

**ECE 5671/6671 – Lab 1**  
**dSPACE DS1104 Control**  
**Workstation & Simulink**  
**Tutorial**

**December 17, 2018**

Equipment needed:

- dSPACE I/O box
- Power Electronics Drive Board (PEDB) with ribbon cable and +12V supply
- Box of cables
- DC generator (DCG), frame mounted

## I. Introduction

The purpose of this tutorial is to provide an introduction to the dSPACE ControlDesk and Matlab/Simulink software to explain their use with the Power Electronics Drive Board (PEDB) and the Motorsolver machines. In general, there are five major parts in this setup that will be used in the lab. These five parts are as follows:

1. Software (Matlab/Simulink and dSPACE). The dSPACE software is configured based on parameters set in Matlab and Simulink (through an \*.sdf file).
2. DS 1104 R&D Controller Card, this card is installed in the computers of the lab and connects to the dSPACE I/O box via a master I/O ribbon cable.
3. The Power Electronics Drive Board (PEDB). This is a HiRel inverter board designed to power the electric motors. The board is controlled by dSPACE via a 37-pin DSUB ribbon cable. The PEDB board requires +12V to drive the board's control circuitry and a separate 42V for the power electronics.
4. The electric machine setup (*i.e.*, electric motors and generators).
5. Cables used to interconnect the components (*i.e.*, encoder cable, banana cables, and BNC cables).

The general block diagram of the setup is shown in figure 1. In this setup, users input the desired signal (*i.e.*, voltage) from the ControlDesk (user-interface) to the dSPACE I/O box and inverter board via DS 1104 R&D Controller card. Users can also measure and transfer the resulting data (*i.e.*, speed, position, current) back to the dSPACE software in real time; the data can then be stored in a \*.mat file and transferred to MATLAB for post processing.

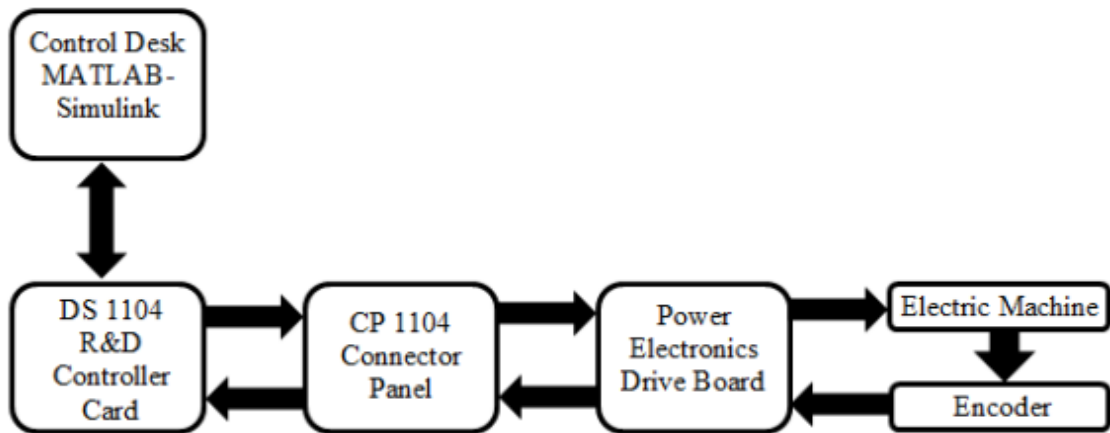


Figure 1: Setup block diagram

The physical appearance of these parts and equipment are shown in figures 2-6:

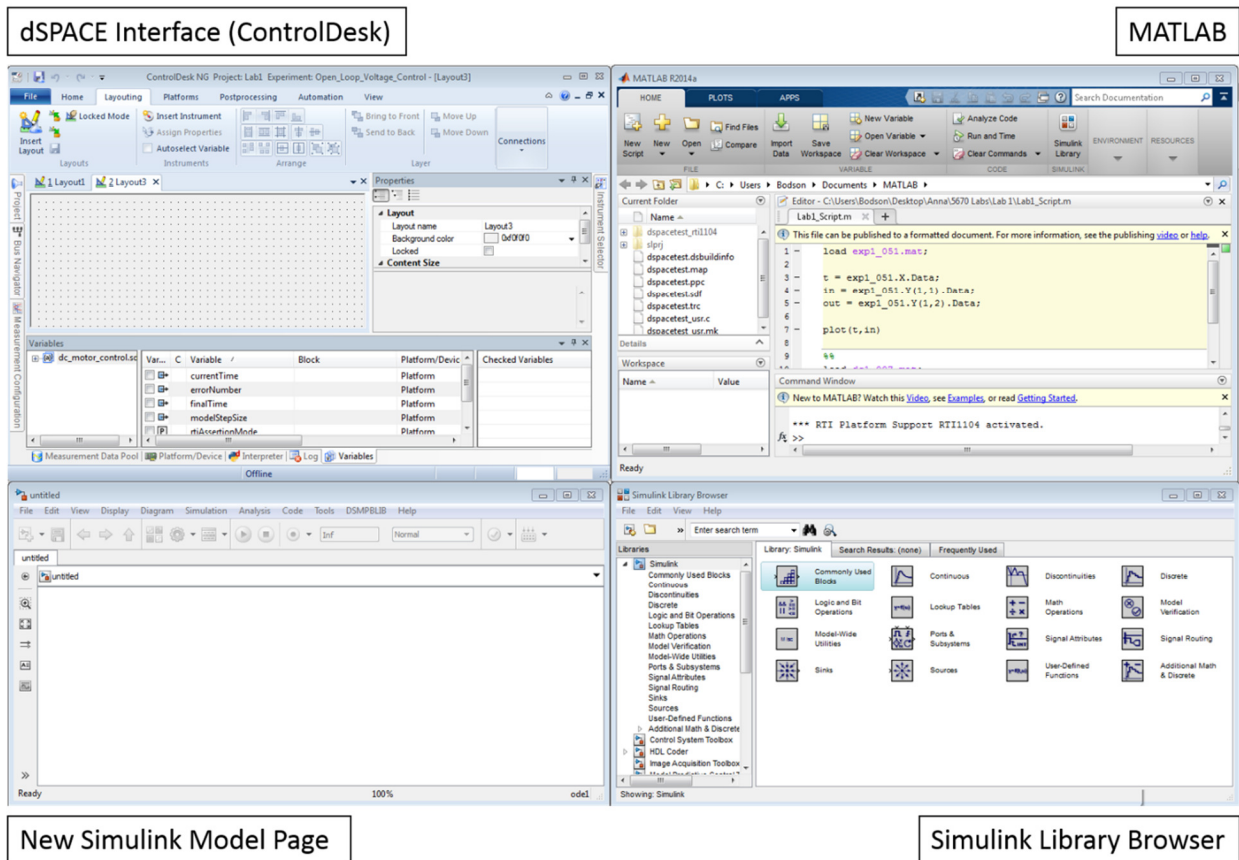


Figure 2: ControlDesk, Matlab and Simulink interface



Figure 3: CP 1104 I/O box with attached master I/O ribbon

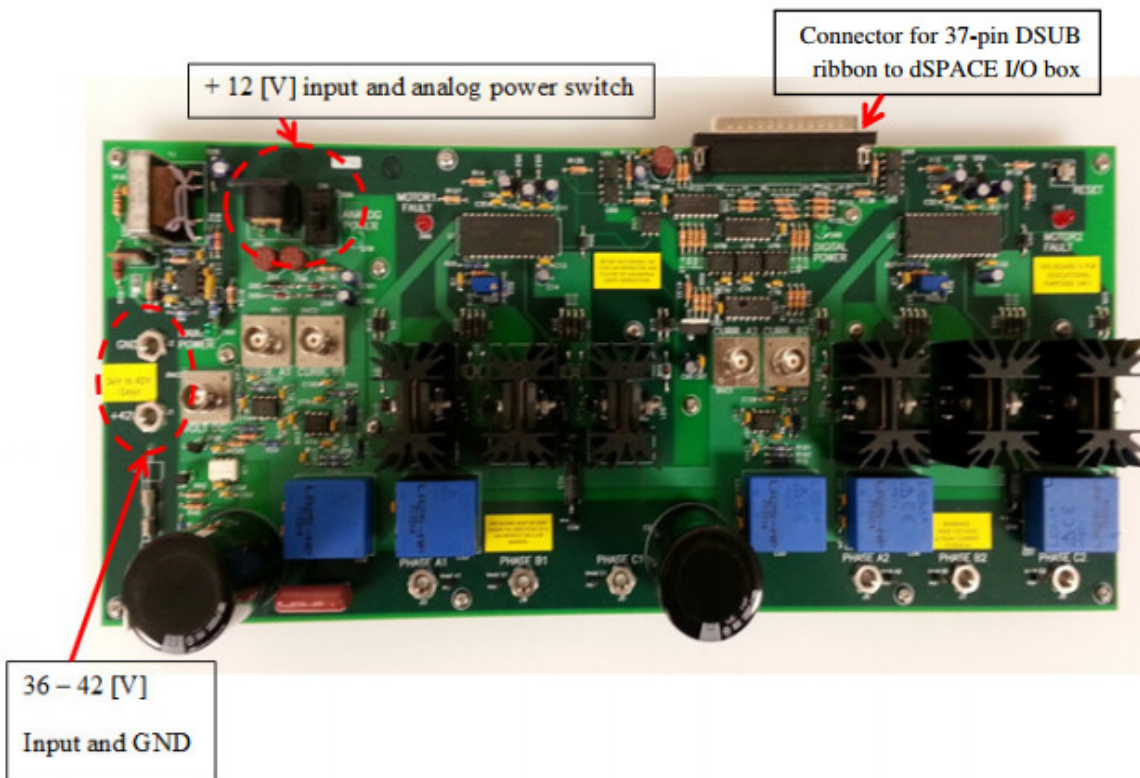
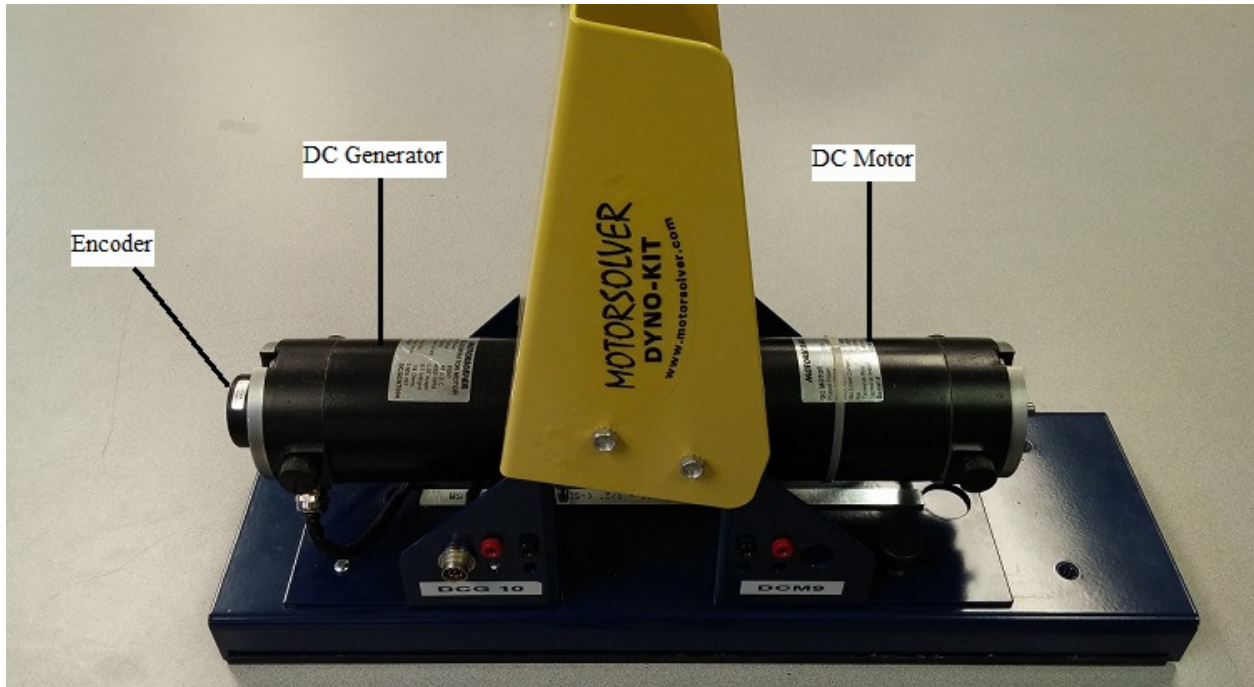
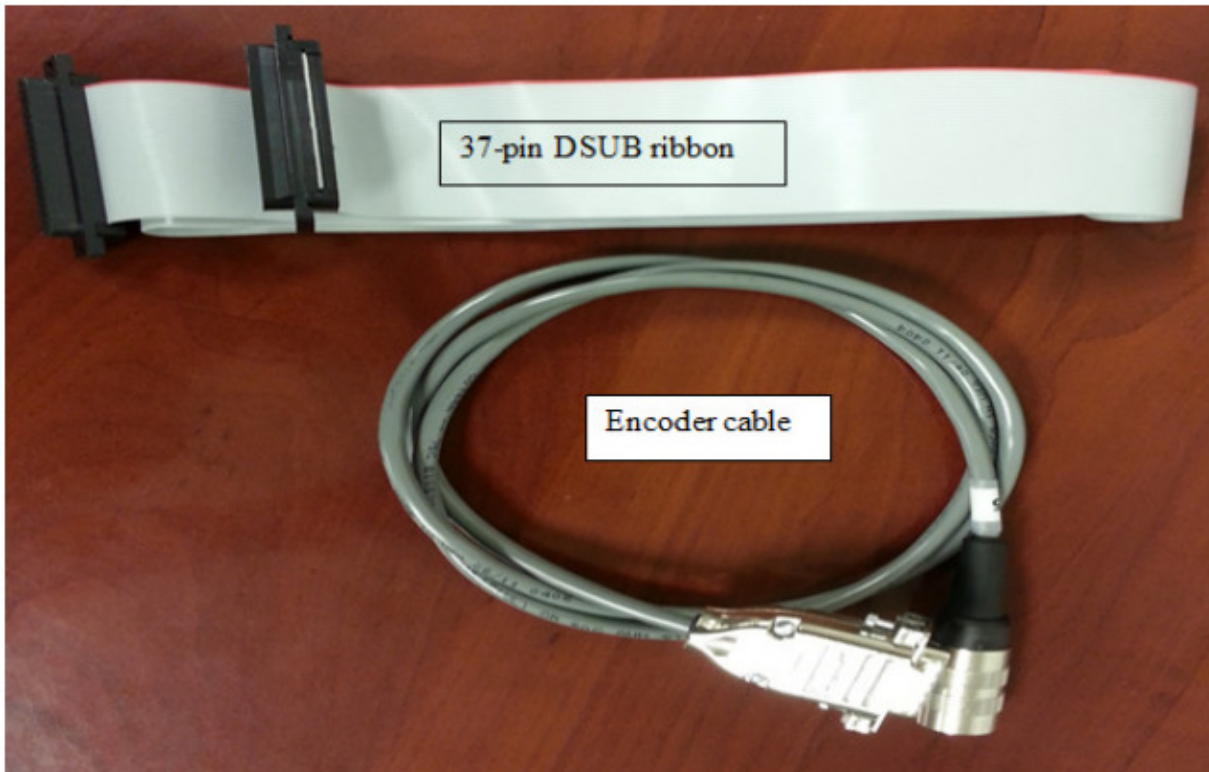


