ECE 5670/6670 - Lab 1

dSPACE DS1104 Control Workstation & Simulink Tutorial

Objectives

The objective of the lab is to provide an introduction to the dSPACE Control-Desk and MATLAB/Simulink software, and to explain their use with the dual power amplifier and the DC motor.

1. Introduction

In this lab, you will need

- Dual power amplifier
- DC motor
- Encoder cable

In general, there are five major parts in the setup that will be used in the lab. These five parts are as follows:

1. Software (MATLAB – Simulink and dSPACE). The dSPACE Control-Desk is operated based on parameters set in MATLAB and Simulink (*.sdf file).
2. DS 1104 R&D Controller Card from dSPACE, this card is installed on the motherboard of the computers in the lab and connects to the dSPACE breakout box via a master I/O ribbon cable.
3. The Dual Power Amplifier. This amplifier is supplied with 120V AC, and is capable of producing up to 25V DC at 4 Amps from each output. The amplifier takes a signal from the BNC on the front (which we will connect to the dSPACE breakout box) and amplifies it with a gain of 5 to the banana plugs on the back.
4. The electric machine setup (*i.e.,* electric motors).
5. Cables used to interconnect (*i.e.,* encoder cable, banana cables, and BNC cables).

The physical appearance of these parts and equipment are shown in Figs. 1-3.

For the purposes of this tutorial, the steps needed to create a Simulink model and layout file in dSPACE are outlined. In these examples, a Simulink model (.mdl or .slx) of a sinusoidal signal and a DC voltage output will be designed and built to run a DC motor. Once the Simulink simulation results are verified for the first model, the model will be modified to output the signal to an oscilloscope using a digital-to-analog output channel. Then, a control panel layout using the dSPACE Control-Desk will be designed (*.lax), serving as a user-interface. Display of the signals on the user-interface will occur via an analog-to-digital input channel. The frequency and amplitude of the output signal can be changed in real-time.

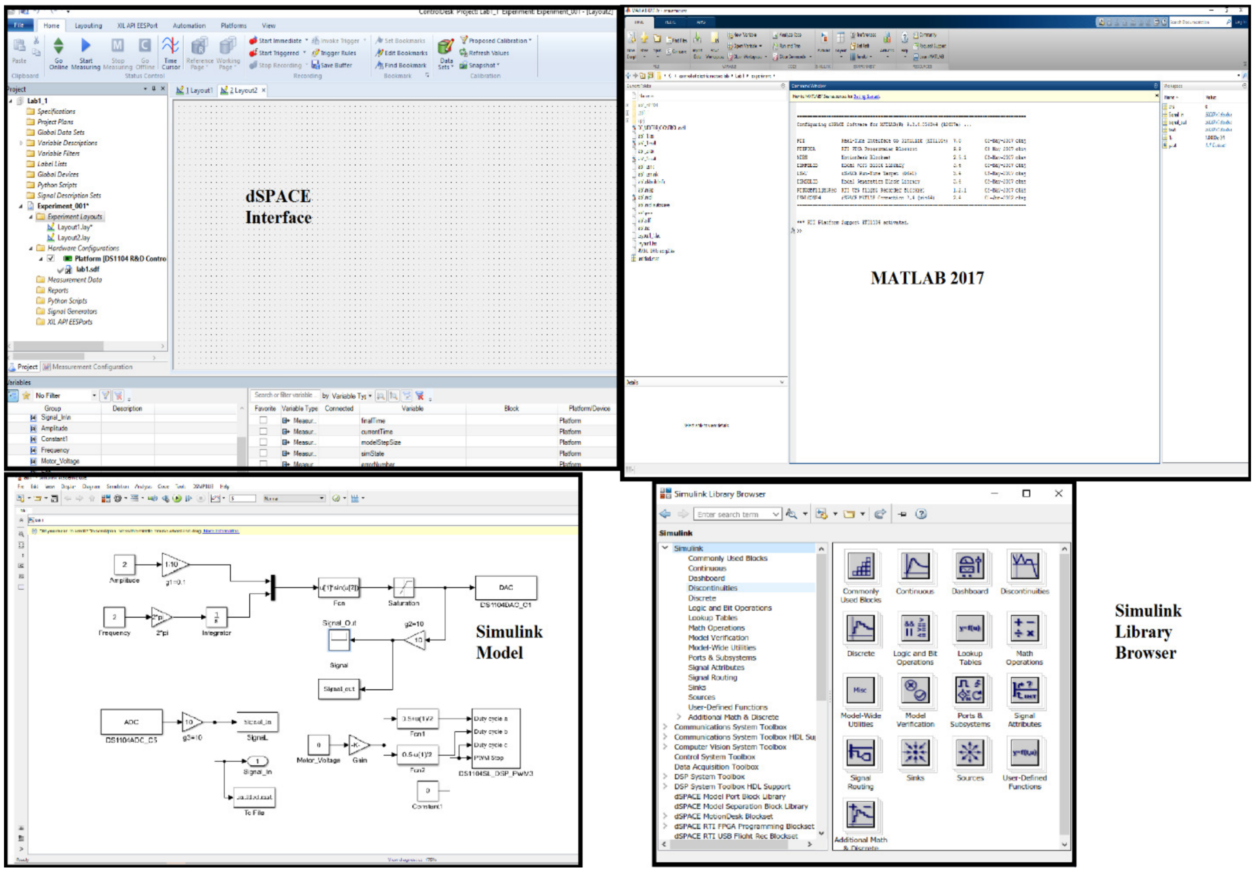


Figure 1: Control-Desk, MATLAB and Simulink windows

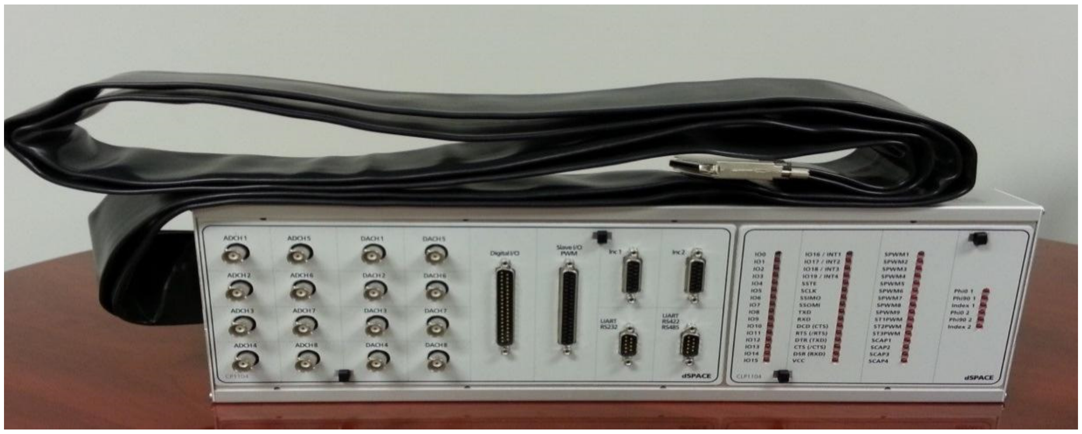
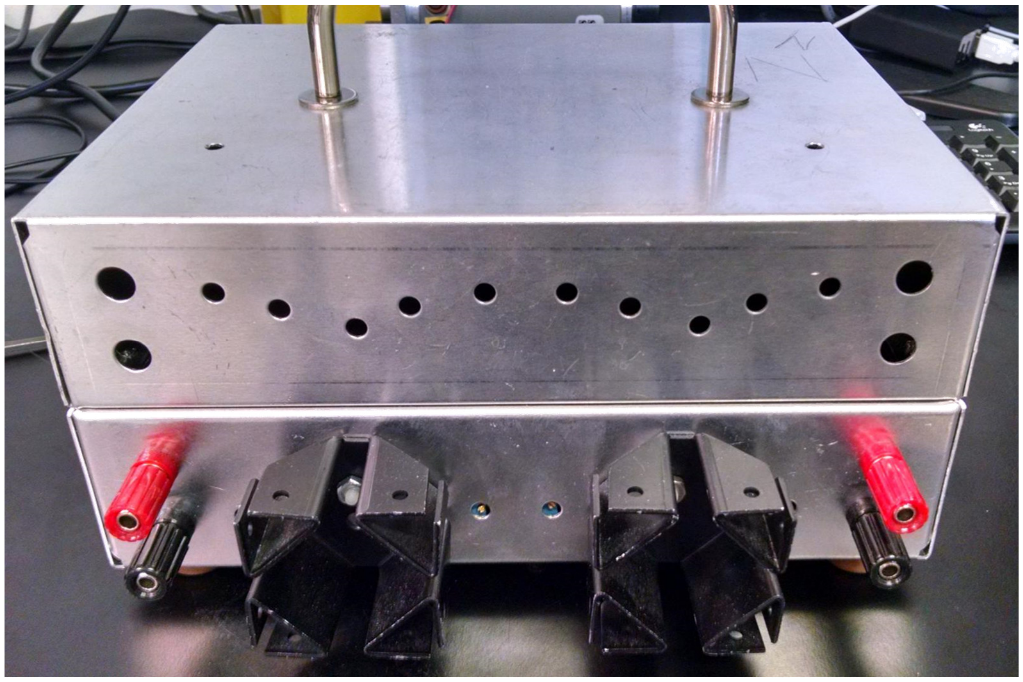
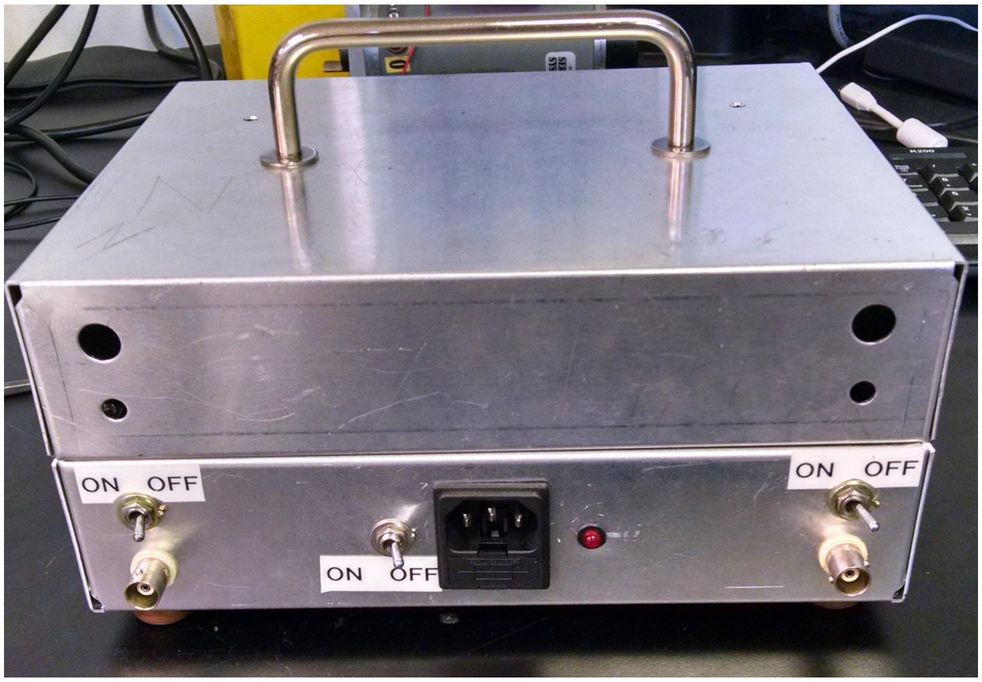


Figure 2: dSPACE breakout box with attached master I/O ribbon



Back side, with banana plug OUTPUTS



Rear side, with BNC INPUTS

Figure 3: Dual Power Amp

In the second example, an open-loop voltage controller will be designed to control the speed of a DC motor. This model allows the users to change speed of the DC motor by varying the applied voltage in the user-interface. Also, this example will introduce techniques to measure the motor speed, and the motor position through analog-to-digital inputs and the Incremental Encoder Interface (INC).

Common Problems

- DO NOT SAVE to the server directory (X: drive). *This includes the desktop.* Save to the local C drive or a USB drive during lab implementation.
- In SIMULINK, Click on *Simulation* >> *Configuration Parameters*
  In the left hand menu select *optimization* and make sure that the *block reduction* box is unchecked, then expand the optimization menu by clicking on the plus sign next to it. Select *signals and parameters* and make sure that the *signal storage reuse* block is unchecked. Next, in the left hand menu select *code generation* and change the system target file to rti1104.tlc.

