DSP Based
Electric Drives Laboratory

USER MANUAL

Department of Electrical and Computer Engineering
University of Minnesota

Revised: August 1st, 2012

CONTENT

Experiment-1
Introduction to the DSP-based Electric-Drives System

1.1 Introduction

There are four major components of the DSP-based electric-drives system, which will be used to perform all the experiments in this course. They are as follows: 1) Motor coupling system, 2) Power Electronics Drive Board, 3) DSP based DS1104 R&D controller card and CP 1104 I/O board and 4) MATLAB Simulink and Control-desk. In this experiment, you will be briefly introduced to the role of above mentioned four components in the DSP-based electric-drives system. An example of speed-control of a DC-motor will be demonstrated. The Simulink file and Control-desk layout will be supplied to perform this experiment. The communication between the four components will be explained while controlling the speed of the motor. Section 1.2 details the DSP-based electric-drives system vis-à-vis the role of the four components listed above. In Section 1.3 a step-by-step procedure to run the DC motor speed-control will be performed.

1.2 DSP-based electric-drives system

Fig. 1.1 shows the block diagram of the DSP-based electric-drives system.

- *Motor coupling system:* This system contains the motor that needs to be characterized or controlled. The system has a mechanical coupling arrangement to couple two electric machines. The motor under test (MUT) or whose speed/torque needs to be controlled, could be either a DC motor or a Three-phase induction motor or a Three-phase Permanent-Magnet AC (PMAC) motor. The system also has an encoder mounted on the machine which is used to measure the speed of the MUT. This can be used for close loop feedback speed-control of the motor. The motor demands a controlled pulse-width-modulated (PWM) voltage to run at controlled speed or torque. The PWM voltage is generated by Power Electronics Drive Board (briefed next); the voltage source thus generated is connected to the motor coupling system as shown in Fig 1.2.

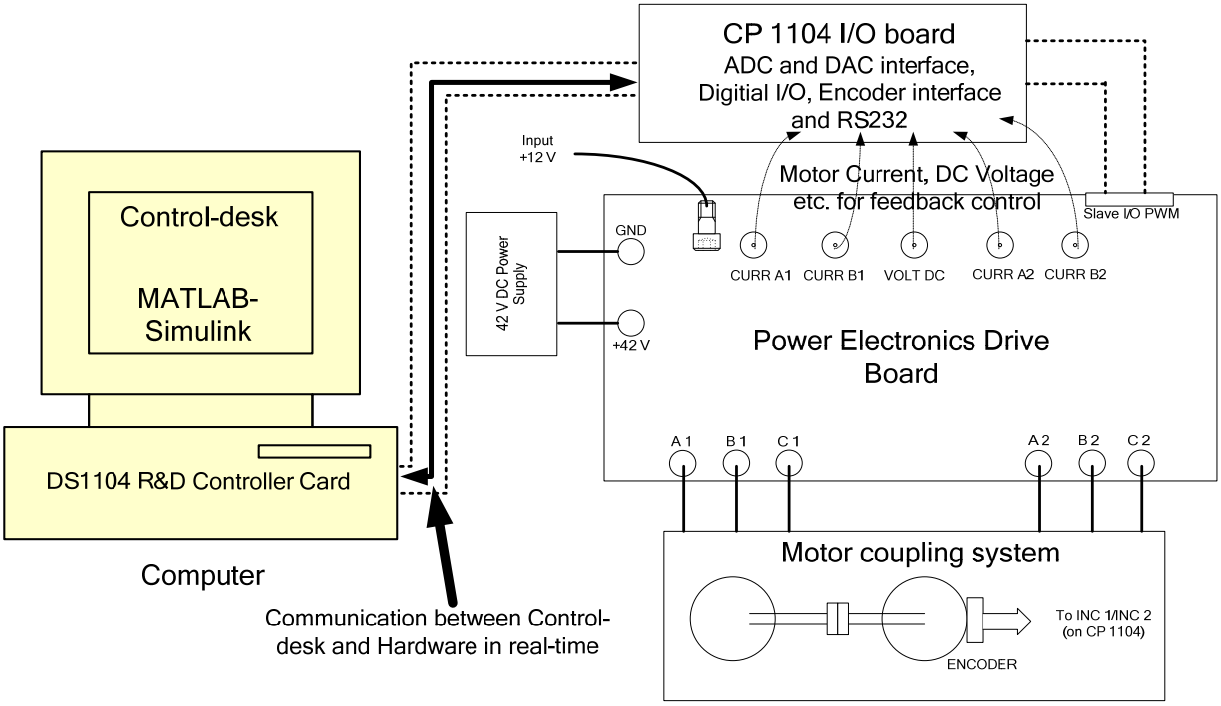


Figure 1.1: DSP-based electric-drives laboratory system

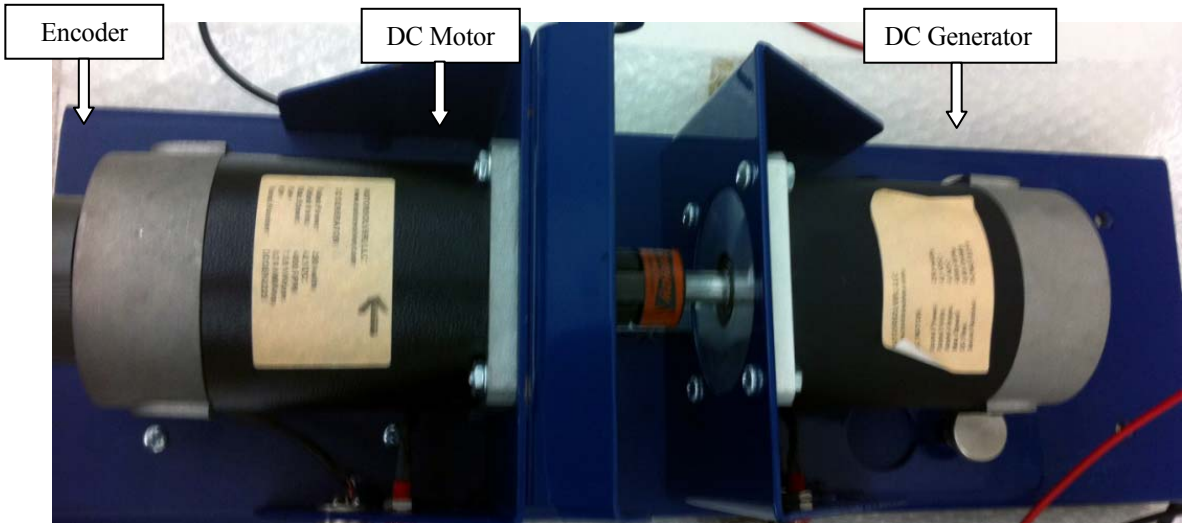


Figure 1.2: Motor Coupling System showing DC Motor, DC Generator and Encoder

- *Power Electronics Drive Board:* This board has the capability to generate two independent PWM voltage sources (A1B1C1 and A2B2C2) from a constant DC voltage source (see Fig 1 in Appendix). Hence two machines can be controlled independently for independent control of variables, at the same time. This board also provides the motor phase currents, dc-bus voltage etc. to control the motor for a desired speed or torque. To generate the controlled PWM voltage source, this board requires various digital control signals. These control signals dictate the magnitude and phase of the PWM voltage source. They are generated by the DS1104 R&D Controller board inside the computer.

- *DS1104 R&D controller Board and CP 1104 I/O board:* In each discrete-time-step, the DS1104 controller board takes some action to generate the digital control signals. The type of action is governed by what we have programmed in this board with the help of MATLAB-Simulink real-time interface. This board monitors the input (i.e. motor current, speed, voltage etc) with the help of CP1104 I/O board in each discrete-time step. Based on the inputs and the variables that need to be controlled (i.e. motor speed or torque); it takes the programmed action to generate the controlled digital signals. The CP1104 I/O board is an input-output interface board between the Power Electronics Drive Board and DS1104 controller board. It takes the motor current, dc-voltage etc. from the Power Electronics Drive Board and also, speed signal (from encoder) from motor coupling system, to the DS1104 controller board. In turn, the controlled digital signals supplied by DS1104 controller board are taken to the Power Electronics Drive Board by CP1104.

- *MATLAB Simulink and Control-desk (Programming DS1104 and control in real-time):* Simulink is a software program with which one can do model-based design such as designing a control system for a DC motor speed-control. The I/O ports of CP 1104 are accessible from inside the Simulink library browser. Creating a program in Simulink and the procedure to use the I/O port of CP 1104 will be detailed in future experiments. At this stage, let us assume that we have created a **control-system** inside the Simulink that can control the speed of a DC motor. When you build the Simulink **control-system** (pressing CTRL+B) by using real-time option, it implements the whole system inside the DSP of DS1104 board, i.e. the **control-system** that was earlier in software (Simulink) gets converted into a real-time system on hardware (DS1104). Simulink generates a *.sdf file when you build (CTRL+B) the **control-**

system. This file gives access to the variables of **control-system** (like reference speed, gain, tuning the controller etc) to separate software called Control-desk. In this software a control panel (see Fig 1.3) can be created that can change the variables of **control-system** in real time to communicate with DS1104 and hence change the reference quantities such as the speed or torque of the motor.

1.3 Demonstration of Speed Control of a DC motor

The system for the speed-control of a DC motor is shown in Fig 1.3. Note that the currA1 (i.e. phase-current of DC motor) and encoder signal (speed of DC motor) is fed back to the DS1104 board via CP 1104. The requirement of feeding back phase-current and speed of the motor will be studied in Experiment-4. For now, assume that these two quantities are required to control the speed of DC motor. Perform the following steps to run the experiment. The communication between the four components (explained in section 1.2) is detailed in each step, wherever necessary.

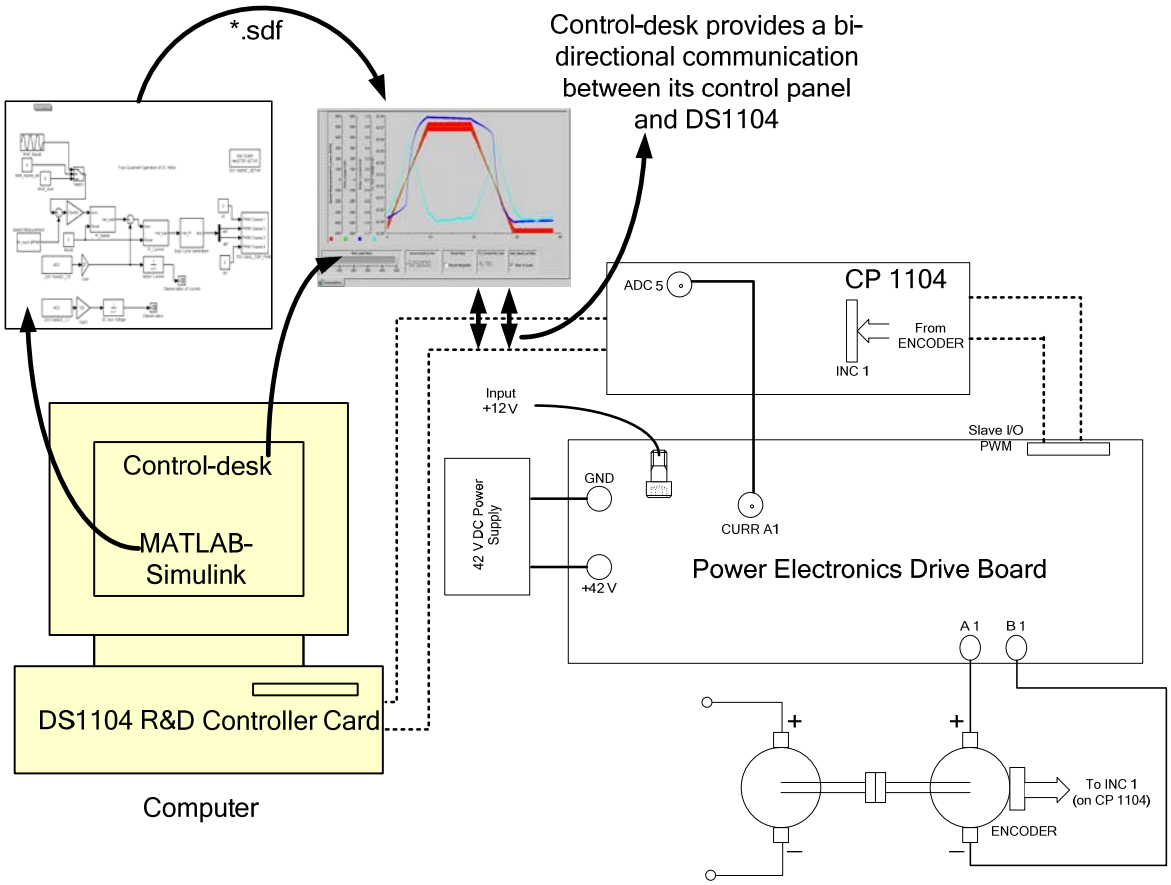


Figure 1.3: Demonstration of DC motor speed-control

- Connect the circuit as shown in Fig 1.3. You are given with files **Exp1.mdl** (Simulink **control-system** file) and **Exp1.lay** (control panel in Control-desk). Create a new folder on desktop as **Exp1** and bring these two files into that folder. Open MATLAB Simulink and set the folder **Exp1** as the path of the current working directory. Verify in the command window for the correct path (Fig 1.4). Open the Simulink file **Exp1.mdl** as shown in Fig 1.4.

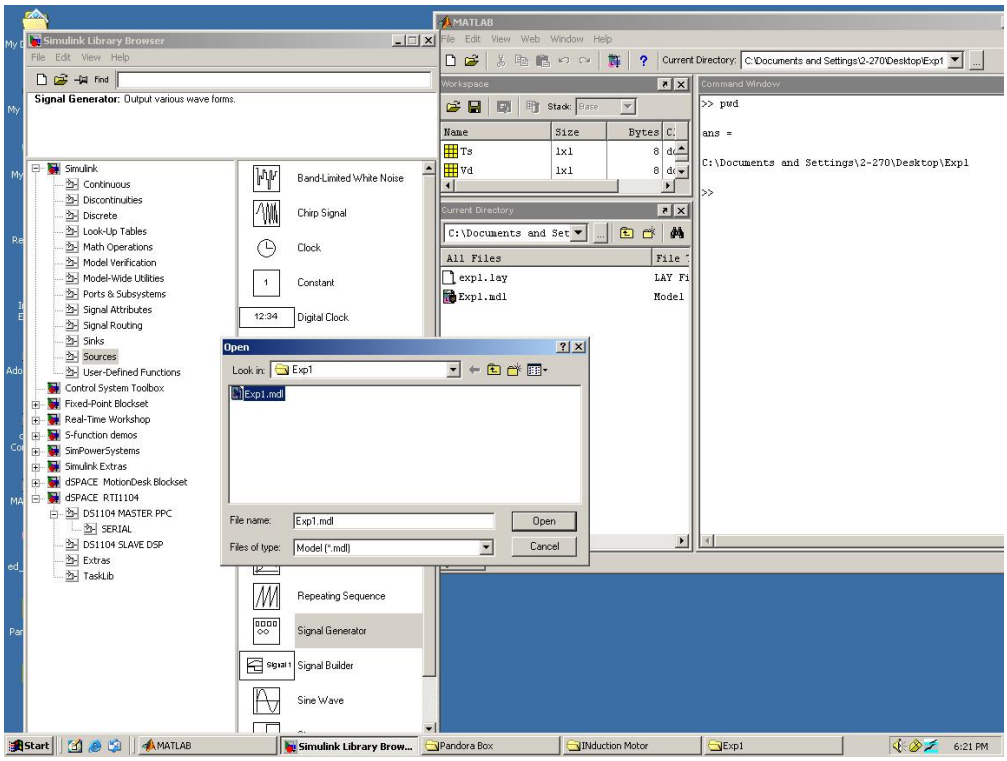


Figure 1.4: Opening the Simulink file Exp1.mdl, changing the path of current working directory

- The Simulink file **Exp1.mdl** will look as shown in Fig 1.5. Open the simulation parameters from the tools menu and set the parameters as shown in Fig 1.5. The fixed step size is the same as the discrete-step, which will be used by the DSP DS1104. This means that in every "discrete-step" the whole program (i.e. **control-system** in this case) will be executed, I/O data will be exchanged and the decision making will be done inside DS1104.

- Type Ts=1e-4 in the Matlab prompt. Press CTRL+B to build the **control-system** in real-time now. Refer to Fig 1.6, note the sequence: 1) Compilation of C-code that is generated by Simulink, which will be used to implement **control-system** in hardware DS1104, 2)

Generation of Exp1.sdf file, which will be used later on by Control-desk, to access the variables of **control-system**.

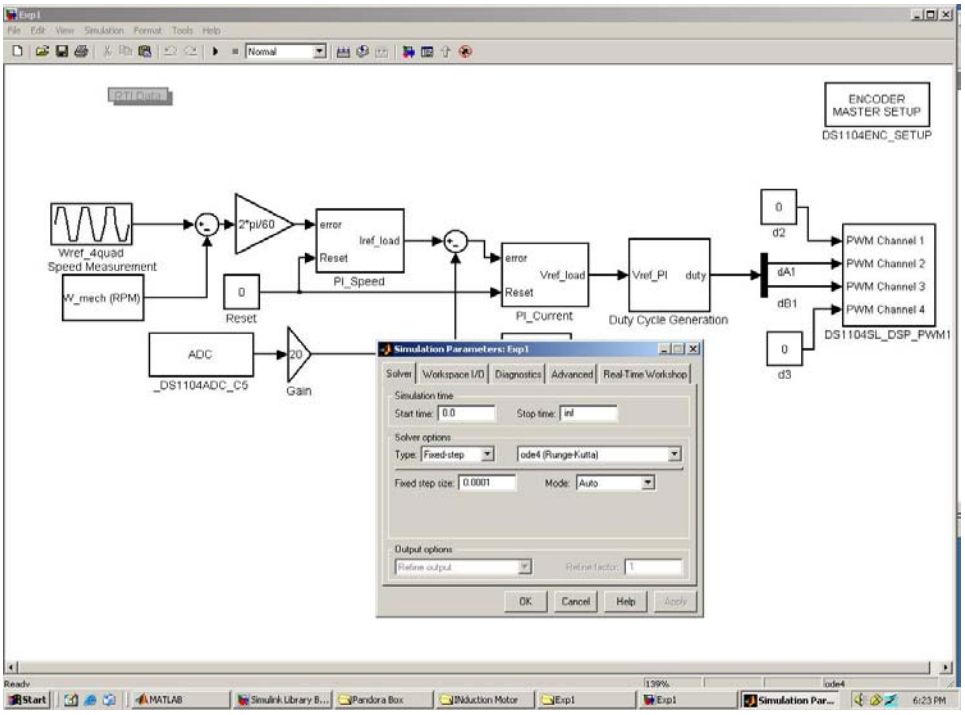


Figure 1.5: Simulink file Exp1.mdl, setting the simulation parameter

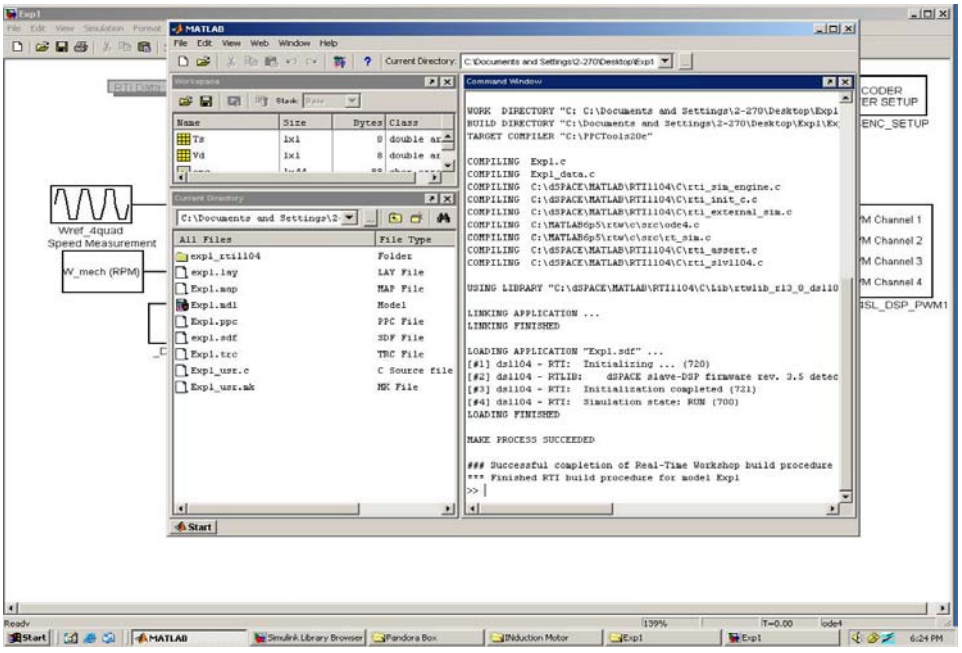


Figure 1.6: Building the Simulink program, real-time implementation

- Check the folder **Exp1**; this should contain the file **Exp1.sdf** in addition to the other system files. Leave them as such for the proper operation of Control-desk. Open Control-desk by double clicking on the **dSPACE Controldesk** icon. The icon should be located on the desktop of the computer; the opened control-desk is shown in Fig 1.7 (ignore the lower panel window shown in Fig. 1.7 at this moment, and also the pop-up window showing Exp1.lay). Click **File->Open Variable File**. A pop-up window will appear, locate the **Exp1.sdf** file in the directory **Exp1** and click **Open**. Now you should be able to see the lower panel window shown in Fig 1.7. Click **File->Open**, the pop-up window will appear again asking for layout file this time. Locate the file **Exp1.lay** as shown in Fig 1.7 and click **Open**. Click **yes** if a pop-up box opens asking for data-connection. The layout thus opened should look as shown in Fig. 1.8. At this stage the control panel of the control-desk is ready to communicate or transfer data with DS1104 via CP1104. Click **Start (PLAY)** and switch to the **Animation mode**. The motor will start running. It will rotate in the positive direction for some time and in the negative direction for some time, i.e. the direction of rotation alternates. Right click on the graph and select **edit capture** setting, and set the capture setting as shown in Fig 1.9.

