Laboratory guide 1: Introduction to the DC servomotor experimental setup

Riccardo Antonello* Francesco Ticozzi*

March 30, 2021

# 1 Activity goal

The purpose of this laboratory activity is to become familiar with the DC servomotor unit used in the next laboratory activities. At the end of this activity, you should be able to implement and configure a Simulink model that is suitable for a "real–time simulation", namely a simulation that runs in real–time and envisages a direct interaction with a physical device.

# 2 Experimental setup

The experimental setup consists of:

1) **Quanser SRV–02 servomotor**. This is the physical device to be controlled (*plant*). It consists of a DC motor (① in Fig. 1c) with a built–in *planetary* gearbox (② in Fig. 1c) capable of driving a mechanical load attached to its output shaft (② in Fig. 1b). The mechanical load considered in this course is a simple disc inertia (i.e. *inertial load* – ① in Fig. 1e). An incremental optical encoder is directly connected to the output shaft of the SRV–02 unit for measuring the position of the mechanical load (② in Fig. 1d). A potentiometer (③ in Fig. 1d) is also connected to the shaft through a gear coupling (that includes an anti–backlash mechanism – ③ in Fig. 1b), and can be used as an alternative load position sensor. The DC motor is driven by a linear voltage driver based on a power operational amplifier (① in Fig. 1d). The simplified electrical schematic of the voltage driver is reported in Fig. 2d. It is rather immediate to deduce that the voltage driver behaves – from "Motor Input" to "Motor Current A" output – as a first order low–pass filter with DC gain

$$k_{\mathrm{drv}} = \frac{R_2}{R_1 + R_2}\left(1 + \frac{R_3}{R_4}\right) \approx 0.6 \tag{1}$$

and cut–off frequency

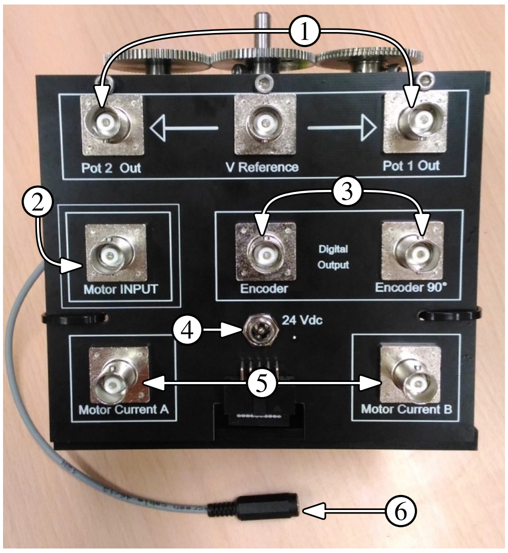
$$f_c = \frac{1}{2\pi\,(R_1 // R_2)\,C_1} \approx 1.2\,\mathrm{kHz} \tag{2}$$

A *shunt resistor* is connected in series to the motor to sense the *armature current* (i.e. the current flowing through the rotor windings).

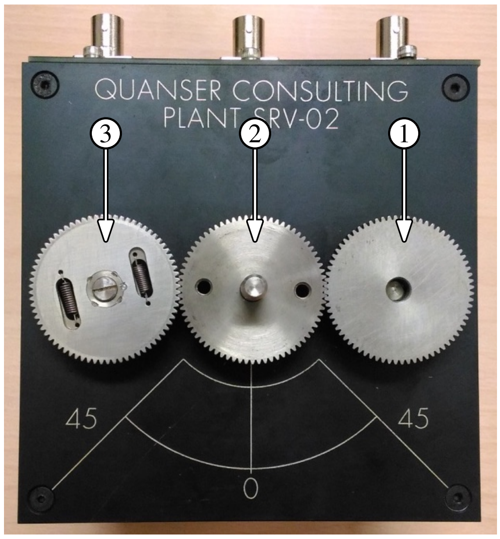
2) **National Instruments Data Acquisition Board (NI-DAQ PCI-6221 or PCIe-6321)**.

The PCI–6221 board (installed on PCFA05 ÷ PCFA12 workstations) supports

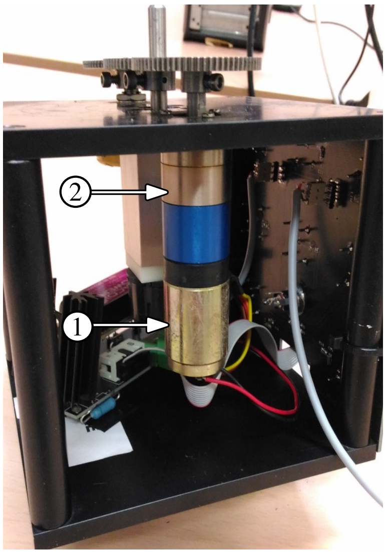
*Department of Information Engineering (DEI), University of Padova.
email: {antonello, ticozzi}@dei.unipd.it



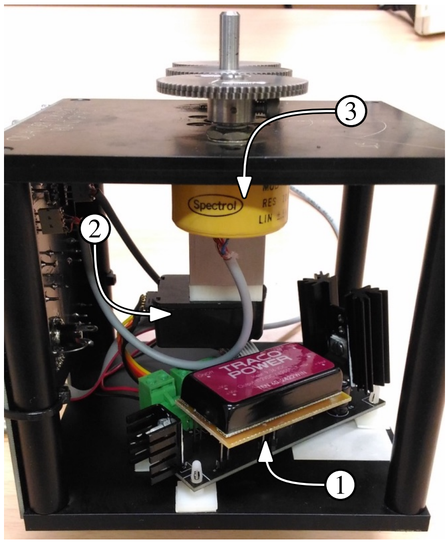

(a)



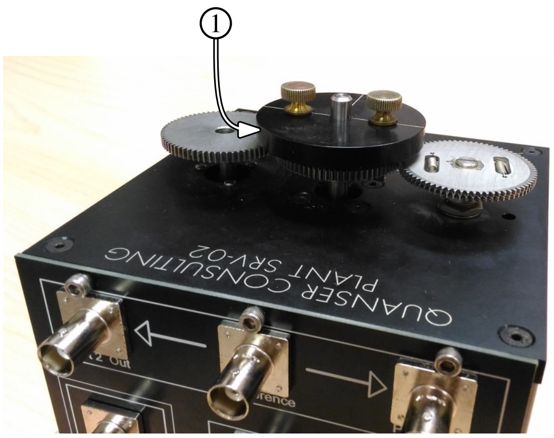

(b)




(c)

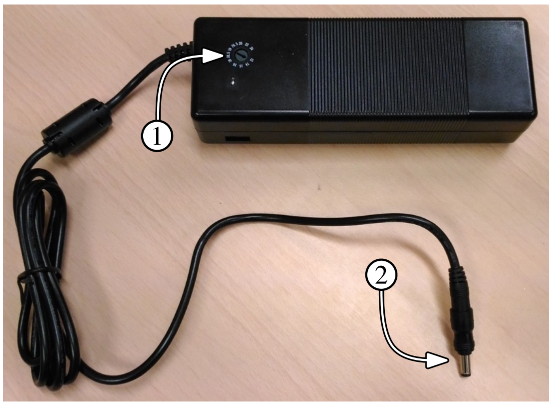



(d)

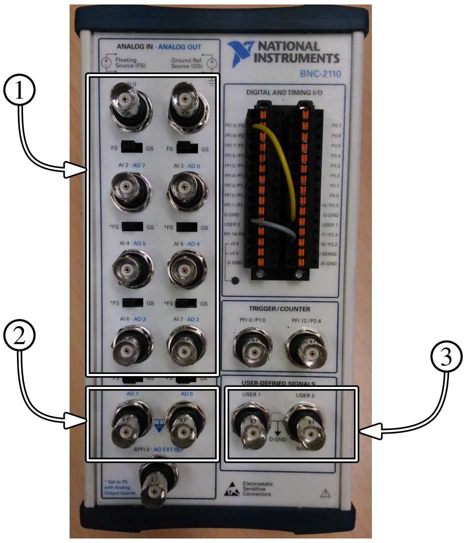



(e)

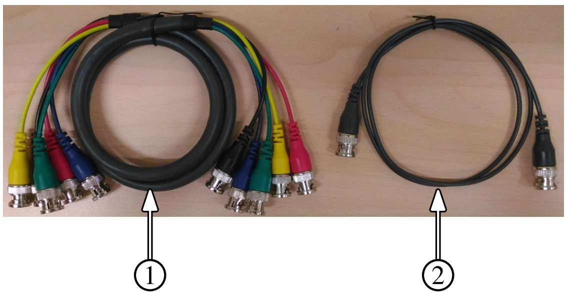
Figure 1: Experimental setup details.