2. Digital-to-Analog Converter

In order to illustrate the digital-to-analog channel functionality, a Simulink model will be designed to generate a sinusoidal signal. This signal will then be observed on a digital scope through a digital to analog channel (DACH).

i. Sine wave Simulink model

Begin by generating a sinusoidal signal using Simulink. (Make sure to select DS1104 when prompted upon opening MATLAB.) Open a new model page in Simulink by clicking on model **File > New > Model** in MATLAB, or by clicking on the Simulink icon and then ***Blank Model***. Now that the Simulink page is open, the Simulink library browser can be accessed by clicking on the Simulink icon or through **View > Library Browser**. This library contains all of the necessary blocks to design Simulink models, which can be added to the model by copy and paste or drag and drop. Save the model in a folder that you can have access to later. The model will be used when building the user-interface in the dSPACE Control-Desk. Place the Simulink blocks as follows, using Fig. 4 as a reference.

a. Drag and drop two ***constant*** blocks from the Simulink library to the model from **Commonly Used Blocks > Constant.** Change the name of these two constants to ***Amplitude*** and ***Frequency***. The value of these constants can be changed by double-clicking on the blocks. Change the values of the constants to zero.
>>**Double-click > Block Parameters: Constant > Main > Constant value: > 0.**

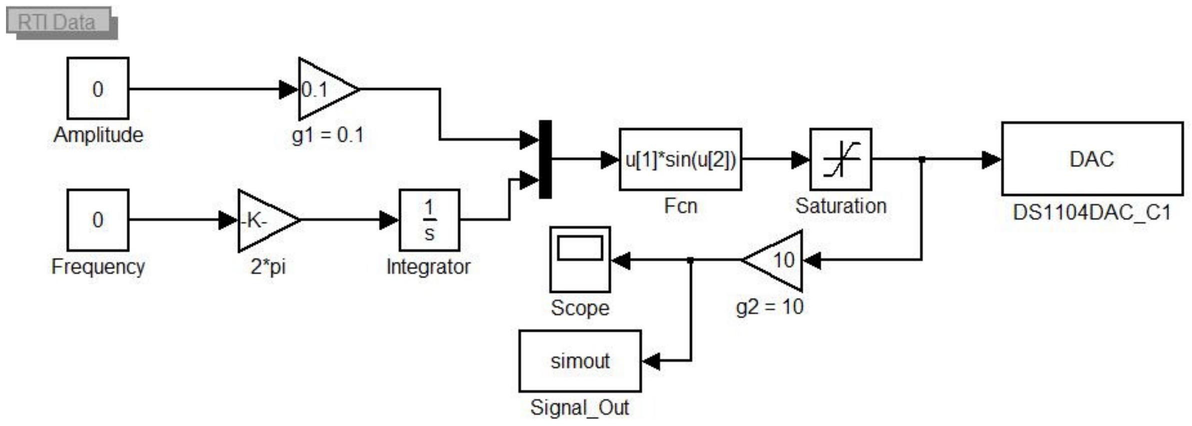


Figure 4: Simulink model for sine wave – blocks connection

b. Drag and drop two ***gain*** blocks from **Simulink Library Browser > Simulink > Commonly Used Blocks > Gain** to the model. The value of these gain blocks can be changed similarly to the constant blocks. Change the value of one of them to (1/10) and the other one to (2*pi).
>> **Double-click > Block Parameters: Gain> Main > Gain: > (1/10) / (2*pi).**
The gain of 0.1 is inserted because a value of 1 in the D/A converter corresponds to a 10V output (the maximum possible). The gain of 2π is used to convert from Hz to rad/s.

c. Add an ***Integrator*** and a ***Mux*** block from **Simulink Library Browser > Simulink > Commonly Used Blocks > Integrator.** Also, add a ***function*** block from **Simulink Library Browser > Simulink > User-Defined Functions > Fcn** to the model**.** Double click on the Fcn and change the expression to **u[1]*sin(u[2]).**
>> Double-click > Block Parameters: Fcn > Expression: > u[1]*sin(u[2]).
This will create a sinusoidal wave form with amplitude of u[1] and angle of u[2] in radians.

d. Add a ***saturation*** block from **Simulink Library Browser > Simulink > Commonly Used Blocks > Saturation.** This block will limit the output signal amplitude. The upper and lower limits can be changed by double-clicking on the saturation block. Set the limits to -1 and 1, which correspond to the scaled minimum/maximum values of the A/D and D/A ports of ±10V.
>>Double-click > Block Parameters: Saturation > Main > Upper limit: > 1 and **Lower limit: > −1.**

e. Drag and drop a ***Digital-to-Analog Channel (DAC)*** block from **Simulink Library Browser > dSPACE RTI1104 > DS1104 MASTER PPC > DS1104DAC_C1.** This block will allow you to view the output signal from CP1104 I/O board on a digital scope in dSPACE. The number at the end of this block name refers to the number of the digital-to-analog channel on the CP1104 I/O board. You can change the channel number by double-clicking on the block.
>>Double click > DS1104DAC_C1 > Unit: > Channel number > 1.
Note: In this tutorial, we will use DACH 1 on the breakout box.

f. Drag and drop a ***To Workspace*** block from **Simulink Library Browser > Simulink > Sinks >To Workspace**. Be sure to change the saved data format to ***Array*** using:
>>Double-click > To Workspace > Save format > Array.

g. Add another ***gain*** block as described in *b*. Change the value of this gain to (**10**). This gain emulates the gain of the D/A converter.

h. Connect all of the blocks and change the names as shown in Fig. 4. In order to connect two blocks, hover the mouse over the small arrowhead on the side of the block. When the mouse becomes a cross, click and drag over to the block where the connection is desired. A solid line should form with a filled black arrowhead at the connected end as shown in Fig. 4. Also note that if there is a red dashed line, no connection has been made. Connections to the middle of a line can be done in the same way as between blocks.
This setup will output a sinusoidal signal from DACH 1 on the I/O board. Note that if you wish to rotate blocks for the sake of a clearer diagram, you can **Right-Click Block > Format > Rotate Block > Clockwise** or simply select the block and type **Ctrl+R.**

ii. Building the Simulink model

In order to build the Simulink model, first define the sampling period. In the Simulink toolbar:
>> Toolbar > File > Model Properties > Model Properties > Callbacks > InitFcn*
In the Model Initialization Function window, type: **Ts=1e-4;**
This will set the sampling period to 100μs or the sampling frequency to 10kHz. Note that other variables that you use in the Simulink diagram can be initialized in this manner. This can be helpful if several constants have the same value.

a. Change the simulation time to infinity from the Configuration Parameters in the Simulink toolbar:
>> **Toolbar > Simulation > Model Configuration Parameters > Solver > Simulation time > Start time / Stop time > 0.0 / inf**
Also, change the fixed-step size and solver, under solver option, to 0.0001 [s] and ode1(Euler) respectively in the same window.
>> Type:> Fixed-step / ode1(Euler) > and **Solver: > 0.0001**

b. To verify the functionality of the model before building it, the sine wave can be observed in the simulation with the use of a scope block. Drag and drop a ***Scope*** block from **Simulink Library Browser > Simulink > Commonly Used Blocks > Scope.** Connect the scope to the gain block g2=10, as shown in Fig. 4. Change the value of frequency and amplitude constant blocks to 2, as described in ***II.i.a.*** Click on the play button in the Simulink toolbar above the model. The signal can be seen by double-clicking on the scope block. Fig. 5 shows the sine wave in Simulink. In order to show this plot, change the stop time from **inf** to **5** then click play, the signal should appear when finished. Use the ***autoscale*** (binoculars) icon in the scope window to view as shown.
Note: If only a small portion of the waveform is showing on the scope, the sample limit needs to be turned off. In the scope window select **View > Configuration Parameters > Logging >** uncheck **Limit Data Points to Last**. Run the simulation again by pressing the play button.

Save a copy of the simulated plot for your lab report. This is accomplished by performing a print screen, or by clicking the printer in the scope window and printing to a file (*e.g.*, PDF type).

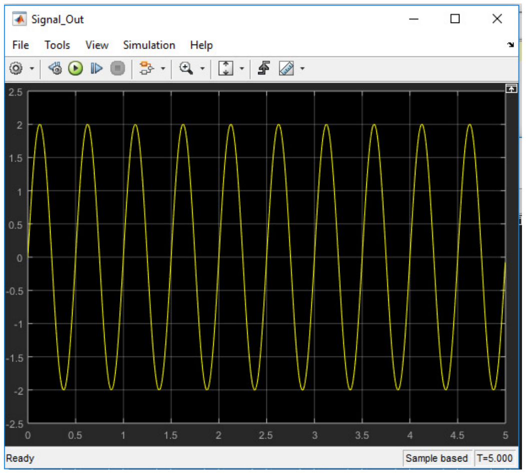


Figure 5: Sine wave output from the Simulink model seen on the scope

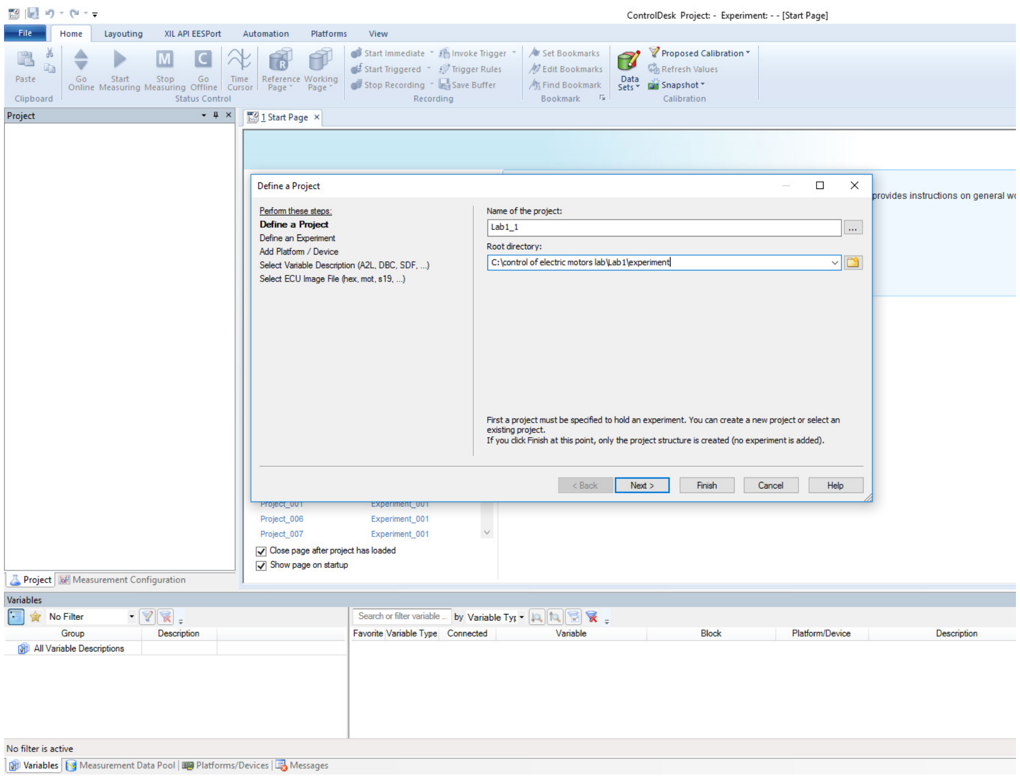
c. Before building the model, **make sure** to change the values of the *Frequency* and *Amplitude* blocks back to **zero** and change the stop time back to **Inf**. Also, make sure that the path to the current directory in MATLAB is the one that the Simulink model is saved in.
d. In the Simulink main page, hold the **Ctrl key** and press **B** on the keyboard. This will build a .sdf file that has all of the variables used in this model. The .sdf file will be used in dSPACE Control-Desk to map the inputs and outputs.

3. dSPACE Control-Desk

Open the dSPACE program by double-clicking on **dSPACE ControlDesk 6.1** icon in Desktop or from the Start menu.

iii. Load the .sdf file

Go to **File > New > Project + Experiment >**. You should see the window of Fig. 6. Give the project a name. In the **Root directory:** copy the location of the **.sdf** file you build earlier. Press **Next**. Then, give a name for the Experiment. Go to the next page by pressing **Next**. In this page (**Add Platform / Device**) make sure you select **DS1104 R&D Controller Board** in **Supported Platform/Device Types** and choose **ds1104** in **Available Platforms/Devices** option. Your window should look like Fig. 7.