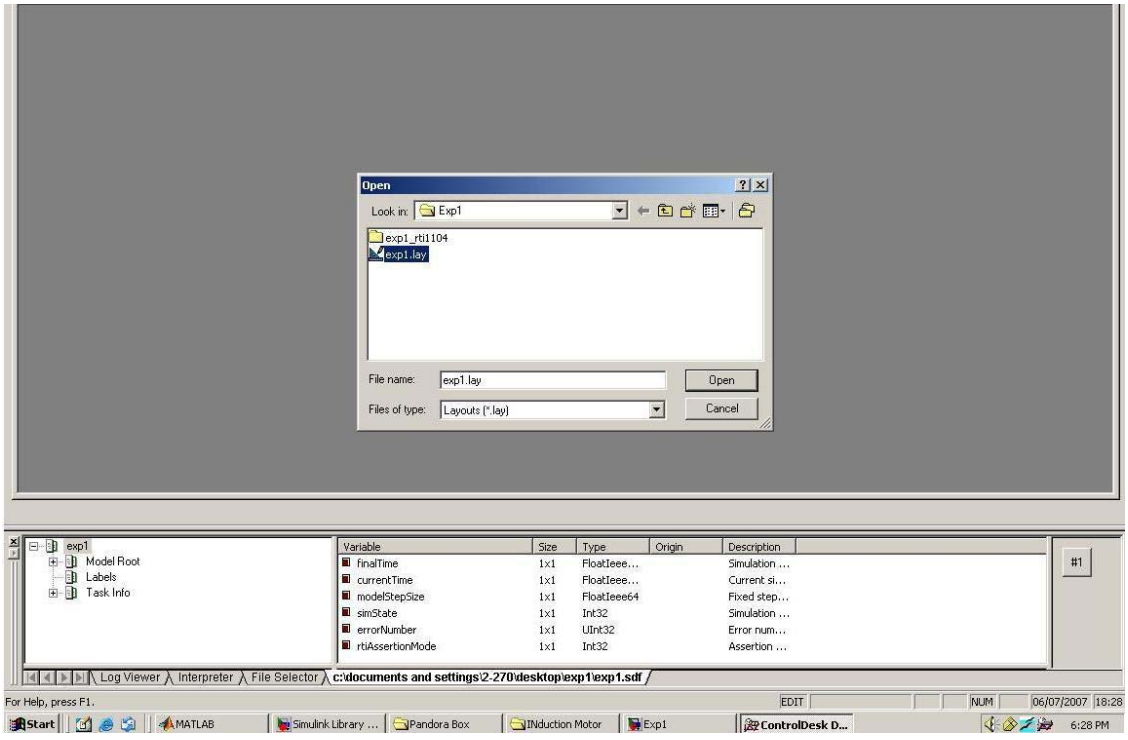


Figure 1.7: Control-desk panel, opening the layout

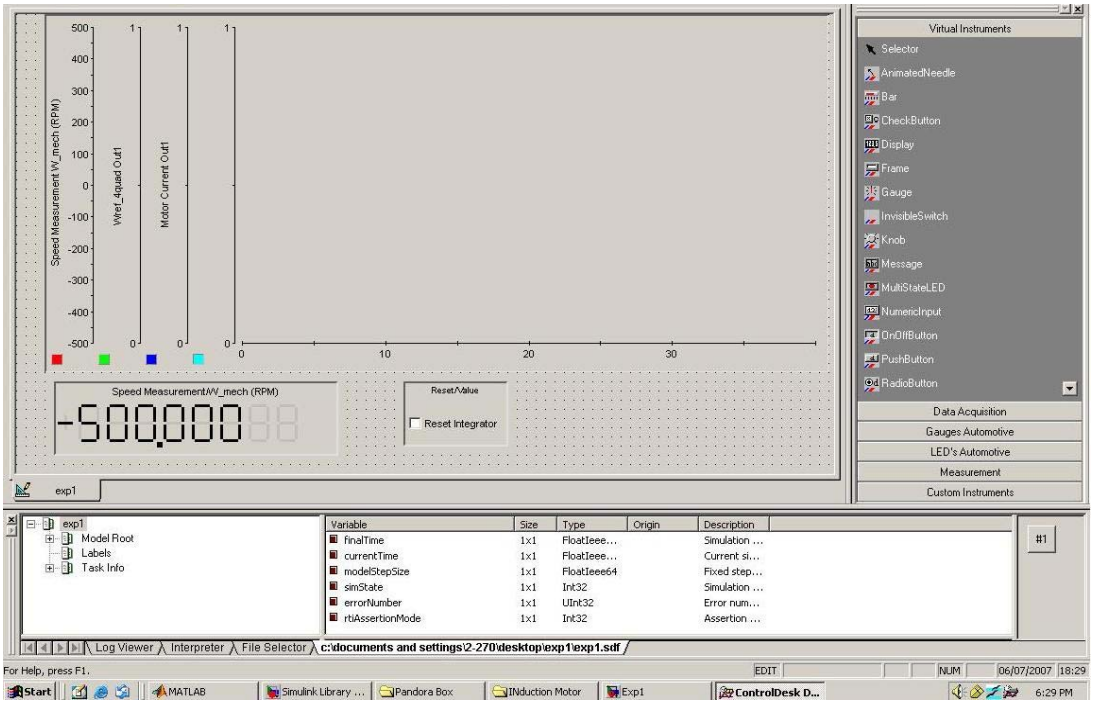


Figure 1.8: Control panel layout

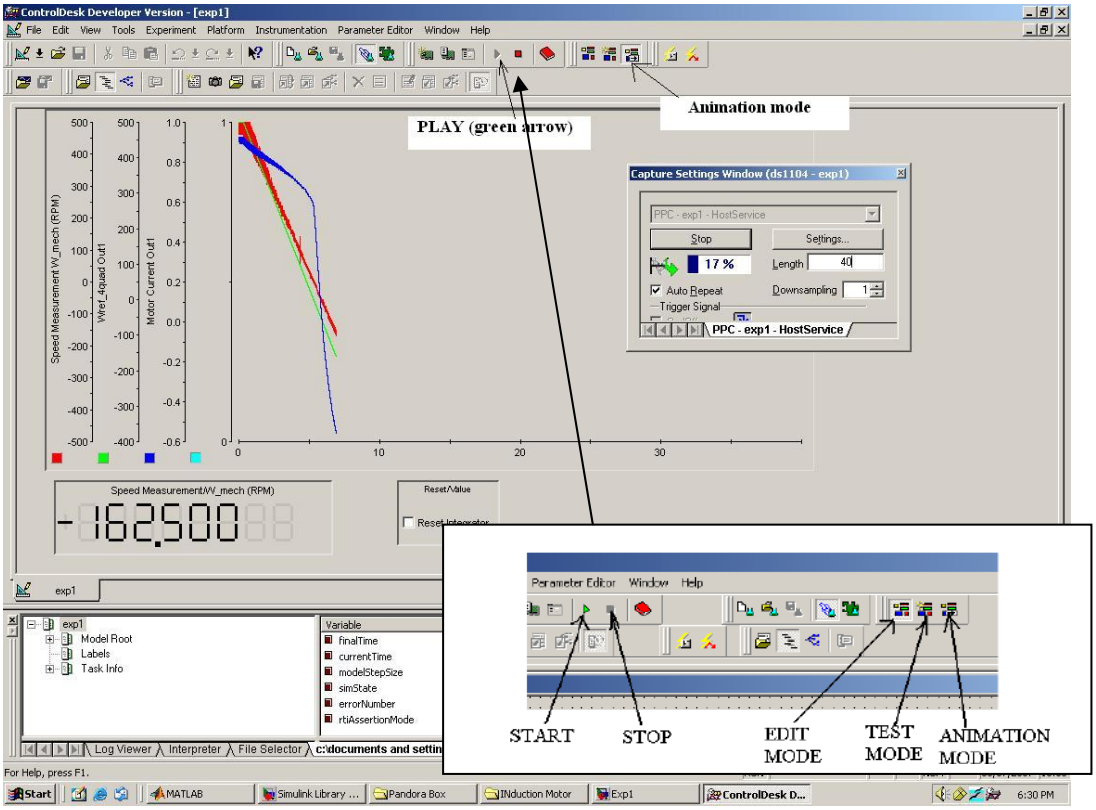


Figure 1.9: Various control buttons, starting and stopping the system

- You should be able to observe the speed of the motor on control panel (Fig. 1.9). The waveform for motor phase current, reference speed and actual speed is also shown. Press the button **Edit mode** and then press **STOP**. Take the waveform as shown in Fig 1.10. Study the explanations detailed next. The control-desk is able to access the following data with the help of **Exp1.sdf** file:

  - **Actual Speed of the motor** from **W_mech** subsystem which is inside the Simulink **control-system**. You can open the subsystem **W_mech**; you will observe how the input port INC1 (DS1104ENC_POS_C1) of **CP 1104** is utilized to read the actual speed of the motor. In the actual system, this port is connected (hardwired) to DS1104 though. But, this port is also a part of Simulink **control-system**; hence it will be listed as a variable inside the **Exp1.sdf**. Since the control-desk has access to the variables of **control-system** through **Exp1.sdf**, hence the control-desk can read the port INC1, modify the data and send it back to any of the output port of CP 1104, if necessary.

  - **Motor current, reference speed and actual speed** can be observed in the same manner, the communication among components is same as explained above.

  - Note that, in this demonstration, we are only observing the variables of **control-system** such as speed of the motor and current. It is also possible to change the variables of **control-system** in real-time from the control panel. In the future experiments, this will be done to give a reference speed command to run the motor at a desired speed. This reference speed command will be changed in real-time to change the speed of the motor.

- *Sequence of events, when **Start** button is pressed on the control panel:*

  - DS1104 will be commanded to generate the controlled digital signals as per the speed and phase-current of the motor in real-time. The information about the speed and current of the motor is available to DS1104 via CP 1104, which is connected to the Power-Electronics-Drive-Board (for current feedback) and Motor coupling system (for speed feedback).

- The controlled digital signals will be received by CP 1104 from DS1104. Digital I/O of CP 1104 is connected to the Power-Electronics-Drive-Board; hence the board will start generating the PWM voltage source.

- The motor will receive the PWM voltage at its terminals and hence start rotating. It will speed up; the current in the winding will increase. Since the speed and the current of the motor thus increased are fed back to DS1104 in real-time, the DS1104 will take the next action as per **control-system**. DS1104 will change the pattern of digital signal to change the speed of the motor such that the motor will achieve the speed as commanded in the **control-system** (block **Wref_4quad**). Note that, the speed command in **control-system** is alternating; hence the motor alternates its direction of rotation. The instantaneous motor speed and current is shown in Fig 1.10.

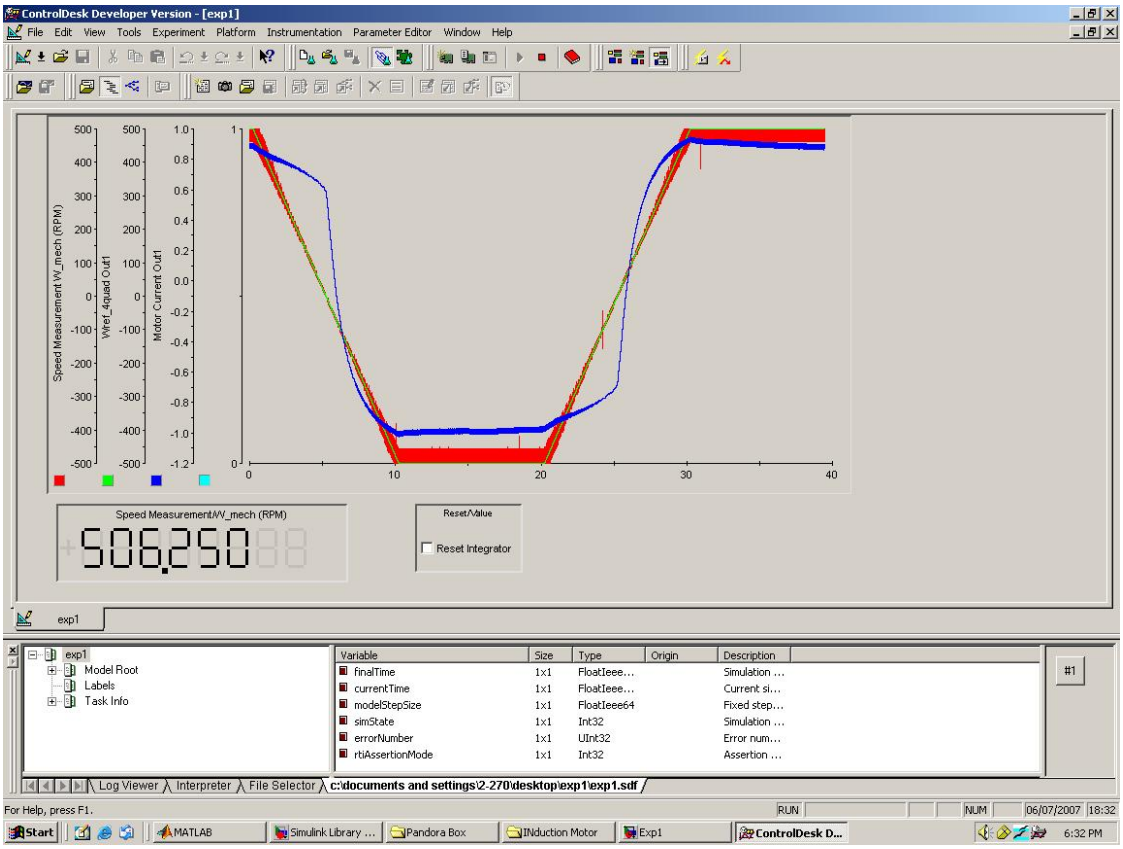


Figure 1.10: Motor actual speed (red), reference speed (green) and phase current (blue)

1.4 Lab Report and Reading assignment

- List the sequence of events i.e. the communication between the four major components when **STOP** button is pressed in the control panel.

- For this experiment draw a flowchart indicating step-by-step procedure to create a real-time model in Simulink, which is followed by controlling a DC motor from control-desk.

- Study thoroughly appendix-A, pay special attention to the Power-Electronics-Drives-Board features. Make a note of voltage and current scaling in the drive board.

- Draw a block diagram of Power-Electronics-Drives-Board indicating the inputs (like power supplies, digital input, resets etc) and outputs (like PWM voltage, motor current, dc-bus voltage etc).

Experiment – 2
Simulation and Real-time Implementation of a Switch-mode DC Converter

2.1 Introduction

In the previous experiment, a demonstration highlighting various components of the electric drives laboratory was performed. Real-time simulation file (*.mdl) and a Control-desk layout file (*.lay) were provided.

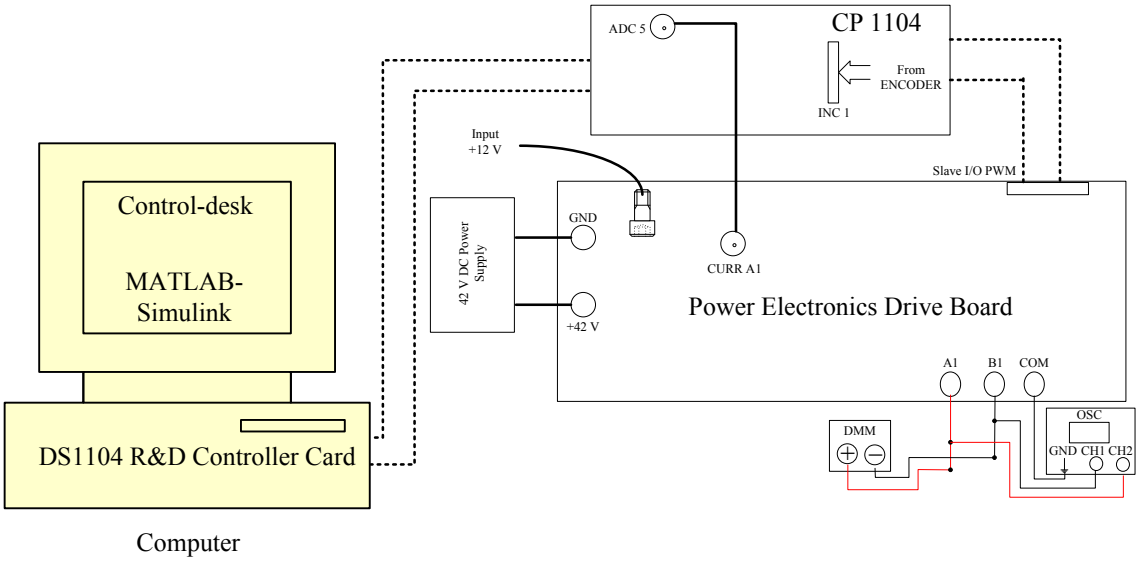


Figure 2.1: System Connections

In this experiment, a Simulink model (*.mdl) of a DC switch-mode power converter will be built. After verifying the simulation results with Simulink model, the model will be modified to control the output voltage of the converter in real-time. A control panel using dSPACE Control-desk will be designed (*.lay) that will serve as a user-interface to regulate the output voltage of the converter.

In section 2.2, theoretical background to implement DC switch-mode power converter in Simulink is briefed. Section 2.3 gives step-by-step instructions to simulate the converter in Simulink. In

section 2.4, the Simulink model is modified for real-time implementation and step-by-step instructions to design the control panel using Control-desk are given.

2.2 Theoretical Background of DC Switch-mode converter

2.2.1 *Switching Power-Pole Building Block*

The switching power-pole building block has been explained in Section 1-6-1 of [1]. Depending on the position of the bi-positional switch, the output pole-voltage v_A is either V_{in} or 0. The output pole-voltage of the power-pole is a switching waveform whose value alternates between V_{in} and 0 depending on the pole switching function q_A. The average output voltage $\bar{v}_A$ of the power-pole can be controlled by controlling the pulse width of the pole switching function q_A.

$$\bar{v}_A = \frac{T_{up}}{T_s} V_{in} = d_A V_{in} \tag{1}$$

$T_{up} \rightarrow$ pulse – width of q_A

$T_s \rightarrow$ switching time period

2.2.2 *PWM of the Switching Power-Pole*

As seen in section 2.2.1, in order to control the average output voltage of the switching power-pole, the pulse width of the pole switching function q_A needs to be controlled. This is achieved using a technique called Pulse-Width Modulation (PWM). This technique is explained in section 12-2-1 of [1]. To obtain the switching function q_A, a control voltage $v_{cntrl,a}$ is compared with a triangular waveform v_{tri} of time period T_s. Switching signal $q_A = 1$ if $v_{cntrl,a} > v_{tri}$; 0 otherwise. As in [1],

$$v_{cntrl,a} = d_a \hat{V}_{tri} \tag{2}$$

Using equations (1) & (2) and assuming $\hat{V}_{tri} = 1V$,

$$v_{cntrl,a} = \frac{\bar{v}_{aN}}{V_d} \tag{3}$$

Where $\bar{v}_{aN} = \bar{v}_A$ = average pole-output voltage with respect to negative DC-bus voltage.

2.2.3 *Two-pole DC Converter*

The two-pole switch-mode DC converter utilizes two switching power-poles as described in the previous sections. The output voltage of the two-pole converter is the difference between the individual pole-voltages of the two switching power-poles. The average output voltage $\bar{v}_o = \bar{v}_{ab}$ can range from $-V_d$ to $+V_d$ depending on the individual average pole-voltages.

$$\bar{v}_o = \bar{v}_{ab} = \bar{v}_{aN} - \bar{v}_{bN} \tag{4}$$

To achieve both positive and negative values of $\bar{v}_o$, a common-mode voltage equal in magnitude to $V_d / 2$ is injected in the individual pole-voltages. The pole-voltages are then given by:

$$\bar{v}_{aN} = \frac{V_d}{2} + \frac{\bar{v}_o}{2} \tag{5}$$

$$\bar{v}_{bN} = \frac{V_d}{2} - \frac{\bar{v}_o}{2} \tag{6}$$

Solving equation (1) to (6),

$$d_a = v_{cntrl,a} = \frac{1}{2} + \frac{1}{2}\frac{\bar{v}_o}{V_d} \tag{7}$$

$$d_b = v_{cntrl,b} = \frac{1}{2} - \frac{1}{2}\frac{\bar{v}_o}{V_d} \tag{8}$$

The above equations will be implemented in Simulink.

2.3 Simulation of DC Switch-mode Converter

2.3.1 *Triangular waveform*

As explained in section 2.2.2, to modulate the pulse-width of the switching signal in a power converter, a control voltage has to be compared with a triangular waveform signal. This triangular waveform will be generated in Simulink, using the **Repeating Sequence block**.

- Create a new directory for the experiment (say *Expt02*).
- Start Matlab and set the path to this directory.
- Type “simulink” at the command prompt and create a new model from **File>New model**.
- Access the Simulink library by clicking **View > Library Browser.**
- In the Library Browser expand the **Simulink** tree and click on **Sources.** Drag and drop the **Repeating Sequence** block into your model.
- Simulink blocks usually have properties that can be modified by double-clicking on the blocks. Double click on the **Repeating Sequence** block and edit the properties as:
    - Time values: [0 0.5/fsw 1/fsw]
    - Output values: [0 1 0]
- Where “fsw” is the switching frequency (10 kHz) set as a global variable in the Matlab prompt. Type >>fsw = 10000
- Add a **Scope** to the model from **Simulink** → **Sinks.**
- Connect the output of **Repeating Sequence** block to the input of the **Scope.**
- The simulation model is now ready. However before running the simulation parameters need to be changed. Go to **Simulation** menu and select **Configuration Parameters.** Set the parameters to the following values:
    - Stop time : 0.002
    - Fixed step size : 1e-6
    - Solver Options: fixed step, ode1 (Euler)
- Run the simulation by clicking on the triangular button on the top. Double click on the scope

after the simulation finishes. The result should look similar to the one shown in Fig. 2.2.