Figure 4: Power Electronics Drive Board (PEDB)

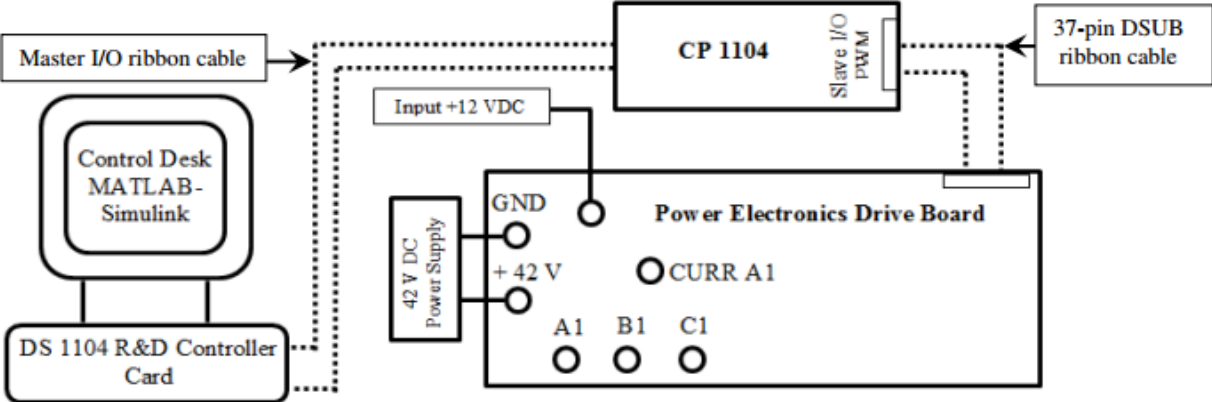


**Figure 5: Electric machine setup with DC motor, DC generator and encoder**



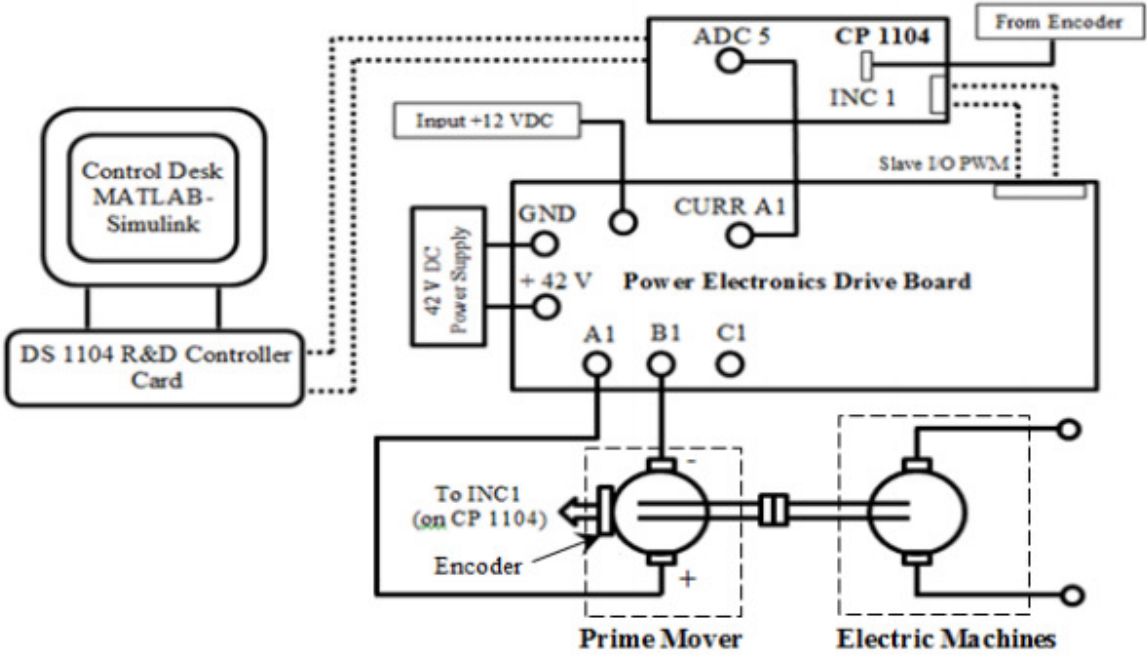
**Figure 6: Encoder and 37-pin DSUB ribbon cables**

The connections between the DS1104 R&D controller card, the breakout box CP1104, and the power electronics drive board are shown in figure 7. Please note that this connection is needed for all labs.



**Figure 7: Physical connection of PEDB to CP1104 breakout box and DS1104 R&D Controller card**

Also, physical connections between the electric machines, PEDB and CP 1104 breakout box for this lab are shown in figure 8.



**Figure 8: Physical connection between electrical machines, PEDB, and CP 1104 breakout box**

For the purposes of this tutorial, the steps needed to create Simulink models and layout files in dSPACE are outlined with the use of two examples. In these examples, a Simulink model (\*.mdl) to generate a sinusoidal signal and another model to apply a DC voltage output to a DC motor are designed and built. Once the first Simulink model is verified in simulations, the model is modified to output the signal to an oscilloscope using a digital-to-analog output channel. Then, a control panel layout is designed (\*.lax), serving as a user interface for the dSPACE ControlDesk. 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.

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



**\*\*Common Problems\*\***

- DO NOT SAVE files to the server directory (X: drive). This includes the desktop. Save to a folder on the local C drive or a USB drive during lab implementation. Also set the Matlab working directory to that same folder.
- In general, refer to the list of common problems and solutions found on the lab web page. If the solution to the error is not found, request assistance from the TA.

## II. 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 an oscilloscope through a digital-to-analog channel (DACH).

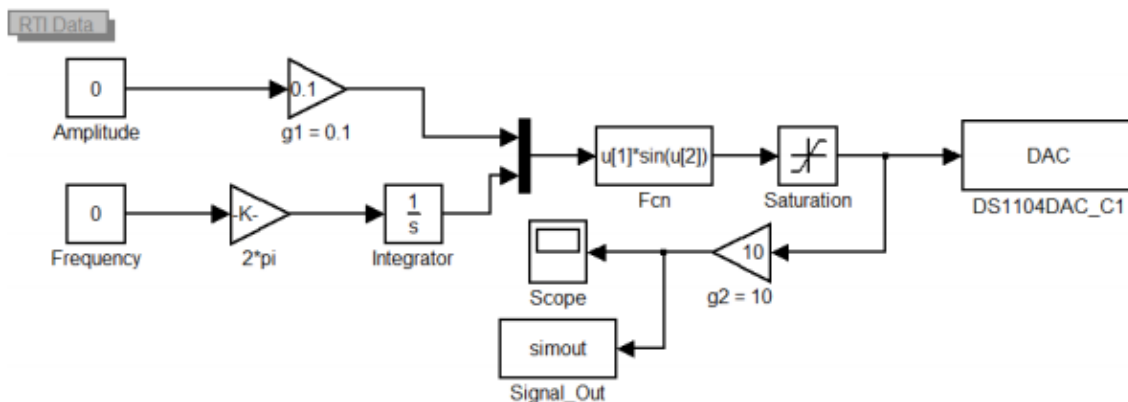
### *i. Sine wave Simulink model*

Start Matlab and make sure to select DS1104 when prompted. Begin by creating a Simulink model to generate a sinusoidal signal. 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 new model icon (). Now that the Simulink page is open, the Simulink library browser can be accessed by clicking on the *Library Browser* 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. Place the Simulink blocks as follows, using figure 9 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 constant values to zero.  
**>>Double click > Source Block Parameters: Constant > Main > Constant value > 0.**
- 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 value of one of them to (1/10) and the other one to (2\*pi).  
**>> Double click > Function Block Parameters: Gain> Main > Gain > (1/10) / (2\*pi).**
- c. Add an *Integrator* and a *Mux* block from **Simulink Library Browser > Simulink > Commonly Used Blocks > Integrator / Mux**. 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 > Function 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,. In turn, these values correspond to  $\pm 10V$  (the dSPACE ADCH and DACH ports scale signals by a factor of 10).  
**>>Double click > Function Block Parameters: Saturation > Main > Upper limit/ Lower limit > 1 / -1.**
- e. Drag and drop a *Digital-to-Analog Channel (DAC)* block from **Simulink Library Browser > dSPACE RT1104 > DS1104 MASTER PPC > DS1104DAC\_C1**. This block will allow you to view the output signal from CP1104 I/O board on an oscilloscope. 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 > Channel number > #.** For this tutorial, use DACH 1 on the I/O board.
- f. Drag and drop a *To Workspace* block from **Simulink Library Browser > Simulink > Sinks > To Workspace**. Change the saved data format to Array through:  
**>>Double click > To Workspace > Save format > Array.**
- g. Add another *gain* block as described in b. Change the value of this gain to 10.



- h.* Connect all of the blocks and change the names as shown in figure 9. 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 figure 9. 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. If you wish to rotate blocks for the sake of a more visually clear connection, you can **Right Click the Block > Format > Rotate Block > Clockwise** or simply select the block and type **Ctrl+R**.