ControlDesk Project: - Experiment: - - [Start Page]

Define a Project

Perform these steps:
Define a Project
Define an Experiment
Add Platform / Device
Select Variable Description (A2L, DBC, SDF, ...)
Select ECU Image File (hex, mot, s19, ...)

Name of the project:
Lab1_1

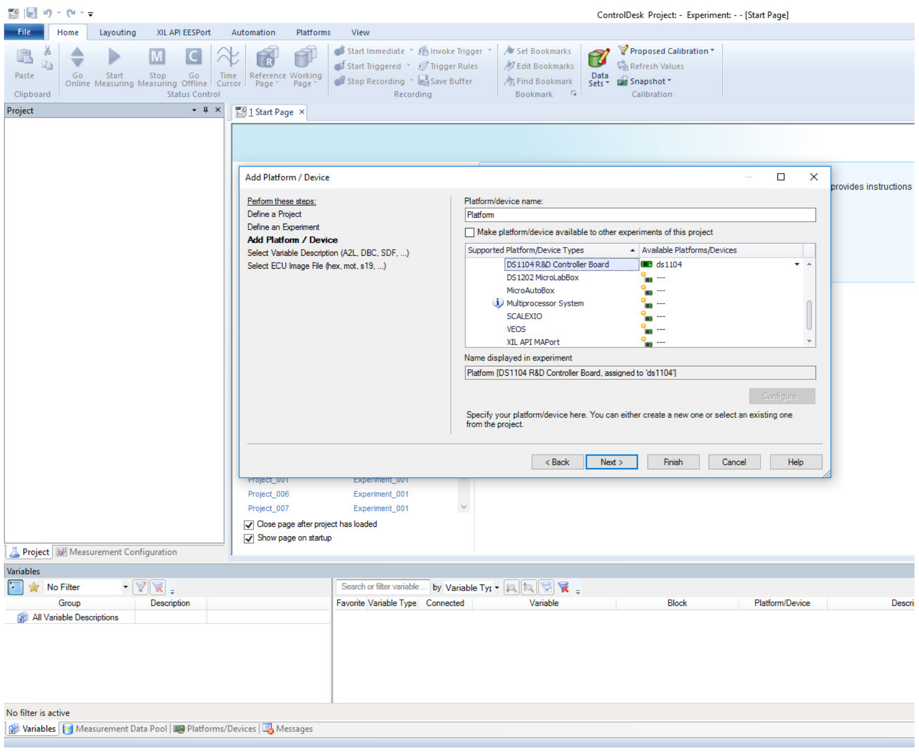
Root directory:
C:\control of electric motors lab\Lab1\experiment

First a project must be specified to hold an experiment. You can create a new project or select an existing project.
If you click Finish at this point, only the project structure is created (no experiment is added).

Project_001	Experiment_001
Project_006	Experiment_001
Project_007	Experiment_001

[✓] Close page after project has loaded
[✓] Show page on startup

Figure 6: Starting a dSPACE project


ControlDesk Project: - Experiment: - - [Start Page]

Add Platform / Device

Perform these steps:
Define a Project
Define an Experiment
Add Platform / Device
Select Variable Description (A2L, DBC, SDF, ...)
Select ECU Image File (hex, mot, s19, ...)

Platform/device name:
Platform

[ ] Make platform/device available to other experiments of this project

Supported Platform/Device Types	Available Platforms/Devices
DS1104 R&D Controller Board	ds1104
DS1202 MicroLabBox	---
MicroAutoBox	---
Multiprocessor System	---
SCALEXIO	---
VEOS	---
XIL API MAPort	---

Name displayed in experiment
Platform [DS1104 R&D Controller Board, assigned to 'ds1104']

Specify your platform/device here. You can either create a new one or select an existing one from the project.

Project_001	Experiment_001
Project_006	Experiment_001
Project_007	Experiment_001

[✓] Close page after project has loaded
[✓] Show page on startup

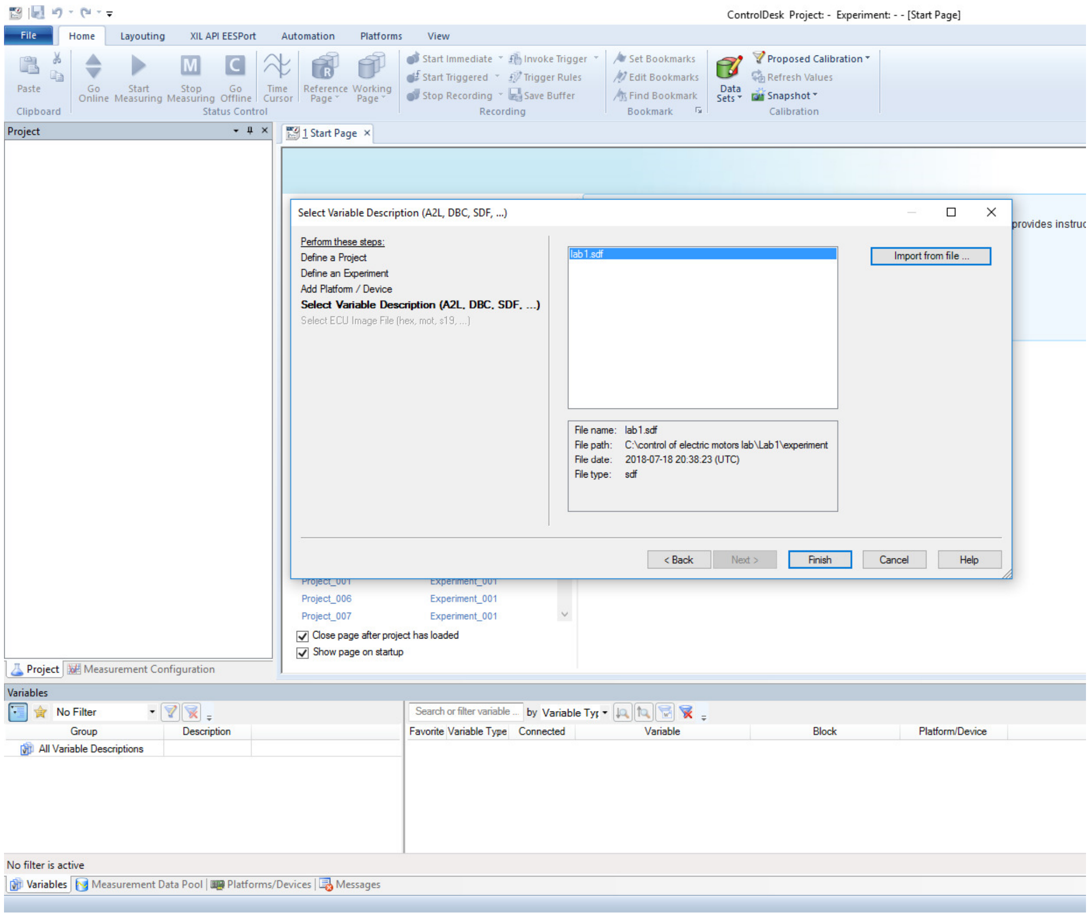
Figure 7: Setting up a dSPACE platform

In the next page (go after pressing **Next**), select **Import from File**. Browse and select your **.sdf** file. Fig. 8 should appear. Click **Finish**. This will load all the variables built in the Simulink model into dSPACE, where they can be accessed through the window at the bottom of the dSPACE program.

Note that only one .sdf file can be open in dSPACE at a time and that by opening or reopening the file you can update variables to their latest values as saved in Simulink without restarting dSPACE.

iv. Create a new layout

a. A new layout window should appear as shown on Fig. 9. Otherwise, click **Insert Layout** to open a new layout, as shown in Fig. 10. If the Instrument Selector does not show automatically, it can be accessed through **View > Controlbars > Instrument Selector**. The instruments in the instrument selector window can be added to the layout window by clicking on an instrument name and drawing it in the layout window. Click anywhere in the layout window and drag the mouse until the instrument is the desired size.



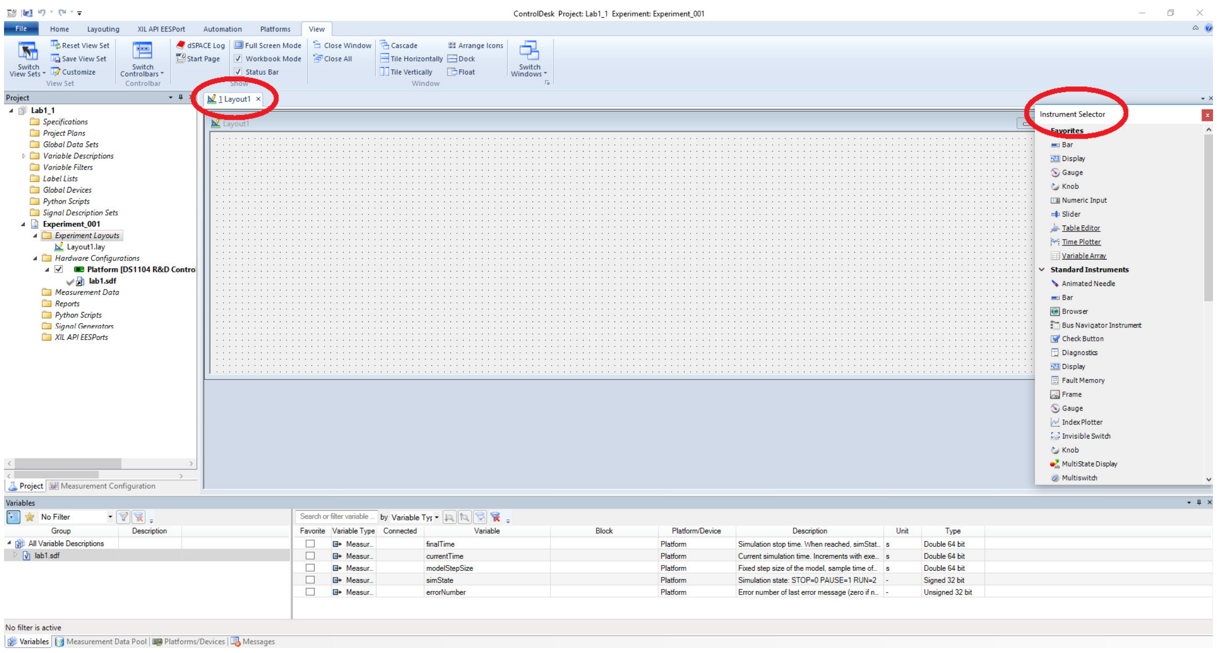
Select Variable Description (A2L, DBC, SDF, ...)

Perform these steps:
Define a Project
Define an Experiment
Add Platform / Device
Select Variable Description (A2L, DBC, SDF, ...)
Select ECU Image File (hex, mot, s19, ...)

lab1.sdf

File name: lab1.sdf
File path: C:\control of electric motors lab\Lab1\experiment
File date: 2018-07-18 20:38:23 (UTC)
File type: sdf

Figure 8: Importing .sdf file.