2.3.2 *Duty Ratio and Switching Function*

For a desired average output pole-voltage $\bar{v}_{aN}$, the control voltage $\bar{v}_{cnrl,a}$ is given by equation (3). Equation (3) is implemented in Simulink and the control voltage thus generated is compared with the triangular signal generated in the last part.

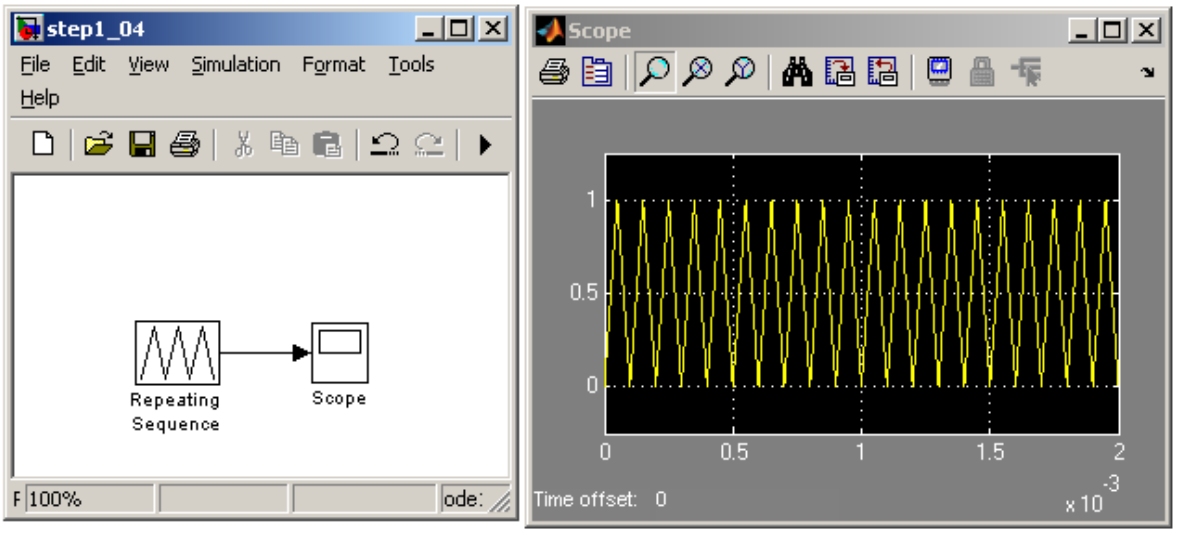


Figure 2.2: Triangular Waveform with 10 kHz frequency

The desired voltage $\bar{v}_{aN}$ is set by a **Constant** block with value one, and can be varied with a **Slider gain** from '0' to the maximum DC-bus voltage V_d (V_d = 42V in the model). The control voltage is generated by dividing $\bar{v}_{aN}$ by V_d. This is done by using a **Gain** block (of value $1/V_d$) at the output of the **Slider gain.**

Comparison of the triangular signal and the control voltage is done using a **Relay** block. The triangular signal is subtracted from the control voltage. The **Relay** block output is then set to '**1**', when the difference is positive and '**0**' when the difference is negative.

To create the model, follow the steps below:

- Open Simulink and create a new model.
- Copy and paste the model of triangular waveform generator from section 2.3.1 (Fig 2.2).

- Add these parts to the model
  - **Constant** block from **Simulink** → **Sources.**
  - **Slider Gain** from **Simulink** → **Math Operations.**
  - **Gain** from **Simulink** → **Math Operations.**
  - **Sum** from **Simulink** → **Math Operations.**
  - **Relay** from **Simulink** → **Discontinuities.**
- Now change the properties of these blocks as follows:
  - Change the **Slider Gain** limits as shown in Fig. 2.3.
  - Change the value of **Gain** to $1/V_d$ (V_d will be set to 42V from the command prompt later).
  - Change **Sum** block signs to $|-+$.

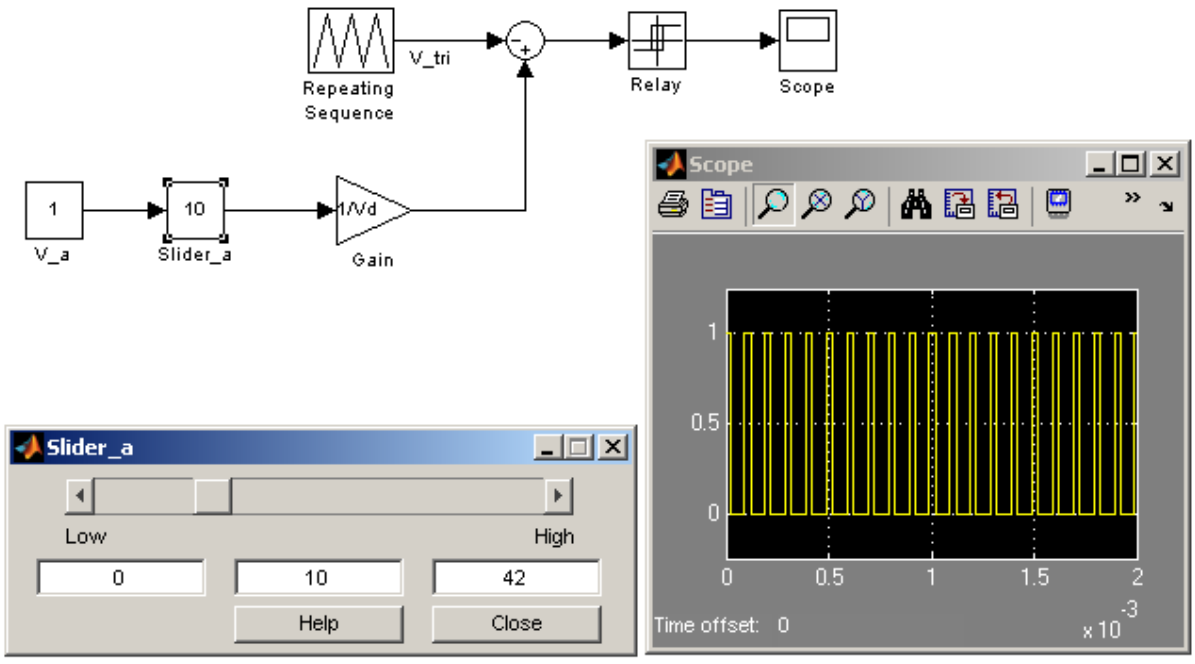


Figure 2.3: Switching Function generation for single pole converter

- Rename the blocks and connect them as shown in Fig. 2.3.
- In the Matlab prompt, type: fsw = 10000, Vd= 42.
- Set the simulation parameters as in section 2.3.1 and save the model.
- Run the simulation and save the waveform for the switching function. (Fig. 2.3)

2.3.3 *Two-pole Converter Model*

Equations (7) and (8) describe the control voltages of the two poles A & B depending on the desired output voltage $\overline{v}_o = \overline{v}_{ab}$. These equations will be implemented in Simulink (Fig. 2.4). Also, the switching power-poles will be modeled using a **Switch.** The **Relay** blocks provide the switching functions for the poles q_a and q_b. Depending on the value of the switching function, the **Switch** outputs the pole-voltage as follows:

For $q_a = 1$, switch output (Pole A) $= v_{aN} = V_d$

For $q_a = 0$, switch output (Pole A) $= v_{aN} = 0$

Create the Simulink model as shown in Fig. 2.5. The **Switch** block can be found in **Simulink** → **Signal Routing.** Change the threshold voltage of the switch to 0.5 (Fig 2.4), or to any number greater than 0 but less than 1. Can you tell why?

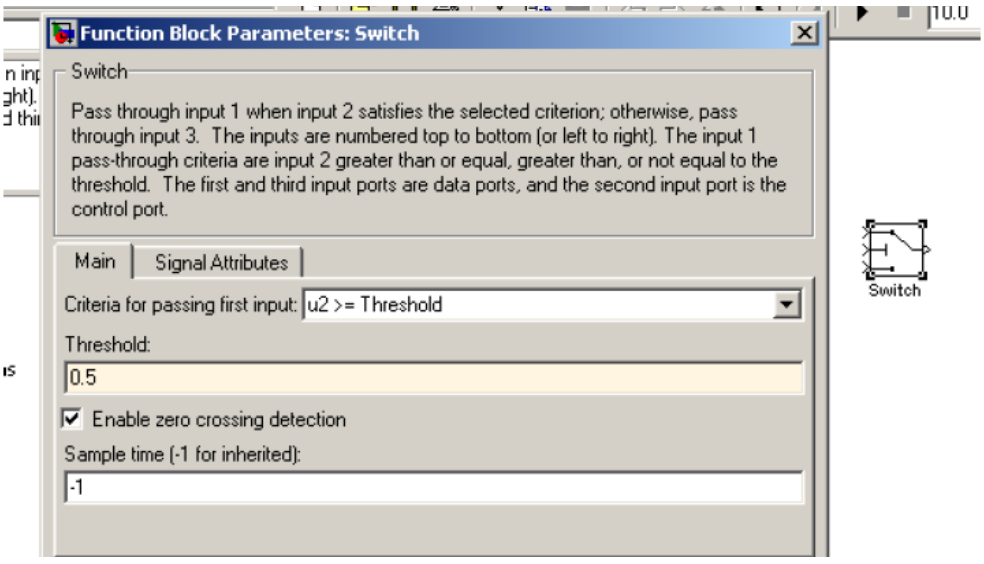


Figure 2.4: Settings for switch block

- Set the simulation parameters and values of fsw and Vd as in section 2.3.2. Run the simulation.

- Collect the following results:

  - Switching function *q* (*t*) for pole 'A' of the two-pole converter.

  - Simulation results of a two pole converter model for two different values of **V_ab**, one positive and one negative.

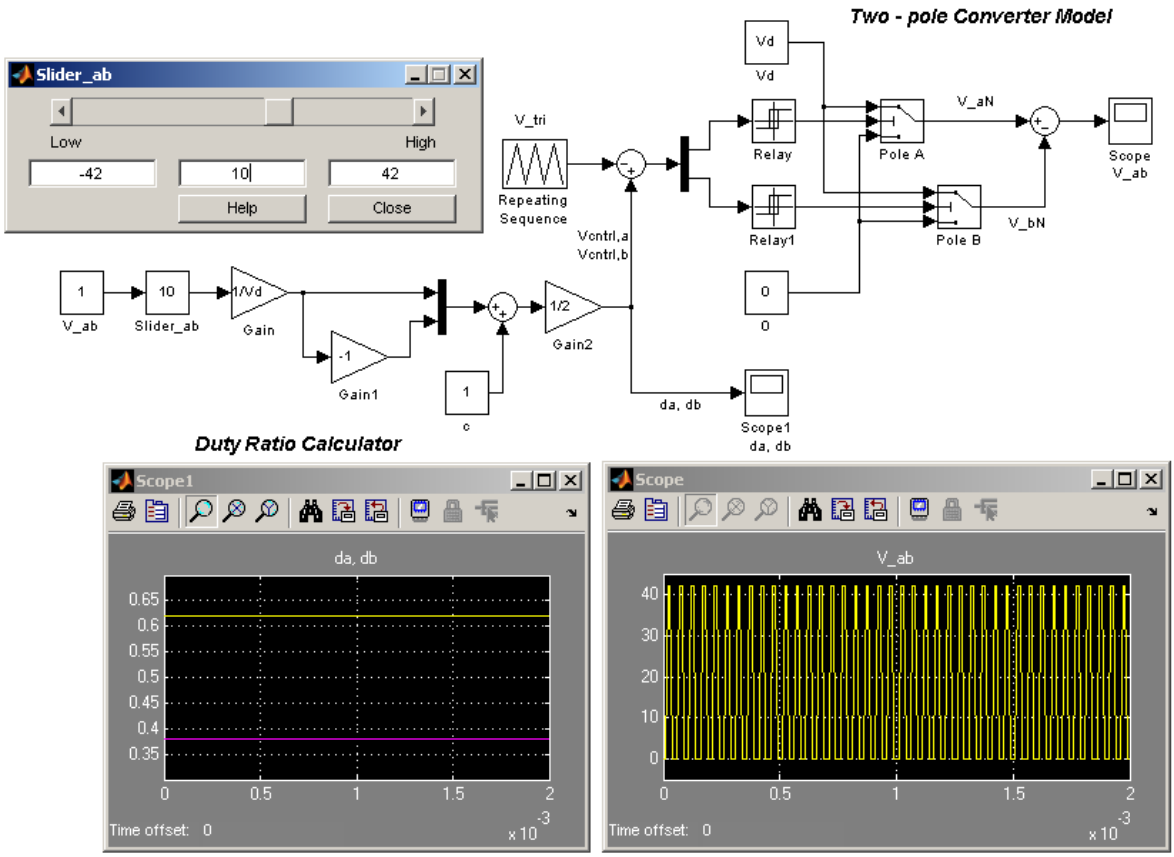


Figure 2.5: Two Pole Switch-Mode Converter Model in Simulink

2.4 Real-time Implementation of DC Switch-mode Converter

Having simulated the two-pole DC switch-mode converter, it will now be implemented in real-time on DS1104. This means that the converter will now be implemented in hardware and its output voltage amplitude will be controlled in real-time using an interface (made possible by the use of dSPACE Control-desk). As explained in experiment-1, real-time implementation involves exchange of signals between the dSPACE Control-desk interface, DS1104 and the Power-

Electronics-Drives-Board. In this experiment, the output voltage reference will be set from the Control-desk interface. The duty ratios for the two poles will be calculated from this output voltage reference inside DS1104. PWM will be internally performed and the switching signals thus generated will be sent to the power electronics drives board through the CP1104 I/O interface. Make connections as shown in Fig. 2.1.

dSPACE provides a block called **DS1104SL_DSP_PWM3**, which embeds the triangular waveform generator and the comparator for all converter poles. The inputs for **DS1104SL_DSP_PWM3** are the duty-ratios for the poles. In Fig. 2.5, the lower part of the model is called the ***Duty Ratio Calculator.*** This part of the model will again be used in the real-time model to generate the pole duty ratios. The triangular wave generator and comparison using relays will be replaced by **DS1104SL_DSP_PWM3,** as these functions are internal to the block. Two legs of the drives board (refer appendix 'A') will replace the two poles (modeled using the **Switches** in Simulink).

- Create the real-time model as shown in Fig. 2.6. Use the ***Duty Ratio Calculator*** from section 2.3.3.

- For the **DS1104SL_DSP_PWM3** block, set the switching frequency as 10000 Hz and the dead-band to '0'. (**DS1104SL_DSP_PWM3** from dSPACE RTI1104 →Slave DSP)

- Make the following changes in the Configuration Parameters.
  - Simulation>Configuration Parameters
    Change the stop-time to ***inf,*** fixed step size to 0.0001
  - Simulation>Configuration Parameters>Real time workshop
    Set the 'system target file' to rti1104.tlc
  - Simulation> Configuration Parameters >Optimization>
    uncheck 'Block Reduction'

- Set Vd = 42V in the Matlab command prompt.

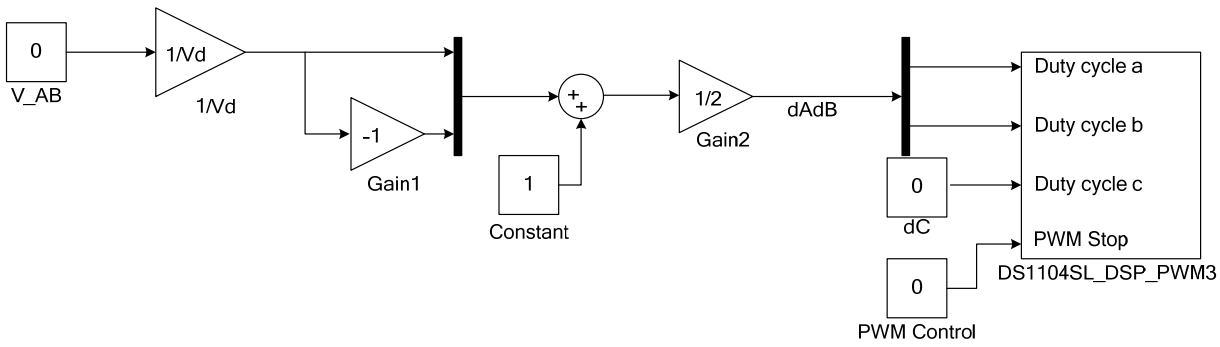


Figure 2.6: Control of a two-pole switch-mode converter in real-time

Once the real-time model is ready, it can be implemented on the DSP of DS1104 by building the model. As explained in experiment-1 building the model will broadly cause:

1. Compilation of C-code (generated by Simulink) and its hardware implementation on DS1104.

2. Generation of a variable file (with extension .sdf) that allows access to the variables and signals in the real-time Simulink model.

    - Build the Simulink model by pressing (CTRL+B). Observe the sequence of events in the Matlab command window.

    - Once the real-time model is successfully built, open Control Desk (icon on PC Desktop).

    - Using the File menu, create a **New Experiment** and save it in the same working root as the real-time Simulink model. Create a **New Layout** using the File menu again. Two new windows will appear in the Control Desk workspace. The one called **Layout1** will contain the instruments used for managing the experiment. The second window is a library, which will let us drag and drop the necessary controls for the experiment into the **Layout**. You can also open the existing **exp2.lay** file from Lab2_Summer2011 folder.

    - Now, select **File>Open Variable File.** Browse to the directory containing the real-time Simulink model. Open the .sdf file (e.g. For Simulink model named twopole.mdl, the variable file will be twopole.sdf).

- After opening the variable file, notice that a new tab in the lower window called **Variable Manager** appears below the layout (Fig 2.7). The variables of the real-time Simulink model file are under the tree **Model Root.** Expand **Model Root,** observe the variables and relate them with the real-time Simulink model.

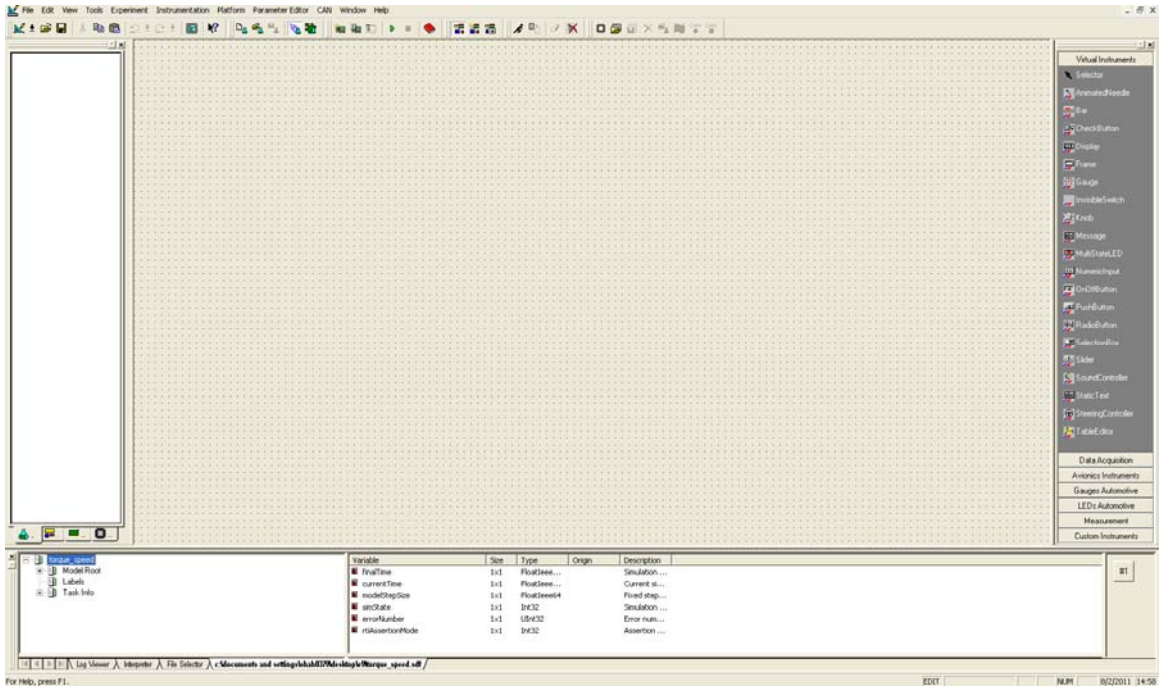


Figure 2.7: New Layout Window for Instrumentation and Control

- Now a user-interface that allows us to change input variables & system parameters (in real-time) and also observe signals will be created. The input variable in this experiment is the output voltage of the switch-mode converter. The duty ratios generated by the ***Duty Ratio Calculator*** will be the signals that will be observed in the layout. The actual pole voltages will be observed directly from the power electronics drives board using an oscilloscope.

- In order to change the reference output voltage and observe the duty ratios, suitable parts need to be added to the layout. These parts are available in the window to the right of the layout. The output voltage reference **V_AB** will be set using a **Slider** and a **Numerical Input.** Both these parts are found under **Virtual Instruments.** Click and draw these parts in the layout. The duty ratios will be observed in a **Plotter** available in **Data Acquisition.** Select **Plotter** and draw it in the layout.