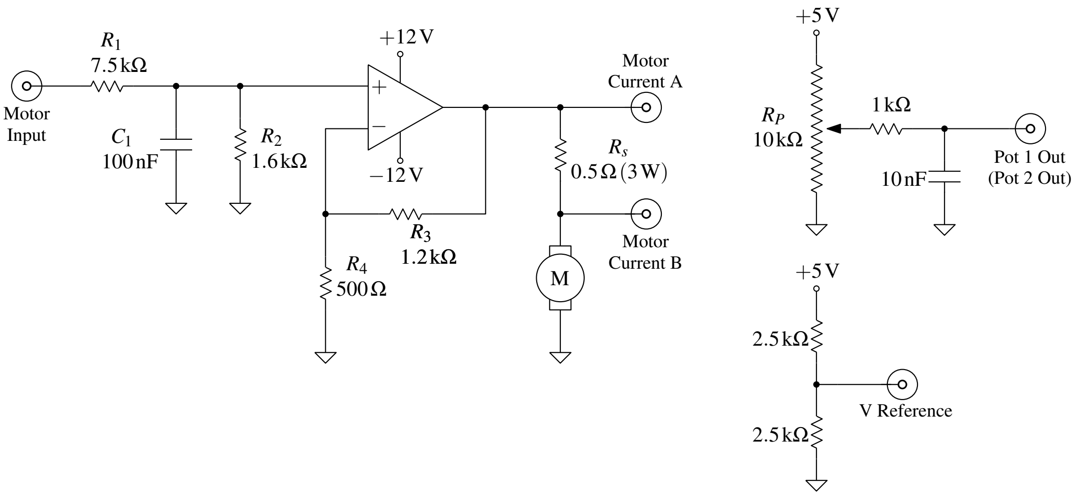

(a)




(b)




(c)




(d)

Figure 2: Experimental setup details (cont'd).

- 16 single-ended or 8 differential analog inputs channels with 16 bit ADC resolution, maximum sample rate (per channel) of $250\,\mathrm{kS/s}$ and configurable ADC input range (available options: $\pm 0.2\,\mathrm{V}$, $\pm 1\,\mathrm{V}$, $\pm 5\,\mathrm{V}$, $\pm 10\,\mathrm{V}$).
- 2 analog output channels with 16 bit DAC resolution, fixed $\pm 10\,\mathrm{V}$ output range and maximum sample rate of $833\,\mathrm{kS/s}$ for single–channel output mode and $740\,\mathrm{kS/s}$ for dual–channel output mode.
- 24 digital I/O channels.
- 2 general purpose 32–bit counters/timers that can be used for PWM generation, encoder counting, frequency measurement, event counting, etc.

The PCIe–6321 board (installed on PCFA01 $\div$ PCFA04 workstations) has similar specifications to PCI–6221, expect for minor differences such as

- the DAC maximum sampling rates: $900\,\mathrm{kS/s}$ for single–channel output mode and $840\,\mathrm{kS/s}$ for dual–channel output mode.
- the increased number of general purpose counters/timers: 4 instead of 2.

3) **National Instruments BNC–2110 terminal board**.

4) **BNC–terminated connection cables** (three single–channel cable + one 5–channel cable for each workbench).

5) **DC power supply** with adjustable voltage output (laptop power adapter).

6) **PC workstation** running **MATLAB/Simulink**. The **Simulink Desktop Real–Time** toolbox (SLDRT) is used to perform a "real–time simulation" that involves a direct interaction with the experimental setup.

The experimental setup parameters that are relevant for simulation and control design purposes are listed in Tab. 1. The nominal values of the parameters are deduced from accompanying data–sheets.

## 3 Real–time simulation

The MATLAB/Simulink environment will be used throughout the course to support all the activities that are generally involved with the typical design flow of a control system. The implementation and experimental testing of a control system will be also done within the same environment, by resorting to the *Simulink Desktop Real–Time (SLDRT)* toolbox. This toolbox provides a real-time kernel for executing Simulink models on a computer running Windows (or Mac OS X). It includes library blocks to interface a Simulink model with several I/O devices that cover a wide range of I/O types, such as analog I/O signals, digital I/O signals, encoder/counter input signals, PWM/frequency outputs and communication protocols (serial, UDP, TCP, etc.). In order to run a Simulink model in real time with the SLDRT support, it is first necessary to convert it into an executable code, that is next automatically loaded into memory and run by the SLDRT kernel whenever the "real–time simulation" is started. The whole conversion process of a Simulink model into an executable code is managed by MATLAB and is transparent to the user. The process basically consists of two stages: in the first stage, the Simulink model is automatically converted into a C language source code by the *Simulink Coder* toolbox. Then, in the second stage, the generated C–code is automatically compiled using a supported C compiler.

| DC gearmotor nominal parameters | |
|---|---|
| Motor type | brushed DC motor |
| Motor id | Faulhaber 2338S006S |
| Nominal voltage | $6\,\mathrm{V}$ |
| Nominal output power | $3.23\,\mathrm{W}$ |
| Motor efficiency | $0.69$ |
| Armature resistance | $2.6\,\Omega$ |
| Armature inductance | $180\,\mu\mathrm{H}$ |
| No–load speed | $7200\,\mathrm{rpm}$ |
| No–load current | $0.08\,\mathrm{A}$ |
| Stall torque | $1.71\,\mathrm{Ncm}$ |
| Back–EMF constant | $0.804\,\mathrm{mV/rpm}$ |
| Torque constant | $7.68 \times 10^{-3}\,\mathrm{Nm/A}$ |
| Rotor inertia | $3.90 \times 10^{-7}\,\mathrm{kg\,m^2}$ |
| Gearbox type | planetary gearbox |
| Gearbox id | Micromotor SA 23/1 |
| Gearbox ratio | $14$ |
| Gearbox efficiency | $0.80$ |
| Moment of inertia of external 72–tooth gear | $1.4 \times 10^{-6}\,\mathrm{kg\,m^2}$ |
| Moment of inertia of extra disc | $3.0 \times 10^{-5}\,\mathrm{kg\,m^2}$ |