Favorite	Variable Type	Connected	Variable	Block	Platform/Device	Description	Unit	Type
☐	Measur...		finalTime		Platform	Simulation stop time. When reached, simStat...	s	Double 64 bit
☐	Measur...		currentTime		Platform	Current simulation time. Increments with exe...	s	Double 64 bit
☐	Measur...		modelStepSize		Platform	Fixed step size of the model, sample time of...	s	Double 64 bit
☐	Measur...		simState		Platform	Simulation state: STOP=0 PAUSE=1 RUN=2	-	Signed 32 bit
☐	Measur...		errorNumber		Platform	Error number of last error message (zero if n...	-	Unsigned 32 bit

Figure 9: New layout and instrument selector

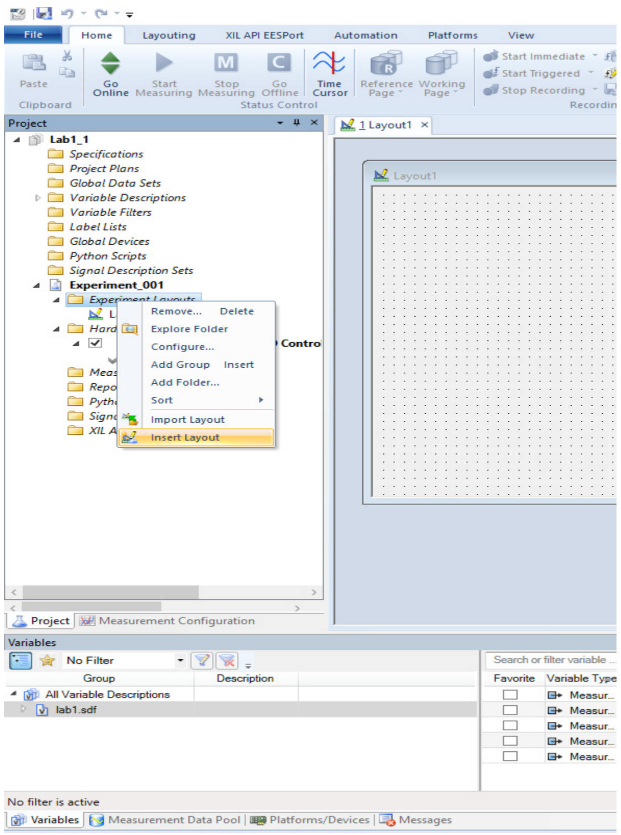


Figure 10: Opening new layout

b. Click on ***Numeric Input*** under **Standard Instruments** section and draw two numerical inputs. Then, click on ***Display*** under **Standard Instruments** and draw two display windows in the layout window. Note that you must click the desired icon in the Instrument Selector before drawing each box. This will allow you to input a number (in this case, frequency and amplitude) and view the entered values in the display windows. The blocks may not look exactly like the ones of Fig. 11 and may include additional text such as “/{%VARIABLE%}”.

c. To see the plot of the sine wave on the Control-Desk, click on ***Time Plotter*** under the **Standard Instruments** in the Instrument Selector section and drag it to your layout.

v. ***Map variables***

To map the variables from the Simulink model to the instruments in the user-interface layout:

a. Click the plus sign next to **Model Root** in the bottom left corner of the dSPACE program, where you will see all the variables and gain names from Simulink model.

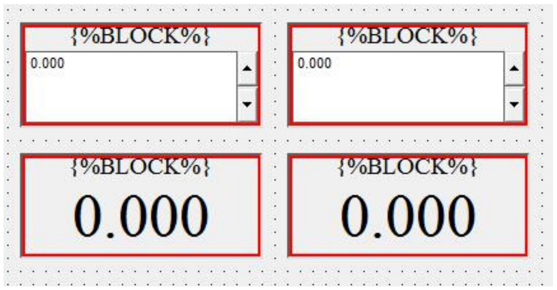
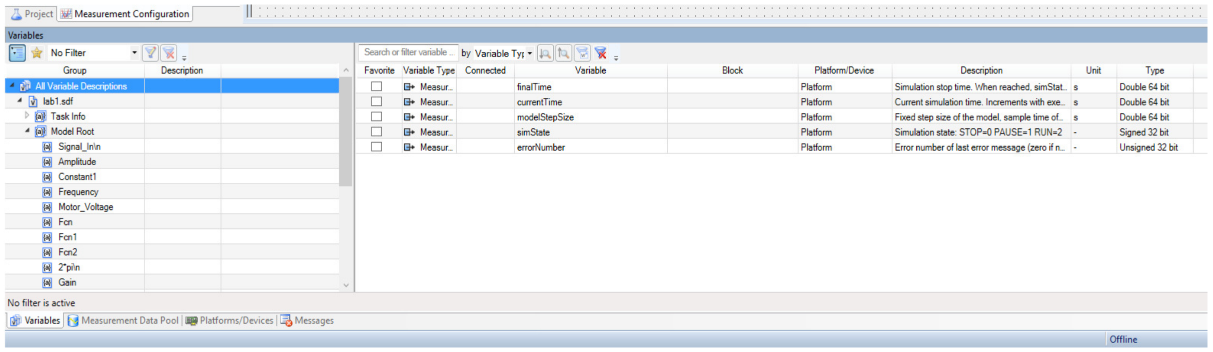


Figure 11: Numerical input and Display windows

Note: If the model root directory is not showing, click **View > Switch Controlbars > Variables.** Then, click on the tab that includes the variable file directory, shown in Fig. 12. It may happen that some signal variables may not be available. In that case, right-click on the signal in the Simulink diagram, choose **Signal Properties**, then check **Test point**. If you don’t see the signal name, insert a gain block with a gain equal to 1 along the signal path or use an output port instead of a Workspace block.



Group	Description
All Variable Descriptions	
lab1.sdf	
Task Info	
Model Root	
Signal_In\n	
Amplitude	
Constant1	
Frequency	
Motor_Voltage	
Fcn	
Fcn1	
Fcn2	
2*pi\n	
Gain	

Favorite	Variable Type	Connected	Variable	Block	Platform/Device	Description	Unit	Type
☐	Measur...		finalTime		Platform	Simulation stop time. When reached, simStat...	s	Double 64 bit
☐	Measur...		currentTime		Platform	Current simulation time. Increments with exe...	s	Double 64 bit
☐	Measur...		modelStepSize		Platform	Fixed step size of the model, sample time of...	s	Double 64 bit
☐	Measur...		simState		Platform	Simulation state: STOP=0 PAUSE=1 RUN=2	-	Signed 32 bit
☐	Measur...		errorNumber		Platform	Error number of last error message (zero if n...	-	Unsigned 32 bit

Figure 12: Windows at the bottom of the dSPACE Control-Desk

b. Click on ***Amplitude***. You will see the parameter name ***value*** appears in the window next to the main window. Click on the parameter ***value*** and drag and drop it into one of the numerical input and display windows. Follow the same steps for ***Frequency***. The color of the numerical input and display windows will change from red to gray as shown in Fig. 13.

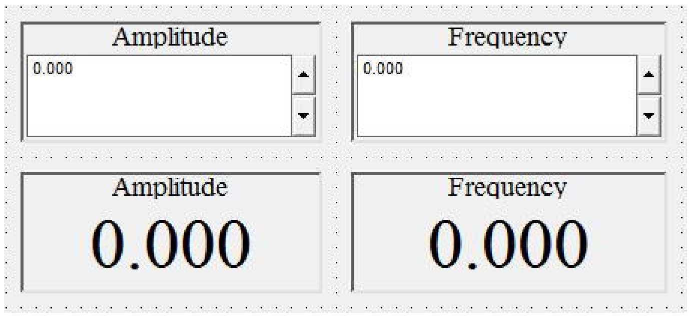
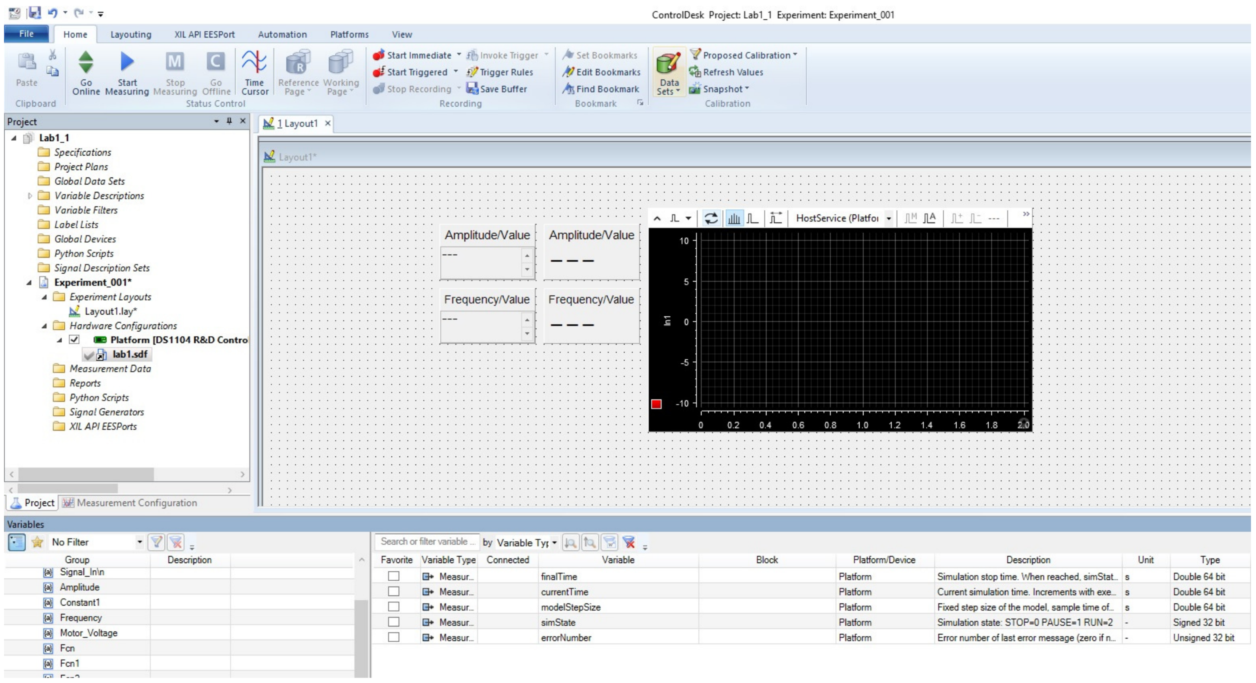


Figure 13: Numerical Input and Display after mapping the variables Amplitude and Frequency



Group	Description
Signal_In\n	
Amplitude	
Constant1	
Frequency	
Motor_Voltage	
Fcn	
Fcn1	

Favorite	Variable Type	Connected	Variable	Block	Platform/Device	Description	Unit	Type
☐	Measur...		finalTime		Platform	Simulation stop time. When reached, simStat...	s	Double 64 bit
☐	Measur...		currentTime		Platform	Current simulation time. Increments with exe...	s	Double 64 bit
☐	Measur...		modelStepSize		Platform	Fixed step size of the model, sample time of...	s	Double 64 bit
☐	Measur...		simState		Platform	Simulation state: STOP=0 PAUSE=1 RUN=2	-	Signed 32 bit
☐	Measur...		errorNumber		Platform	Error number of last error message (zero if n...	-	Unsigned 32 bit

Figure 14: Overall view of the layout window

c. Click on the name of the ***To Workspace*** block, named **Signal Out** in this tutorial, and drag and drop the parameter **In1** to the plotter in the layout window. This will display the signal being output from DACH1, a sine wave. The layout window should look like Fig. 14. Note that the plotter window may not look exactly the same.

vi. Run the Control-Desk

To run the Control-Desk, click on the **Go Online** and **Start Measuring** button in the **Home** tab. The button will switch from green to grey. This will run the code of the Simulink block diagram.

Change the ***Amplitude*** and ***Frequency*** to different values. You should see the sine wave on the plotter array, as shown on Fig. 15. To change the scales of the ***Time Plotter***, select the plotter and then right click and choose **Axes/ Signals Properties** as shown in Fig. 16.