**Figure 9: Simulink model for sine wave – blocks connection**

*ii. Building the Simulink model*

In order to build the Simulink model, first define the sampling period. In the Matlab Command Window, enter  $T_s = 1e-4$ , which will set the sampling period to  $100\mu s$  or the sampling frequency to 10kHz.

- a.* Change the simulation time to infinity from the Configuration Parameters in the Simulink toolbar:

**>> Toolbar > Simulation > Configuration Parameter > 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.

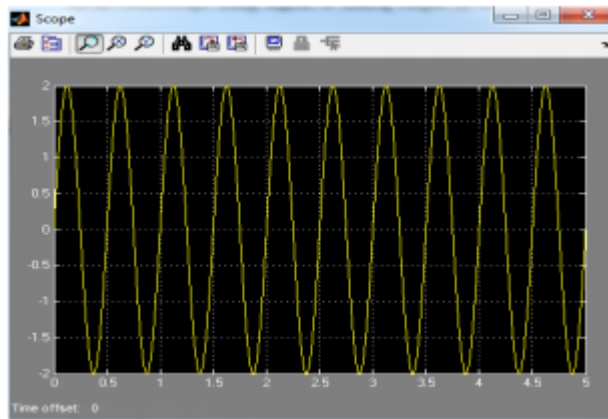
**>> Toolbar > Simulation > Configuration Parameter > Solver > Solver options > Type / Solver > Fixed-step / ode1(Euler) > 0.0001.** Make sure to refer the checklist on the lab website for further model configuration parameter settings.

- 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 with the value of 10 (where the *To Workspace* block is connected), as shown in figure 9. 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. Figure 10 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** 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 must be turned off. In the scope window select **Parameters > History > uncheck limit data point**. 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, snipping tool, or by clicking the printer in the scope window and printing to a file.



**Figure 10: 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 *system description file* (*filename.sdf*, where *filename* is the name of the mdl file). The sdf file contains the code downloaded to the DS1104 board and is used by dSPACE ControlDesk to map the variables of the Simulink diagram to the variables displayed or controlled in the layout.

### Common error messages while building the Simulink model:

When using Matlab R2014a, dSPACE chooses to run on the R2013b environment. The diagnostic viewer will prompt you to revert back to 2013b with the following message:

```
The Simulink parameter handling was NOT reverted back to the R2013b behavior. This is currently not supported by RTI. To revert back to the R2013b behavior please run: revertInlineParametersOffToR2013b
```

Type the command suggested in the viewer, in the command window to revert and build your model again.


If the following message occurs:

```
The activity 'Stop Application' cannot be started because the following activities are still pending:  
'Online Calibration' on client 'ControlDesk NG 5.2'
```

While making edits on a Simulink model that has already been opened on the dSPACE ControlDesk, ensure that the ControlDesk window is offline. Do this by clicking on the *Go Offline* button (refer figure 17).

## III. dSPACE ControlDesk



Open the dSPACE program by double clicking on the (  ) icon or from the Start menu.

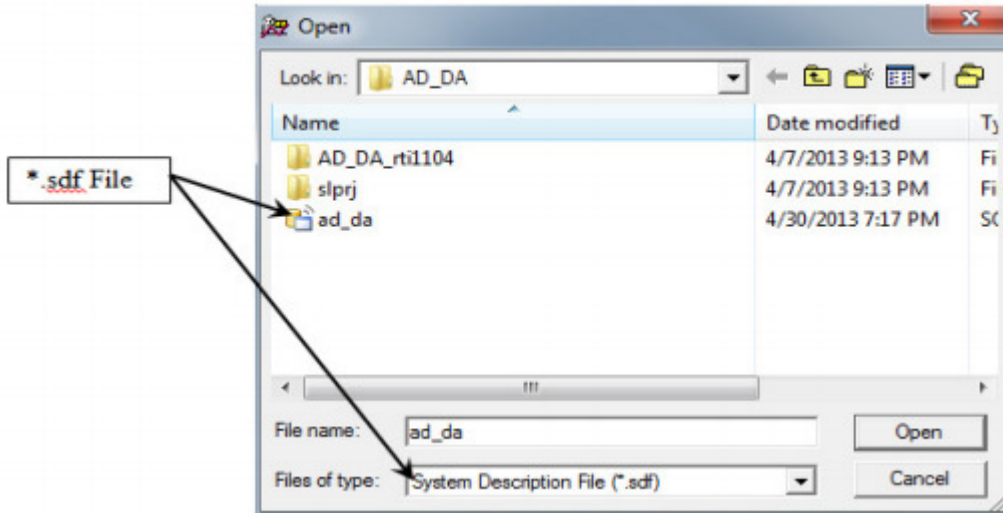
### *iii. Create a new project + experiment*

#### **File > New > Project + Experiment**

This will open a new window which will take you through the necessary steps to create a new project + experiment which you will use in the lab experiment.

- Give the project a name and set the location for the root directory.
- Give the experiment a name
- Check that the “DS1104 R&D Controller Board” is selected.


Open the .sdf system description file through **Import from file... > the folder that the model is saved in > the model name**. Note that only one sdf file can be open in dSPACE at a time.

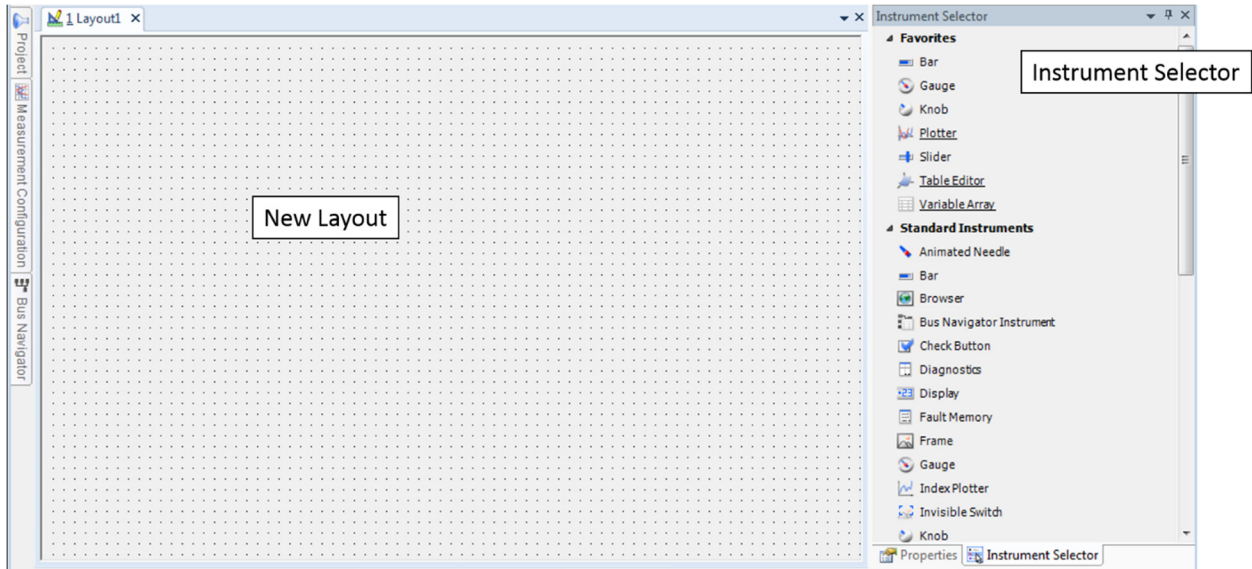


**Figure 11: .sdf file**

Loading a .sdf file will load all the variables created on Simulink into dSPACE, where a layout is built for interfacing purposes. The variables available through Simulink can be seen in figure 14. The window in figure 11 shows the .sdf file being selected for loading. On making any changes to the original Simulink model, by clicking on the **Project tab > Right click on .sdf file > Reload variable description**, you can update variables to their latest values without restarting dSPACE.

*iv. Create a new layout*

- a. If a new layout window does not automatically appear, open a new layout from **Layouting > Insert Layout** (  ). A new window will show up. There is a tab on the right hand side of the screen called “Instrument Selector” as shown in figure 12. You may need to click on the thumb tack in the top right hand corner to keep the menu visible. If the Instrument Selector tab does not show automatically, it can be accessed through **View > Switch 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, which means clicking anywhere in the layout window and drag the mouse until the instrument is the desired size.

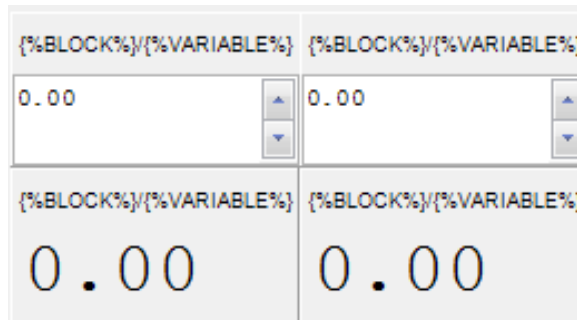


**Figure 12: New layout and instrument selector**

- b. Click on *Numeric Input* under *Standard instruments* section and draw two numerical inputs, and then click on *Display* and draw two display windows as well in the layout window. Note that you must click the desired icon in the Instrument Selector before drawing each box.

>>**Instrument Selector > Standard Instruments > Numeric Input / Display.**

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 below and



**Figure 13: Numerical input and Display windows**

may include additional text such as “/{%VARIABLE%}”. This is not important in identifying what the block displays. When you add variables to these boxes, they will automatically replace “{%BLOCK%}” with the variable name. To add your own caption, right click on the instrument you wish to name and select instrument properties (or simply select the instrument and click on the *Properties* tab on the right). Click on the browse button opposite the **Captions/operating elements**. Click on **caption** and edit the default caption to your satisfaction. To remove an element, select the instrument and click **Ctrl+Delete**.

- c. To see the plot of the sine wave on the ControlDesk, click on *Plotter* from the Standard Instruments list and draw a plotter in the layout window.

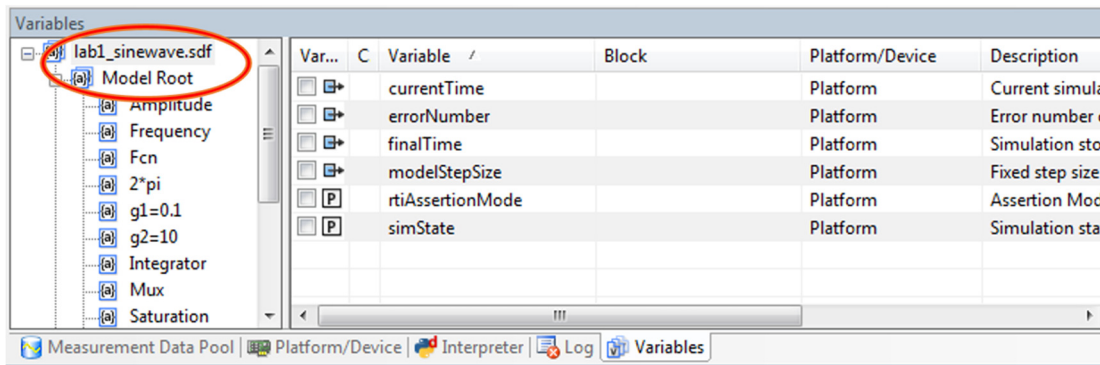
**>>Instrument Selector > Standard Instruments > Plotter**

**v. Map variables**

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

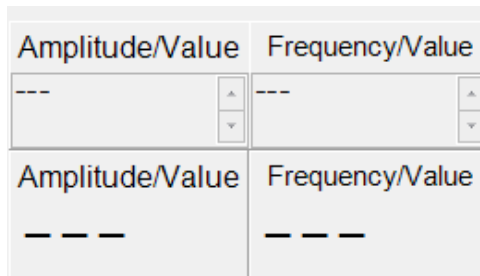
- a. Click the plus sign next to *Model Root* in the bottom left corner of the dSPACE program, where you will see all the variables and gain names from Simulink model.

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 figure 13. 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**. Another solution is to insert a gain block with a gain equal to 1 along the signal path.



**Figure 14: Windows at the bottom of the ControlDesk window**

- 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. Do the same steps for *Frequency*. Your layout should



**Figure 15: Numerical Input and Display after mapping the variables Amplitude and Frequency now look similar to figure 15.**

- 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 y-axis plotter in the layout window. This will display the signal being output from DACH1, a sine wave. The layout window should look like figure 16. Note that the plotter window may not look exactly the same.