- Now appropriate variables will be assigned to the parts. Under **Model Root,** locate **V_AB** and select it. It will have a parameter called **Value** (right side panel) which corresponds to the value of the **Constant** block **V_AB** in the real-time Simulink model. Drag and drop **V_AB/Value** into the **Slider** and also on **Numerical Input** one-by-one. Now, the value of **V_AB** can be changed in real-time using these two parts. Similarly, to observe the duty ratios in real-time, assign the two outputs **(Out1** and **Out2)** of the **De-mux** (the one following the **Gain2** block) to the plotter. The experiment is now ready; it should look as shown in Fig 2.8. Start the experiment by clicking the Start button and select the animation mode (Fig 2.8)

- Turn the power supply ON and observe the pole-voltages on the oscilloscope. Vary the output voltage reference **(V_AB)** using the **Slider** or the **Numerical Input.** Observe the changing duty-ratios and pulse-widths of the pole-voltages.

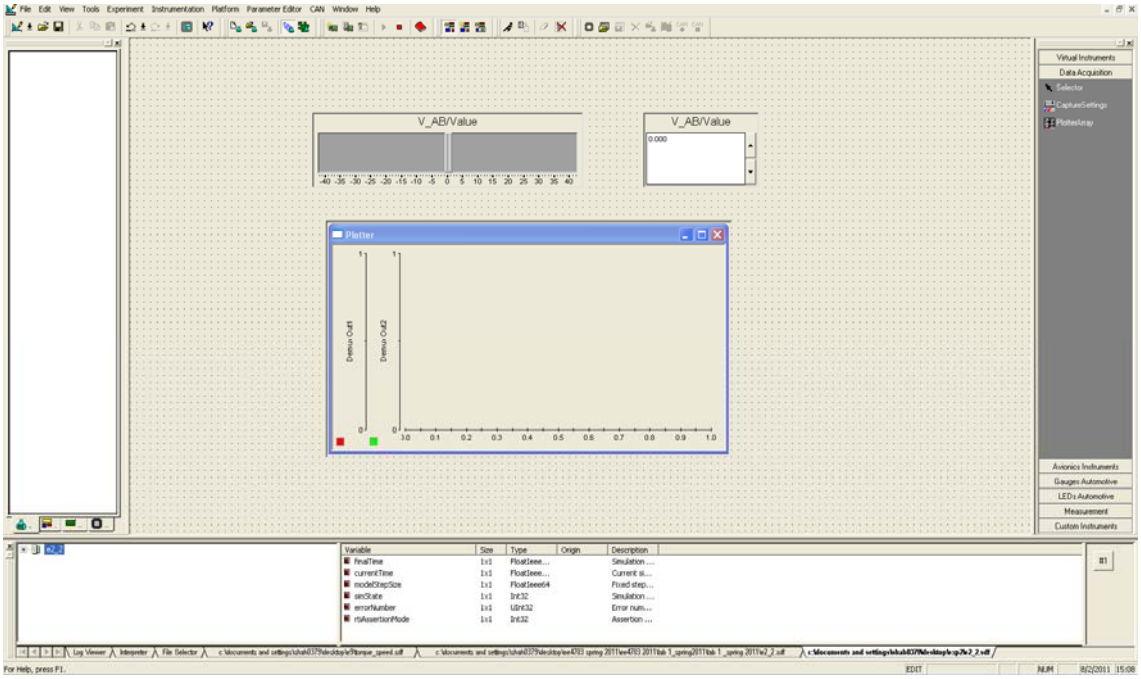


Figure 2.8: Control Desk layout for Switchmode DC Converter

2.5 Lab Report

- Include the following results in your report

1) Section 2.3.2: Run the simulation and save the waveform for the switching function. (Fig. 2.3)
2) Section 2.3.3:
   a) Duty ratios da and db for the two-pole converter.

   b) Simulation results of two-pole converter model for two different values of **V_ab**, one positive and one negative.

- Record the output voltage waveform on the oscilloscope for $\mathbf{V_{A1}}$ and $\mathbf{V_{B1}}$ w.r.t. COM (two probes will be used) and obtain by subtraction on the scope the values of $\mathbf{V_{A1B1}}$ set in section 2.4.

- Record the corresponding duty ratio waveforms for the above values.

- Measure the output voltage frequency and comment on the result obtained (**Hint**: relate the frequency set in the PWM block to the frequency of the voltage observed on the oscilloscope).

E x p e r i m e n t – 3
Characterization of DC Motor: Part 1

3.1 Introduction

The output voltage control of a two-pole DC-Switch-mode-converter was implemented in real-time, in the last experiment. The purpose of the real-time implementation was to obtain a variable DC-voltage at the output of the power converter, while controlling its amplitude with a dSPACE-based Control-desk user interface. In this experiment, a DC-motor will be connected to the output of the power converter. With this arrangement, a variable voltage can be applied to the terminals of the DC-motor. We will observe that by changing the magnitude of the applied voltage, the speed of the motor can be varied. This is also referred as open-loop voltage controlled DC-motor. The electrical parameters of the motor can be calculated by the open-circuit and blocked rotor tests and the voltage vs speed characteristics can be verified.

The objectives of this experiment are

1) To observe open-loop speed control of a DC motor
2) To calculate the motor back-emf constant k_E
3) To calculate the electrical parameters (R_a and L_a) of the motor using the blocked rotor test
4) Verify the voltage vs speed characteristics of the DC motor

3.2 Control of a DC Motor in Open Loop

Varying its supply voltage can change the speed of a DC motor. The model of output voltage control of the switch-mode dc converter was discussed in Experiment – 2 and the same will be used.

- Use the model for the two-pole switch-mode converter (Fig 3.1) OR download the file 'two_pole.mdl' from online.

- Change the name of the Constant block from **V_ab** to **V_motor**; this will be the input voltage of the DC-motor.

3.2.1 *Adding current measurement blocks*

For measuring the current, **Channel 5** of the A/D converter (**ADCH5**) on CP 1104 will be used. Remember that the data have to be scaled by a factor of 10. In addition, the current sensor outputs 1V for 2 amps of current; therefore it actually needs to be scaled by 20 (shown in Fig. 3.1).

- *Drag and drop the **DS1104ADC C5** block from the dSPACE library. (dSPACERTI1104→DS1104 MASTER PPC→ **DS1104ADC C5**)*

- *Connect a **Gain** block at its output and set its value at 20.*

- *Connect a **Terminator** at the output of the **Gain** block (rename this block '**motor**_**current**')* and label the signal as Ia.

3.2.2 *Adding speed measurement blocks*

To measure speed we shall use the **DS1104ENC_POS_C1** block from the dSPACE library. This block provides read access to the delta-position and position of the first encoder interface input channel. The delta position represents the scaled difference of two successive position values of a channel. To receive the radian angle from the encoder the result has to be multiplied with $\frac{2\pi}{\text{encoder\_lines}} = \frac{2\pi}{1000}$ where, **encoder_lines** is 1000 for the encoders used in the laboratory setup.

The delta-position scaled to a radian-angle has to be divided by the sampling time to obtain the speed, as in:

$$\omega = \frac{d\theta}{dt} = \frac{\Delta\theta}{t_{k+1} - t_k} = \frac{\Delta\theta}{T_s} \tag{1}$$

- Drag and drop the **DS1104ENC_POS_C1** block from the dSPACE library. In addition the encoder set-up block **DS1104ENC_SETUP** is to be added to the model. Both these blocks are in dSPACERTI1104→DS11DS1104ENC_POS_C104 MASTER PPC.

- Connect a **Terminator** block to the Enc position which is located in DS1104ENC_POS_C1 (Simulink→Sinks→Terminator)

- Connect a **Gain** block at Channel 1 output (i.e. Encdelta position) and set its value as $\frac{2\pi}{(Ts*1000)}$ where, T_s is the sampling time set in the simulation parameters under the fixed-step box. The output of this block is the motor speed in rad/s. However, at low speeds, there will be oscillations in the measured speed values. Hence an averaging to get more accurate readings are needed.

- Download the file 'Avg_Block.mdl' and copy it to your folder. Connect it as shown in Fig.3.1. The output of this block is the average speed in radians/sec.

- Add another **Gain** block (rename this 'speed_rpm') in series with this to convert the rad/s value to RPM. Change the gain value to $\frac{60}{2\pi}$.

Your real-time model is now ready and should look like in Fig 3.1.

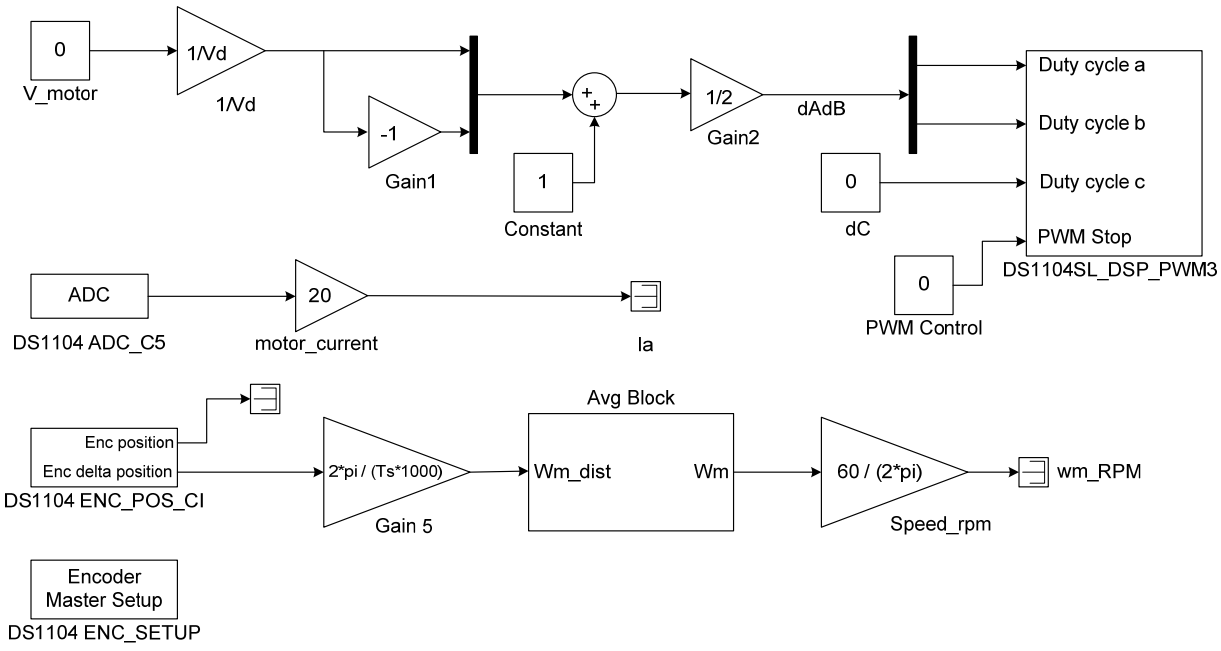


Figure 3.1: Real time Model for Open-Loop Speed Control of a DC Motor.

- Make the following changes:

  Simulation → Configuration Parameters

  →Solver→ Start time=0, Stop time = inf

  Type: Fixed-step, Solver: ode1 (Euler)

  Fixed-step size: 1e-4

  →Optimization→ in Simulation and code generation, uncheck everything except 'Implement logic signals as Boolean data'

  →Real - Time Workshop → System target file → rti1104.tlc

- Enter in command prompt:

  >>Vd = 42; (Enter the value of the DC supply voltage here)

  >>Ts = 1e-4;

- Build it (Ctrl+B) (Make sure the current directory is the same as the location of the .mdl file) In Matlab main window, you will see,
  'MAKE PROCESS SUCCEEDED'

3.2.3 Connections on the Board as per Fig. 3.2

Couple the DC generator and DC motor under test (MUT). Connect the armature of the DC Motor to the output of two converter poles A1 and B1. Connect the **CURR. A1** (phase-current measurement port) on the drives board to the **Channel ADCH5** of CP 1104 I/O board. Also, connect the encoder output (mounted on the DC-motor) to the **INC1 9-pin DSUB** connector on CP 1104 I/O board. Connect the MUT to a DMM to measure the value of E_b.

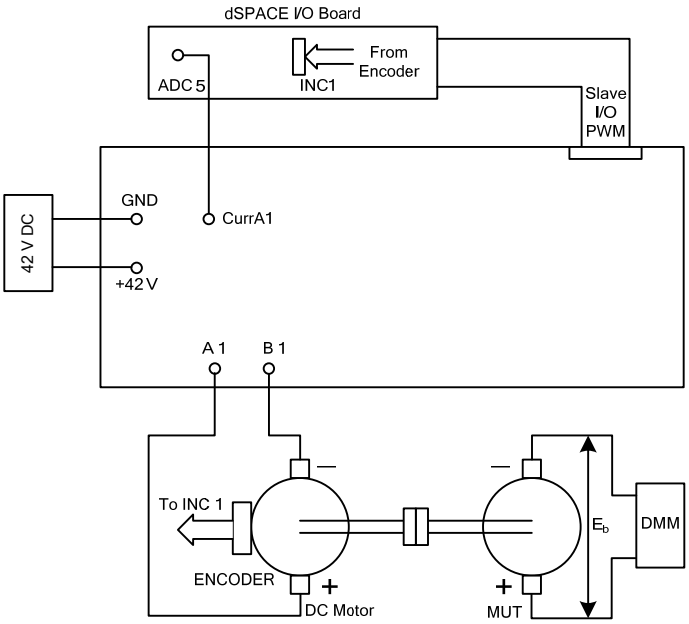


Figure 3.2: Connections for measurement of k_E

3.2.4 Creating Control Desk Layout

- Open dSPACEControlDesk

- Open Variable file (.sdf) select the generated .sdf file.

- File → New → Layout

- Select and draw the following as shown in Fig 3.3.

  Virtual Instruments → Slider, Numerical Input, Display

  Data Acquisition →PlotterArray

- Drag and drop the appropriate values into the Slider, Numerical Input etc.

  OR you can download the **ke.lay** file and drop the appropriate values into the instruments.

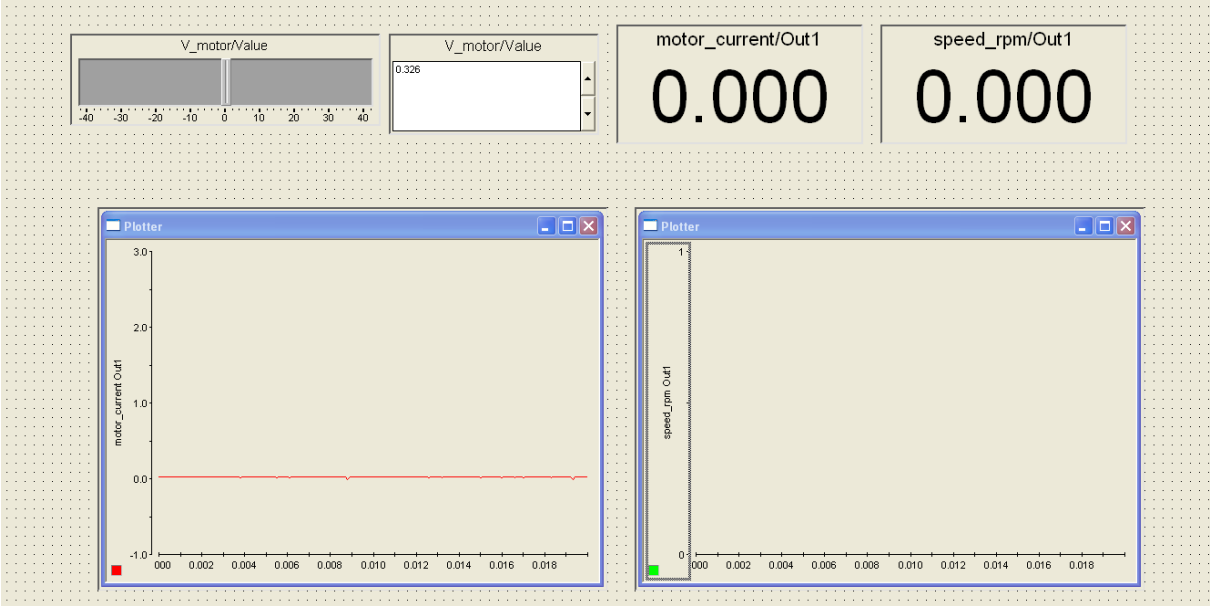


Figure 3.3: Layout files for measurement of k_E

3.3 Determination of Ke (Open Circuit Test)

The back emf that is generated in the motor is directly proportional to the speed of the motor (Equation (2)). In this section, the MUT is driven at a certain speed by another motor (in this case a DC generator running as a motor). The open circuit voltage (E_b) is measured using a DMM. For 10 different values of speed from 0 rpm till around 2000 rpm, measure the value of E_b. Enter this in Table 1. Plot these values in Matlab and find the slope of this line (k_E).

NOTE: Do not exceed $V_{(A1\ B1)}$=20V (~2000 rpm)

$$E_b = \omega\, k_E \qquad (2)$$

Table 1: ***(Take 10 readings from 0 to 2000rpm)***

Motor Speed (RPM)	Measured Eb

3.4 Determination of electrical parameters (Blocked Rotor Test)

$$V_a = R_a I_a + k_E \omega + L_a \frac{di_a}{dt} \quad (3)$$

To estimate the armature inductance, the motor must be held a standstill, $\omega = 0$. If the rotor is blocked and a step voltage is then applied to the armature terminals, the current increases exponentially in time and equation (3) becomes:

$$V_a = R_a I_a + L_a \frac{di_a}{dt} \quad (4)$$

The solution for this differential equation is (5),

$$i_a = \frac{V_a}{R_a}(1 - e^{-t/\tau}) \quad (5)$$

Where, $\tau = \frac{L_a}{R_a}$

The current increases exponentially to the final value $\frac{V_a}{R_a}$. The slope of the curve measured at t=0, is dependent on the value of L_a and R_a is given by $\frac{di_a}{dt} = \frac{V_a}{L_a}$.

NOTE: Do not exceed $V_{(A1\ B1)}$ = 3V in the blocked rotor test. Think why?

3.4.1 Real-Time model

A step voltage can be given to the motor using the **SHUTDOWN and RESET** signal on the drives board. The **SHUTDOWN** signals are controlled by the ***DIGITAL I/O*** channels 11 and 12. When ***IO11/12*** is **0** (OFF state) the switching signals are inhibited and the switches are opened. Setting ***IO11/12*** to **1** (ON state) and resetting (***IO10***) resumes the regular operation of the converters. The ***IO10/11/12*** digital channels will be added as slave bit out blocks for our model from the slave library. In addition two constant locks and two **BOOLEAN** conversion blocks (Search 'data type conversion' in settings change *output data type* to *Boolean*) should be added with **SD1** and **SD2** using the same signal. The model should like the one shown in Fig. 3.4

dSPACE RTI1104→ SLAVE BIT OUT → DS1104SL_DSP_BIT_OUT_C11

change the *Channel Number* to 11/12.

Similarly, Add a **RESET** button on channel 10. Your model must look like Fig. 3.4

Build the model (Ctrl+B).

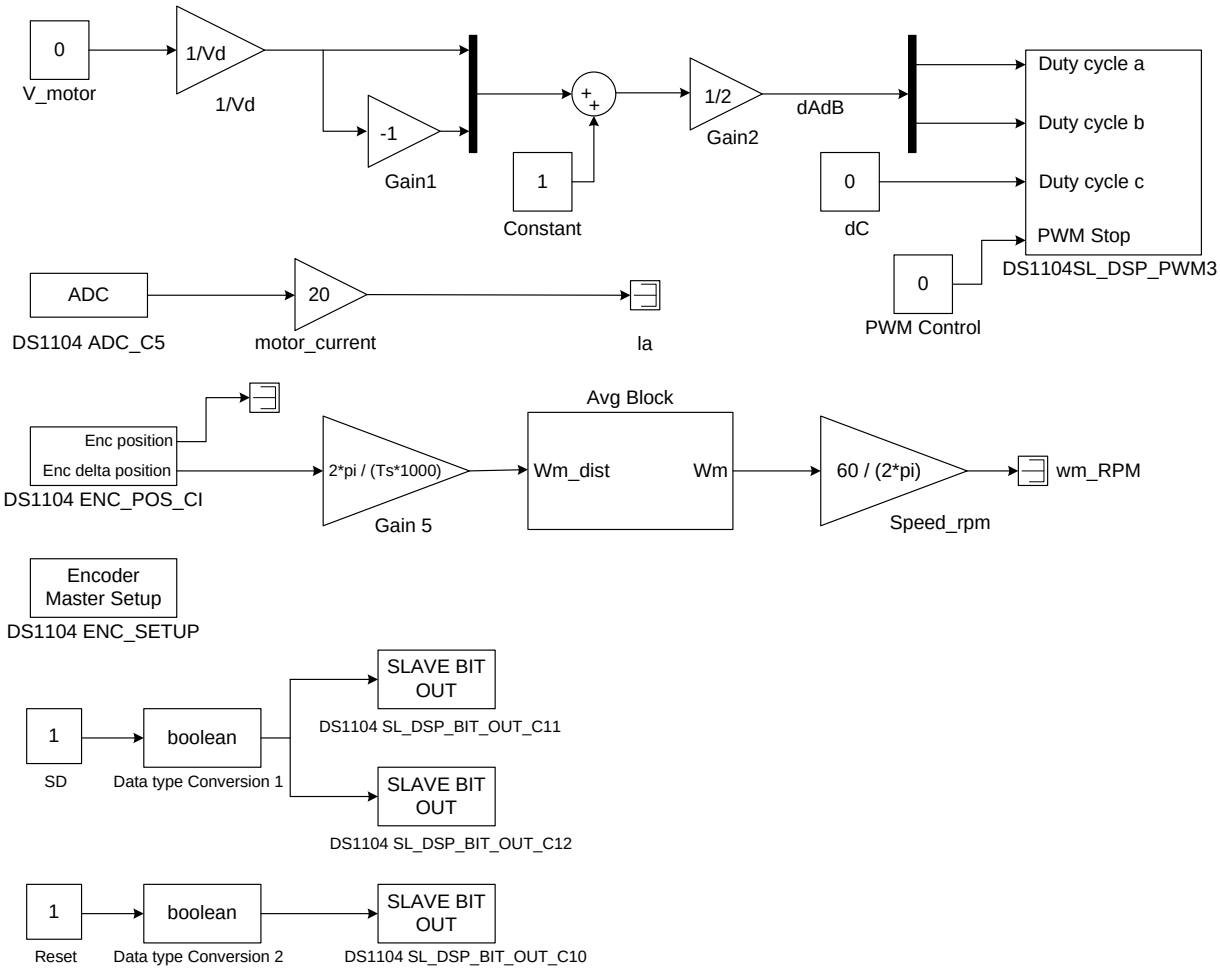


Figure 3.4: Simulink model for blocked rotor test

3.4.2 Creating Control Desk Layout

Reload the variable file and modify the layout to include a CheckButton for SD and Reset. (Virtual Instruments→ CheckButton) as in Fig 3.5 or open the **exp3sdreset.lay** file.