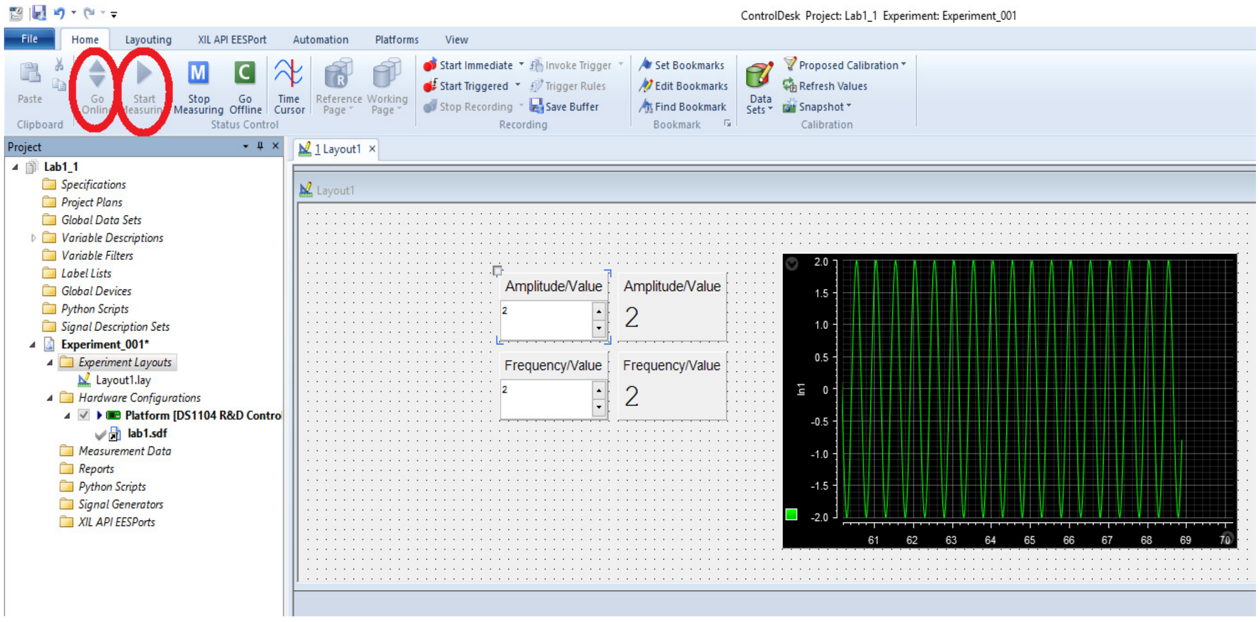


Figure 15: Control-Desk while running

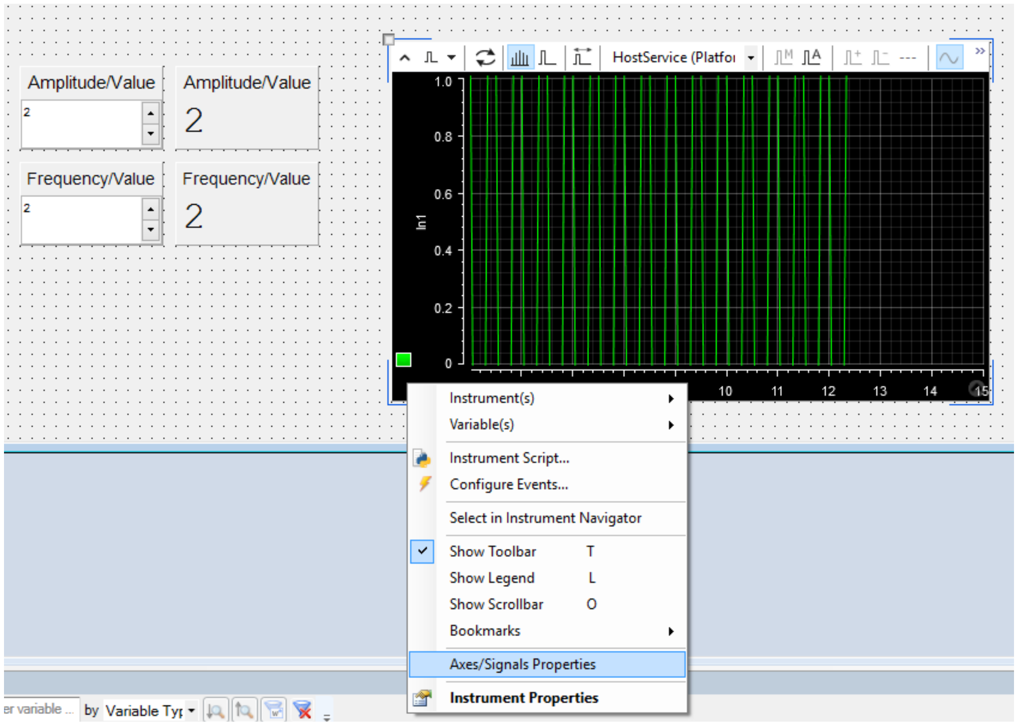


Figure 16: Rescale Signals in the Plotter

Now that the code of the Simulink diagram is built and functioning properly, the signal can also be seen on an oscilloscope via a BNC cable. Connect DACH1 to an oscilloscope; the signal showing on the oscilloscope should look like the one shown in the plotter array in the user-interface (see Fig. 15). Capture a screenshot of the sine wave on the oscilloscope for the report. One way to do this is to use the Open Choice Desktop software. Open the software and click on the “Select Instrument” button in the top left hand corner. Select the instrument with the name that starts with “USB::..” and hit OK. Next, click the “Get Screen” button on the left hand side to capture the oscilloscope screen.

After finishing the experiment, **set all the inputs to zero again** and then put the Control-Desk into offline mode by hitting **Go Offline**. This is an important step as when you go offline, it just makes the software to go offline. However, the process keeps running. So, the last entered value keeps on going. Building a new process and overwriting the ongoing process can be hazardous. Stopping the program before rebuilding a new sdf file, from a modified Simulink file is necessary.

4. Analog-to-Digital Converter

An Analog-to-Digital Channel (ADCH) will now be used to read the output sine wave from DACH1 back to the user-interface. Reopen the Simulink model previously built.

a. In Simulink, drag and drop an *analog-to-digital* channel block from >>**Simulink Library Browser > dSPACE RTI1104 > DS1104 MASTER PPC > DS1104ADC_C5.** The number of this channel can be changed similarly to the digital-to analog-channel. There are a total of 8 ADCH channels on the I/O board, of which the last four can be accessed through this block by changing the port number. The first four channels are multiplexed and can be accessed through >>**Simulink Library Browser > dSPACE RTI1104 > DS1104 MASTER PPC > DS1104MUX_ADC.** In this tutorial, we will use A/D channel number 5.

b. A voltage of 10V on the A/D channel is scaled to the signal with value equal to 1. Therefore, add another ***gain*** and ***To Workspace*** (same as for ***Signal_Out*** in section ***II.i.f***) blocks to the model and change the gain value to (**10**) and ***To Workspace*** to ***Signal_In***. This will convert the normalized value of the output to its actual value. Connect the blocks as shown in Fig. 17.

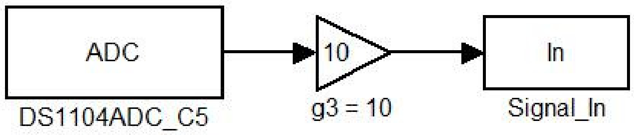


Figure 17: Digital to Analog Channel (ADCH) block

An overall view of the Simulink model for this example can be found in the appendix, Figure 33. Take a screenshot of your complete Simulink model for the report. One way to do so is to type:

```
saveas(get_param('file','Handle'),'figure.tif')
```

in the MATLAB command window, where ***file*** is the name of the mdl file (or Simulink window in general), and ***figure.tif*** is the name of the file that will be saved (in the MATLAB directory). Other formats are also possible (see MATLAB documentation).

vii. Read the signal back to the Control-Desk

After adding the blocks of Fig. 17 to the Simulink model, build the model again as described in section ***ii***. This will update the .sdf file and add the new blocks' names. Open the .sdf file into dSPACE, as done in section ***iii*** and it will replace the existing variable file. Draw another plotter array window and drag and drop the variable name of the ***To Workspace*** block, called ***Signal_In*** in this tutorial, to the plotter array. Use a BNC cable to connect the DACH1 to ADCH5. If you would like to see the signal simultaneously on the user-interface and oscilloscope, use a BNC splitter. Run the program by pressing the start button. The output signal to DACH1 and input signal to ADCH5 are shown in Fig. 18.

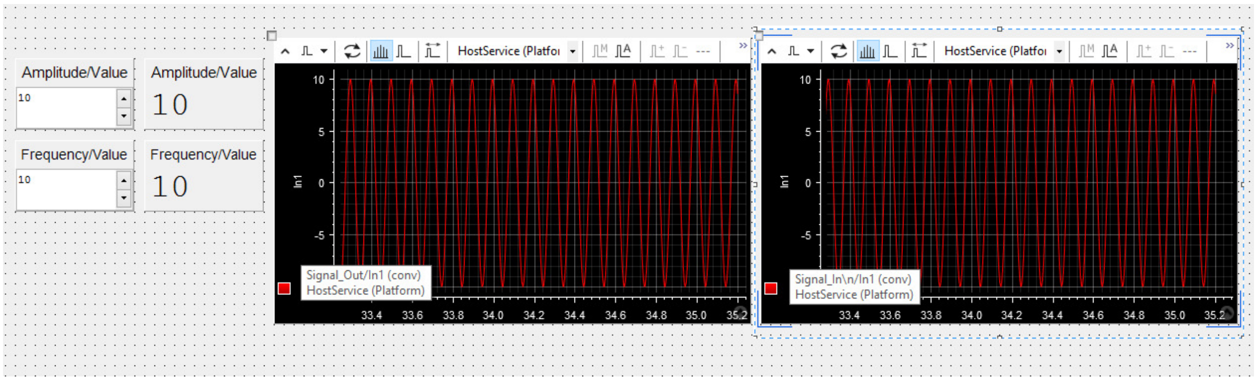


Figure 18: Layout file showing sine wave applied to DACH1 and read back through ADCH5

Also, the signals can be plotted in one plotter array, as shown in Fig. 19, by dragging and dropping both signals into one plotter. Observe that the sine wave measured from ADCH5 is lagging the original sine wave by one sampling period, as expected.

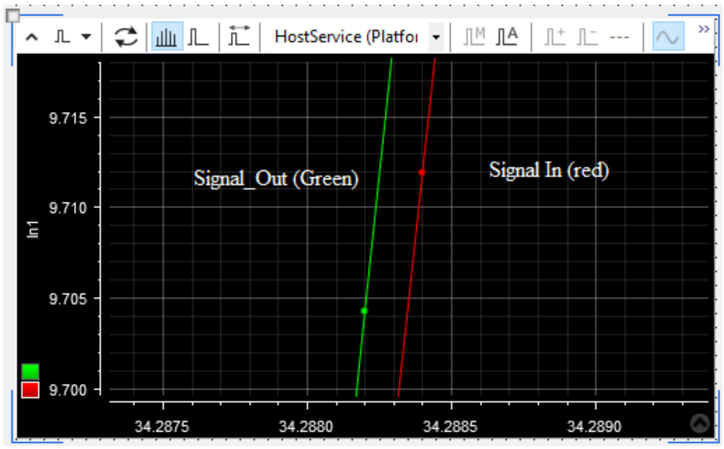
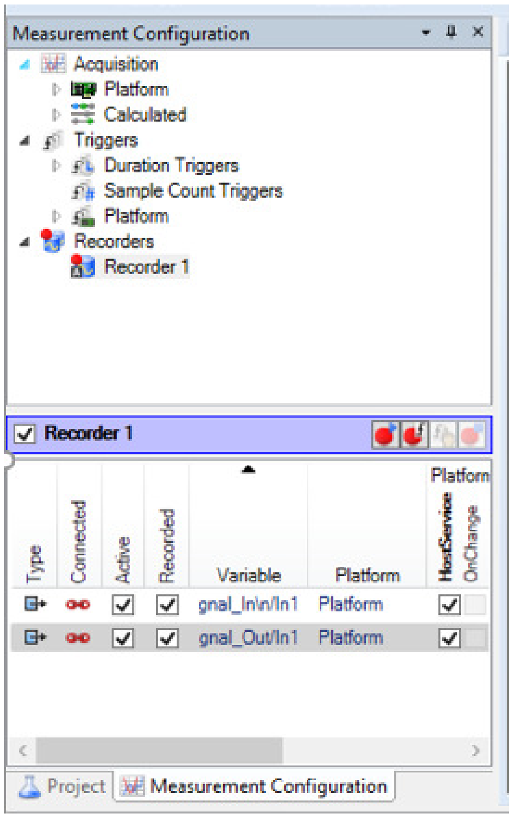


Figure 19: Capture of the plotter array showing both signals together

viii. Save the data

a. Settings: The captured data on the plotter array can be saved as a Matlab data file (.mat). There are different ways to do so. The simplest way is through the *Measurement Configuration*. This display can be added to the dSPACE layout from >>**View > Switch Controlbars > Measurement Configuration**, as shown in Fig. 20. Data is saved through the use of a ***Recorder***. In the ***Measurement Configuration*** tab on the left hand side of the Control-Desk window, as shown in Fig. 20, click on the recorder name, here **Recorder 1** to select the recorder. Variables mapped to any plotters present in the layout will automatically be added to the list of variables for which data will be captured. If there are additional variables that need to be captured, drag and drop the variables from the Variables window to the ***Recorder*** variable list.