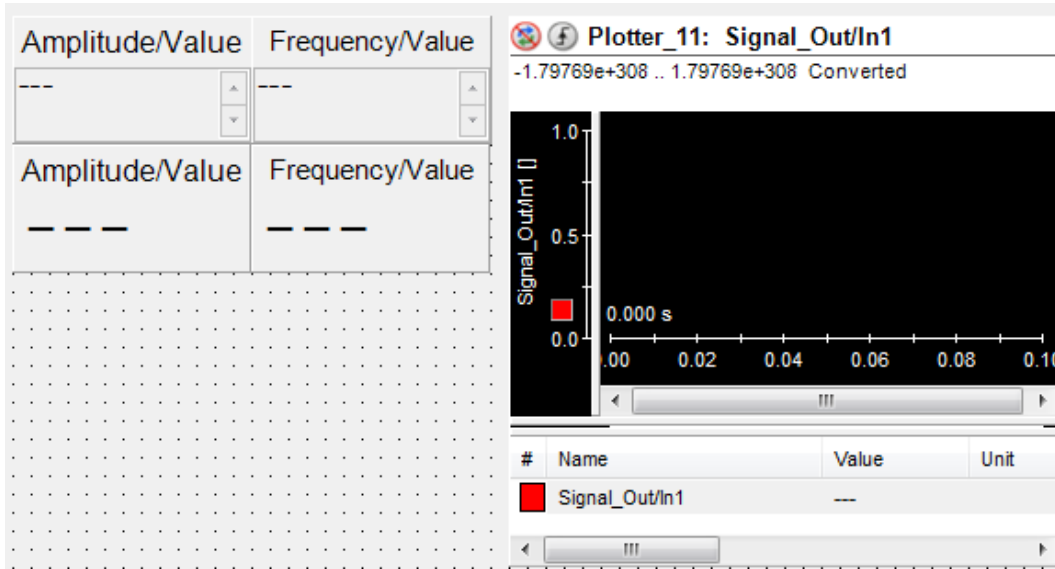


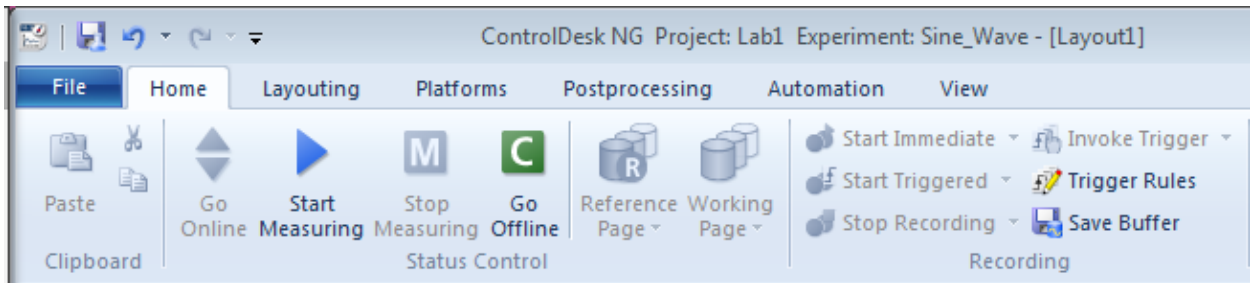
Figure 16: Overall view of the layout window

- d. To change the axis ranges on the plotter in the layout, go to: **>> Properties > Axes**, then click on the browse button seen in front of *Axes* line. To have a fixed axis window, select *Fixed* from *Scaling mode*. Then you can change the max and min of the relevant axis. This can also be handled while saving the data through *Measurement configurations* for the x-axis.

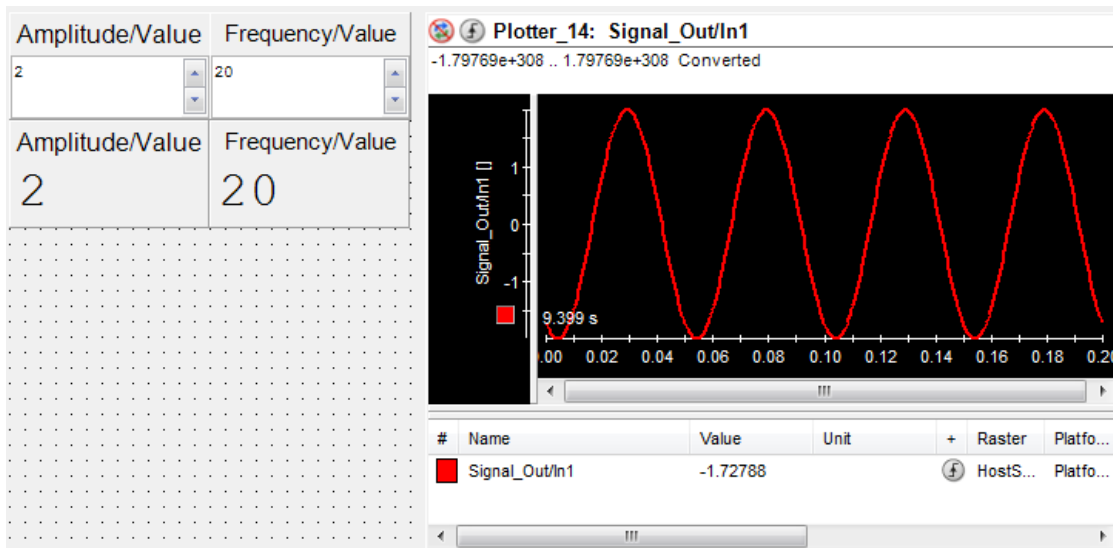
#### vi. Run the ControlDesk

The program/experiment on the ControlDesk runs as soon as the model is built with the dSPACE ControlDesk window open. If this is the first time you are building the model and have opened it fresh on the dSPACE ControlDesk, clicking on the *Go Online* for the first time starts the program. The *Start Measuring* button will only initiate the plotters to start displaying real time data. The *Start Measuring* button will switch from blue to grey and the *Stop Measuring* button from grey to blue.

Change the Amplitude and Frequency to different values. You should see the sine wave on the plotter array, as shown in figure 18. Note that if the time scale is uncomfortably large or small, you will need to change the sample time length using the *Measurement Configuration* menu described in section *viii* of this tutorial.



**Figure 17: Start Measuring and Stop Measuring buttons**



**Figure 18: ControlDesk while running**

To stop the measurement on the plotter, click the *Stop Measuring* button (the blue button will then turn grey) as in figure 17. This will only prompt the plotter to stop displaying real time data. But the program is still running on the dSPACE card even if you click on the “Go Offline” button. We shall address this issue in the future labs. However, zero out all your input values and press the “Go Offline” button to disengage the control desk.

Now that the program is built and functioning properly, the signal can also be seen on an oscilloscope via a BNC cable. Having connected the breakout box to the back of the computer (**make sure that the connector at the back of the computer is fully engaged!**), connect DACH1 to an oscilloscope. When you go online and vary the amplitude and frequency, the signal showing on the oscilloscope should look like the one shown in the plotter array in the user-interface in figure 18. 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.



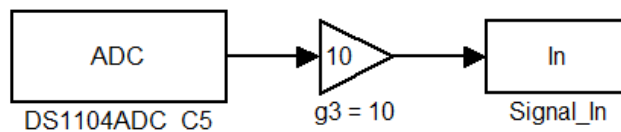
Zero out the input values again and stop the plotters by pressing the stop measuring button. In particular, if you need to modify the Simulink model or the layout, stop the online mode by pushing CTRL+F8 or by clicking the *Go Offline* button on the home tab.

**Important note:** Even after going off-line, the process created in Simulink will still be running on the board. In all the experiments, it is important to leave the system in a safe state when going off-line (for example, with zero voltages applied to the motors). The process running on the dSPACE board can also be stopped and restarted by clicking on the “Platform/Devices” tab at the bottom of the ControlDesk. The process is running if a green triangle appears in front of the board’s name, and stopped in the case of a red square. The state can be changed by right-clicking on the board’s name and clicking “Stop RTP” while the dSPACE control desk is offline (a red square appears indicating that the program has been stopped). If this is done while the Control-Desk is online, error messages will be displayed. The icon returns to “Run state” automatically when the control desk is engaged online again. Note that, when compiling a Simulink diagram using Ctrl-B while the control desk is open with the .sdf file loaded (assuming that the model was previously built already to generate the requisite .sdf file), the code is not only downloaded, but also immediately started. For this reason, the code should also be initialized with safe start-up values. As mentioned before, this will be addressed in the future labs.

## IV. 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 similar 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 figure 19.



**Figure 19: Analog to Digital Channel (ADCH) block**

- c. An overall view of the Simulink model for this example can be found in the appendix, figure 40. Take a screenshot of your complete Simulink model for the report. One way to do so is to type:

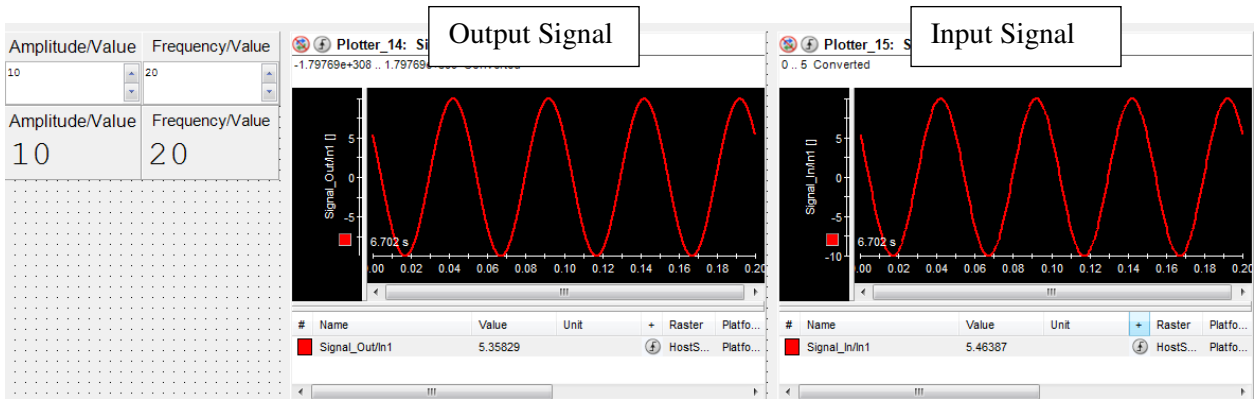
```
saveas(get_param('file','Handle'),'figure.tif')
```

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 where the picture will be saved (in the Matlab directory). Other formats are also possible (see Matlab documentation).

### **vii. Read the signal back to the ControlDesk**

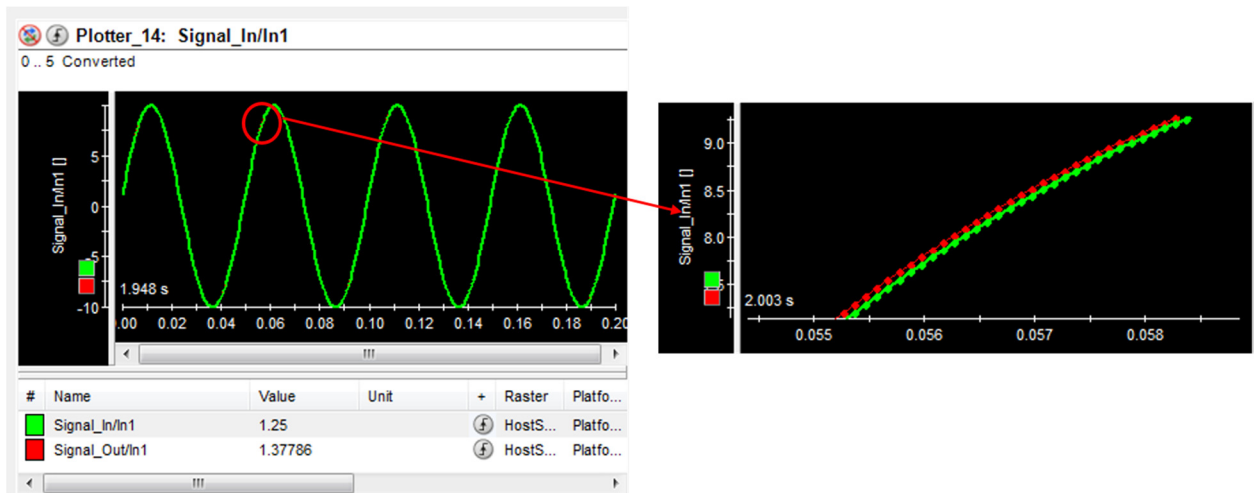
After adding the blocks of figure 19 to the Simulink model, build the model again as described in section *ii*. This will update the .sdf file and add the new blocks' names. Reload the updated .sdf file into dSPACE by clicking on the **Project tab (in the navigator window on the left hand side of the Control- Desk window)**> **Hardware Configurations** > **right click the .sdf file** > **Reload Variable Description** and it will replace the existing loaded 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 going online. Initiate the plotters by pressing *Start Measuring* button. The output signal to DACH1 and input signal to ADCH5 are shown in figure 20.