| Sensors data | |
|---|---|
| Sensor type | optical incremental encoder |
| Sensed quantity | output shaft angular position (incremental) |
| Sensor id | Hewlett-Packard HEDS-5540#A06 |
| Pulses–per–rotation (ppr)[(1)] | 500 (all units except "MOTORE 8" and "10")<br>1024 (on "MOTORE 8" and "10" units) |
| (1) $\times 4$ times larger when adopting a quadrature encoding mode for counting the encoder pulses. | |
| Sensor type | potentiometer "1" |
| Sensed quantity | output shaft angular position (absolute) |
| Sensor id | Spectrol 138-0-0-103 |
| Resistance | $10\,\mathrm{k\Omega}$ |
| Angle range | $\pm 176$ deg |
| Supply voltage | $5\,\mathrm{V}$ |
| Sensor type | potentiometer "2" |
| Sensed quantity | flexible joint angular displacement (absolute) |
| Sensor id | Spectrol 357-0-0-103 |
| Resistance | $10\,\mathrm{k\Omega}$ |
| Angle range | $\pm 170$ deg |
| Supply voltage | $5\,\mathrm{V}$ |

| Voltage driver data | |
|---|---|
| Driver type | linear voltage driver (LM1875 power op-amp) |
| Driver gain | $\approx 0.6$ |
| Driver cut-off frequency | $\approx 1.2\,\mathrm{kHz}$ |

| I/O board data | |
|---|---|
| Board id | NI PCI-6221 (on PCFA05 ÷ PCFA12 workstations)<br>NI PCIe-6321 (on PCFA01 ÷ PCFA04 workstations) |
| ADC range | $\pm 0.2\,\mathrm{V}$, $\pm 1\,\mathrm{V}$, $\pm 5\,\mathrm{V}$, $\pm 10\,\mathrm{V}$ |
| ADC resolution | $16\,\mathrm{bits}$ |
| DAC range | $\pm 10\,\mathrm{V}$ |
| DAC resolution | $16\,\mathrm{bits}$ |

Table 1: Experimental setup data.

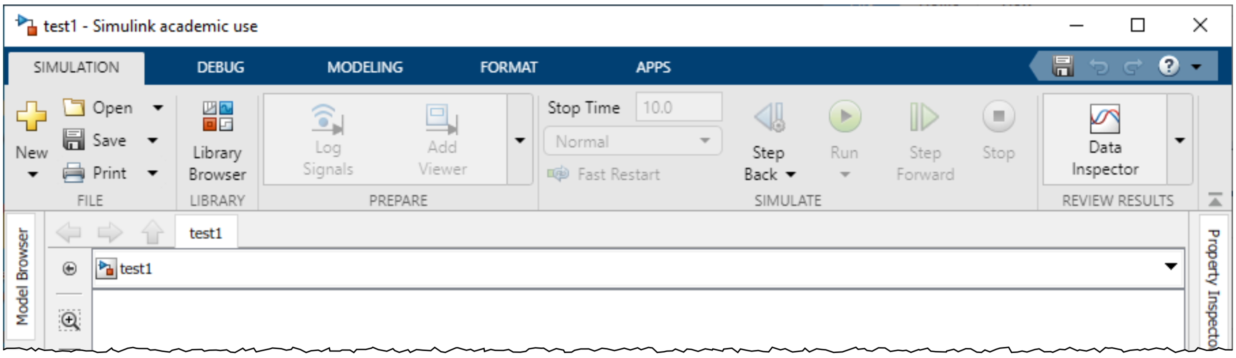
## 3.1 Configure a Simulink model for Simulink Desktop Real-Time

Below are reported the steps required to configure a Simulink model to run in real–time with the support of the SLDRT toolbox:

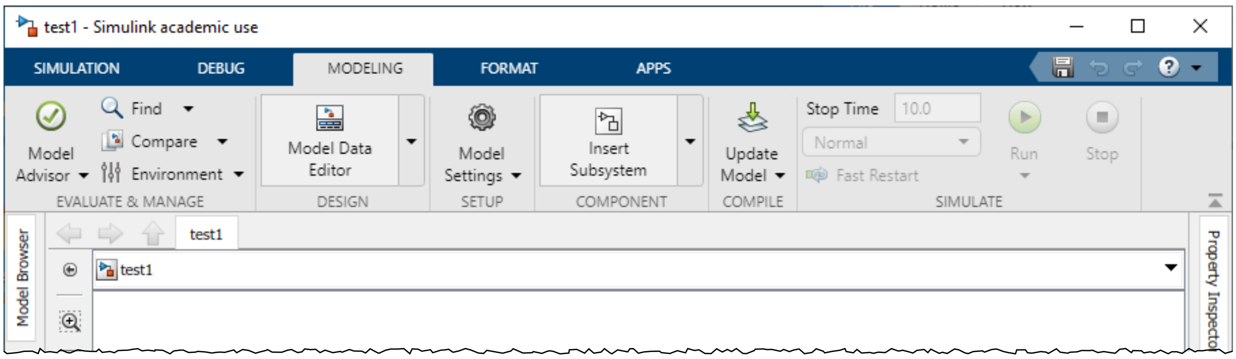
1) Change the current MATLAB working directory to your working directory, by using the MATLAB command cd, or the *Current Directory* field in the MATLAB window toolbar.

2) Open a new Simulink model, either by selecting *New* → *Model* from the MATLAB window toolbar, or by invoking `slLibraryBrowser` from the command window and then using the *New Button* in the toolbar of the *Simulink Library Browser* window.

3) Save the Simulink model in the working directory, by selecting **Save** → **Save as ...** from the **Simulation** tab on the Simulink model window. Note that the Simulink Coder generates some auxiliary files in the current working directory during the code generation process. Therefore, ensure to save the Simulink model file in your current working directory, to prevent the generation of the auxiliary files in other folders.




4) To configure a model for Simulink Desktop Real-Time, open the *Configuration Parameter* window by clicking **Model Settings** on the **Modeling** tab of the Simulink model window.

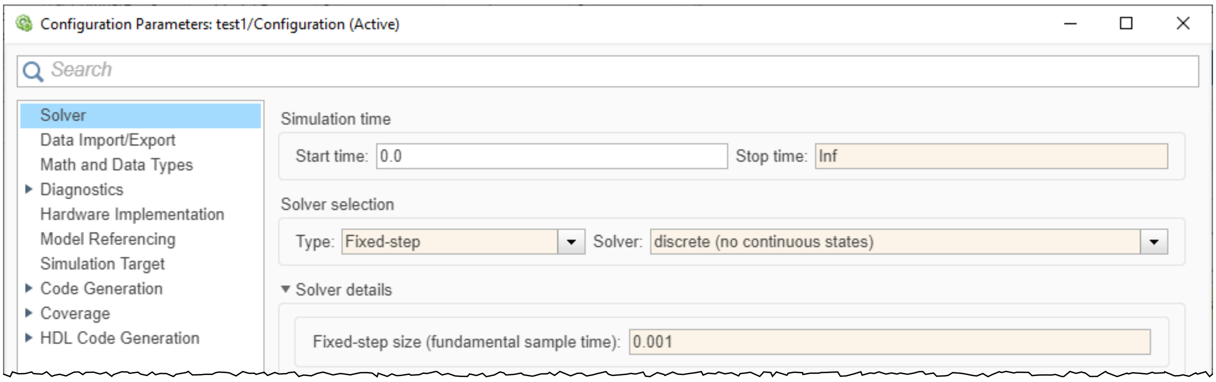



5) Click the **Solver** node. In the **Stop time** field, set an appropriate final time; use `Inf` to run the simulation indefinitely, until the stop button is pressed by the user.

From the **Type** list, select **Fixed-step**, because the *Simulink Coder* does not support variable step solvers. From the **Solver** list, select a solver. If your Simulink model contains continuous-time blocks, then select an appropriate *ode* solver to numerically solve (integrate) them: for example, choose a general-purpose solver such as `ode5 (Dormand-Prince)`. If instead your Simulink model contains only discrete-time blocks, then select the `discrete (no continuous state)` solver.