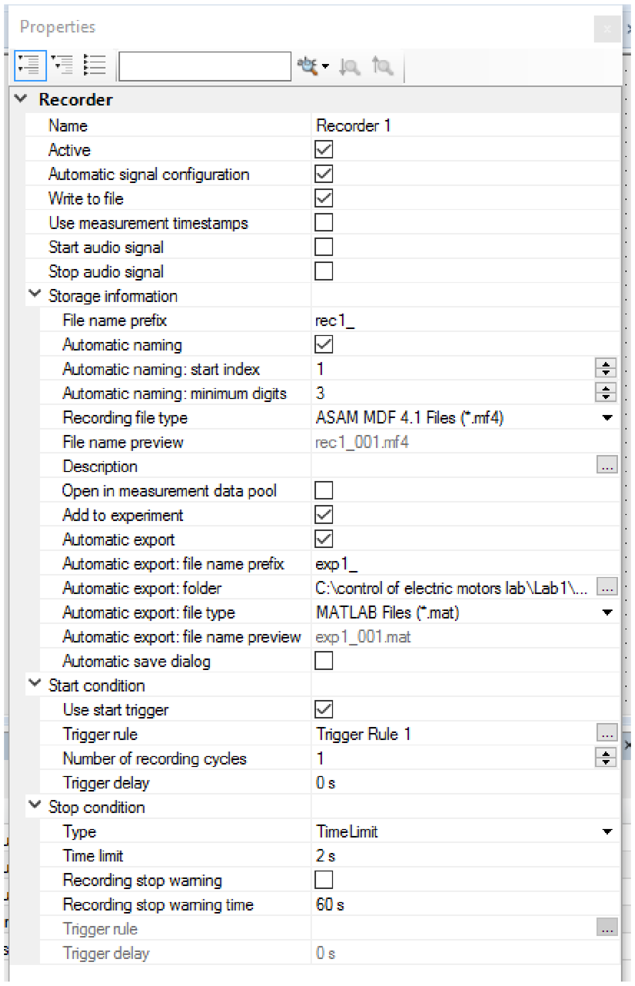
						Platform	
Type	Connected	Active	Recorded	Variable	Platform	HostService	OnChange
		✓	✓	gnal_In\n/In1	Platform	✓	
		✓	✓	gnal_Out/In1	Platform	✓	

Figure 20: Measurement Configuration

To export the recorded data to a .mat file, click on the name of the recorder as in Fig. 21 and open the Properties tab on the right-hand side of the screen in Fig. 21, or right-click on the name of the recorder, then **> Properties**. Check the **Automatic Export** checkbox. The file name prefix will be a part of the file name for each .mat file you save. The numerical suffix will be incremented by one each time you save a file. You can see what the file name will be by looking at the file name preview. Browse to the location that you want your .mat files to be saved at. And be sure to change the file type to **MATLAB Files (*.mat)**. You can also check the box for ***Automatic Save*** dialog. If this box is checked, dSPACE will prompt you to save the recorded data set. If this box is unchecked, dSPACE will automatically save a set of data each time you start the recording.

If you have another program with the same signal names, you can re-use its recorder. For this, right-click on the **Recorder 1> Export>** Save as Types: **XML files: >** Give a new name. To load a previously-saved recorder, right-click on **Recorder> Import>** and find the recorder you want to re-use.

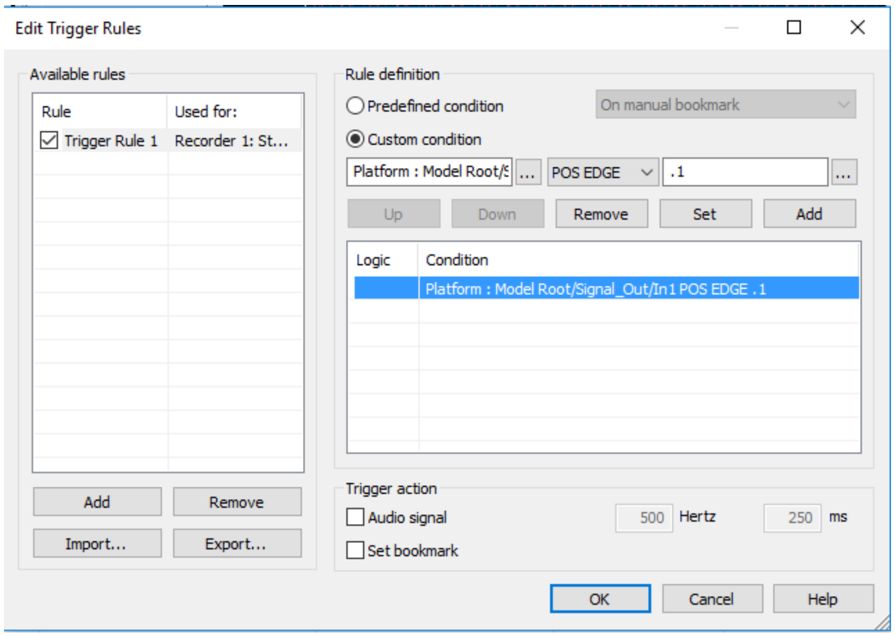
b. **Triggering the recording**: It is important to note that the ***Recorder*** and ***Trigger rules*** only control data capture. They will not affect when voltages are applied, nor will they start/stop any signals from being applied or read. With this in mind, you can use the ***Trigger Rules*** to start the recording of desired variables when some variable threshold is crossed. There are multiple ways to save data using a trigger signal. One method is to use a Start Trigger. Under the **Properties tab > Start Condition,** check the **Use Start Trigger** checkbox. Click on the browse button for the **Trigger Rule.** This should open up a new window called **Edit Trigger Rules,** as shown in Fig. 22, where you will set the conditions and thresholds that will start the data capture.


Properties

Recorder	
Name	Recorder 1
Active	✓
Automatic signal configuration	✓
Write to file	✓
Use measurement timestamps	☐
Start audio signal	☐
Stop audio signal	☐
Storage information	
File name prefix	rec1_
Automatic naming	✓
Automatic naming: start index	1
Automatic naming: minimum digits	3
Recording file type	ASAM MDF 4.1 Files (*.mf4)
File name preview	rec1_001.mf4
Description	
Open in measurement data pool	☐
Add to experiment	✓
Automatic export	✓
Automatic export: file name prefix	exp1_
Automatic export: folder	C:\control of electric motors lab\Lab1\...
Automatic export: file type	MATLAB Files (*.mat)
Automatic export: file name preview	exp1_001.mat
Automatic save dialog	☐
Start condition	
Use start trigger	✓
Trigger rule	Trigger Rule 1
Number of recording cycles	1
Trigger delay	0 s
Stop condition	
Type	TimeLimit
Time limit	2 s
Recording stop warning	☐
Recording stop warning time	60 s
Trigger rule	
Trigger delay	0 s

Figure 21: Properties of Recorder

Click **Add** to create a new trigger rule. Select the button for **Custom condition**. Select a variable to trigger off of using the browse button (3 dots) next to the textbox. Next, there is a drop down menu which will allow you to choose the type of condition to use. To the right of the drop down menu is another text box and accompanying browse button. You may either type in a value for the threshold of the trigger or you may select another variable. In this case, since we are using 10V as our amplitude, a threshold of 0.1 will ensure that the threshold is crossed. Since you want to make sure data capture begins on the rising edge, select **POSEDGE** from the drop down menu. Note that you can also use the **Trigger delay** option to set a specific time after the start signal to begin the capture of data. This can be done in the ***Properties*** tab under **Start conditions**. You can also set the length of the data capture under the Stop conditions section. Select **Type > Time Limit** and for Time limit specify the amount of time for capturing data. You can use trigger similarly to create a stop data recording condition as well instead of time limit. To capture data, open the Measurement Configuration tab and select the Recorder you will be using. It should look similar to Fig. 23.


Edit Trigger Rules

Available rules

Rule	Used for:
☑ Trigger Rule 1	Recorder 1: St...

Rule definition

○ Predefined condition — On manual bookmark
◉ Custom condition

Platform : Model Root/S — POS EDGE — .1

Logic	Condition
	Platform : Model Root/Signal_Out/In1 POS EDGE .1

Trigger action

☐ Audio signal — 500 Hertz — 250 ms
☐ Set bookmark

Figure 22: Edit Trigger Rules Window

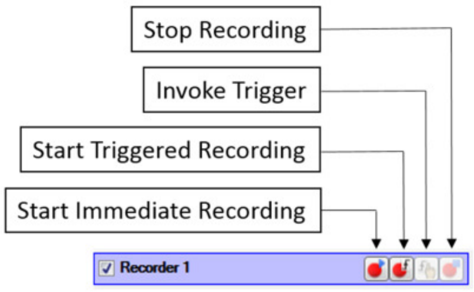


Figure 23: Data Capture Buttons

Since you will be using a trigger to capture data, select the **Start Triggered Recording** button. The blue background will now turn yellow. The recorder is now waiting for the defined trigger threshold before it begins to collect data. Once the threshold is reached, it will begin to capture data. If a time limit was set for the recorder, it will automatically stop once it reaches the time limit. If the ***Automatic Save*** dialog box is checked, you will be prompted to save this set of data. Once you have recorded the desired data and saved it to a .mat file as described previously, you will then be able to open it in MATLAB using the **load** command. However, the format is not particularly convenient to view and analyze the data. Therefore, we suggest the use of a macro to convert the data to a more tractable form. For this purpose, do not change the name of the file once it is saved in the MATLAB directory.

The trigger settings can be saved for a later experiment as well. As shown in Fig. 22, click on **Export> Save as type**: **Trigger Rules Files (*.txt) >**. Give it a new name. To load the saved trigger settings, click on **Import>** Browse and find the trigger file that you want to load.

c. Importing the data in MATLAB: There are different ways to view the data saved from the Control-Desk in MATLAB. The preferred option for these labs is to use the Matlab macro **Mat_Unpack.m**, which can be downloaded from lab web page. In order to use **Mat_Unpack**, the m-file must be saved in the folder containing the file and the MATLAB directory must be set to this folder. In the command line, type: **Mat_Unpack** and hit enter. There will then be a prompt reading, "Enter .mat file to load from current working directory." Enter the name of your saved data file without the '.mat' ending. You will be see the list of variables along with time variable which should be named 't'. Once this is done, you can plot or manipulate your variables using the names listed.

   You can also save the variables in a new .mat file, keeping only those variables that you want, and avoiding the need to use **Mat_Unpack** again for this data. In the Matlab window, type ***save exp1.mat var1 var2***, where *var1* and *var2* are the variables that you want to save and *exp1.mat* is the file name. Later, typing ***load exp1.mat*** will copy the variables back into the workspace.

   Using the method described previously for the saving of data, create 2 plots using MATLAB: one for the sinusoid output on DACH and one for the sinusoid input on ADCH. Label and save a copy of these plots for your report. Make sure that you have captured all plots and screenshots for your report before moving on to the next experiment!

5. Open-Loop Voltage Controller

In this part of the tutorial, we will control the speed of a DC machine using Simulink and dSPACE Control-Desk through an open-loop voltage controller. For the purpose of this tutorial, a DC motor with an encoder mounted on its shaft will be used.

ix. Open-loop voltage control Simulink model

The Simulink model of Fig. 24 allows the user to control the speed of the DC motor by changing the voltage applied to the motor (labeled **Motor_Voltage**). Note that there are two blocks: a ***saturation*** block (with upper and lower limit of +25 and -25 respectively) and a ***gain*** block (with a gain of 1/50) between the input voltage and the DAC connection. This compensates for the gain of 10 of the D/A and the gain of 5 from the amplifier.

>> Save the model as *'dc_motor_control'* and make sure that the MATLAB directory path to the current folder is the one that the model is saved in.
>> Type ***Ts =1e-4***; in **Toolbar > File > Model Properties > Callbacks > InitFcn***
>> Make necessary changes to simulation time and the step size as described in ***ii****(h).*
>> Build the model using **Ctrl+B**.

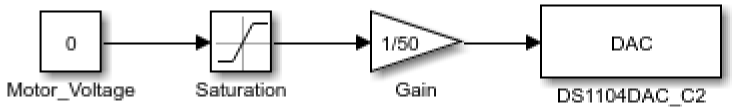


Figure 24: Simulink model to produce a DC voltage

x. *DC motor dSPACE Control-Desk*

Now that the model is ready, you can design the user-interface. Create a new layout and load the .sdf file (dc_motor_control.sdf) as described in section ***iii****(a)*. Map the ***Motor_Voltage*** using a ***Numeric input*** and a ***Display*** from the Instrument Selector. Also use it as a trigger variable as discussed in *viii.b* with a POSEDGE value of 2V.