**Figure 20: Layout file showing the sine wave applied to DACH1 and read back through ADCH5**

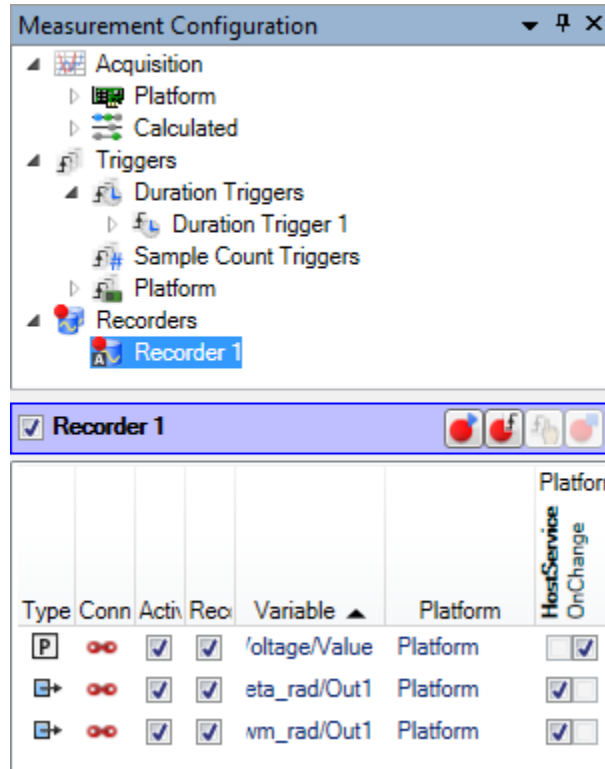
Also, these signals can be plotted in one plotter array by dragging and dropping both signals into one plotter, as shown in figure 21. The figure shows that the sine wave measured from ADCH5 is lagging the original sine wave by one sampling period, as expected.



**Figure 21: Capture of the plotter array showing both signals together**

**viii. Save the data**

The captured data on the plotter array can be saved as a .mat file. Data is saved through the use of a *Recorder*. In the *Measurement Configuration* tab on the left hand side of the ControlDesk window, as shown in figure 21, 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.



**Figure 22: Measurement Configuration Tab and Recorder in Detail**

To export the recorded data to a .mat file, click on the name of the recorder as in figure 22 and then open the *Properties* tab on the right hand side of the screen as shown in figure 23, or, **right click on the name of the recorder > 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.

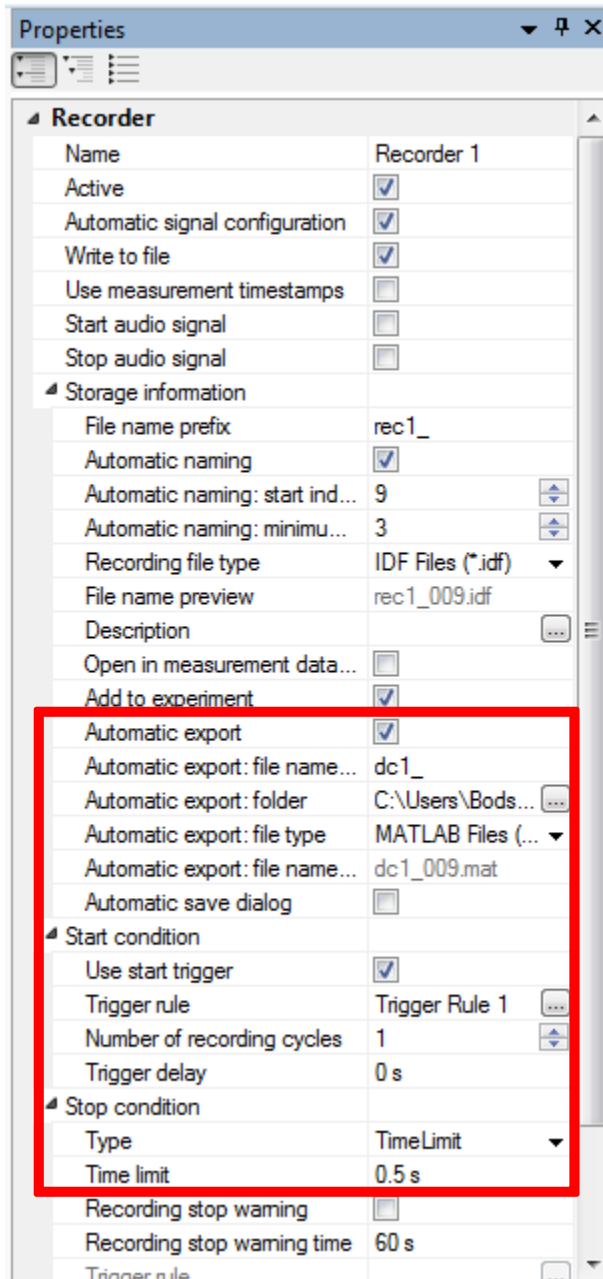
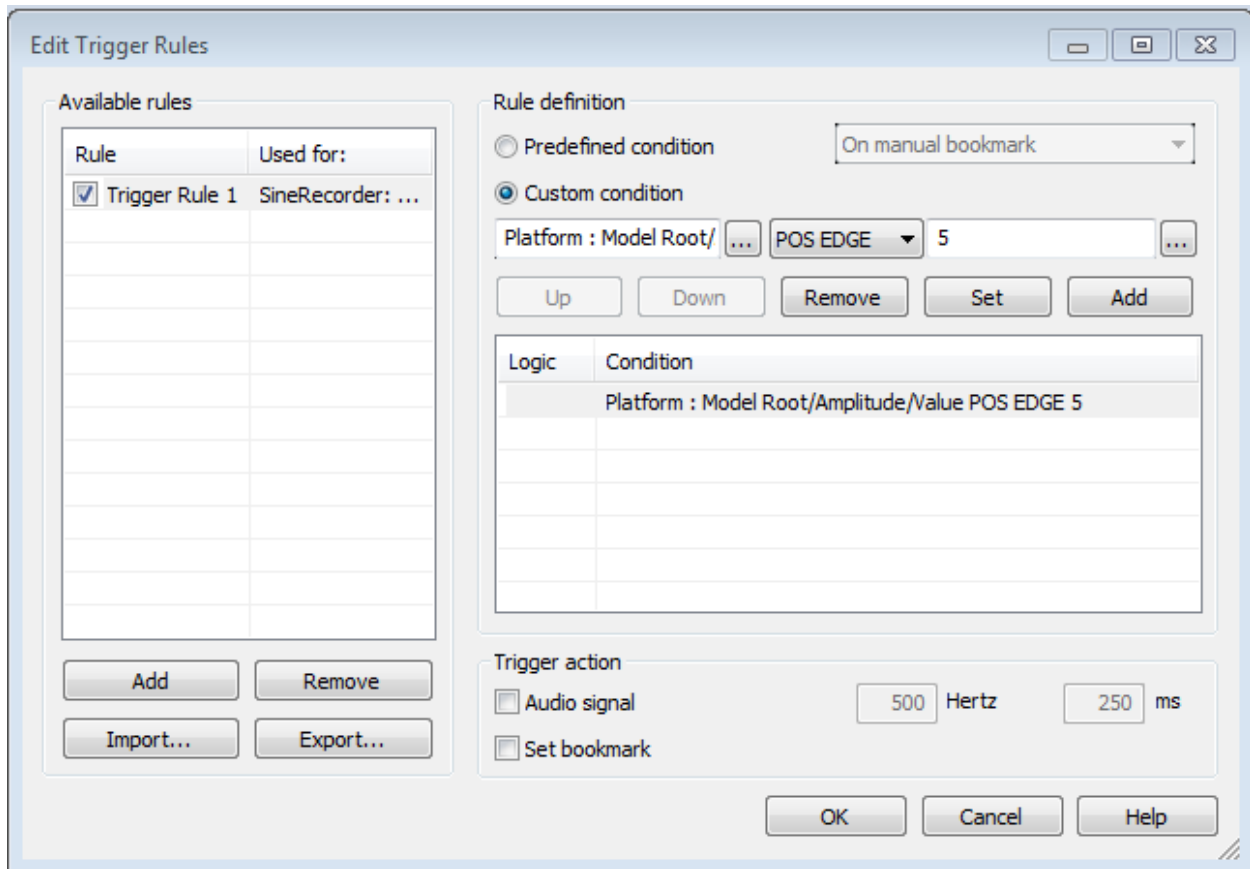


Figure 23: Properties Tab for a Recorder

## Triggering the recording

It is important to note that the *Recorder* and *Trigger rules* only controls data capture. It will not affect when voltages are applied nor will it start/stop any signals from being applied or read. With this in mind, you can use the *Trigger Rules* to capture desired variables starting 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 in figure 24. This is where you will set the conditions and thresholds that will start the data capture.



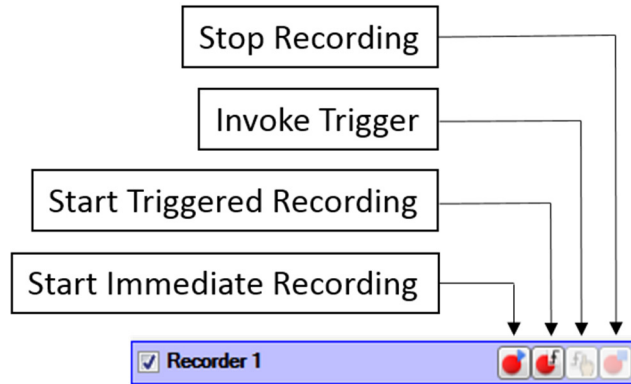
**Figure 24: Edit Trigger Rules Window**

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 5 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 > TimeLimit** and for Time limit specify the amount of time for capturing data.

To capture data, open the Measurement Configuration tab and select the Recorder you will be using. It should look similar to figure 25.



**Figure 25: Recorder Data Capture Buttons**

Since we will be using a trigger to capture data, select the *Start Triggered Recording* button. The blue background will now turn yellow as in figure 26. 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.



**Figure 26: Recorder Waiting for Trigger**

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.

## Importing the data in Matlab: Mat\_Unpack.m

The *Mat\_Unpack.m* file should be downloaded from the lab web page and saved in the folder containing the data file, which should also be the MATLAB working directory. 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 then be prompted to give a name to each saved variable. Type in a name you would like and hit enter for each variable. There will also be a prompt for an additional time variable that you need to name as well. Once this is done, you can plot or manipulate your variables using the names you just gave them.

You can also save the variables in a new .mat file using *save ('expl', 'var1', 'var2')* where *var1* and *var2* are the variables that you want to save and *expl* is the .mat file name. Later, typing *load expl.mat* will copy the variables back into the workspace. In this manner, you will not need to run Mat\_Unpack when you want to analyze the data again.

Using the data save method previously described, create 2 plots using MATLAB: one for the sinusoidal output on DACH and one for the sinusoidal 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!