Under **Additional options**, in the **Fixed step size (fundamental sample time)** field, enter a sample time. For example, enter $0.001$ for running the Simulink model in real-time with a $1\,\mathrm{ms}$ sample time.

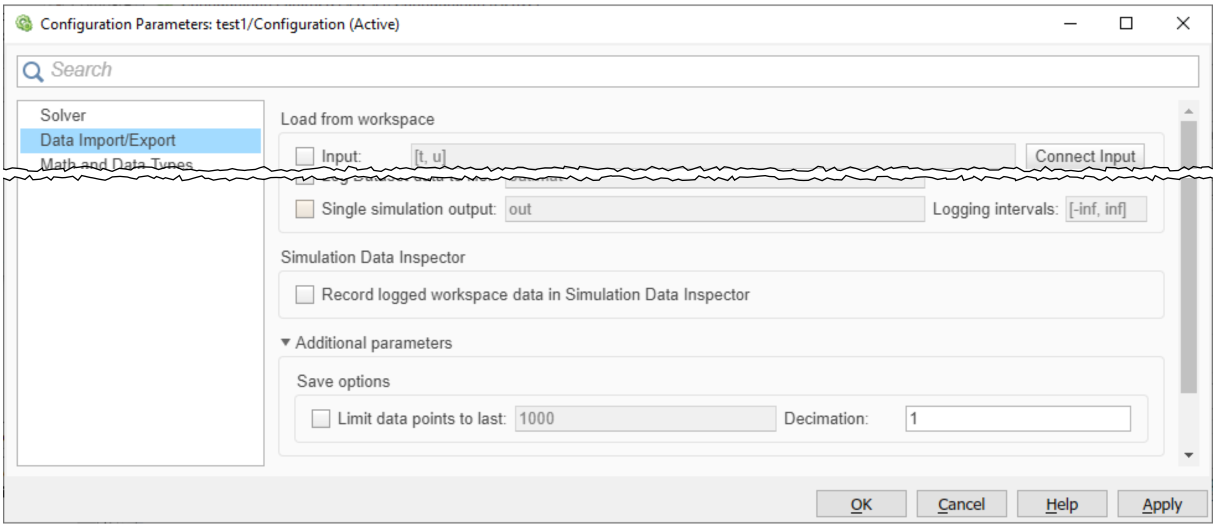
Press the **Apply** button to confirm the new simulation settings.



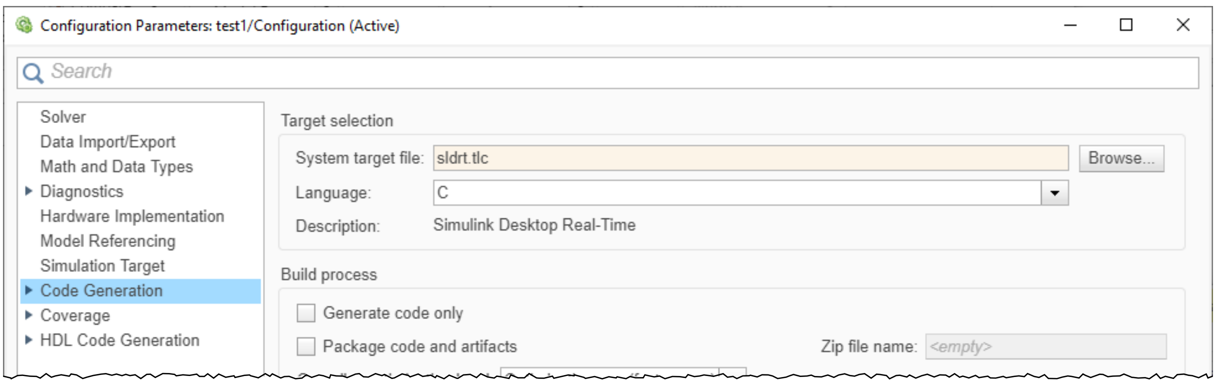

6) Click the **Data Import/Export** node. Uncheck the **Single simulation output** to export simulation data logged by *Scope* and *To Workspace* blocks as independent variables in the MATLAB workspace. Otherwise, a single container variable (of type `Simulink.SimulationOutput`) will be generated.

Under **Additional parameters**, uncheck **Limit data points to last** to save all the data produced by the simulation.

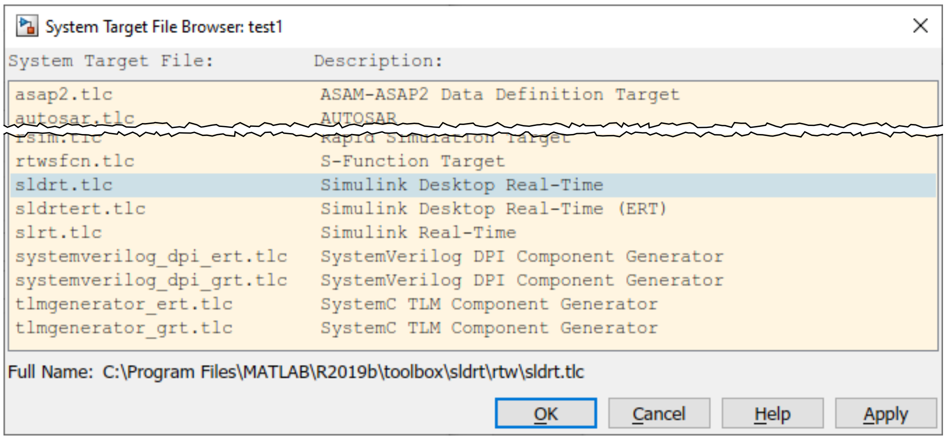
Press the **Apply** button to confirm the new simulation settings.




7) Click the **Code generation** node. In the **Target selection** section, click the **Browse** button at the **System target file** list.

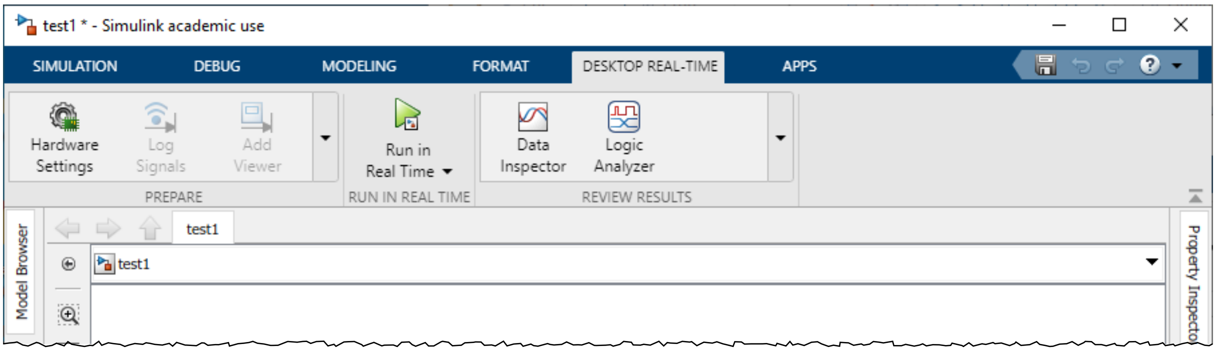



In the **System Target Browser** select `sldrt.tlc` (i.e. the *system target file* for building a Simulink Desktop Real-Time application), and press the **OK** button.

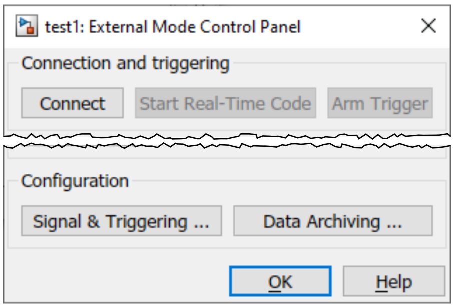



8) Press the **OK** button in the *Configuration Parameter* window to close it and confirm all the new simulation parameters settings.