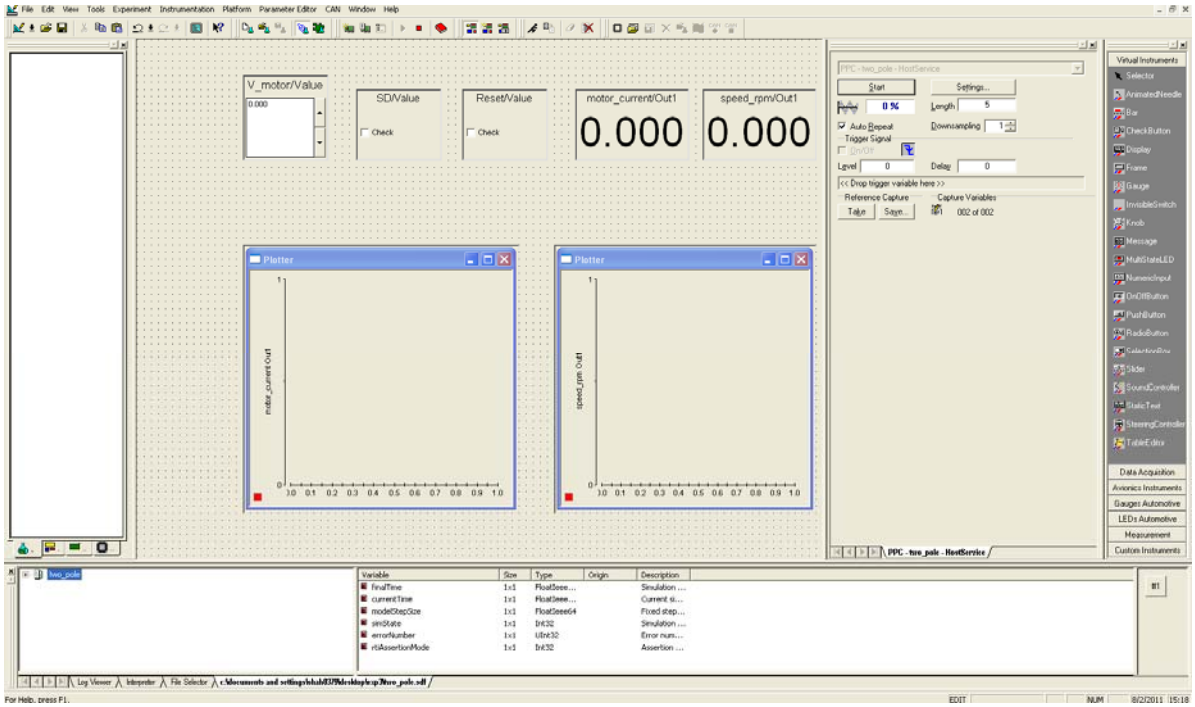


Figure 3.5: Layout for blocked rotor test

3.4.3 Connections on the Board

Make connections for blocked rotor test as in Fig 3.6.

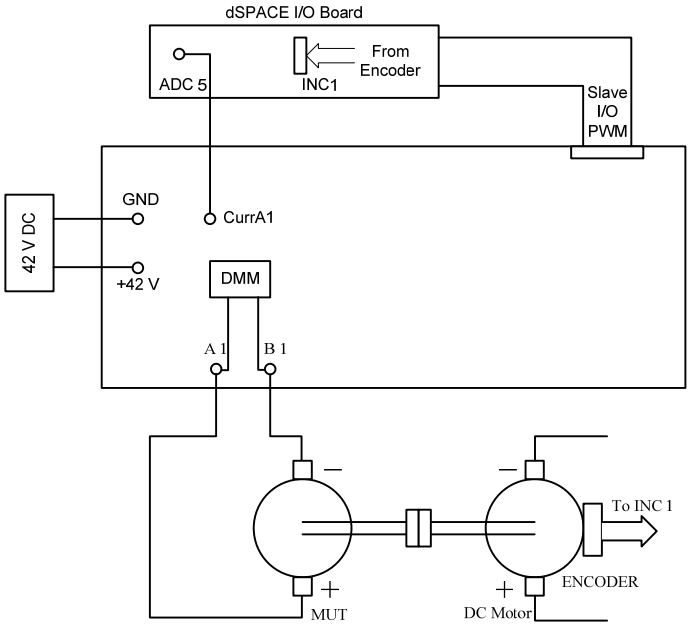


Figure 3.6: Connections for blocked rotor test and no load characteristics

3.4.4 Inductance determination

1. Open View Control bars/Capture Setting Window. Change the setting in the capture setting window as shown in Fig 3.7. Drag the reset signal from the model root values into the grey box situated below the level-delay set boxes. Check the box called ON/OFF, check the edge direction, and the set the level value to 0.5. Also set the length to 0.4.

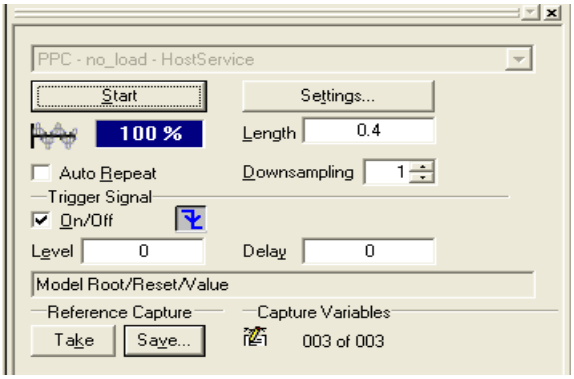


Figure 3.7: Capture setting window

2. Block the rotors firmly.

3. Uncheck and then recheck the SD control. This button works as a switch to connect and disconnect the machines from the power supply.

4. Set the V_motor to a low value (not exceeding 3 V) and uncheck Reset to give a step input voltage. The current should increase exponentially and reach a constant steady state value.

5. Now, you will observe that every time you uncheck the Reset control in the layout, the plot area will display the current and it will stop when it reaches the maximum measurement time. The **Length** is set to 0.4. This will set the data capture time as 0.4s which is large enough to observe the whole transient process in current. The screen shots are shown in Fig 3.8 & 3.9.

6. Check and uncheck SD and Reset to make some measurements. After you are satisfied with the data displayed go to the Capture Settings Window and press the SAVE button. The dialog box will ask you to name the .mat file that will contain the graphic data in all plot areas.

7. Download the file ‘br.m’ which has instructions on plotting from the .mat file and additional code to calculate the value of, L_a and, R_a from the graph.

8. Measure the voltage between terminals Phase A1 and B1 using the DMM when the rotor is blocked.

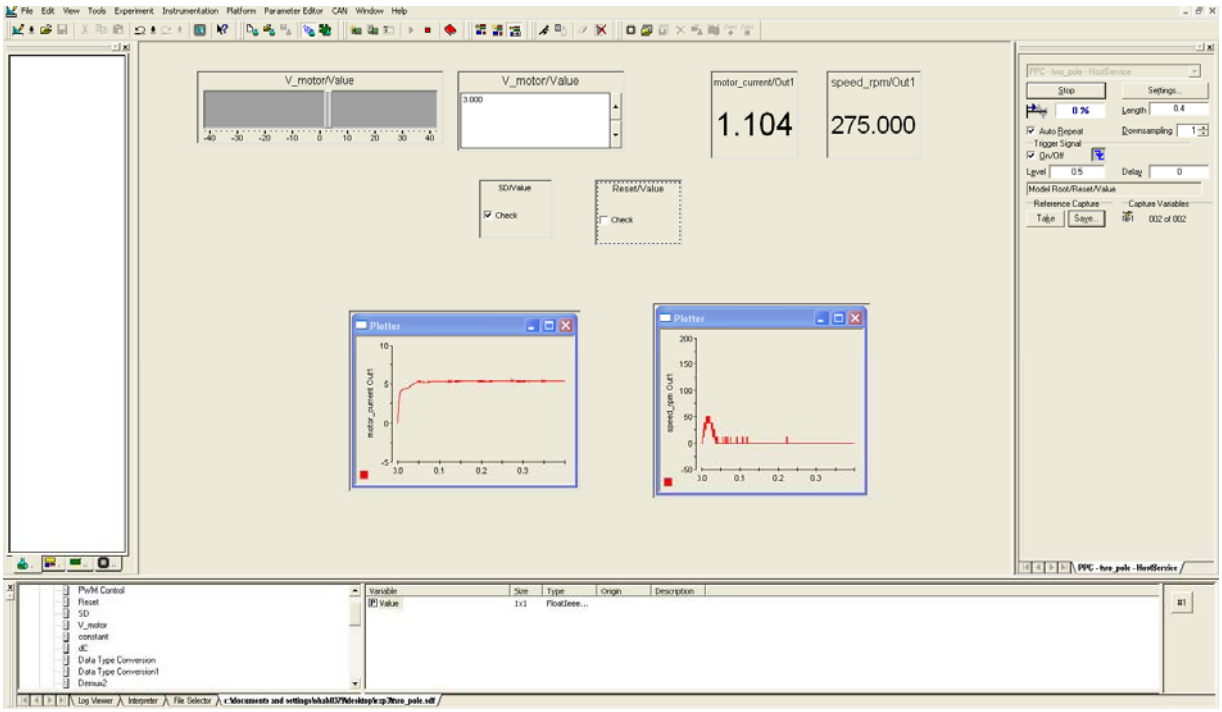


Figure 3.8: Screen shot of dSPACE control desk

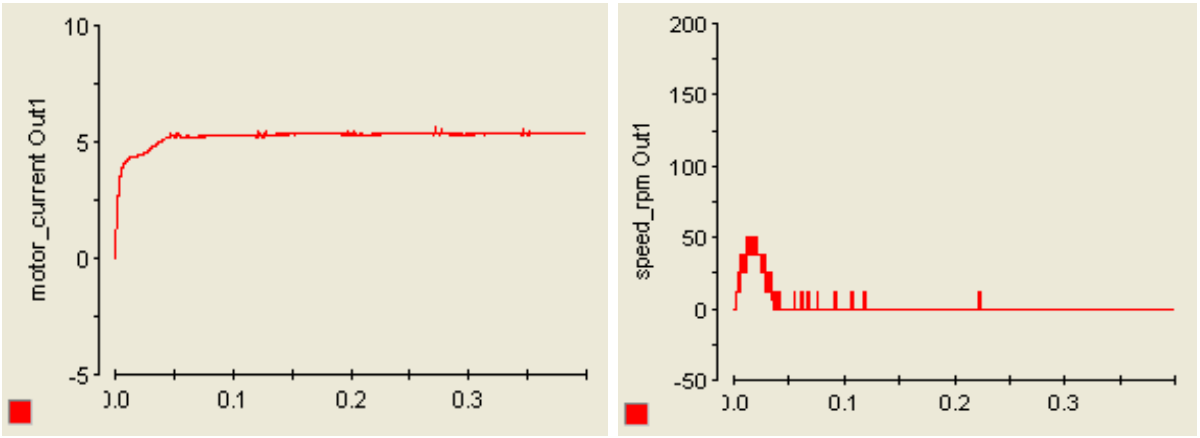


Figure 3.9 Waveform of motor current and speed (rpm) as observed from the dSPACE control desk.

3.5 Open-loop speed control (voltage vs speed characteristics)

In steady state, with voltage V_a applied to the armature terminal of a DC-motor, following equation can be written:

$$V_a = R_a I_a + k_E \omega \quad (6)$$

From equation (6), the armature voltage can be calculated in real-time to run the DC-motor at a desired speed ω (rad/s). Note that, there is no feedback here, we are calculating the equivalent amount of voltage that need to be applied, to run the motor at a desired speed. Hence this type of speed control can be termed as open-loop voltage control. The values of armature resistance R_a and the back-EMF constant k_E should to be known beforehand. The calculated and measured speed is compared at the end of this section.

3.5.1 Measurements

The connections for this section are the same as in section 3.4.3. Record the values of current and speed for different values of voltages specified in Table 2.

NOTE: Use the DMM to measure the actual motor voltage between terminals Phase A1 and B1.

Table 2.

V_{motor}	(V)	0.5	1	3	5	7	10	15	20
$V_{\_A1\text{-}B1}$	(V)								
Speed	(RPM)								
Current	(A)								

3.6 Lab report: *(10 points)*

1) In Section 3.3, *(3 points)*
   a) Table with values of Motor speed and Measured Eb. *(1pt)*

   b) Plot the values of E_b vs speed and *(1 pt)*

   c) Calculate the value of k_E. *(0.5 pt)*

   d) Does the line pass through the origin? Should it? *(0.5 pt)*

2) Section 3.4, *(3.5 points)*

   a) Why do we apply a reduced voltage to the motor? *(0.5 pt)*

   b) Attach the plot of motor current when a step voltage of 3V is given to the motor *(1 pt)*

   c) Calculate the value of R_a from this plot. *(1 pt)*

   d) Calculate the value of L_a from the initial slope of the current. *(1 pt)*

3) Section 3.5, *(2.5 points)*

   a) Plot the voltage vs speed characteristics of the DC motor. *(1 pt)*

   b) Comment on the graph. *(0.5 pt)*

   c) Using the value of R_a, k_E, and measured I_a, calculate speed and plot the measured speed and calculated speed on the same graph. *(1 pt)*

   Report :(1 point)

E x p e r i m e n t – 4
Characterization of DC Motor: Part 2

4.1 Introduction

In the previous experiment, the electrical parameters R_a, L_a and k_E of the DC motor were determined. In this experiment the mechanical characteristics B and J will be determined. The torque speed characteristics will be verified.

4.2 Open loop control of DC-motor with load

The Simulink model used in Experiment 3 will be used in this experiment.

- Create a new folder **Expt 4**.
- Start **Matlab** and change the directory path to **Expt 4**
- Open the **Simulink** model used in the last experiment or download the file ***'no_load.mdl'. Note the speed output is in radians/sec.***

To this model, appropriate blocks for controlling the active load (a DC-generator whose electromagnetic torque will be varied) need to be added.

4.2.1 *Adding a DC-load* (LOAD) *to the DC-motor* (MOTOR)

To determine the DC-motor steady state characteristics, a second DC-motor will be axially coupled to the motor under test (MUT). The second motor will be open loop voltage-controlled, similar to the MUT.

- The terminals of the load (DC-motor) should be connected to PHASE A2 and PHASE B2 terminals on the power board. Set the supply voltage (V_d) to be close to 42V.
- Ensure a firm mechanical coupling between the motors.

Add necessary blocks to make the model shown in Fig 4.1. All the required blocks can be copied from the file **'components.mdl'.** Or you can build it by following the instructions (a) to (f) below.

a) Add a second set of voltage-control blocks in the Simulink model and connect the duty-cycles to the 2 and 3 inputs of ***DS1104SL_DSP_PWM***.

b) Set the switching frequency as 10000 Hz in the PWM block.

c) Double click the ***DS1104SL_DSP_PWM*** block, go to **PWM Stop and Termination**, uncheck "*Set all Ch*" and then click on "*Set all*".

d) Connect the ***CURR. A2*** channel to *ADCH6* on the controller box

e) Make the load current ***(CURR. A2)*** available in the Simulink model by copying the first current measurement blocks, and then double clicking on ***DS1104ADC_C5*** and changing it to channel 6.

f) Save the model. The model should like the one shown in Fig. 4.1.

- In Matlab, enter Vd=42 (Enter the Vd supply voltage here) and Ts=1e-4

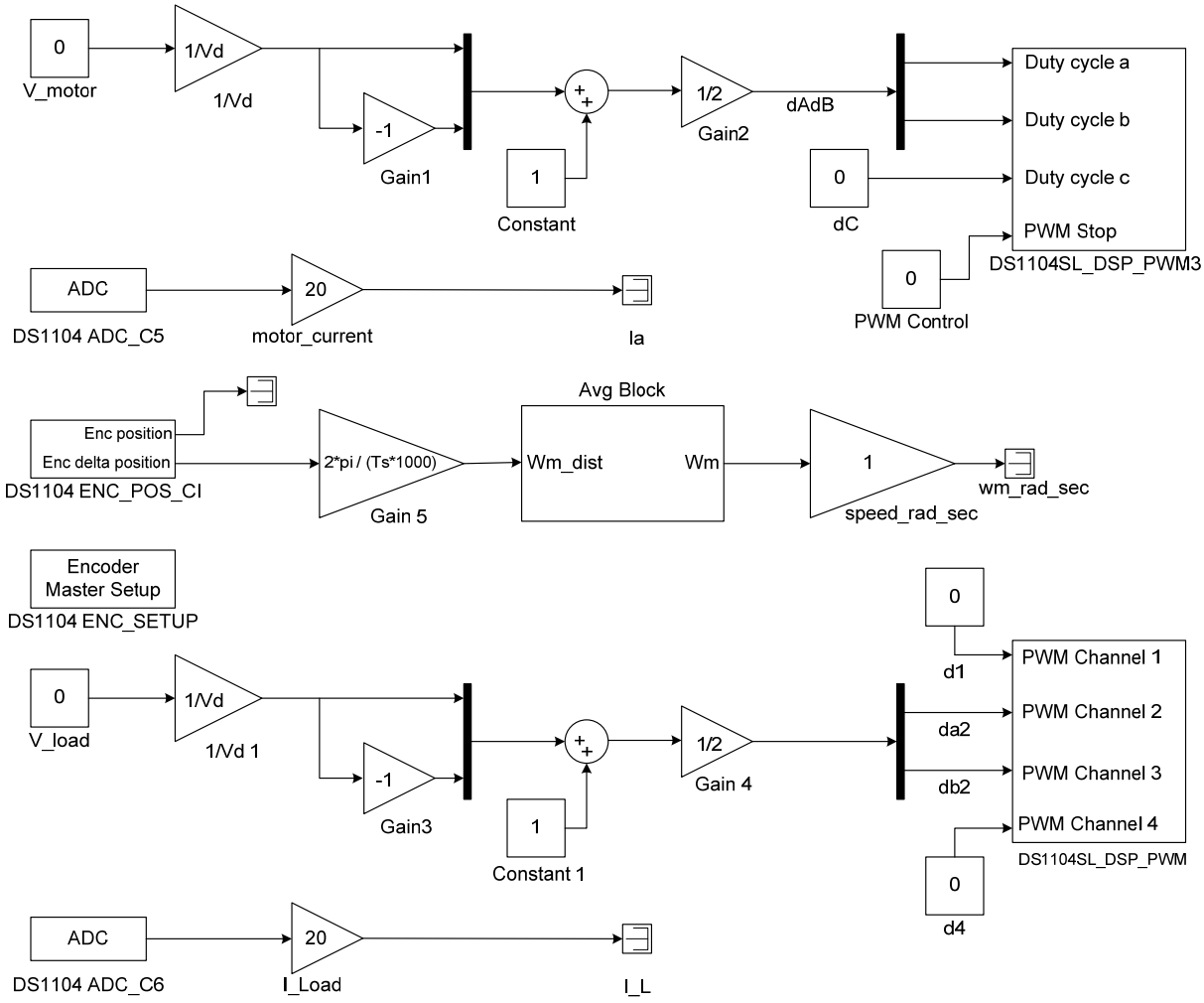


Figure 4.1: Real-time Simulink model for motor and load control

4.2.2 *Creating the Control Desk Interface*

- Build (CTRL+B) the Simulink mode.

- Open Control Desk and 'Open Variable File' the .sdf file that was generated. Create a new layout (or download **exp4_1.lay** and drag the necessary the values as per Fig 4.2) and draw two **Slider Gain** controls and two **Plotters**.

- Drag and drop the **V_motor** and **V_load** variables to the Slider gains.

- Assign one plotter to display '*motor_current'* and *'I_Load'* currents and one plotter to display the speed *'speed_rad_sec'*.

- In order to record the numerical values of currents and speed, add three **Displays** to the layout and assign them the speed and current variables. The control desk layout should look like as shown in Fig. 4.2.