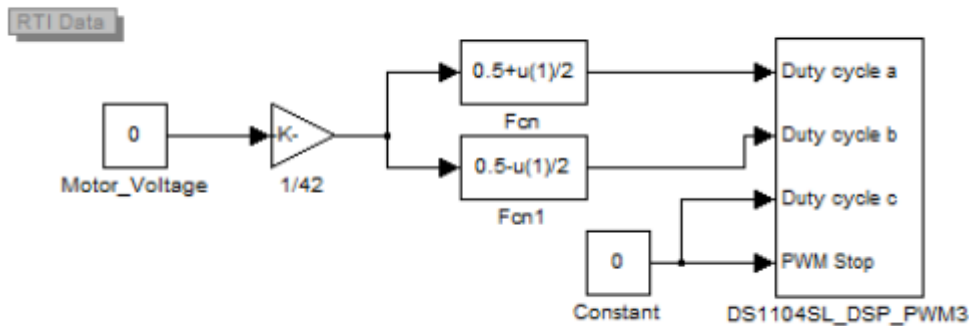
## V. Open-Loop Voltage Controller

In this part of the tutorial, you will control the speed of a DC machine using Simulink and dSPACE ControlDesk with an open-loop voltage controller. The voltage applied to the DC machine is based on a Pulse Width Modulation (PWM) modulation technique. For purpose of this tutorial, a DC generator (DCG) with an encoder mounted on its shaft will be used as the prime mover. Note that the machines labeled DCM and DCG are identical motors, but only the DCG machines have an encoder mounted on them.

### *ix. Open-loop voltage control Simulink model*

The Simulink model of figure 27 allows the user to control the speed of the DC motor by changing the voltage applied to the motor (labeled Motor\_Voltage). Note that the two boxes that lead into *Duty cycle a* and *Duty cycle b* are user-defined functions in Simulink. A value from 0 to 1 in *Duty cycle a* corresponds to a voltage  $V_a$  from 0 to 42V on channel A of the PEDB (assuming a DC supply of 42V). A DC motor voltage of 10V is obtained by applying  $21+5 = 26V$  on one side of the motor and  $21 - 5 = 16V$  on the other. A DC motor voltage of -10V is obtained with reverse commands. This technique makes it possible to apply voltages of positive or negative polarity to the motor, despite the fact that a single channel (A, B, or C) of PWM can only apply positive voltages.





**Figure 27: Simulink model of a DC motor controller**

The DS1104SL\_DSP\_PWM3 block can be found:

**>>Simulink Library Browser > dSPACE RTI1104 > DS1104 SLAVE DSP> DS1104SL\_DSP\_PWM3.**

Double click on this block and make the following changes (applicable for all lab experiments):

>> Set the PWM frequency to 10 [KHz] (10000 Hz)

>> Set the Deadband to 0 [ $\mu$ s]

>> Under the initialization tab select 'suspend to' option and make sure all the channels with bar (/a, /b, /c) are set to TTL High and the others to TTL Low.

>> Do the previous setup for PWM Stop under PWM Stop and Termination tab.

>> 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  $T_s = 1e-4$  in the Matlab Command Window; this sets the sampling time to 100 $\mu$ s.

>> Make necessary changes to simulation time and fixed step-size as described in *ii.a*.

>> Build the model Ctrl+B.

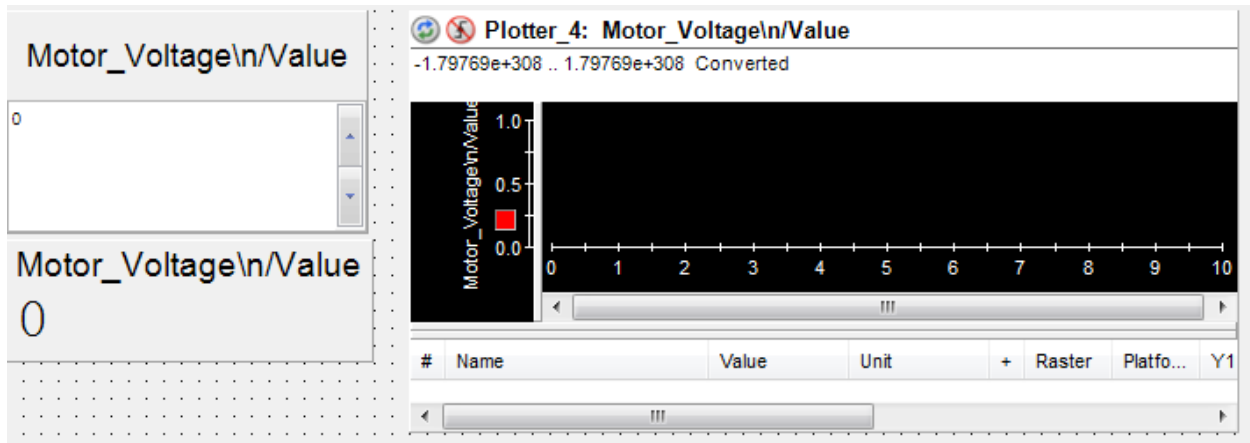
### ***x. DC motor dSPACE ControlDesk***

Now that the model is ready and you have loaded the .sdf file of the model, you can design the user-interface. Create a new layout and create a new experiment to load the .sdf file (dc\_motor\_control.sdf) under the lab 1 project as described in section *iii*.

>> **File > New > New Experiment**

This will create a new experiment under the Project structure Lab 1. Remember that only one experiment can be active at a time in a Project. If you would like to activate another experiment in the same project structure, go to the Project tab, right click on the experiment to be activated and click **Activate**.

Map the Motor\_Voltage to a numerical input and a display as discussed previously. Add a plotter and map the Motor\_Voltage to this plotter. This will come in handy to use the *Start Measuring* button and see the variations in the variable. The motor can, however, be run without a plotter, simply by going online and increasing the motor voltage value once the hardware connections are set up. Refer to figure 28.



**Figure 28: Capture of user-interface for DC motor control**

### ***xi. Power the PEDB and hardware connection***

- a. Ensure that the older connections are undone. Before running the ControlDesk, make sure that the PEDB is powered properly. This board requires two separate power supplies, as shown in figures 29-30. Connect the +12V adapter to the socket on the inverter board and change the switch to the ON position, refer to figure 4.

Prior to connecting the main power supply to the board; turn on the power supply, adjust the output value to 42V, and turn off the power supply. The power supply will save the last setting before it was turned off.

Connect the positive (red terminal) on the power supply to the positive port on the inverter board and ground (black terminal) to ground port on the PEDB, refer to figure 4. Turn ON the power supply. These steps will prevent applying a voltage higher than the rated voltage to the inverter board in case the previously saved value is higher than 42V.



**Figure 29: +12V Adapter**



**Figure 30: Power Supply**

Connect the PEDB and CLP 1104 I/O box using the ribbon cable. Then, connect the PEDB phase A1 port to the positive (red) terminal and B1 port to the negative (black) terminals on the DC motor. Refer to figure 8 for the hardware connection.

- b. Now that the hardware connections are set, run the layout file by clicking on the *Start Measuring* button and change the value of the voltage (Motor\_Voltage) applied to the DC motor. The DC motor should start rotating counterclockwise and you can vary its speed by changing the value of the voltage in dSPACE.

## ***xii. PEDB control functions***

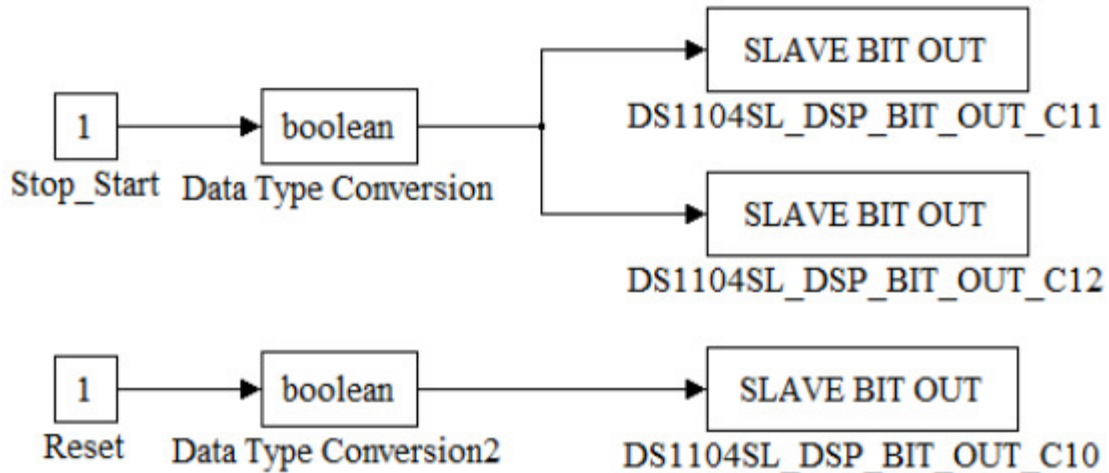
- a. To drive the DC motor for any given step voltage, the *Shutdown* and *Reset* signals on the PEDB are used. This will allow any desired step voltage to be input to the electric machine while it is initially at rest. Also, this feature enables you to stop the electric machine instantaneously without stopping the program or inputting zero voltage in the user-interface. The shutdown signals are controlled by the digital I/O channels 11 and 12 embedded in the drive board interface. These channels are controlled by a Boolean (1 or 0) input. Initially, the states of these switches are set to be one, meaning the switches for the shutdowns are closed. When the state changes to zero in dSPACE ControlDesk, the switching signals are inhibited and the switches are open. Similarly, the *Reset* signal is controlled by the digital I/O channel 10.

This feature allows you to reset the data collection and inverter board in case of motor fault. First, convert the I/O inputs to Boolean. This can be done with the use of a Data Type Conversion found under **Simulink Library Browser > Simulink > Commonly Used Blocks > Data Type Conversion**. Change the output data type in this block to Boolean.

**>> Double click Data Type Conversion > Output data type > Boolean.**

Now that the I/O inputs are converted to Boolean, the signal can be sent to the PEDB with the use of a slave bit out block. This block can be found under **Simulink Library Browser > dSPACE RTI1104 > DS1104 SLAVE DSP > DS1104SL\_DSP\_BIT\_OUT\_C0**.

The addition to the model should look like the one shown in figure 31. The number of channels in the slave bit out block can be changed as described in section *i*. Select channels 11 and 12 for *Shutdown* and 10 for *Reset*. After adding these blocks, rebuild the model as discussed previously (section *ii*) and reload the .sdf file into dSPACE Control-Desk (section *vii*).



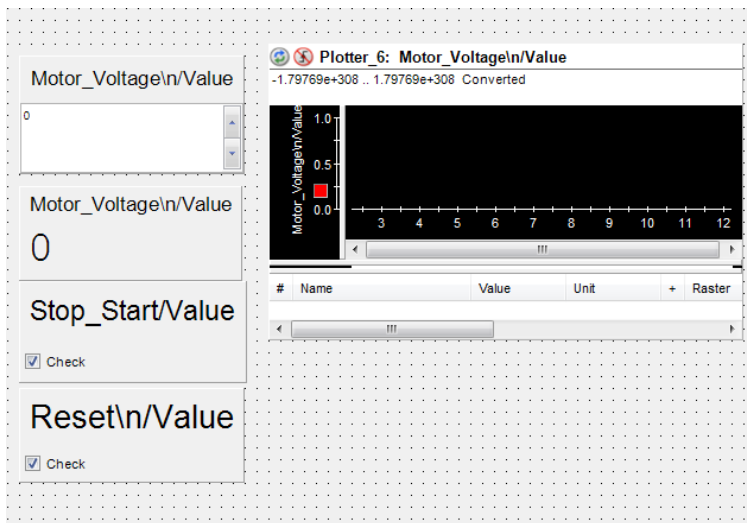
**Figure 31: Simulink model of shutdown and reset**

- b. The *Shutdown* signal, here represented by the *Start\_Stop* constant, will allow us to control the operation of the motor through the ControlDesk interface. Go offline on the dSPACE ControlDesk while editing the layout. Add two check boxes to the user-interface through:

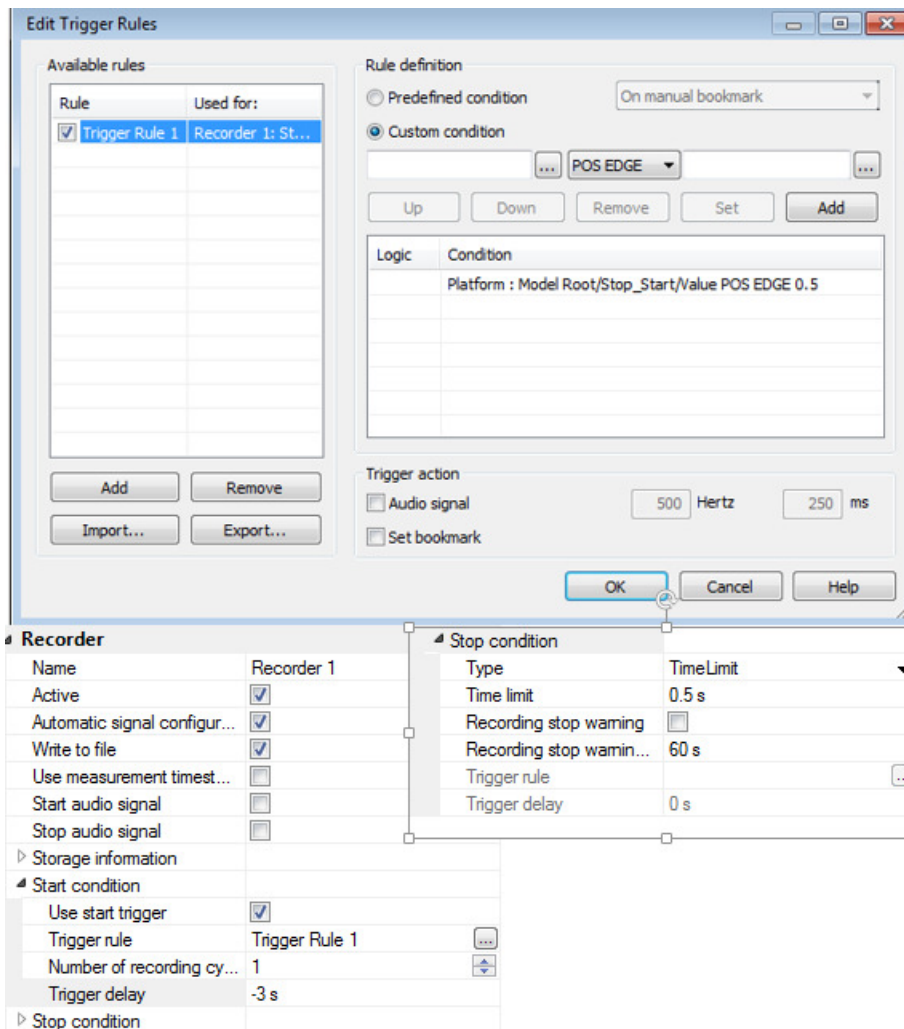
>>**Instrument Selector > Standard Instruments > Check Button.**

Drag and drop the variable *Stop\_Start* to one check box and the variable *Reset* to the other one, as shown in figure 32.

Note: Unchecking the *Start\_Stop* button while the ControlDesk is online ensures that the motor does not rotate even when a voltage is applied in the layout. While it is unchecked, however, the PEDB indicates a *motor fault*. In order to clear this fault, simply check the *Start\_stop* button and the *Reset* button. If the fault still persists (which may happen if the order of engaging the check buttons is different), push the manual *Reset* button on the PEDB. **Never push and hold the reset button!** Turn off the power to the PEDB board and back on again. If the fault light comes back on, there is a problem with your experiment. If needed, ask the TA for help.



**Figure 32: Configuration of check buttons**

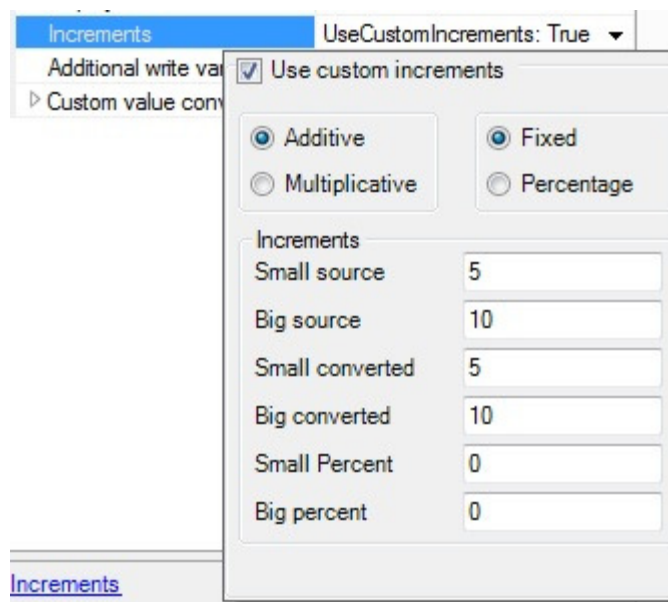


**Figure 33: Data capture**

### xiii. Setting voltage increments

In order to increment the motor voltage by steps of certain magnitude, >> **right click on the Motor\_Voltage numeric input > Instrument properties > Variables > Increments> Click on the drop down menu at the end of the line and check the *Use Custom Increments* box.**

Ensure that the radio buttons are selected for *Additive* and *Fixed*. Under *Increments*, change the *Small source* to the desired step magnitude, here 5, since you will need to increase the motor voltage by 5V in a step. Refer to figure 34.



**Figure 34: Enabling custom numeric increments**

Now, to test this set up and layout, go online on the ControlDesk and uncheck the *Stop\_Start* check box and increase the motor voltage to 5V. At this point, the motor should not rotate. Click the *Stop\_Start* check box, then the *Reset* check box. Now, the motor should start rotating with a step voltage input magnitude of 5V. Stop the motor by clicking the *Stop\_Start* check box. Note that following these sequences is important to properly collect data. **Always stop the motor using the Stop/Start box. Also, if the inverter board shows a motor fault, it can be cleared by checking and unchecking the reset box or by following directions mentioned in the note in section *xii.a*.**

To set up the measurement data collection system, use the *Recorder* (refer section *viii*). Choose the *Start\_Stop* from the *Model Root* to be the trigger variable. Set it up such that a positive edge at the threshold level of 0.5 triggers the recorder to save data. On choosing to use a start

condition, i.e., a trigger, the option to use *Trigger delay under start condition* is at your disposal. Enter a trigger delay of -3s. This negative delay ensures that the data begins to be recorded slightly before the trigger is employed and so, no data is lost between the time of trigger and time of recording. Through the stop condition, set up the length of the recording to be for 2s as described in section *viii*. Refer figure 33. Make sure that your *Duration Trigger* is set to a total of more than 5s (3s before the trigger and 2s after the trigger). You can do this in the *Measurement Configurations* tab. Under **Acquisition**, expand **Platform**; under platform, expand **HostService** and select **Duration Trigger 1**. It should be enabled with an active selection in the check box in the bottom on the same panel. Here, you can enter the duration trigger value.

#### *xiv. Motor position and velocity*

An encoder is mounted on the DC generators used in the lab. The encoder position block can be found in:

**>>Simulink Library Browser > dSPACE RTI1104 > DS1104 Master PPC> DS1104ENC\_POS\_C1.**

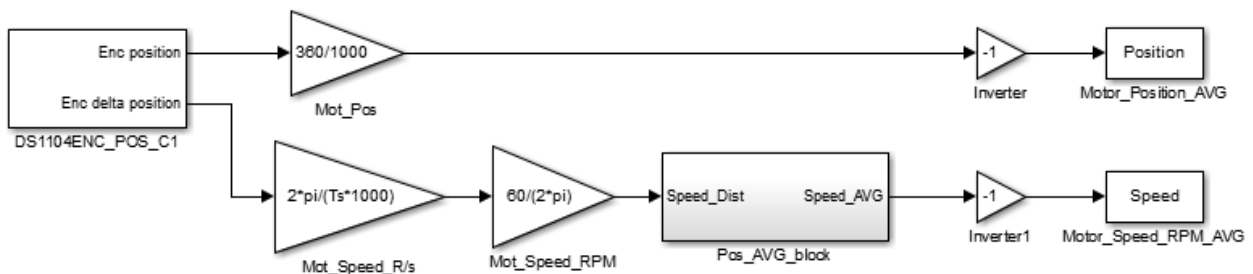
This block provides read access to the position and delta position of the two encoder interface channels. The number of these channels can be changed as described in section *i*. Also add the block Encoder Master Setup to the model. This block sets the global specifications for the channels of the encoder 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 is made with a disk having one thousand lines per-revolution, which is 0.36 degree-per-line. To convert the line count to degrees, multiply the line count by the constant 0.36. The encoder delta position output is used to compute the angular velocity of the machine in rad/sec by dividing it by the sampling time as in figure 35. The two blocks on the right, *Position* and *Speed* are “To Workspace” blocks.



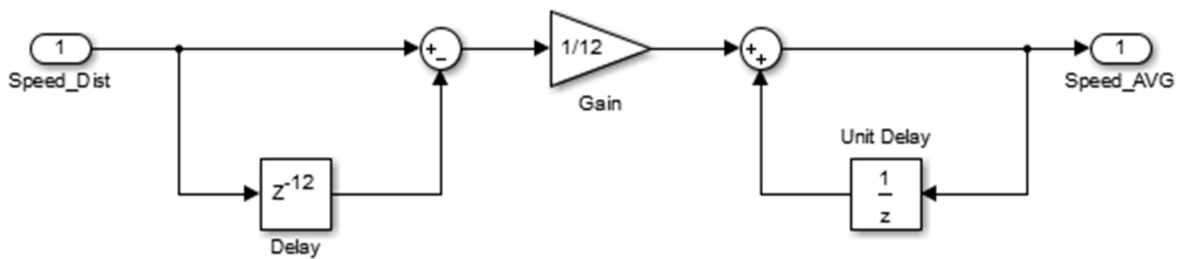
**Figure 35: Encoder position and velocity read block**



**Notes:**

- The actual encoder resolution is  $\frac{1}{4}$  the number of degrees per line, or 0.09 degrees.
- The inverters in figure 35 are in place because the encoder reads a negative position for a positive voltage applied to the motor. In order to obtain position and speed values with appropriate signs, the encoder data is inverted.

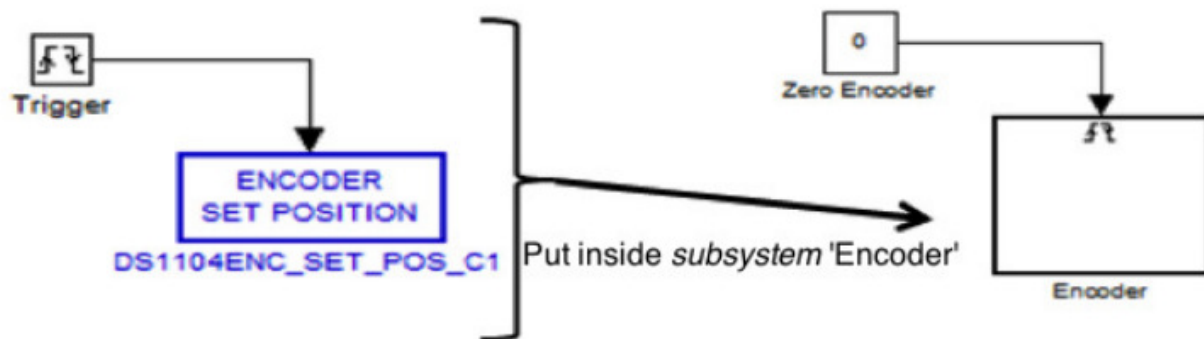
An averaging block is used to smooth the velocity reconstructed by differentiation of the quantized position measurement. This block is shown in figure 36 and it can be added to the model using a subsystem block. The sample size in this averaging block is 12 ( $n=12$ ).



**Figure 36: Averaging block**

The subsystem block can be added from **Simulink Library Browser > Simulink > Commonly Used Blocks > Subsystem**. Build the averaging block in the figure above as a subsystem and add it to encoder average position block. Make all the necessary connections and changes as shown in figure 35. 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 '-'.

**xv. Zero encoder position**



**Figure 37: Zero encoder block**

To reset the encoder position and measure the position of the motor without including the previous line count from the encoder, you can send a constant zero to the encoder set position block using a trigger, as shown in figure 37.

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 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 channel in this block matches the channel number selected for the encoder position block (C1 or C2)

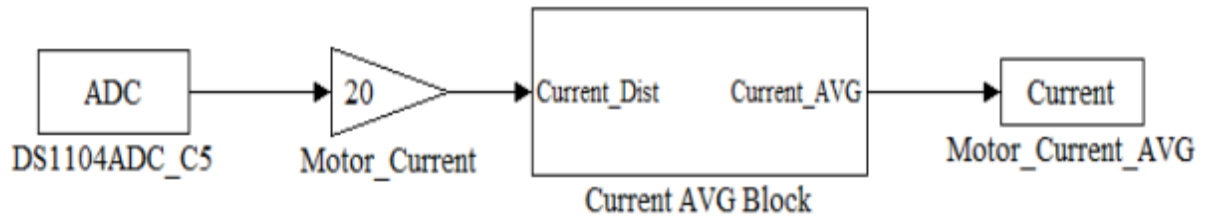
Connect the blocks and feed the trigger (outside the subsystem) with a constant zero, as shown in figure 37.