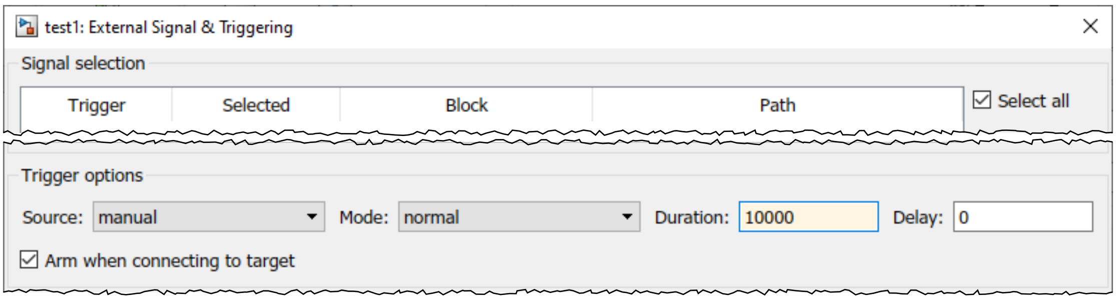
9) In the Simulink model window, on the **Desktop Real-Time** tab, click **Prepare** $\rightarrow$ **Control Panel** to open the *External Mode Control Panel*.




10) In the *External Mode Control Panel*, click the **Signal & Triggering** button to open the *External Signal & Triggering* window.




11) In the *Trigger options* section, in the **Duration** field enter length (number of sample points) of the buffer used to exchange data between Simulink and the real-time application. For example, with a sample time equal to $0.001$, a Duration value of $10000$ samples allows to store $10\,\mathrm{s}$ of simulation data.




Click the **OK** button to save changes and close the *External Signal & Triggering* window. On the *External Mode Control Panel*, click **OK** to close the window.

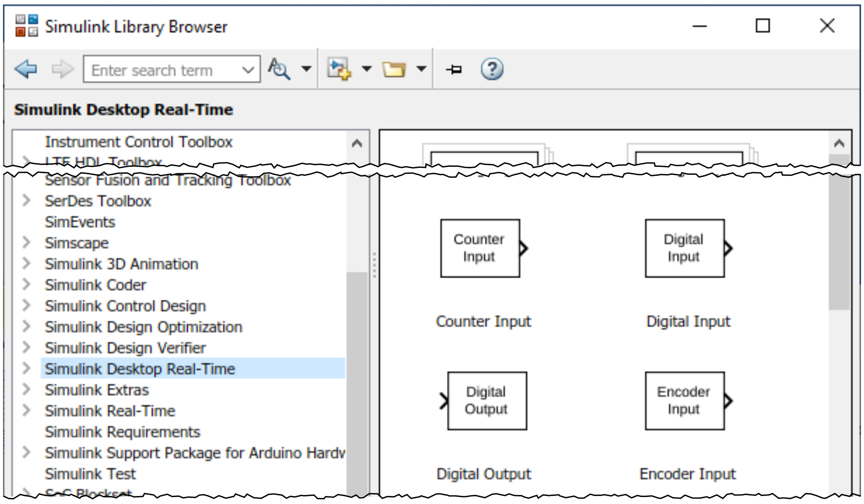
Note: when completely full, the buffer is erased before saving new data. This implies that the data saved in the buffer are at most the last "Duration" samples produced by the simulation.

12) On the **Simulation** tab of the Simulink model window, click the **Save** button to save the model with the new configuration settings.

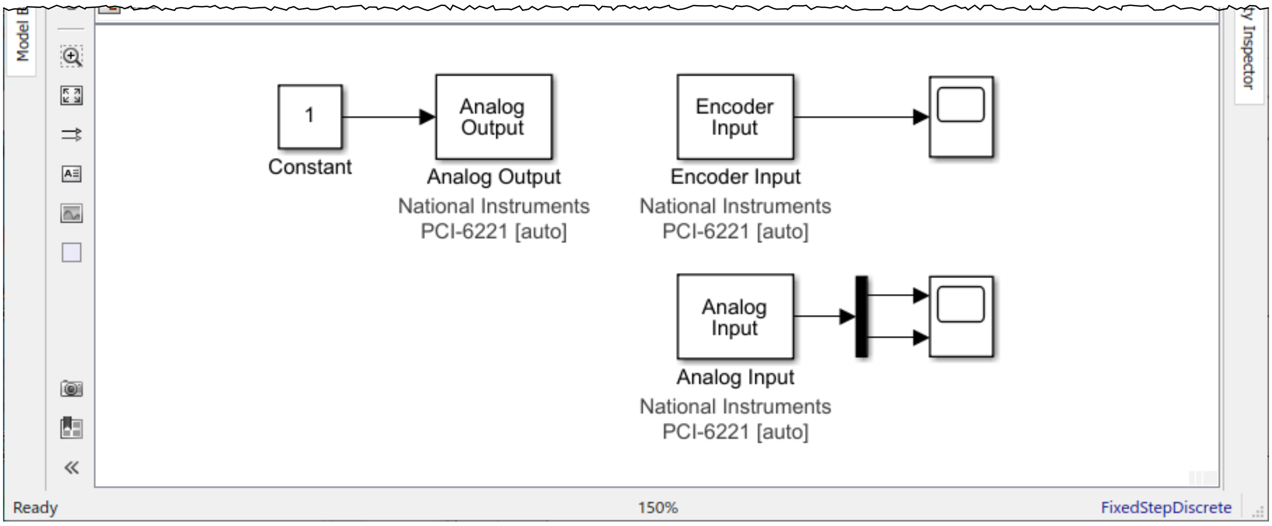
## 3.2 Interface the Simulink model with the hardware

The SLDRT blockset allows to connect a Simulink model to a range of I/O devices, including the NI-DAQ board installed on the laboratory PC. The following example shows how to use the SLDRT blocks to interface a Simulink model with the Quanser SRV-02 servomotor:

1) In the **Simulink Library Browser**, select the **Simulink Desktop Real–Time** blockset node.




2) Drag & Drop the **Analog Input**, **Analog Output** and **Encoder Input** blocks of the SLDRT in your Simulink model window.




The **Analog Input** block reads the value of a specified input channel (or list of channels) of the Analog-to-Digital Converter (ADC) integrated in the selected I/O board.

In this example, the block is used to (simultaneously) acquire the "Motor Current A" and "Motor Current B" signals of the Quanser SRV–02 unit. These are the two components of the Analog Input block output. They can be separated with a *Demux* block, and displayed on a *Scope*.

The **Analog Output** block writes a value on a specified output channel (or list of channels) of the Digital-to-Analog Converter (DAC) integrated in the selected I/O board.

In this example, the block is used to apply a desired voltage at the "Motor Input" of the Quanser SRV–02 unit. Use a *Constant* block to specify the voltage value to generate with the DAC.