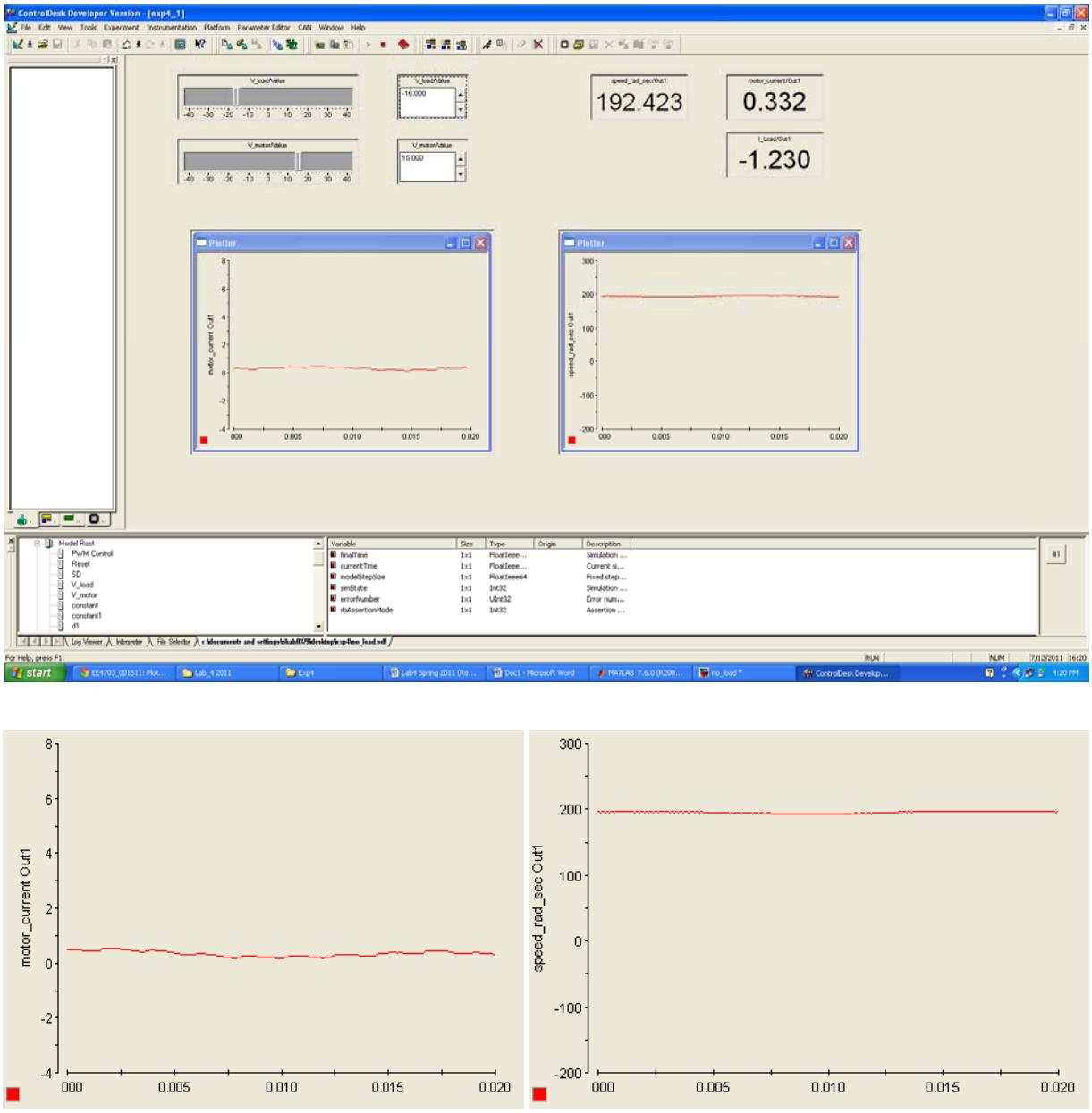


Figure 4.2: Control-desk layout (showing motor current and motor speed)

4.2.3 *Theoretical Background*

In this experiment, you will plot the torque-speed characteristics of a DC-motor, using the second motor as load. For a constant **V_motor** voltage, the load is varied using **V_load.** The motor current and speed are recorded in a table. A set of measurements is obtained for different supply voltages.

The steady-state mechanical characteristics of a DC-motor are the dependency between the electromagnetic torque (N-m) and the electrical speed (rad/s). Since the dependency is linear, the characteristics will be straight lines for the entire voltage range 0 - V_{rated} and is independent of the load. The motor equations reflect this linearity:

$$V_{motor} = R_a I_a + k_E \omega \quad (1)$$

$$T_e = k_T I_a \quad (2)$$

The steady-state model for the load can be approximated with a friction-type model, where the torque is proportional to the speed, and a constant friction torque is always present:

$$T_e = T_L + B\omega + T_{friction} \quad (3)$$

where, all terms in the right hand side are load related. For our setup, where the load is a voltage controlled DC-machine, the load torque T_L is, in fact, the electromagnetic torque developed by the second DC-motor.

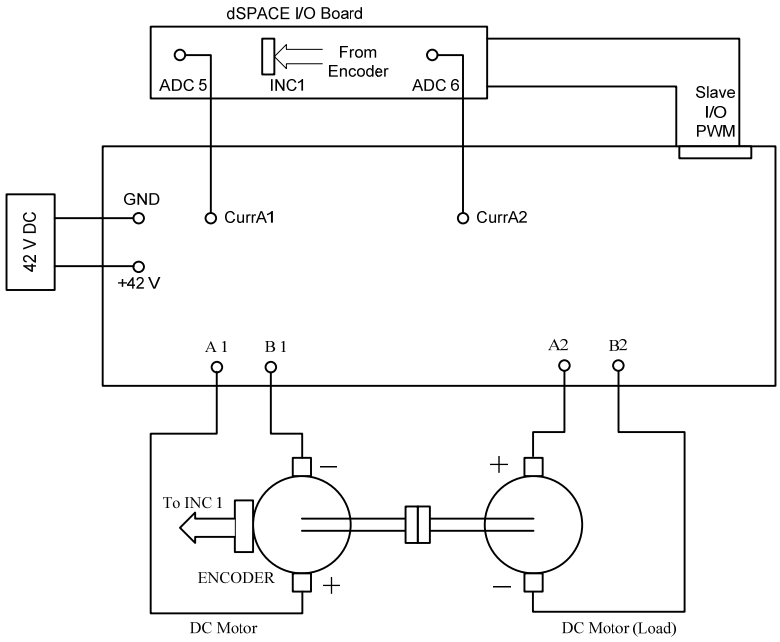


Figure 4.3: Connections for torque speed characteristics

4.2.4 *Drawing the torque-speed characteristics:*

- Make connections as per Fig 4.3

- Maintain the MOTOR voltage at constant levels {10V and 15V}(*measure the actual motor voltage using DMM. Turn off the power. Now connect the load side motor to Phase A2 and Phase B2 and connect the DMM to the load side motor. Always ramp up the voltage or you may get a fault.*). Adjust the LOAD voltage reference such that the MOTOR current takes the following values at each voltage: {0; 1; 2; 3; 4; 5 A}. Make sure that neither motors draws more than 5 A. *(If V_motor is set to 10V, set V_load to -10V at the start to prevent overcurrents)*

- Record the motor speed (ω_rad/sec), the LOAD current **(IL)** and LOAD voltage **(V_load)** required to obtain the specified MOTOR currents. All current measurements will be multiplied with k_E to obtain the corresponding torque values as per equation (2). **(The sign of speed will be positive for the motor with the encoder and opposite for the motor without encoder.)**

- Draw the MOTOR and LOAD characteristics using the data acquired during the measurement process.

Table 1 Torque – Speed Characteristics

V-motor[V]	V-motor[V] Measured by DMM	V-load[V]	V-load[V] Measured by DMM	I-motor[A]	I-load[A]	ω[rad/s]
15V		-15V		0A		
15V						
15V						
15V						
15V						
15V						
10V		-10V		0A		
10V						
10V						
10V						
10V						
10V						

4.3 Open Loop Control of DC Motor without Load (to calculate B and $T_{friction}$)

For determining the friction parameters, the MOTOR will be run under no-load conditions as done in the previous experiment (Experiment 3). Since the MOTOR has to overcome only friction (T_L = 0), the electromagnetic torque ($T_e = k_E I_a$) will follow the linear friction model (see equation (3)) in steady-state. Fill in the TABLE 2 using the values obtained in experiment-3. Plot the values of speed (X-axis) vs T_e and find the slope of the line and the y-intercept.

Table 2 No-load torque speed characteristics (to calculate B and $T_{friction}$)

V_{motor}	(V)	0.5	1	3	5	7	10	15	20
$V_{\_A1\text{-}B1}$	(V)								
Speed	(rad/sec)								
Current	(A)								
Torque	(Nm)								

Use the values obtained from Table 2 in experiment-3 and calculate the torque.

4.4 Determination of Inertia

In this section the moment of inertia (J [kg – m^2]) will be determined. The setup consists of two DC-motors, axially coupled and supplied from two converters. One motor is current controlled,

such that it will act as an active load. This motor will be named as LOAD. The second motor (MUT) is open-loop voltage controlled. This motor will be named as MOTOR.

$$T_e = T_L + T_{friction} + B\omega + J\frac{d\omega}{dt} \tag{4}$$

The motor is brought to a no-load steady-state $\left(J\frac{d\omega}{dt} = 0\right)$ speed ω_o, by disconnecting the load ($T_L = 0$). At this point ($t = 0^-$),

$$T_e = T_{friction} + B\omega_o; \quad \text{where } T_d = k_T I_a\left(0^-\right)$$
$$I_a\left(0^-\right) \rightarrow \text{MOTOR current at steady-state no-load speed} \tag{5}$$

To make electrical torque (T_e) equal to zero in the mechanical dynamics equation (4), a complete shutdown of the motor supply is required. At ($t = 0^-$) the whole system is shutdown. This implies that the electromagnetic torques in the MOTOR (T_e) becomes zero. The dynamic equation will be:

$$0 = T_{friction} + B\omega + J\frac{d\omega}{dt} \tag{6}$$

At ($t = 0^+$) i.e. just after shutting down the system, the equation (6) can be written as

$$0 = T_{friction} + B\omega + J\left(\frac{d\omega}{dt}\right)_{t=0^+} \tag{7}$$

Thus,

$$J = \frac{-\left(T_{friction} + B\omega_o\right)}{\left(\frac{d\omega}{dt}\right)_{t=0^+}} = \frac{-k_T I_a\left(0^-\right)}{\left(\frac{d\omega}{dt}\right)_{t=0^+}} \tag{8}$$

By knowing $\omega_0, I_a(0^-)$, and graphically determining the slope $\left(\frac{d\omega}{dt}\right)_{t=0^+}$ of the speed curve at ($t = 0^+$), the system inertia J can be calculated using equation (8).

4.4.1 *Simulink model for dynamic parameter determination*

Include the **SHUTDOWN** and **RESET** signals as done in Experiment 3. Your model should look like Fig. 4.4.

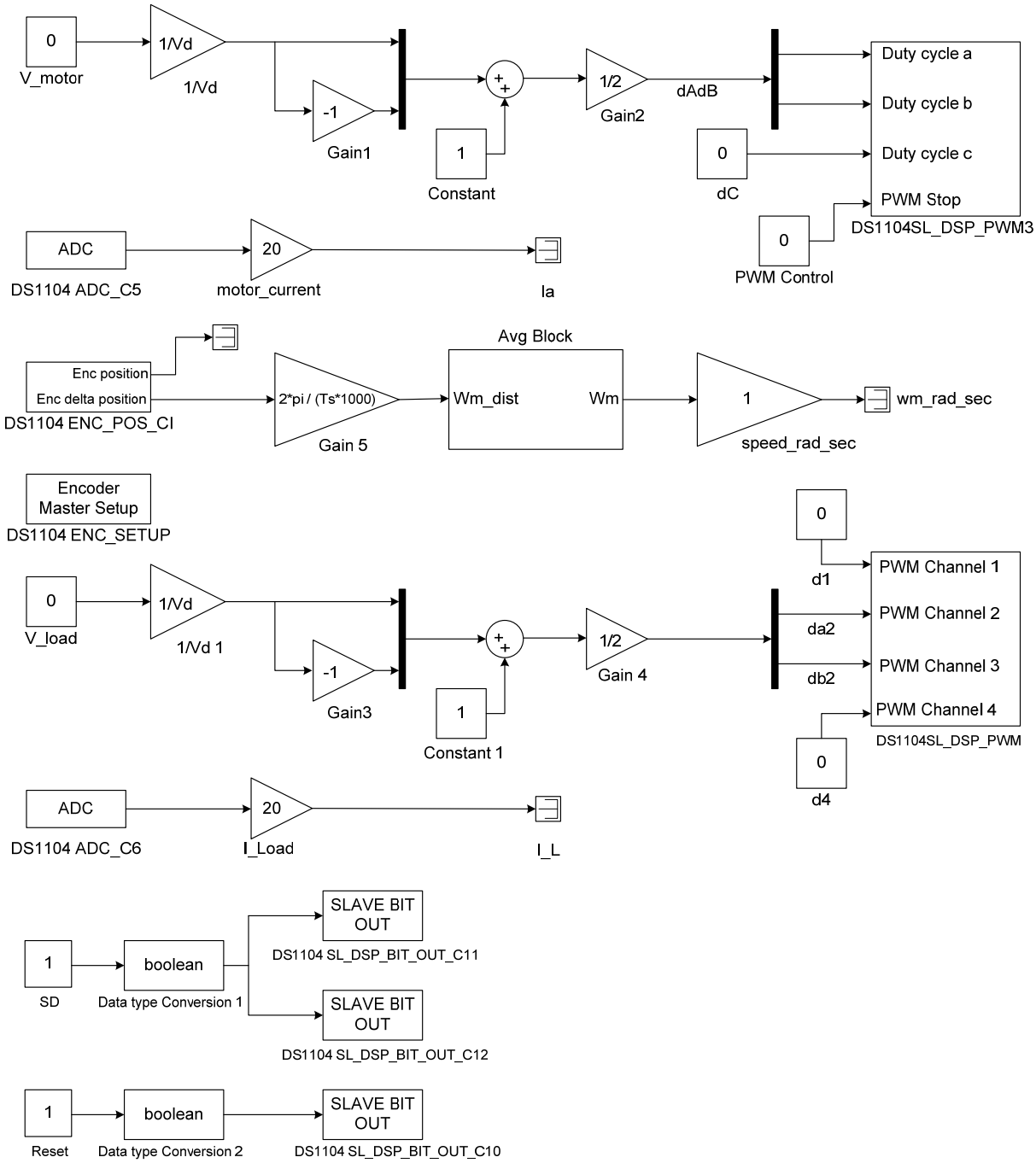


Figure 4.4: Simulink model to determine J.

4.4.2 Control desk layout for dynamic parameter determination

Make connections as in Fig 4.6 for determination of J.

- Start dSPACE ControlDesk and open the generated .sdf file from File → Open variable file.
- Open the Layout file and include two **CheckButtons** assign them to Reset and SD. (Fig 4.5)

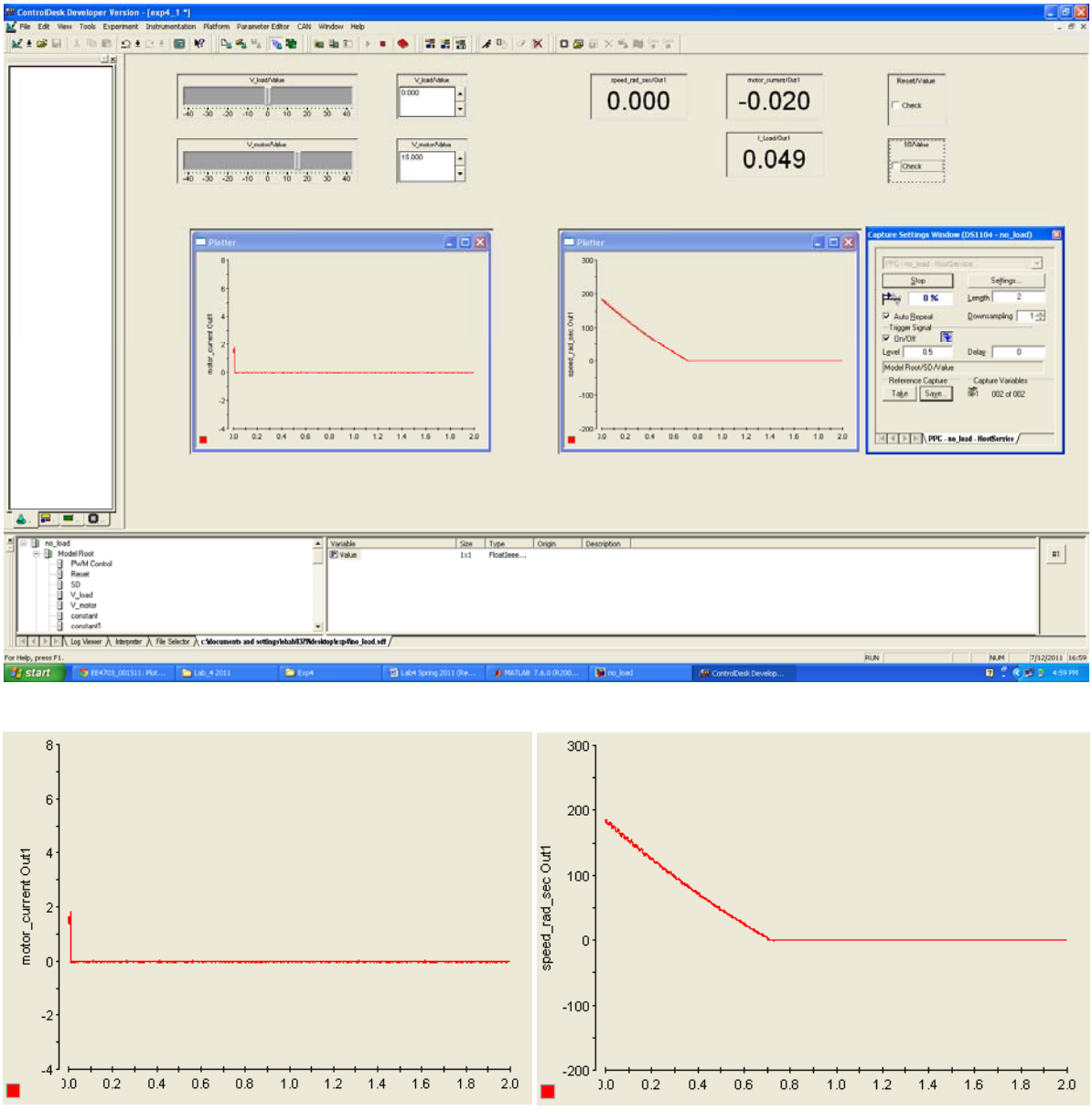


Figure 4.5: dSPACE ControlDesk Layout

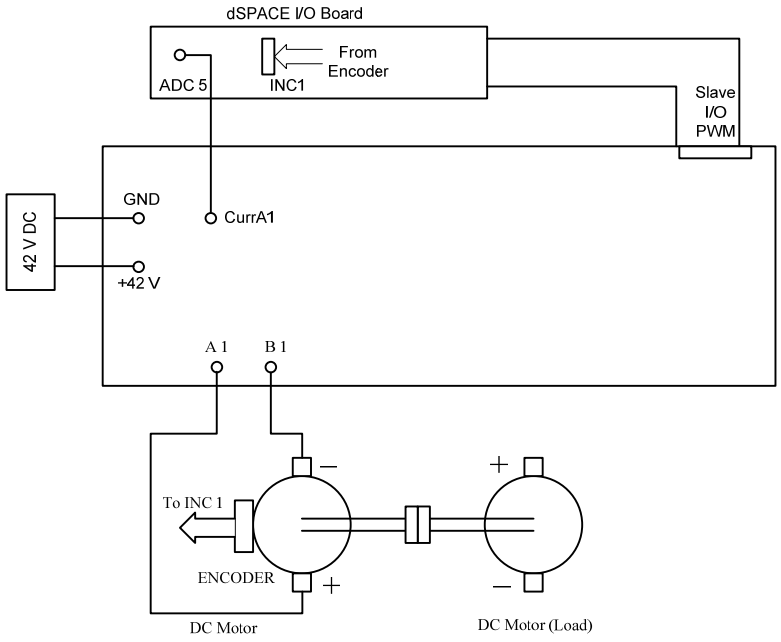


Figure 4.6: Connections to determine J

4.4.3 Inertia determination

- In the Capture Settings Window (View→Controlbars→Capture Setting Window) the following modifications must be made: Increase the display length to 2 seconds and change the Trigger Signal to SD (Fig 4.7).

- Disconnect the LOAD motor from the Power-Electronics-Drive-Board. Check the SD control (uncheck Reset) and increase the voltage on the MOTOR to 15 V.

- Record the speed ω_0 and armature current **I_motor** $= I_a(0^-)$ value at this operating point.

- Uncheck the Shutdown button. This will initiate the display process and, after two seconds, the speed plot will stop and a decreasing exponential curve will be obtained (Fig 4.5).

- Press again the SAVE button in the Capture Settings Window and store the data in a file named **j.mat** file. Plot the .mat file using **'**exp4_plots.m**'** (download this). Calculate the value of J as explained in section 4.4.1.