Save and build the model. Remember to stop the online calibration before building.

#### ***xvi. DC motor current measurement***

In order to measure the current in the DC machine, an analog to digital channel will be used as described in section **IV**. However, the current measured by the current sensors on the PEDB is reduced by a factor of 2, while the A/D itself a reduction ratio of 10. Therefore, the analog-to-digital signal should be multiplied by 20 to compute the actual current value. Since the DC motor is operating using phases A1 and B1 on the Inverter Board, the current through phase A1 (labeled as CURR. A1) is the same as the current through phase B1 (labeled as CURR. B1), but with the opposite sign. You can assign a different channel from what was used for the sine wave example or use the same ADCH5.

To reject the noise in the current measurement, use an averaging block similar to the one used in section **xiii** for the motor angular velocity measurement. For the current averaging filter, set the sample size to 12 by setting the gain to 1/12 and the delay length to 12.



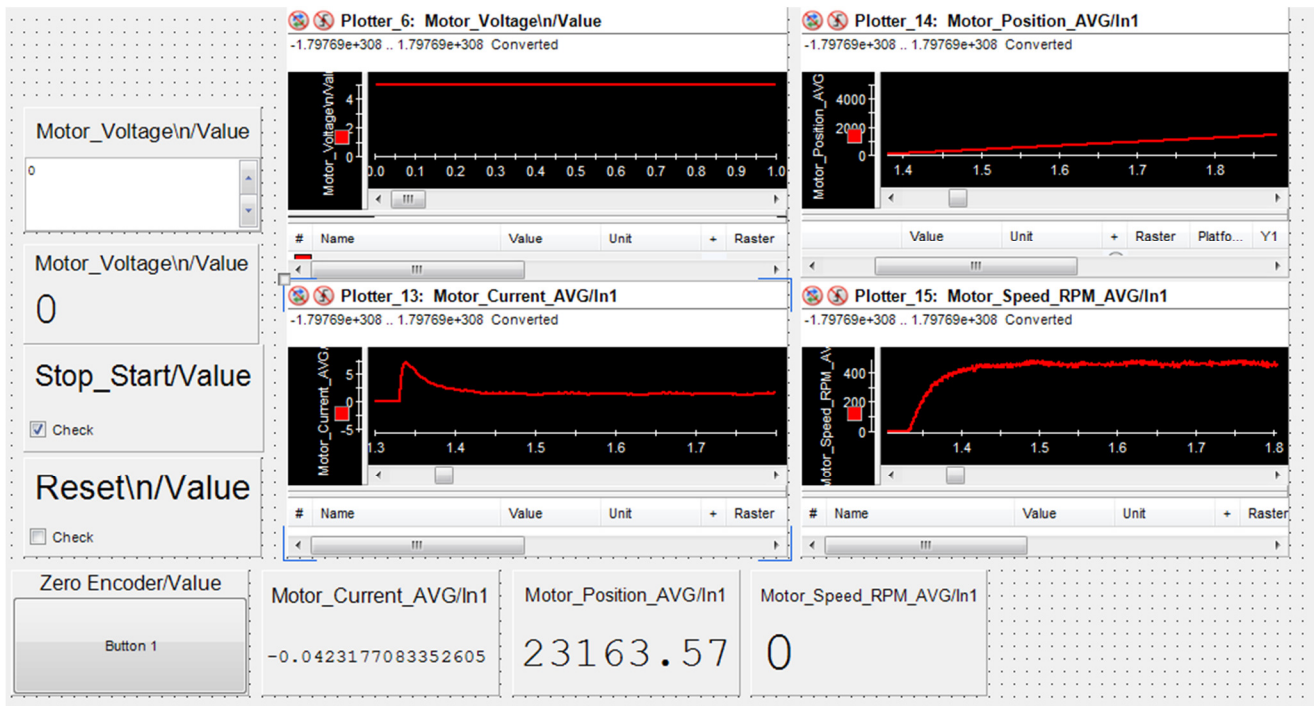
**Figure 38: Simulink model of Analog to Digital Channel (ADCH) to measure the motor current**

Ensure that the *Save format* of all the *To Workspace* blocks is *Array* and not *Timeseries*. This can be done by accessing the properties of the block by double clicking it. In the event that it is *Timeseries*, there will be no variable to map in the model root for that “To Workspace” block. Make sure to build the model again after adding all of the blocks as described in section *ii*. The overall view of Simulink model to run the DC motor can be found in the appendix, figure 41. Capture a screenshot of the Simulink model for your report.

In dSPACE, use an ON/ OFF Button, found under *Standard Instruments*, for sending a signal to zero the encoder position (line count). Modify the ON/ OFF button so that it shows only one button. This can be done by accessing the button properties. Also, add a display window to the user-interface to view the speed of the motor, and drag and drop the motor speed, named *Motor\_Speed\_RPM\_AVG*.

Connect the encoder cable to the encoder mounted on the DC machine and also to the Incremental Encoder Channel (INC) on the CLP 1104 I/O box that was selected in section *xiii*. Finally, connect CURR. A1 on the inverter board to the selected analog-to-digital channel in section *xv* via a BNC cable. Refer to figure 8 for hardware connection.

For this example, you will measure the motor current, position, and velocity. Add 3 plotters with the current, position, and velocity variables. Engage the ControlDesk by going online and start measuring. Uncheck the *Stop\_Start* and *Reset* check boxes and increase the motor voltage to 5V. Click on the *Start Triggered Recording* button. Check the *Stop\_Start* check box. The plotter arrays should capture the motor current, position, and velocity in real time. The captured values for a step input of 5V are shown in figure 39.



**Figure 39: Measurement captured**

At this point, you can make sure that the zero encoder function added to the model is functioning properly. Stop the motor by unchecking the Stop/Start check box. Click the zero encoder button (ON/OFF button) and the motor position will reset to zero. Save the data as mentioned in section *viii*. Create plots of current, position, and velocity for the captured data to submit with your report. If everything in this lab has operated correctly, the lab tutorial has been completed.

## VII. Report Requirements:

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

- Introduction and/or objectives
- Include the equipment numbers of all of the major components used
- Screenshot of the simulated sinusoid signal
- Screenshot of the sinusoid on the lab oscilloscope
- MATLAB plots of sinusoidal waveforms
- MATLAB plots of current, position, and velocity
- Simulink Models
- Conclusion (Describe what worked well and did not work well in this lab, and make suggestions for possible improvements.)

## VIII. Appendix

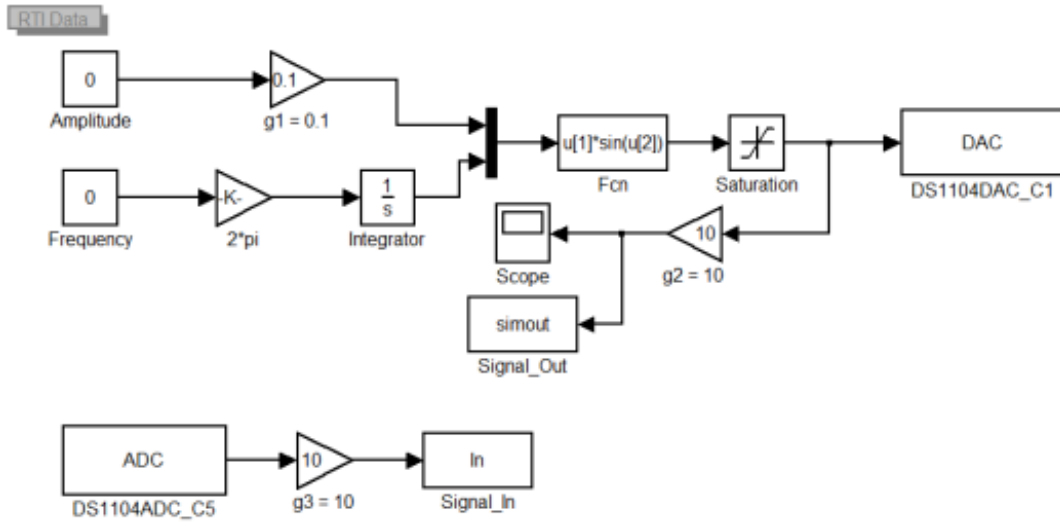


Figure 40: Overview of Simulink model to generate a sine wave

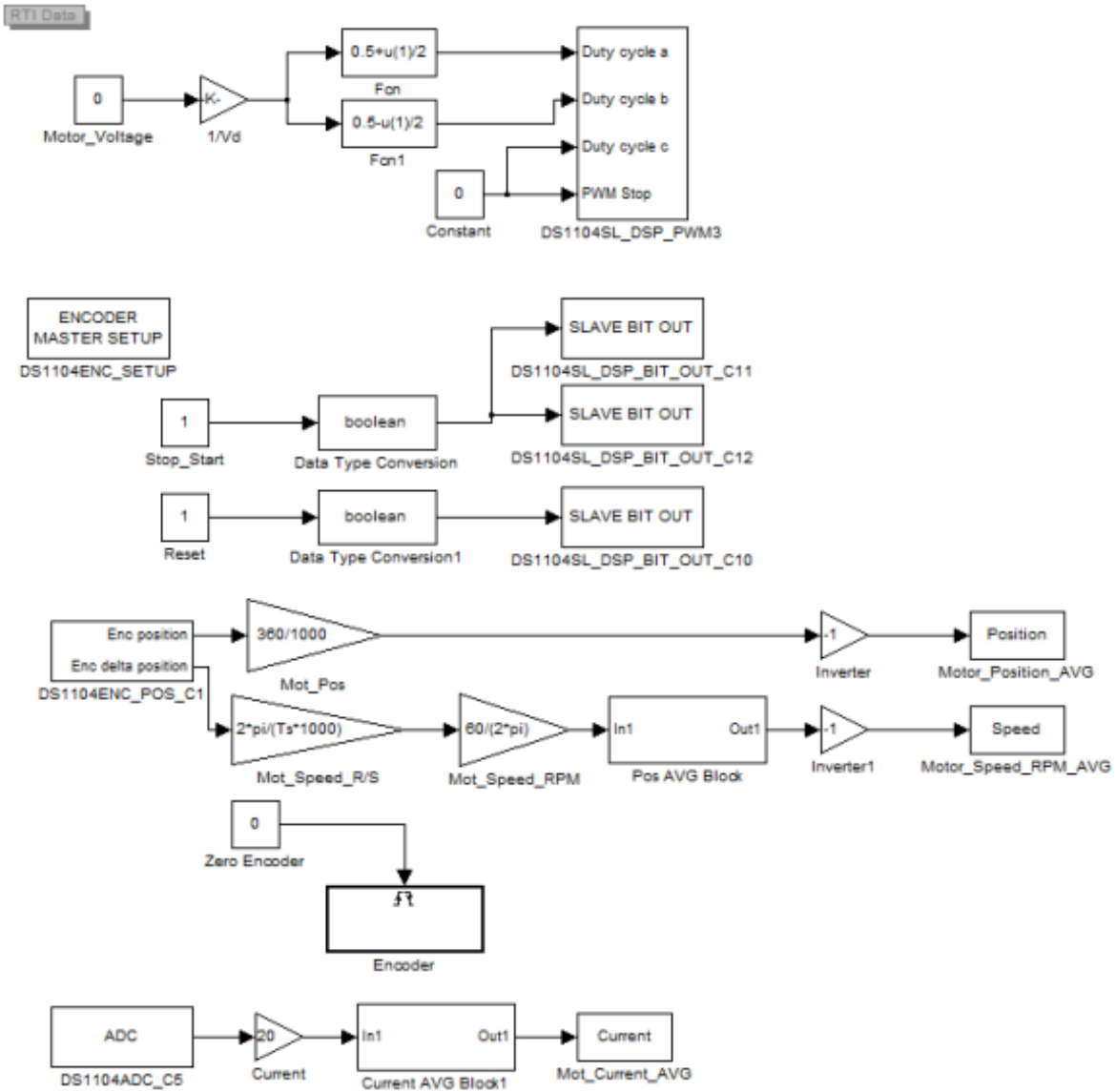


Figure 41: Overview of Simulink model to run the DC motor