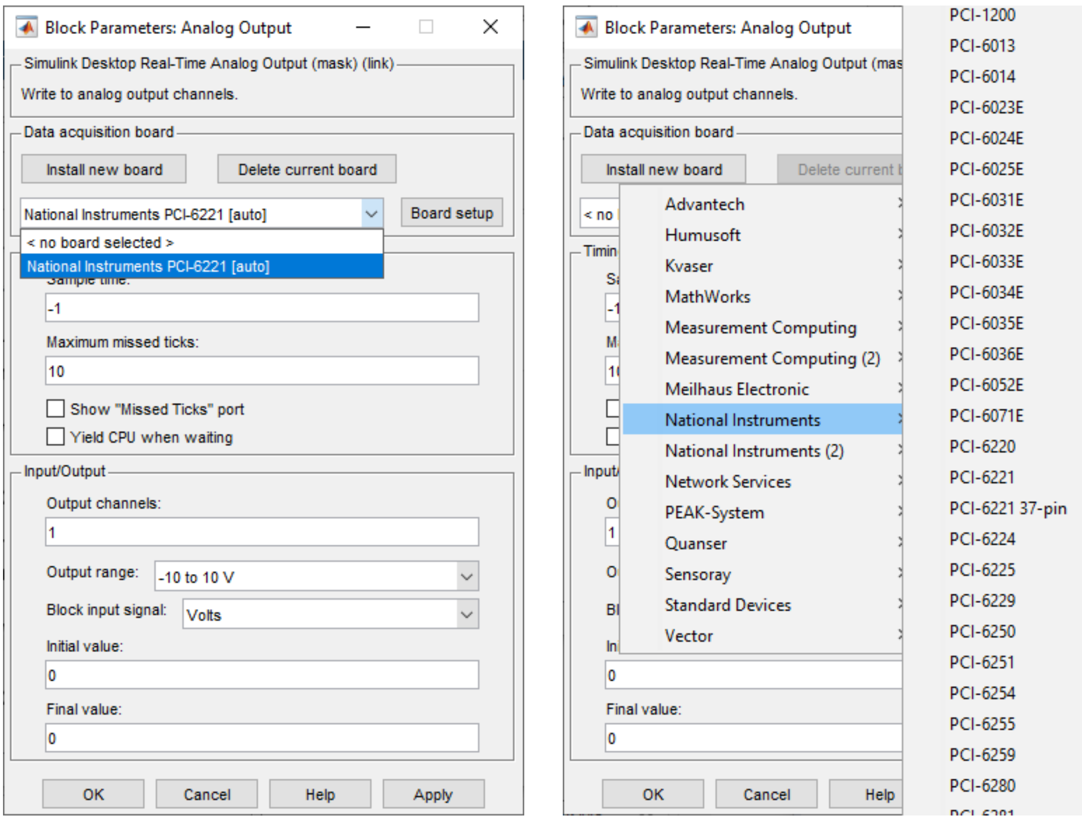
The **Encoder Input** block reads the register of a specified Encoder Pulse Counter integrated in the selected I/O board.

In this example, the block is used to get the pulse count of the incremental optical encoder installed in the Quanser SRV–02 unit. Use a *Scope* to display the output of the Encoder Input block.

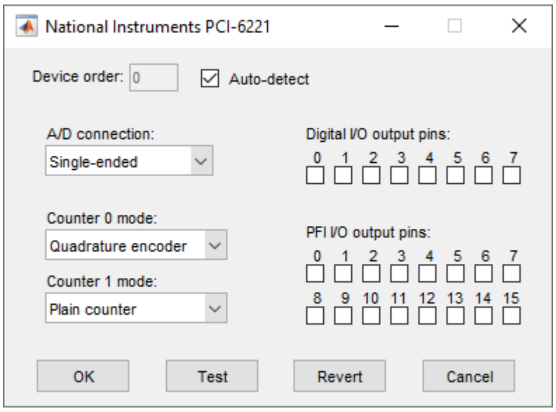
3) Double-click on the *Analog Output* block to open the *Block Parameters: Analog Output* dialog window.

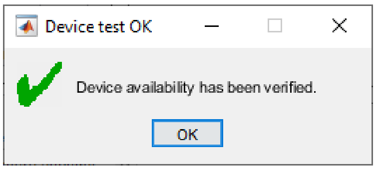
In the *Data acquisition board* section, specify the I/O board associated with the block (either `National Instruments PCI-6221` or `PCIe-6321`, depending on your workstation):

- if the board has been already installed and configured to work with Simulink, simply select the corresponding entry in the drop–down list accessible by clicking the "▼" button. Then click the **Board setup** button to check the board configuration settings.
- if the board has not been installed yet, click the **Install new board** button, and select the appropriate entry in the menu of supported boards.




4) A new window is opened to configure the board settings. In the **Counter 0 mode** drop–down list of the new window, select the `Quadrature encoder` option. This set the encoder counter of the NI-DAQ board to count both the rising and falling edges of the two encoder outputs (Note: this setting is only relevant for the Encoder Input block that will be configured later). Next, click the **Test** button to verify the board installation: the *Board Test OK* appears in case of success. Finally, click the **OK** button to confirm the board installation.





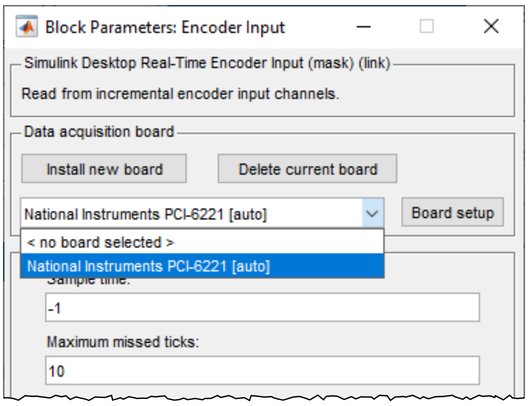

5) In the *Timing* section of the *Analog Output* dialog window, set **Sample Time** to the desired value in seconds, or leave "$-1$" to inherit the sample time from the configuration parameters of the Simulink model.

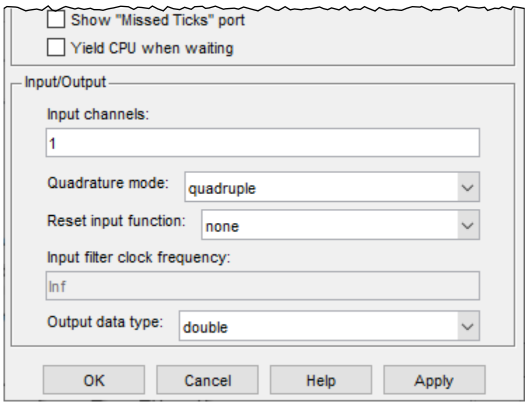
6) In the **Output channels** field of the *Input/Output* section, specify the list of DAC channels to use. Enter "1" in the field for using the first DAC channel, which is connected to the "Motor Input" of the SRV–02 unit via the "AO 0" terminal of the BNC–2110 terminal board.

Set both the **Initial value** and **Final value** fields to zero, in order to set the DAC output to zero at the beginning and ending of the simulation. This ensures that the motor will be turned off at the end of every simulation.

7) Click the **OK** button to confirm the settings and close the *Block Parameters: Analog Output* dialog window..

8) Double-click on the *Encoder Input* block to open the *Block Parameters: Encoder Input* dialog window.






In the *Data acquisition board* section, open the drop–down list (click on the "▼" button) and select the I/O board installed in the previous steps (either `National Instrument PCI-6221` or `National Instrument PCIe-6321`, depending on your workstation) installed in the previous steps.

9) In the **Sample Time** field of the *Timing* section, set the desired sample time value in seconds, or leave "$-1$" to inherit the sample time from the configuration parameters of the Simulink model.