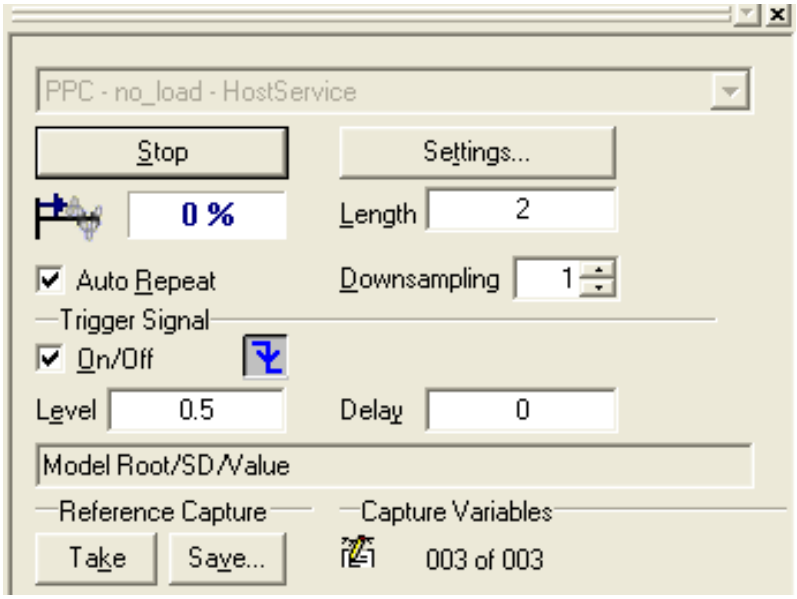


Figure 4.7: Capture Settings

4.5 Lab Report: *(10points)*

Make sure to include plots/ tables with captions. Use proper units, eg. R=10 ohm. Show the steps used in calculations for Questions 4-7

1. a) Plot the torque-speed characteristics of the DC motor and load motor- 2 graphs (Section 4.2.4) *(2points)*.

   b) Comment on the torque speed characteristics in terms of when is the maximum power available for a given voltage? How the curves change with different input voltages? *(1point)*

2. a) Plot the torque speed curve from section 4.3 *(1point)*

   b) Determine the values of B and $T_{friction}$ *(2points)*

3. Determine the value of J *(1point)*

4. For a torque of upto 0.5Nm, plot this motor's steady state torque-speed characteristics for Va = 42V. Label axis with proper units!*(0.5point)*

5. The datasheet of this motor mentions a current limit of 6A. Will you select this motor to drive a load torque of 1Nm, give reason(s) for your answer? *(0.5point)*

6. What is the power loss in the armature of this motor, if it is used to drive a load of 0.5Nm and at 4000rpm? If you assume a lossless friction model, what is the efficiency of power conversion? *(1point)*

7. In the above question, if a switched mode power converter is used to drive the dc motor, what are the values of duty cycle d_a and d_b for a dc bus voltage of 50V? Also what will be the dc bus current? *(1point)*

Experiment – 5
DC Motor Speed Control

5.1 Introduction

In experiment-3 and 4, the speed of the DC-motor was controlled by using an open-loop voltage control. The purpose of this experiment is to design and implement a close-loop speed control of a DC-motor drive. We shall use the same DC-motor for which the parameters were calculated in the previous experiment. At first, the controllers will be designed and tested on a simulation model of the DC-motor. Once the parameters are tuned, the model of the DC-motor will be replaced with the real motor. The tuned controllers will be implemented in real-time on DS1104 to perform the close-loop speed control of the DC-motor.

5.2 Simulink Model of the DC-motor

The model for a DC-motor in frequency domain is derived in Chapter 8 [1].

$$I_a(s) = \frac{V_a(s) - E_a(s)}{R_a + sL_a} \qquad E_a(s) = k_E \omega_m(s) \tag{1}$$

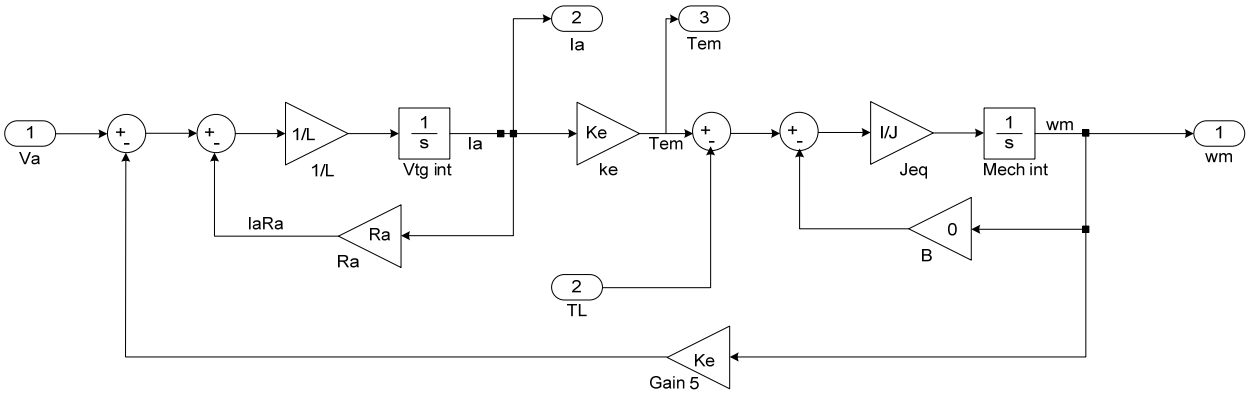
$$\omega_m(s) = \frac{T_{em}(s) - T_L(s)}{sJ_{eq}} \qquad T_{em}(s) = k_T I_a(s) \qquad k_T = k_E \tag{2}$$

Equations (1) and (2) can be easily implemented in Simulink using standard blocks as shown in Fig. 5.1

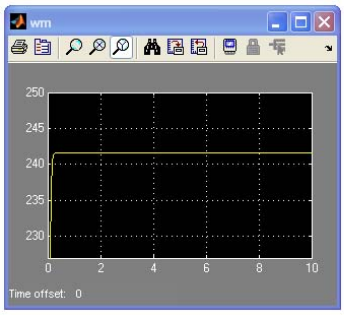
- Download the Simulink model **'dc_motor.mdl'** to the desktop (Fig. 5.1). Convince yourself that it is the model for a dc motor.

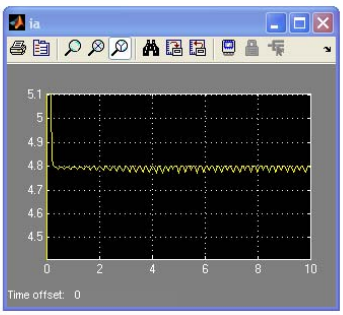
- The representation in Fig 5.1 uses integrators instead of transfer functions. This allows setting the initial conditions for the current and speed state variables. The model also includes the friction coefficient B. However, during simulations, B can be considered zero or its value can be obtained from experiment 4, and the model will be similar to the one described by equations (1) and (2). Create a subsystem by selecting components shown in Fig 5.1 and name it as DC

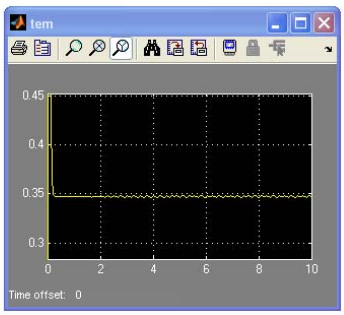
Machine. The simulation block of the DC motor parameters is shown in Fig 5.2.



(a) Simulink Model







(b) Waveform for ωm (c) Waveform for Ia (d) Waveform for Tem

Figure 5.1: Simulink model of DC-motor and waveforms for ωm, Ia and Tem

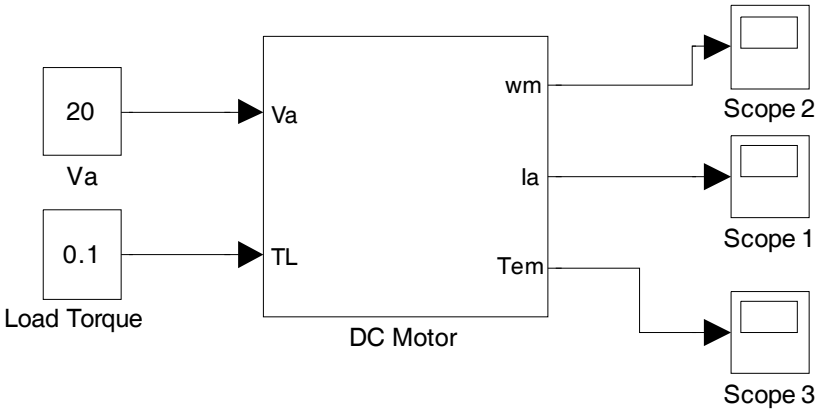


Figure 5.2: Simulation block of DC Motor Parameters

- Now enter the values of DC-motor parameters, which were evaluated in experiment – 3 and 4 in file **'dc_motor_parameters.m'**. Ia_initial, wm_initial are the initial values for the integrator. To observe the zero initial condition response, set these values to zero. Run this file. Make sure your units are all consistent.

- Run the simulation (with default configuration parameters) for the following two cases. Compare the observed values with calculated values. Save the plots and include them in your report.
  a) Va = 20V, Load_Torque = 0.3 Nm
  b) Va = 20V, Load_Torque = 0 Nm

5.3 Controller Design

Once the DC-motor model is built, the controllers can be added and tuned. Start with the current loop for which a PI controller is required.

- Download the file **'components.mdl'**. All the necessary blocks can be copied from here. For the more adventurous, follow the instructions as specified in points a, b, and c to build it!

  a) Build the model for a PI controller see Fig.5.3. Double click the integrator block and enable limit output. Then set the Upper and Lower saturation limits to **+lim/-lim**. The **lim** value should be set to 1(in the .m file) as the absolute maximum value of control voltage is 1, which is the input to Kpwm block. The resultant maximum value of voltage applied to the DC motor will be ±42, which is the rating of the DC motor.

  b) The armature current is fed back to the controller input.

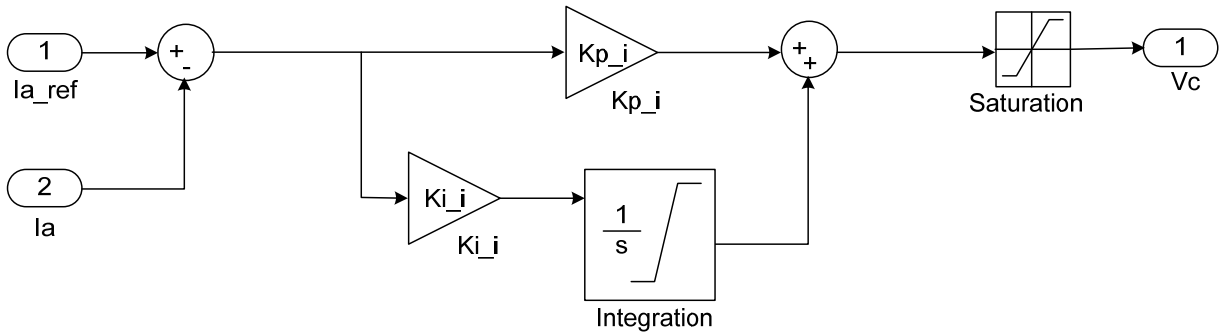


Figure 5.3: PI controller model

c) The Saturation block sets the maximum and minimum limits for the control voltage (in our case ±1).

- Design current controller for a bandwidth of 100 Hz (628.3 rad/sec) (phase margin 90 deg). The parameters of the PI controller (namely Kp_i and Ki_i) are computed using the motor parameters, which were evaluated in the earlier experiment. This procedure is described in section 8-7-1 [1].

- Create the file for a current controlled DC motor as shown in Fig 5.4(a).

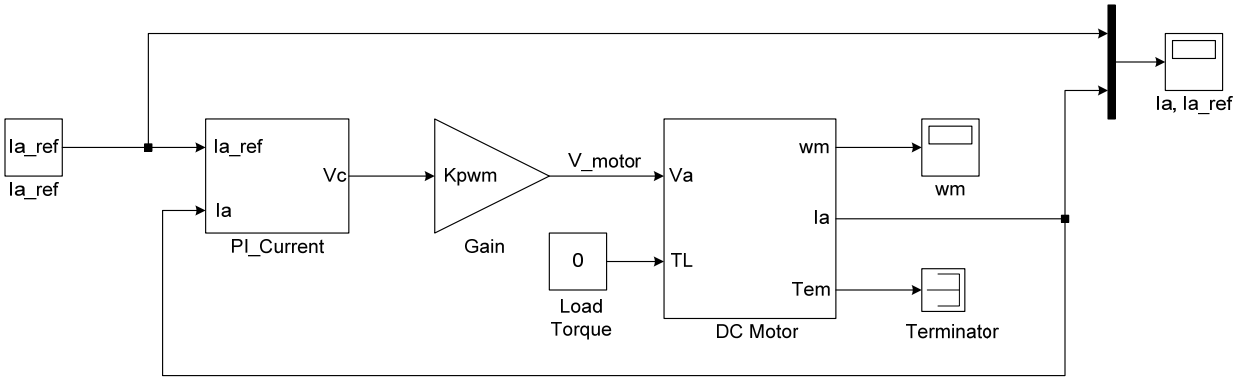
- Running the Simulink model for the current controller with reference current as 2A, results similar to the Fig.5.4 (b) and Fig. 5.4(c) will be obtained.

- Set the value of Kp_i and Ki_i in Matlab prompt (or in the m file 'dc_motor_parameters'). Also set the values of lim = 1, Kpwm = 42, Ia_ref=1. Run the m-file before running the simulation, which will load the values of all the variables. **Run this simulation for a reference current of 1 A (for 0.005 sec, fixed-step, ode1, time step 1e-4).** Save the plots for the report.

- Once the response in current is considered optimal (low overshoot, fast rise-time, zero steady state error), the speed controller can be designed. For designing the speed controller you can assume B=0 but while building the Simulink block, include B.

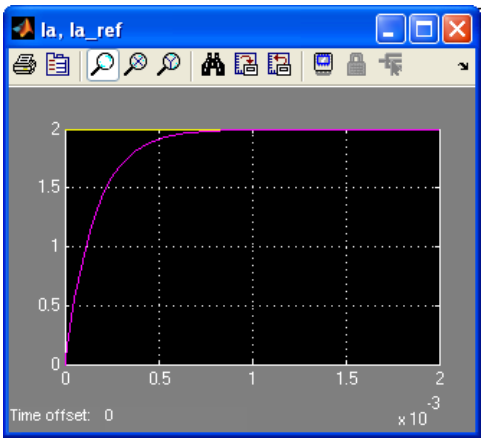
- A similar PI controller for the speed loop ('PI_Speed') will be added to the Simulink model. Design the speed controller for a bandwidth of 10Hz (62.83 rad/sec) (Phase margin 60 deg).

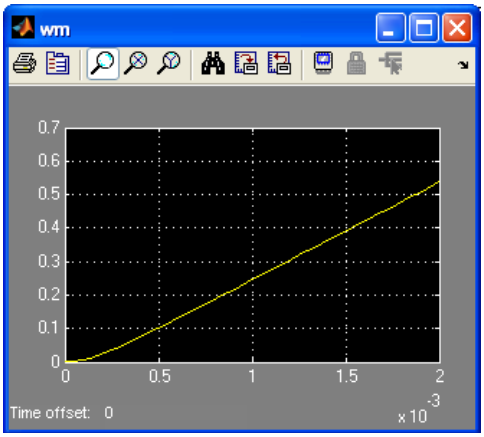
- Follow the procedure described in 8-7-2 [1] to design the speed control loop, using the motor parameters determined in earlier experiment.



(a) Simulink model for current controller

For example the current & speed waveform for a reference current of 2A are as given in Fig 5.4 below





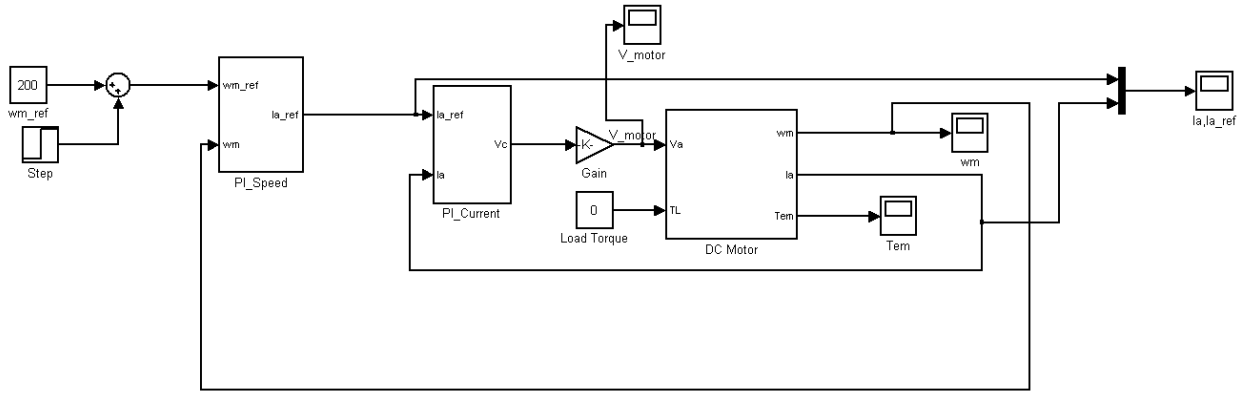
(b) Current waveform for 2A current reference (c) Speed waveform for 2A current reference

Figure 5.4: Simulink model and result for current control loop

The Simulink model for the cascade control and the waveforms for speed and current are shown in Fig.5.5 (a). **Please make sure that wm from DC motor is fed back to wm in PI_speed block and Ia from DC motor is fed back to Ia in PI_current block**. The Speed PI controller has a current limit output of ±5A, necessary to limit the current during transients (both in simulation and real-time systems). To check the controller design, we will give a step change in the speed reference. This is implemented using a constant and step source blocks (Fig 5.5(a)). The results of cascade control are shown in Fig. 5.5(b) and Fig. 5.5(c). If the controller parameters were correctly tuned, then it's time to go on for the next step, and implement the controllers in a real-time system.

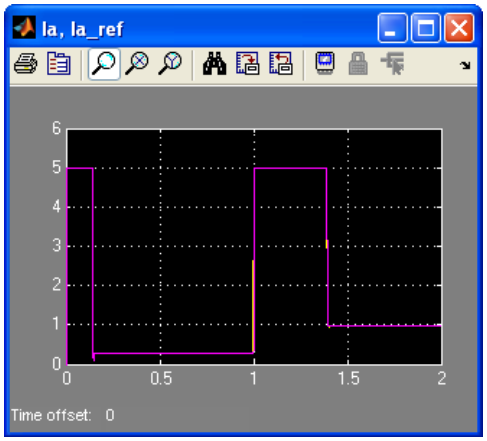
Set the simulation output of speed response for a step input varying from 100 to 300 rad/s. (run the simulation for 2 sec fixed-step, ode1, and time step 1e-4).

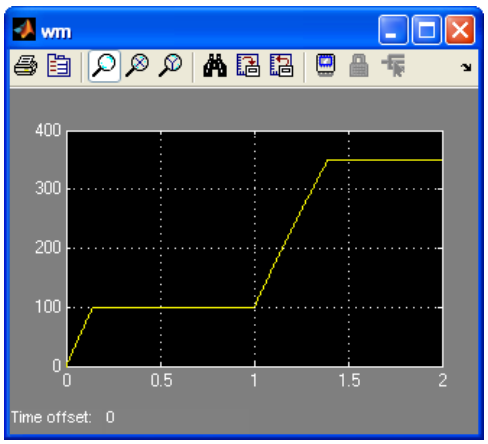
Set the speed reference to be 200 rad/sec and give a step change in the load torque from 0Nm to 0.3 Nm at 1 sec (run for 2 sec). Attach the graph in your lab report.



(a) Simulink model for cascade control

Following are examples of waveform in Fig 5.5 where a step change of 100 to 350 rad/sec are shown.





(b) Current waveform for a step change in speed (c) Speed waveform for a step change in speed

Figure 5.5: Simulink model and result for cascade control

5.4 Real-time implementation of feedback control

For dSPACE implementation, the dc-motor model will be replaced with the real motor and power converter with 42V dc supply will replace Kpwm block. The control voltage to duty cycle conversion was already discussed and implemented in experiment – 2.

All the necessary components to make the model in Fig 5.7 are provided in the file **'components.mdl'.** If you would like to build it the instructions as in a to f are given below:

a) Add the reset block used in the previous experiment.

b) Modify the Speed Control block as shown in Fig. 5.6. Change the integrator block parameters by double clicking on it and changing its external reset to either. Open the Current Controller and change its integrator's reset as was done in the Speed Control. Connect the reset inputs of speed controller and current controller as shown in Fig. 5.6. These changes allow the integrators to start up correctly in the real-time environment.

