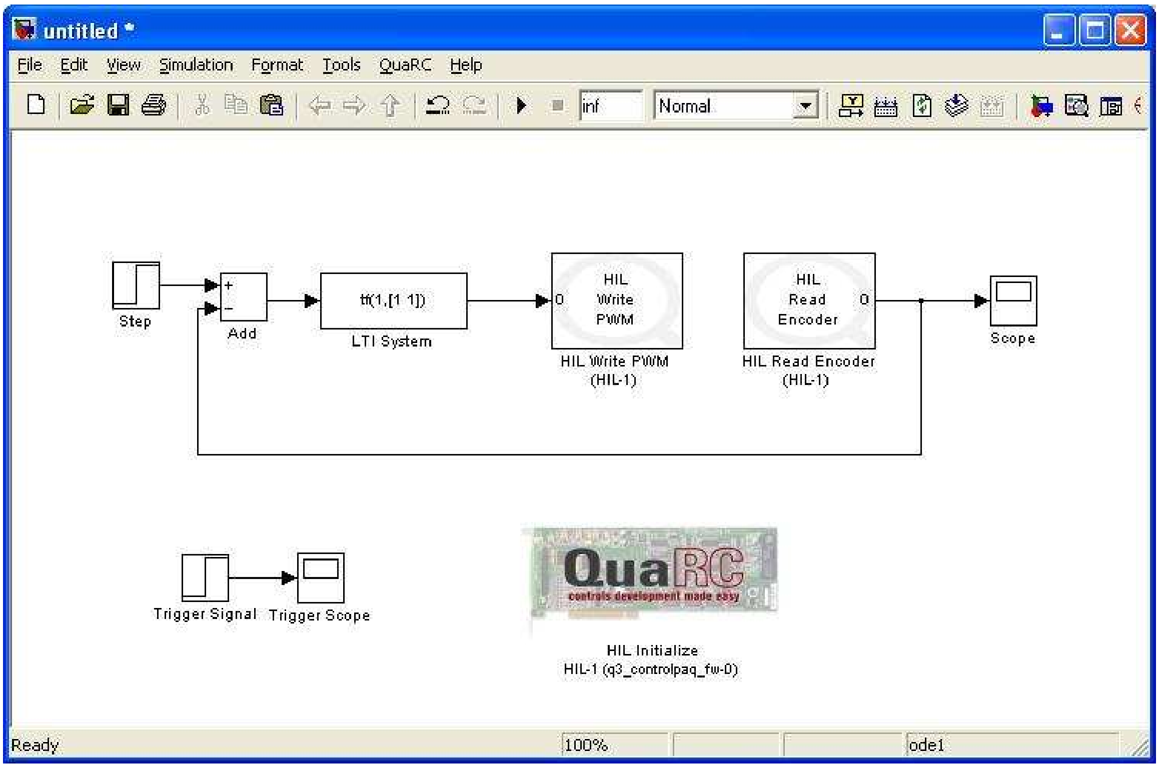
8. Lastly, the model must be set to EXTERNAL mode. In the model toolbar to the right of the final simulation time an the play button, is a pull down menu that usually says NORMAL, click the menu and select EXTERNAL.



9. The model must be BUILT first before it can be run. Building involves generating and compiling the real time code. To do this click on the TOOLS menu and under Real-Time Workshop select BUILD MODEL, or just hit CTRL+B.

10. The Matlab command window will show all of the steps of the build process and indicate if it completes successfully or not.

11. To run the model, you must connect to it by clicking the CONNECT TO TARGET icon to the right of the PLAY button in the model's toolbar.



12. If everything is working correctly the model will start the Real Time Kernel which supersedes Windows and guarantees real time execution. Clicking on the PLAY button will run the model.

13. You will often get an error CHECKSUM MISMATCH. This means that something important in the model has changed since the last real time build, and the model must be built again. Hit CTRL+B. You will find yourself building these models often as you tune your Simulink diagrams.

A word of warning: RTW gives you the capability to change gain parameters in real-time. In other words, changing a gain value while the controller is running will *immediately* change that value. This should be done with great care as the equipment may be damaged by allowing the cart to run against the end of the track or by feeding high frequency signals to the motor.

3 Digital Filter

The following procedure gives a discrete-time controller with the same step response as an analog controller. The procedure is fairly intuitive when you think about it.

Take a continuous time filter,

$$P(s) = \frac{a}{s+a}$$

and examine its output, $Y(s)$, in response to a step input. So,

$$Y(s) = \frac{a}{s(s+a)}$$

This corresponds to the time domain signal,

$$y(t) = 1 - e^{-at}$$

Now suppose we sample this time domain signal with a sampling period of T. This is equivalent to choosing $t = kT$, where k will now be our discrete time index. So our sampled signal is

$$y(kT) = 1 - e^{-akT}$$

In the z domain, this is

$$Y(z) = \frac{z(1 - e^{-aT})}{(z-1)(z - e^{-aT})}$$

We would like for this to be the output of some system, call it $P(z)$, when the input is a sampled step function. A sampled step function has the z transform

$$U(z) = \frac{z}{z-1}$$

Therefore, we want to choose $P(z)$ such that

$$Y(z) = P(z)U(z)$$

We can do this by examination, which yields

$$\begin{aligned} P(z) &= \frac{1 - e^{-aT}}{z - e^{-aT}} \\ &= \frac{z^{-1}(1 - e^{-aT})}{1 - e^{-aT} z^{-1}} \end{aligned}$$

The discrete time filter can now be implemented. This discrete-time transfer function is referred to as the “step-equivalent”.