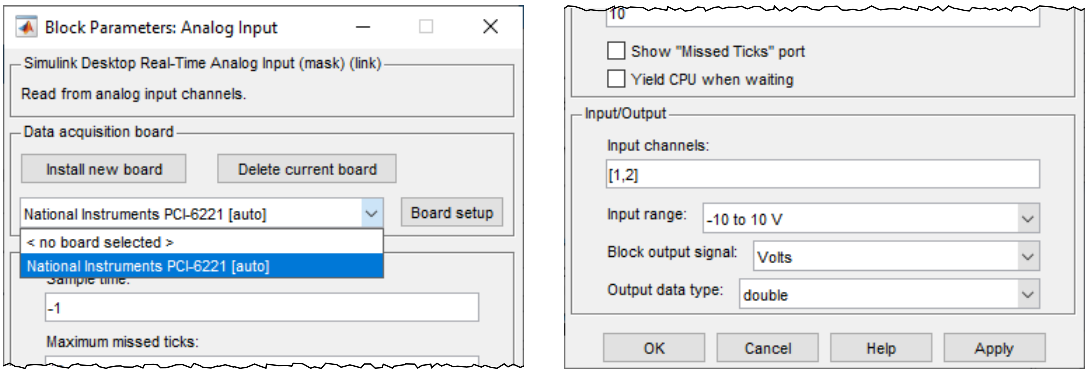
10) In the **Input channels** field of the *Input/Output* section, specify the list of encoder counters to read. Enter "1" in the field for using the counter 0 (note: this is the counter that has been configured to count using a quadrature encoding mode during the board installation and setup), whose inputs are connected to the "Encoder" and "Encoder 90°" outputs of the SRV–02 unit via the "USER 1" and "USER 2" terminals of the BNC–2110 terminal board.

In the **Quadrature mode** drop–down list, select the option `quadruple` to enable the counter to count using the quadrature encoding mode.

In the **Reset input function** drop–down list, select the option `none` to reset the counter register only at the beginning of the simulation. The zero position reference is therefore assumed as the position of the encoder shaft at the beginning of the simulation.

11) Click the **OK** button to confirm the settings and close the *Block Parameters: Encoder Input* dialog window.

12) Double-click on the *Analog Input* block to open the *Block Parameters: Analog Input* dialog window.




In the *Data acquisition board* section, open the drop–down list (click on the "▼" button) and select the I/O board installed in the previous steps (either `National Instrument PCI-6221` or `National Instrument PCIe-6321`, depending on your workstation) installed in the previous steps.

13) In the **Sample Time** field of the *Timing* section, set the desired sample time value in seconds, or leave "$-1$" to inherit the sample time from the configuration parameters of the Simulink model.

14) In the **Input channels** field of the *Input/Output* section, specify the list of ADC channels to use. Enter "`[1,2]`" in the field for using the first two ADC input channels, which are connected to the "Motor Current A" and "B" outputs of the SRV–02 unit via the "AI 0" and "AI 1" terminals of the BNC–2110 terminal board.

In the *Input range* drop–down list, select the appropriate voltage range for the ADC input channels. Select the "`-10 to 10 V`" option to comply with the armature voltage range ($\pm 6\,\mathrm{V}$).

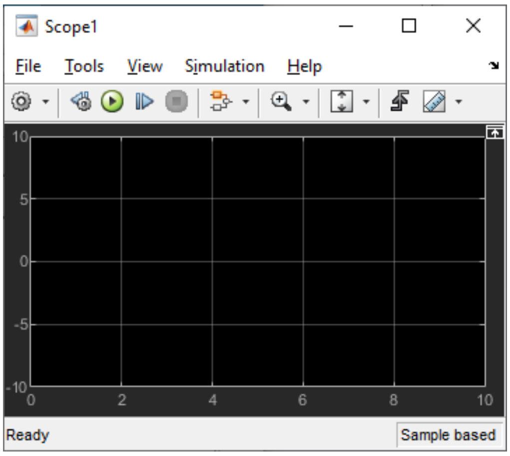
15) Click the **OK** button to confirm the settings and close the *Block Parameters: Analog Input* dialog window.

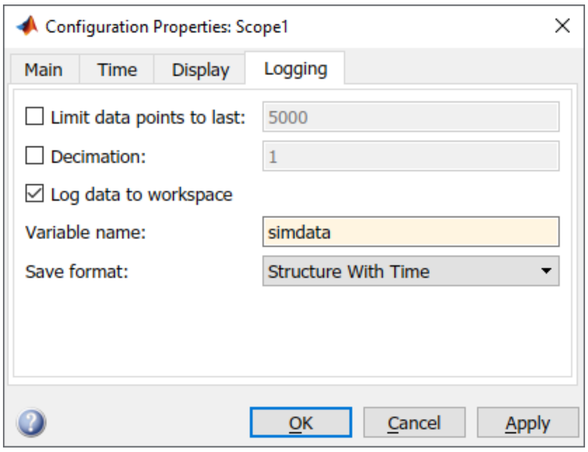
16) Save the Simulink model before proceeding (Note: it is a good practice to *regularly* save your Simulink model, to avoid losing your work in case of an accidental MATLAB crash).

## 3.3 Export simulation results to MATLAB workspace

To export the simulation results to the MATLAB workspace, use either a *To Workspace* block (located under *Simulink* → *Sinks* in the Simulink Browser window), or the *Logging* facility of the *Scope* block. Regarding the latter option, it can be enabled as follows:

1) Double-click on the *Scope* block, and click the **Configuration Properties** button on the Scope toolbar to open the *Configuration Properties* window.




2) Click the **History** tab. Uncheck the **Limit data points to last** checkbox.

Check the **Save data to workspace** checkbox. In the **Variable name** field, enter the name of the variable used for saving the Scope data in the MATLAB workspace. The variable is created in the workspace at the end of the simulation.

3) In the **Format** drop–down list, select the appropriate format for the variable to be generated in the workspace. The *Structure With Time* format is recommended, since it stores both time and data values into a single data structure. The time and data values can be retrieved by simply referencing the appropriate fields of the data structure: for example, if `simdata` is the name of the variable, use the following MATLAB code to access the time and data fields:

- `t = simres.time` to retrieve the time values
- `y = simres.signals(n).data(:,m)` to retrieve the $m^{\text{th}}$ signal in the $n^{\text{th}}$ set of axes of the scope. If a single signal is logged into a scope with a single set of axes, then the form `simres.signals.data` can be used as a shorthand.

## 3.4 Connecting the cables

Use the BNC–terminated cables to connect the SRV–02 unit with the BNC–2110 terminal board:

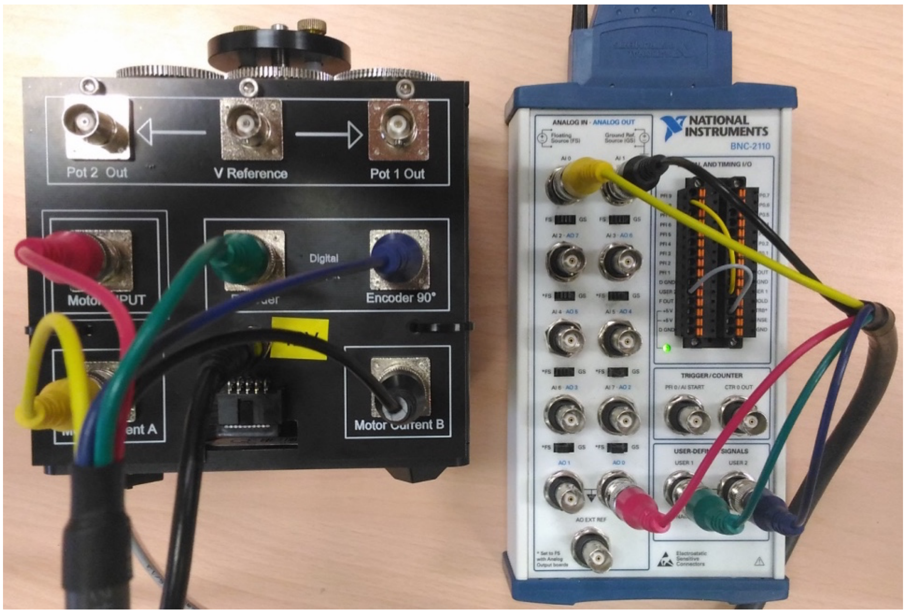
1) Connect the “AO 0” analog output channel of the BNC–2110 terminal board to the “Motor Input” of the SRV–02 unit.