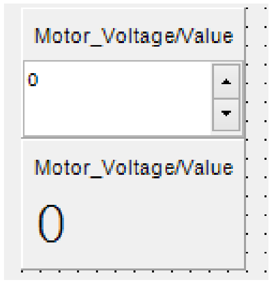


Figure 25: Capture of user-interface for DC motor Voltage

xi. *Power the amplifier and hardware connection*

Plug the dual power amplifier using the supply cord, and keep the main switch in the OFF position. Connect from DACH 1 to one side of the amp (on the backside with a BNC connection) using a BNC-to-BNC cable. Connect the motor to the back of the amplifier on the same side that you plugged the BNC using banana-to-banana cables.

xii. *Motor position and velocity*

An encoder is mounted on the DC motors used in the lab. Plug the encoder cable from the motor to the INC1 port on the dSPACE breakout box. The encoder read-out block can be found at:
>>Simulink Library Browser > dSPACE RTI1104 > DS1104 Master PPC> DS1104ENC_POS_C1.

This block provides read access to the two encoder interface channels *enc position* and *enc delta position* of. The number of these channels can be changed as described in ***i****(a)*. Use an encoder cable to connect the motor to the breakout box and make sure that the channel number matches the number that you chose in the Simulink encoder block. Also add an ***Encoder Master Setup*** block to the model. This block sets the global specifications for the channels of the encoder interface.

>>**Simulink Library Browser > dSPACE RT11104 > DS1104 Master PPC> DS1104ENC_SETUP.**

Make sure that the encoder signal type in the encoder master setup block is set to *single-ended (TTL).*

>> Double click > DS1104ENC_SETUP > Encoder signal type > Type> single-ended(TTL).

The encoder consists of 500 lines per revolution. To convert the line count to radians, use the block of Fig. 26. The two blocks on the right, *term1* and *term2*, are "terminator" blocks, and are used to keep MATLAB from displaying warnings that the signals are disconnected.

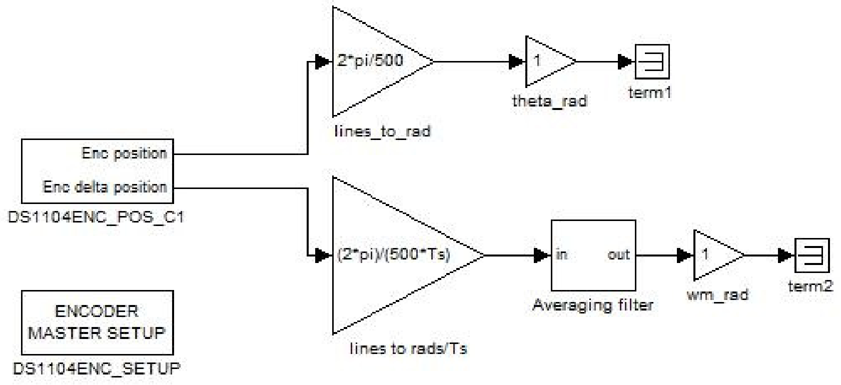


Figure 26: Encoder position and velocity read blocks

An averaging block is used to smooth the oscillation of the velocity signal. This block is shown in Fig. 27 and can be added to the model using a subsystem block. The averaging block produces an output that is the average of the last *nd* input values. You will need to set *nd=30* in

>> Toolbar > File > Model Properties > Callbacks > InitFcn*

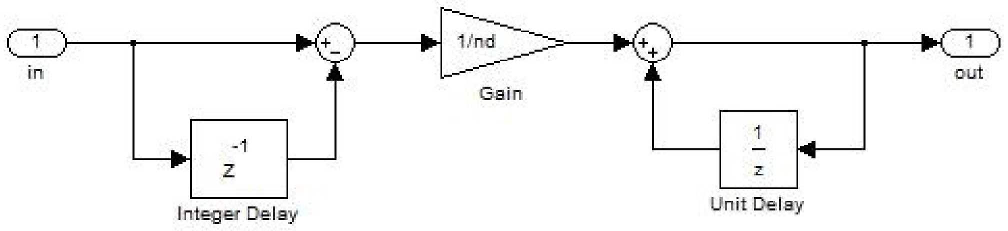


Figure 27: Averaging block

The subsystem block can be created by copying from **Simulink Library Browser > Simulink > Commonly Used Blocks > Subsystem.** Define the averaging block subsystem as shown in the figure above, using a gain block and delay blocks from the library. You can edit the blocks by double-clicking on them. For the summation circles, you can edit addition or subtraction by changing the sign from a '+' to a '−'. In the ***Integer Delay*** block, set the number of delays to *nd*. Add the averaging block to the encoder position block and make all the necessary connections and changes as shown in Fig. 26.

xiii. Zero encoder position

To reset the encoder measurement so that the position of the motor does not include the previous line count from the encoder, you can send a constant zero to the block ***Encoder Set Position*** block using a trigger block, as shown in Fig. 28.

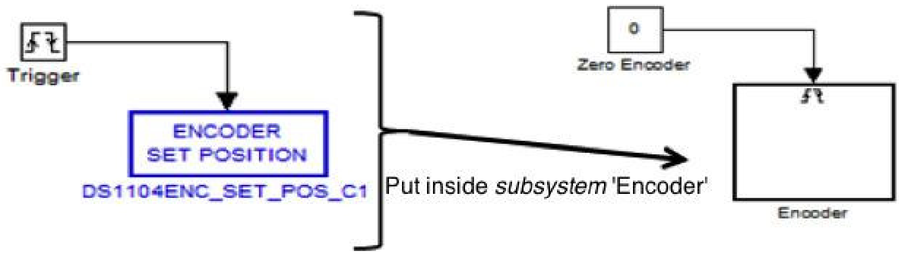


Figure 28: Zero encoder block

Create a new subsystem (renamed 'Encoder') and add a trigger from **Simulink Library Browser > Simulink > Ports & Subsystems > Trigger.** Change the trigger type to ***Either*** through:

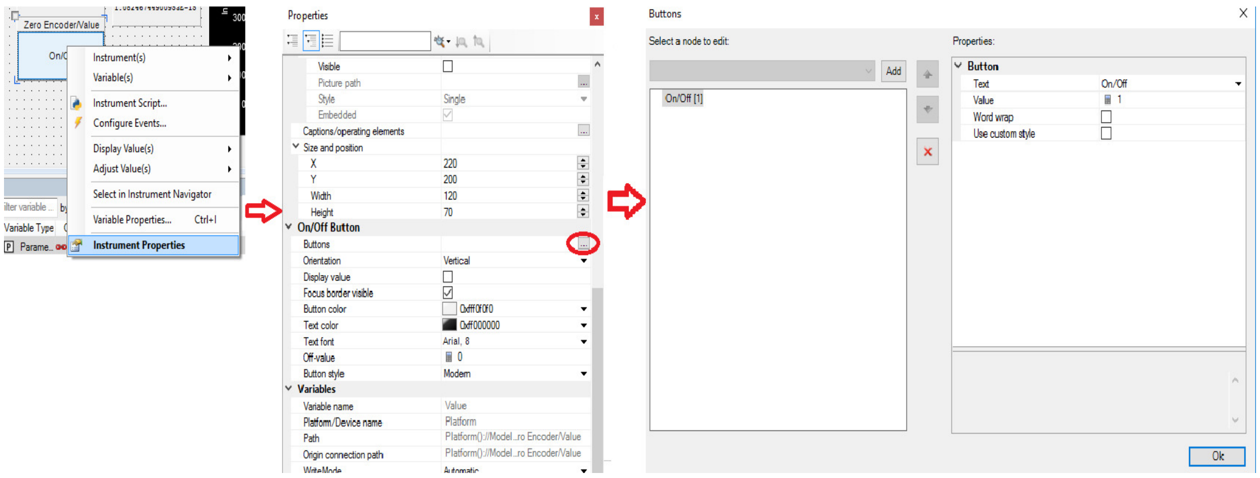
>> Double-click > Trigger-Based Linearization > Parameters > Trigger type> Either.

Add an output port to the trigger by checking the ***Show output port*** in the ***Trigger*** menu. Also add an ***Encoder Set Position*** block to the subsystem from **Simulink Library Browser > dSPACE RT11104 > DS1104 Master PPC> DS1104ENC_SET_POS_C1.**

Make sure the number of the channel in this block matches the number selected for the encoder position block (C1 or C2). Connect the blocks and feed the trigger (outside the subsystem) with a constant equal to zero, as shown in Fig. 28.

Layout:

Use an **ON/OFF Button**, found under **Standard Instruments**, to send a signal to zero the encoder. Modify the **ON/OFF Button** so that it shows only one button, as shown in Fig. 29 (delete one button and keep the other). Add a ***Display*** window to the layout to view the speed of the motor, and drag and drop the motor speed, named ***wm_rad***. Do the same for the ***th_rad*** variable. The layout for this example should look like the one shown in Fig. 30 without the plots. An overall view of user-interface used in this example can be found in the appendix on Fig. 36.



Properties

Visible	☐
Picture path	
Style	Single
Embedded	✓
Captions/operating elements	
Size and position	
X	220
Y	200
Width	120
Height	70
On/Off Button	
Buttons	...
Orientation	Vertical
Display value	☐
Focus border visible	✓
Button color	0xff0f0f0
Text color	0xff000000
Text font	Arial, 8
Off-value	0
Button style	Modern
Variables	
Variable name	Value
Platform/Device name	Platform
Path	Platform():/Model...ro Encoder/Value
Origin connection path	Platform():/Model...ro Encoder/Value
WriteMode	Automatic

Buttons

Select a node to edit: On/Off [1]

Properties:

Button	
Text	On/Off
Value	1
Word wrap	☐
Use custom style	☐

Figure 29: Configuring Zero Encoder button

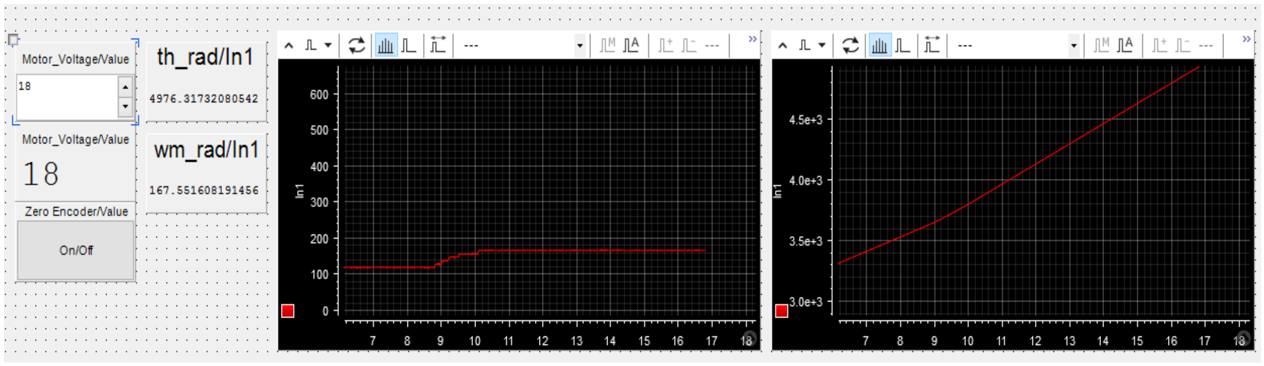
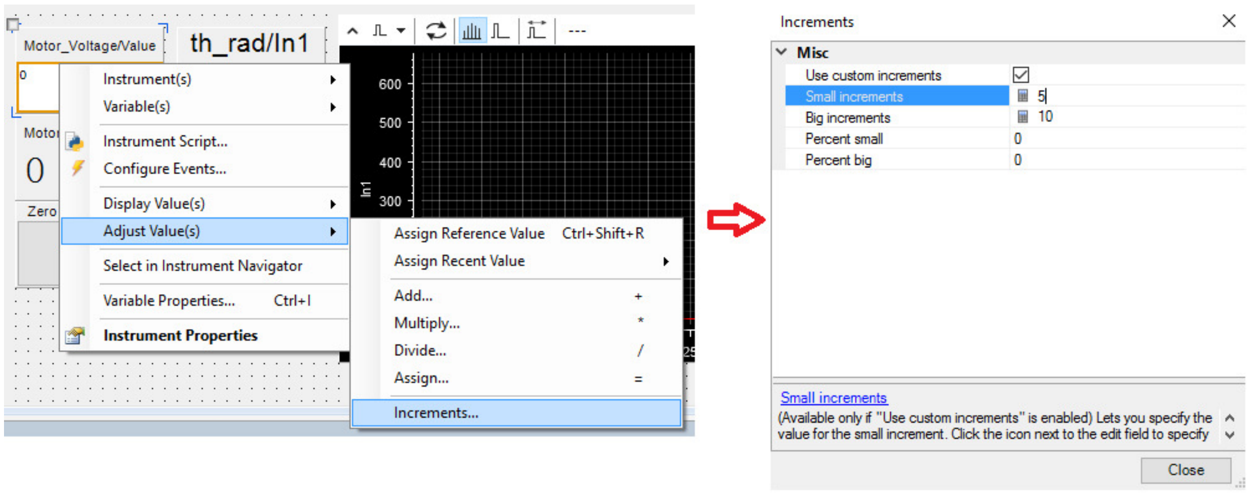


Figure 30: dSPACE layout for DC motor control

Next, run the program by clicking on the ***Go Online*** and ***Start Measuring*** buttons, as described in section *III.vi* and increase the ***Motor_Voltage***. The DC motor voltage will increase by 1V with each click. To make it increase by 5V per click, return to the off-line mode. Then right-click as shown in Fig. 31 and put 5 in ***Small increments***. Now follow *viii (b)* to set ***Motor_Voltage*** as trigger variable and use 0.1 as the trigger value. Set any time between 0.2s to 0.5s as stop time. Push the ***Start Triggered Recording*** button. Then, increase the ***Motor_voltage*** once. It will record data for 0.5s. Reset the voltage to zero and go off-line. Import the data to MATLAB. If the time variable is missing, you may recreate the vector based on the sampling period and the length of the other vectors.