2) Connect the “Motor Current A” and “Motor Current B” outputs of the SRV–02 unit to the “AI 0” and “AI 1” analog input channels of the BNC–2110 terminal board.

3) Connect the “Encoder” and “Encoder 90°” outputs of the SRV–02 unit to the “USER 1” and “USER 2” inputs of the BNC–2110 terminal board.

A direct connection between the two user–defined inputs “USER 1” and “USER 2” and the two (quadrature) inputs of counter 0 is established in the spring terminal block located in the *Digital and timing I/O* section of the BNC–2110 terminal board.

4) VERIFY THAT THE REGULATED OUTPUT VOLTAGE IN THE POWER SUPPLY ADAPTER IS SET TO $12\,\mathrm{V}$ BEFORE PROCEEDING. Connect the power supply to the SRV–02 unit by using the plug–in connector.

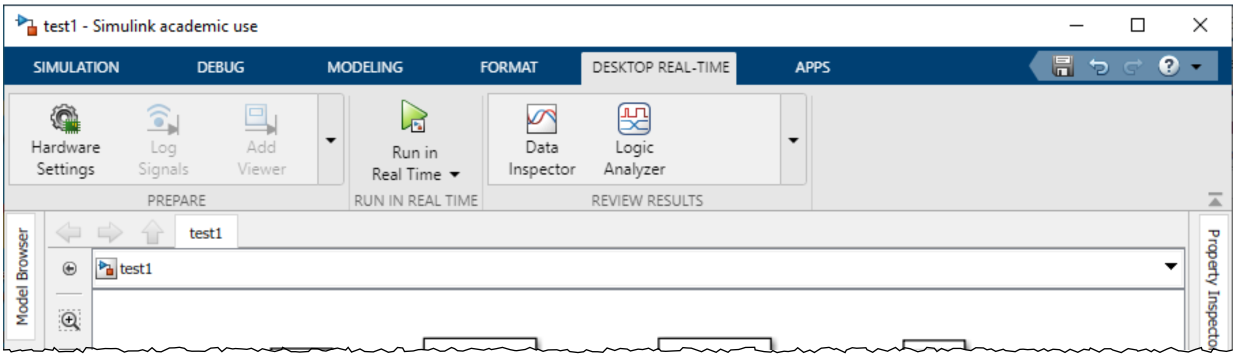



**IMPORTANT PRECAUTION**: when the SRV–02 is turned on, pay attention to not short–circuit the voltage driver terminals by accidental contacts with conductive materials (e.g. rings, metal clips, etc.).

## 3.5 Run the Simulink model in real–time

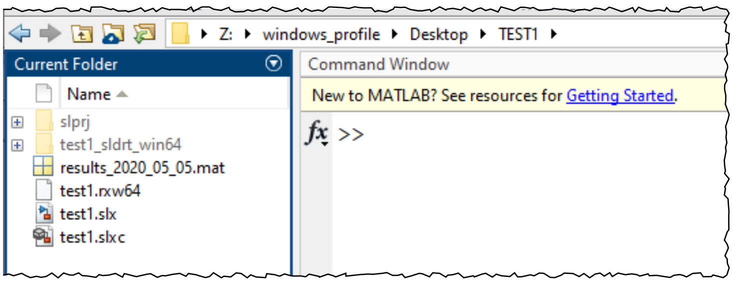
To run the Simulink model in real–time ("real–time simulation"), proceed as follows:

1) On the **Simulink Desktop Real-Time** tab of the Simulink model window, click the **Run in Real-Time** button.




The following sequence of operations is started:

- *Build*: the Simulink Coder is invoked to convert the Simulink model into a set of C-code source files. Then, a C compiler compiles and links the source files to create the real-time application. The C-code source files are created in the current working directory.



The Build operation can be individually invoked by clicking **Run in Real Time** → **Build** on the **Desktop Real-Time** tab of the Simulink model window.

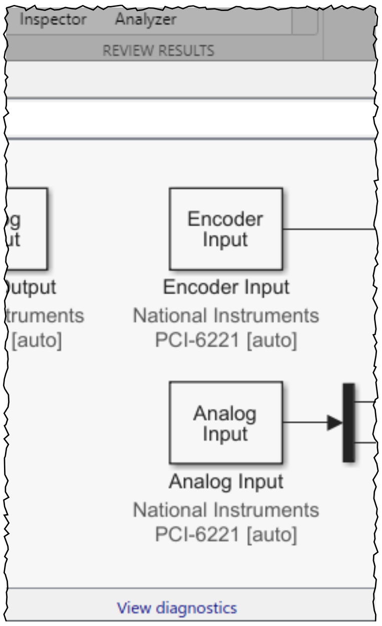
- *Connect*: the Simulink model is connected with the SLDRT kernel, to enable the exchange of commands, parameters and logged data.

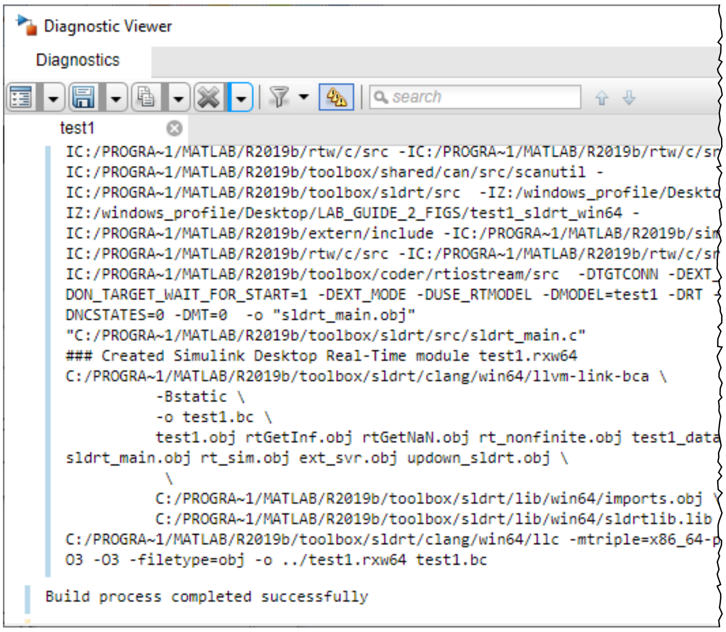
  The Connect operation can be individually invoked by clicking **Run in Real Time** → **Connect** on the **Desktop Real-Time** tab of the Simulink model window.

- *Start*: the real–time application is loaded into memory, and run by the SLDRT kernel.

  The Start operation can be individually invoked by clicking **Run in Real Time** → **Start** on the **Desktop Real-Time** tab of the Simulink model window.

Note: you can check the progress of the building process on the *Diagnostic Viewer* window, accessible by clicking the *View diagnostic* link at the bottom of the Simulink model window.




2) While the simulation is running, you can modify the DC motor armature voltage in real–time by changing the value of the Constant block connected to the Analog Output block.

Because of the presence of the static friction, the motor shaft does not spin for small values of the armature voltage. Slowly increase the value of the armature voltage until the motor shaft starts to rotate. Use the scope to monitor the encoder pulse count and the voltage drop across the shunt resistor.

3) To stop the simulation, push the **Stop** button in the Simulink model toolbar.