At this point, make sure that the zero encoder function added to the model is functioning properly. Click the zero encoder button (ON/OFF button) and the motor position will reset to zero. If everything in this lab has operated correctly, the lab tutorial has been completed.



Use custom increments	✓
Small increments	5
Big increments	10
Percent small	0
Percent big	0

Small increments
(Available only if "Use custom increments" is enabled) Lets you specify the value for the small increment. Click the icon next to the edit field to specify

Figure 31: Setting up the DC Motor voltage

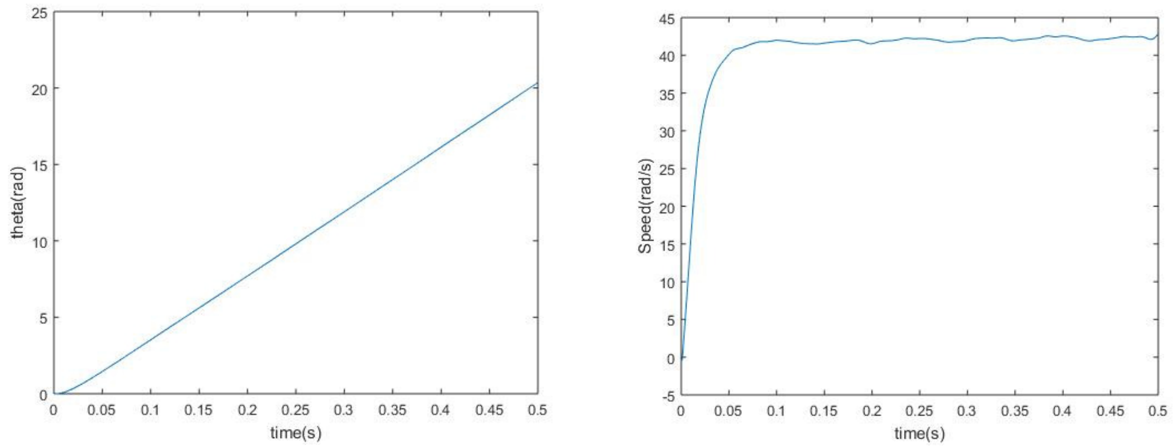


Figure 32: Captured DC motor position and velocity for step input of 5V

6. Report Requirements:

Use the following as a guideline when preparing the lab report:

- Introduction and/or objectives
- Screenshot of the simulated sinusoid signal
- Screenshot of the sinusoid on the lab oscilloscope
- MATLAB plots of sinusoidal waveforms
- MATLAB plots of position, and velocity vs. time
- Screenshots of Simulink Models
- Conclusion with reference to stated objectives. Describe what worked well and did not work well in this lab, and make suggestions for possible improvements.

***Be sure to LABEL the axes of all your plots and to include UNITS on all of your values. Comments should accompany every plot.**

VII. Appendix

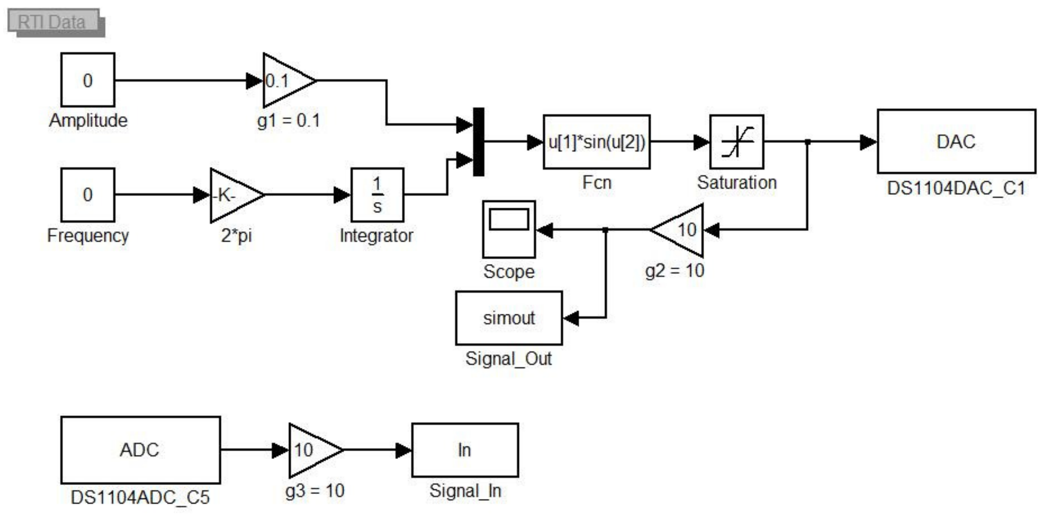


Figure 33: Overview of Simulink model to generate a sine wave

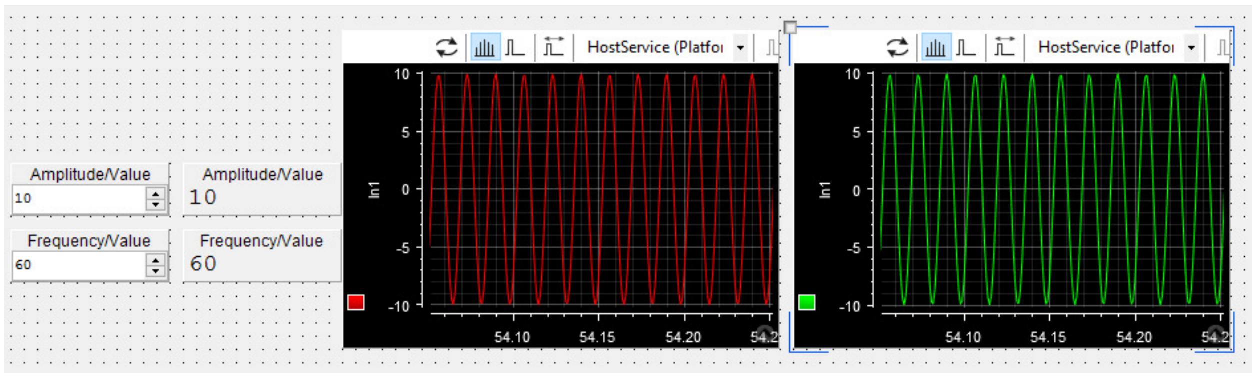


Figure 34: Overview of the user-interface used in sinusoidal example

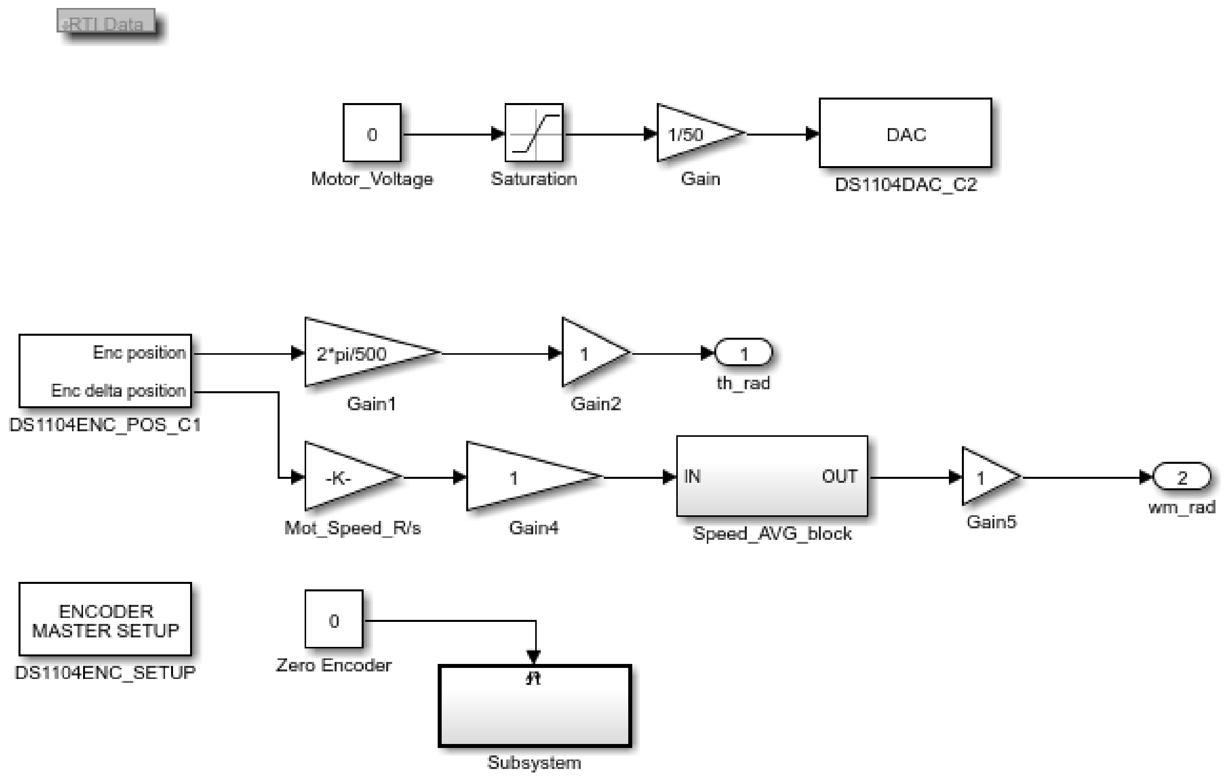
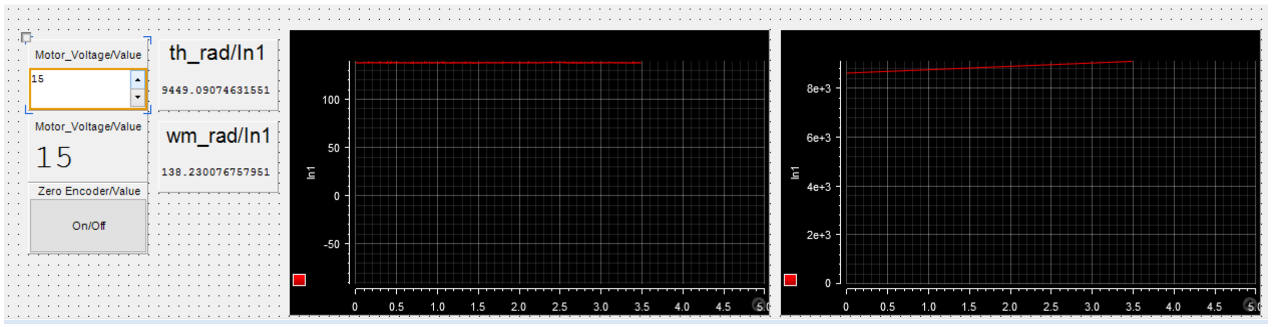


Figure 35: Overview of the Simulink model to run the DC motor



Motor_Voltage/Value
15

Motor_Voltage/Value
15

Zero Encoder/Value
On/Off

th_rad/In1
9449.09074631551

wm_rad/In1
138.230076757951

Figure 36: Overview of the DC motor user interface