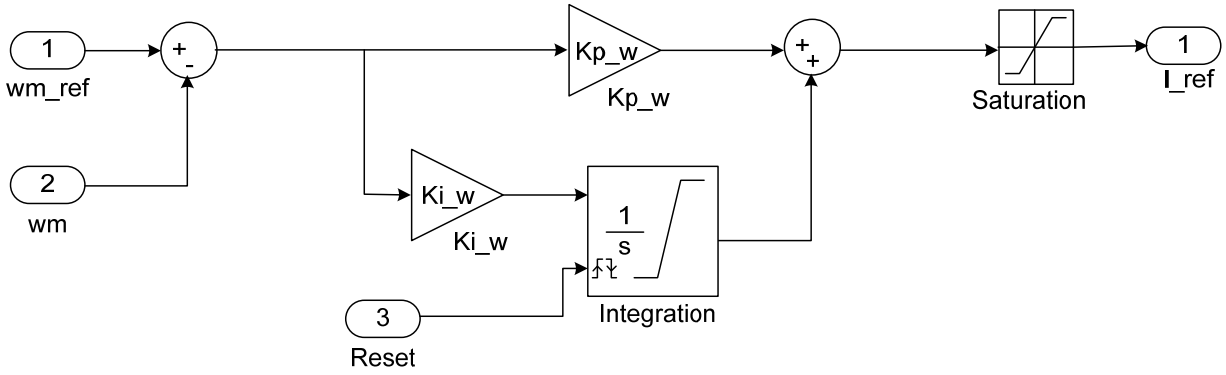


Figure 5.6: Speed Controller

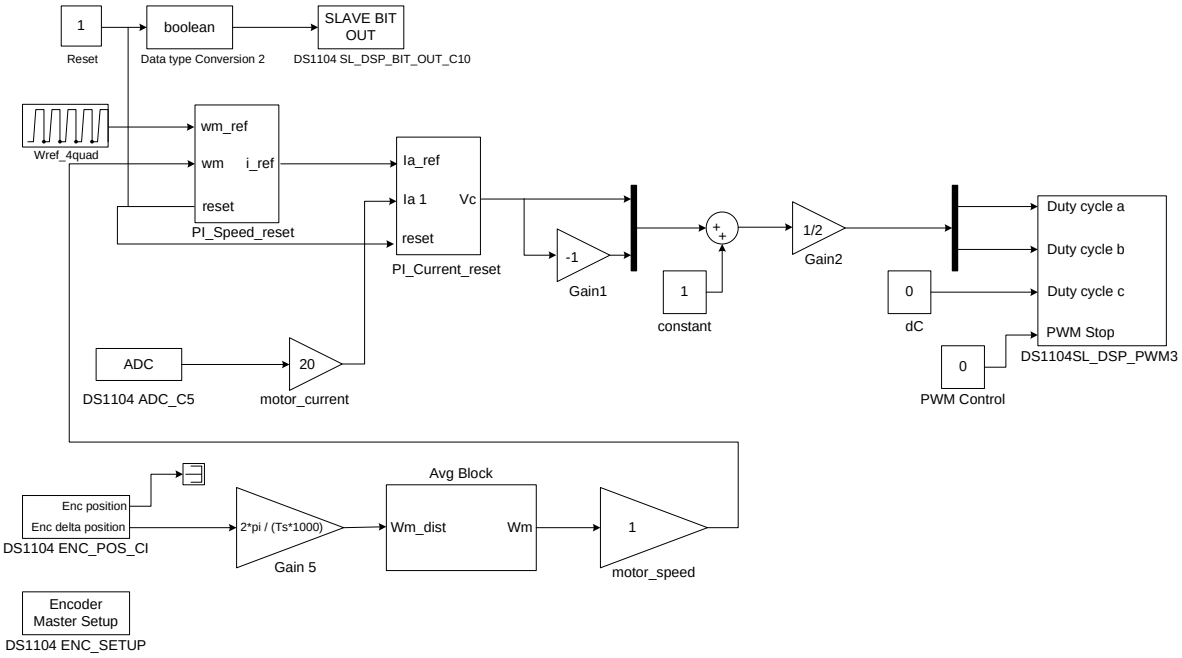


Figure 5.7: Simulink model for real-time implementation of DC motor control

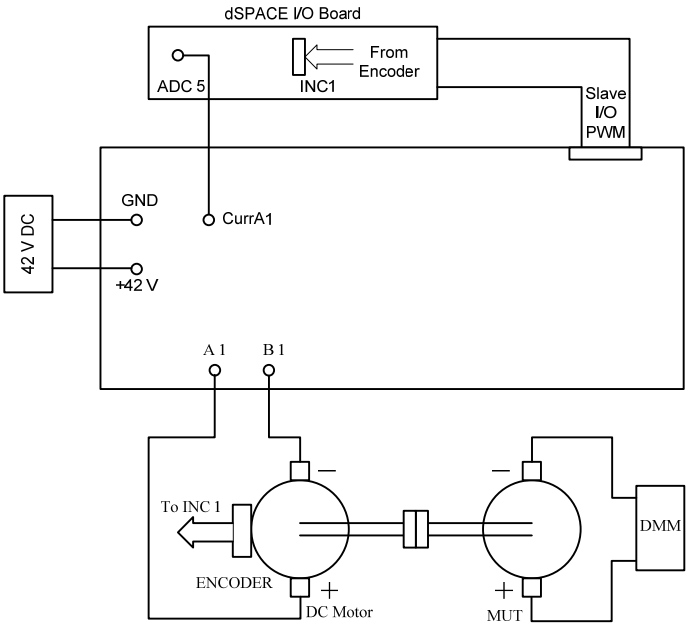


Figure 5.8: Connections on the board

c) Remove the DC-motor mask model and gain Kpwm block.

d) Copy and paste the duty-cycle calculator from the Simulink model used in previous experiments.

e) The current and the speed are to be measured. For measurements use the blocks already designed in previous experiments.

f) Replace the speed ref, wm_ref_step and sum block with a constant block for setting the speed reference.

- At the Matlab prompt, set the sampling time Ts=0.0001 and the dc-bus voltage at Vd=42V. Also set the values of various variables you have defined in the model.

- Set the Simulation→Configuration Parameters

→Solver→ Start time=0, Stop time =inf

Type: Fixed-step , Solver: ode1(Euler)

Fixed-step size:1e-4

→Optimization→in Simulation and code generation, uncheck everything except

'Implement logic signals as Boolean data'

→Real-Time Workshop→System target file→rti1104.tlc

- Fig 5.7 is then the simulation block for DC-motor control. Make the connections on the board as shown in Fig. 5.8. The block wref_4quad gives a periodic step change in reference speed from 100 rad/sec to 300 rad/sec.

- Build (CTRL+B) the model and start dSpace Control Desk.

- Open the variable file (.sdf) and then open the layout file '**dc_motor_speed_control.lay**' and add values as shown in Fig.5.9.

- Run the experiment and compare the real-time results with the simulations.

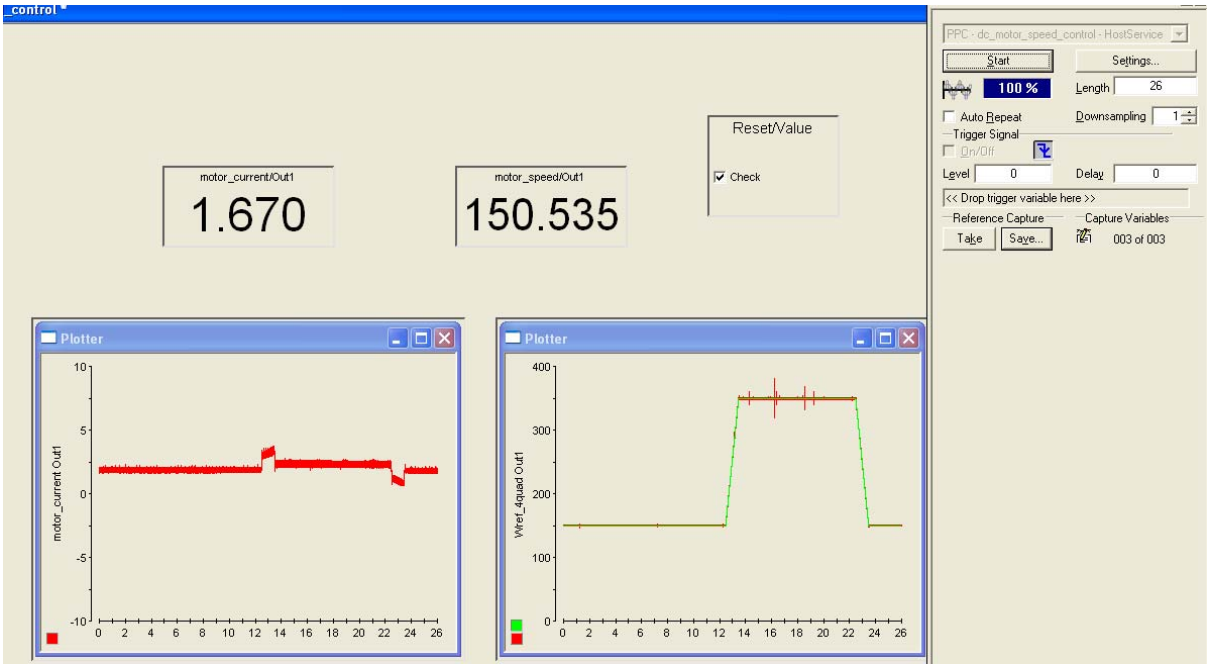


Figure 5.9: Control-desk interface for DC motor control for step from 150 rad/sec to 350 rad/sec

5.5 Lab Report *(10points)*

1. Section 5.2: Run the simulation (with default configuration parameters for 10 sec) for the following two cases. Check the observed values with calculated values (show your calculations). Save the plots and include them in your report. *(2points)*

a) Va=20V, Load_Torque = 0.3 Nm

b) Va=20V, Load_Torque = 0 Nm

2. Section 5.3:

a) Report your calculations for current and speed controller. Note: Design current controller for a bandwidth of 100 Hz (phase margin 90 deg) and speed controller for a bandwidth of 10Hz (Phase margin 60 deg). For designing the speed controller you can assume B=0 but while building the Simulink block, include B. *(2 points)*

b) Attach simulation output of current response for a step input of 1A. *(1 point)*

c) Attach simulation output of speed response for a step input of 200 rad/s from a constant value of 100 rad/s *(1 point)*

d) All the responses required for questions 2b) and 2c) are based on step change in reference signals. These are required more for design purposes. In practical applications, it is more important to know how the system responds to disturbances in load torque. In the simulation, give a step load torque of 0.3 N-m while maintaining a constant speed of 200 rad/s. Observe the response in current and speed and attach the plots. *(1 points)*

3. Section 5.4: Attach the speed and current response for a step change in speed reference as observed through control-desk for a step change from 100 rad/sec to 300 rad/sec. *(3 points)*

5.6 References

[1] “ELECTRIC DRIVES an integrative approach” by Ned Mohan, 2000, MNPERE.

Experiment-6
Four-Quadrant Operation of DC-motor

6.1 Introduction

In the previous experiment, you controlled the motor to run under speed control. The objectives of this experiment are,

1) To observe the four-quadrant operation of a DC motor.

2) To control a motor under torque-control.

3) To couple the speed control motor and torque controlled motor, and observe the effect of a stepped torque.

6.2 Four quadrant operation of a DC motor

The four-quadrant operation is performed by giving an alternating reference-speed command to the DC-motor, from positive speed (200 rad/sec) to negative speed (-200 rad/sec) with a constant ramp. The speed controller designed in the previous experiment is used to track the instantaneous reference-speed command. Fig. 6.1 shows the Simulink model file of the system, and Fig 6.2 shows the hardware connections. To keep the same inertia the same as the ones used to design the speed-control-loop in the Experiment-5, another DC-motor is coupled to the DC-motor whose four-quadrant operation is desired. The terminal voltage Va, current Ia, and speed & torque relations are given below.

$$I_a = \frac{V_a - E_b}{R_a}; \qquad E_b = k_E.\omega_m \tag{1}$$

$$T_{em} = k_T I_a; \quad k_T = k_E \tag{2}$$

Real-time implementation

Modify the file obtained from Experiment-5, which was used for speed control of the dc motor to give a stepped waveform that goes from -200 rad/sec to -200 rad/sec (Fig 6.1). **Save the waveform for the reference speed and measured speed and label the quadrant of operation.**

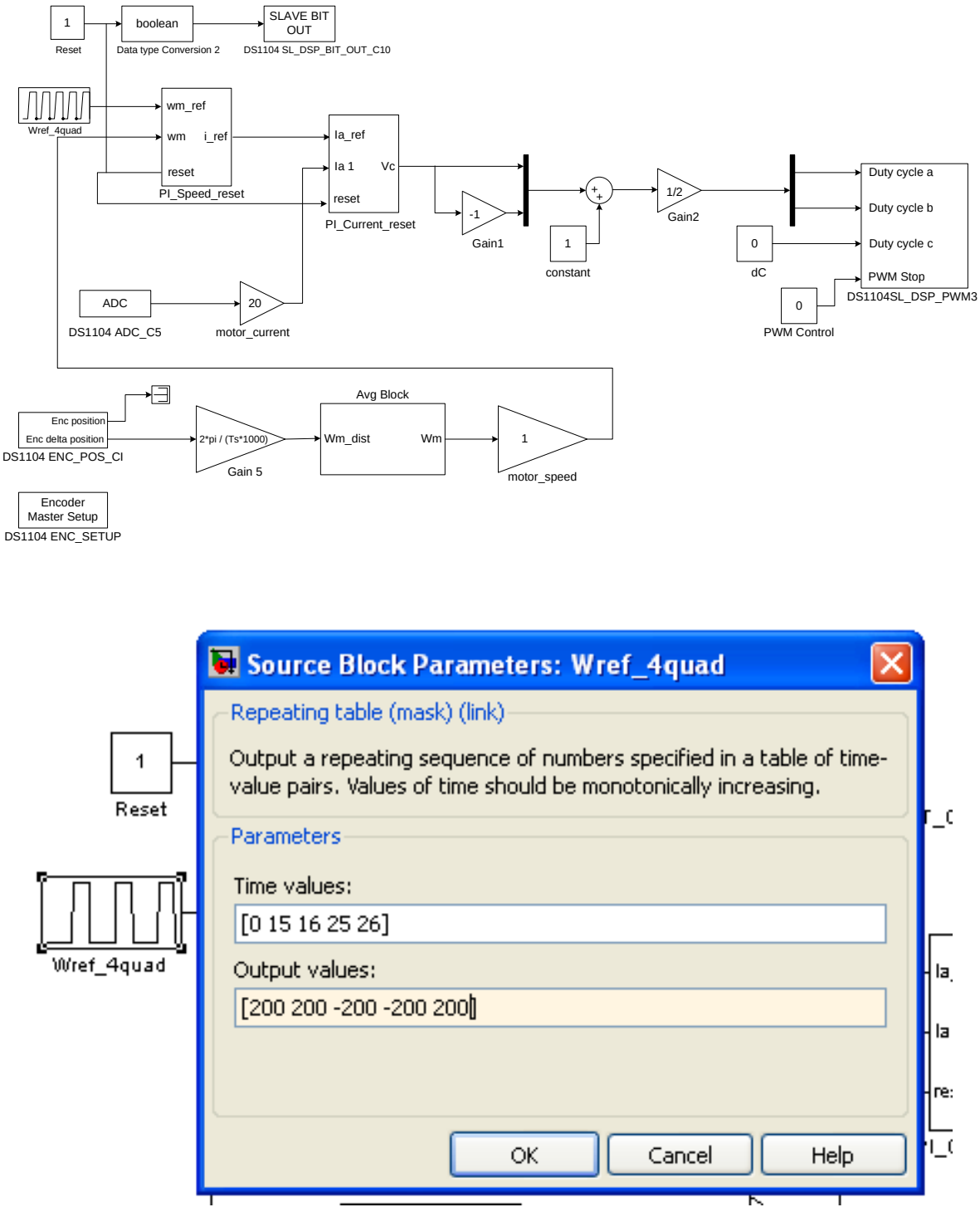


Figure 6.1: Simulink model for four-quadrant operation of DC motor

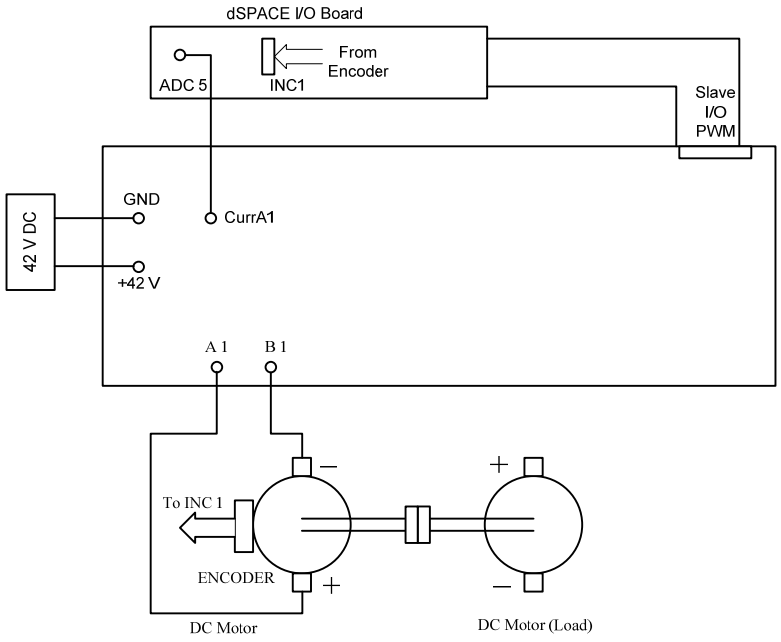


Figure 6.2: Connections for four-quadrant operation of DC motor

6.3 DC motor under torque control

Part a: Modeling of a constant load

The Simulink model for constant torque on DC motor is shown in Fig. 6.3. Open the file **'torque_control.mdl'.** Assign suitable values to the PI current controller that was obtained from the previous experiment-5. Enter the value of Vd = 42 and Ts=1e-4 at command prompt. Build the model (Ctrl+B) and start dSpace ControlDesk. Open the (.sdf file) and the layout file "**torque_control.lay**" setup the capture setting window (View→Controlbars→Capture setting window) as shown in Fig 6.4 and the connections are shown in Fig. 6.5.

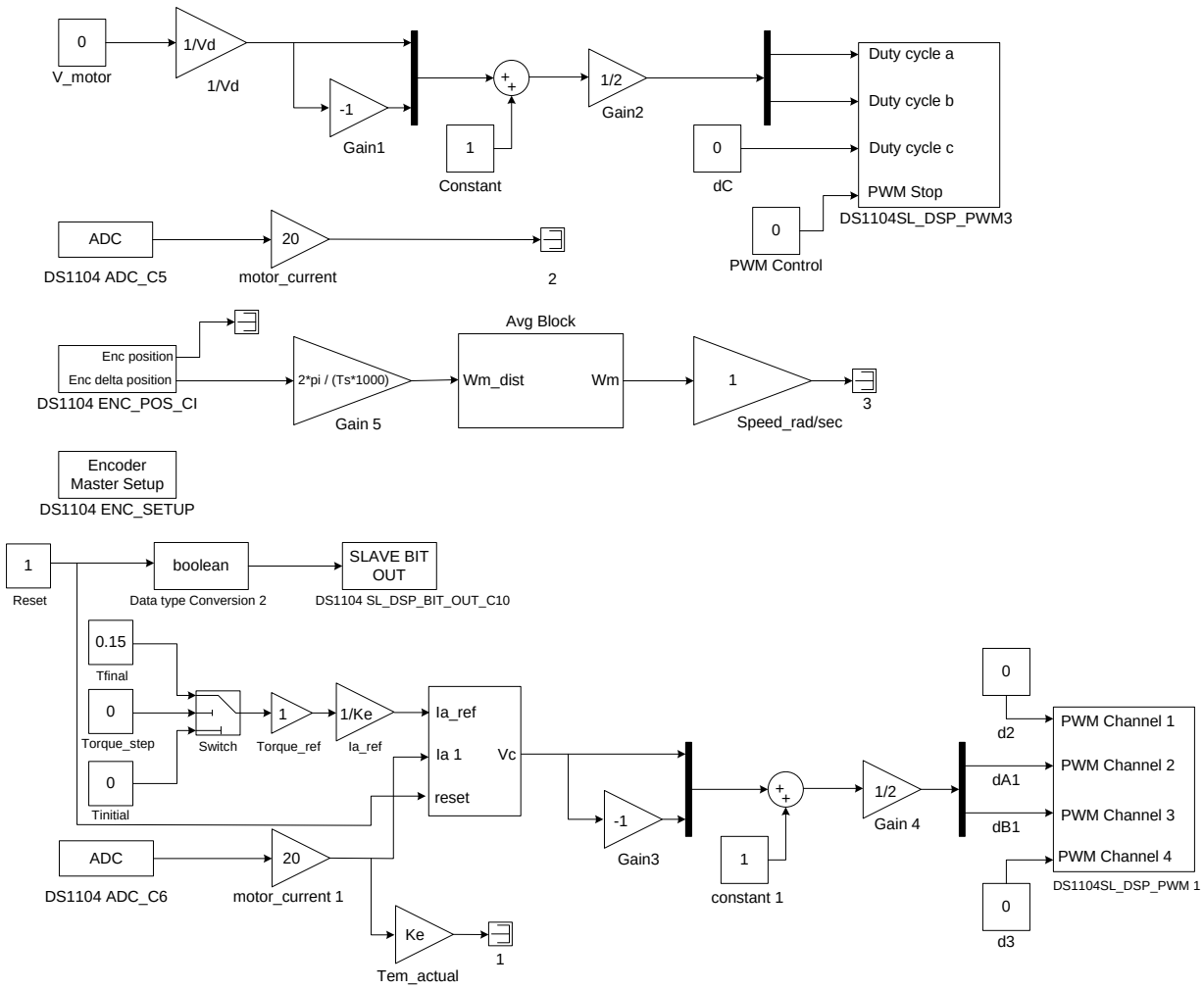


Figure 6.3: Simulink model for constant torque on DC motor

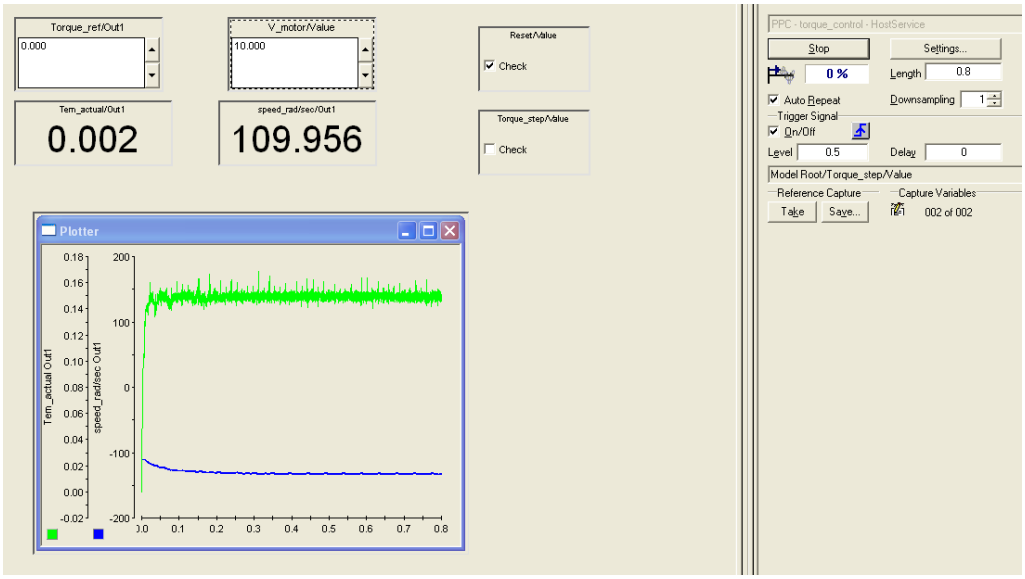


Figure 6.4: dSpace ControlDesk layout for constant torque on DC motor

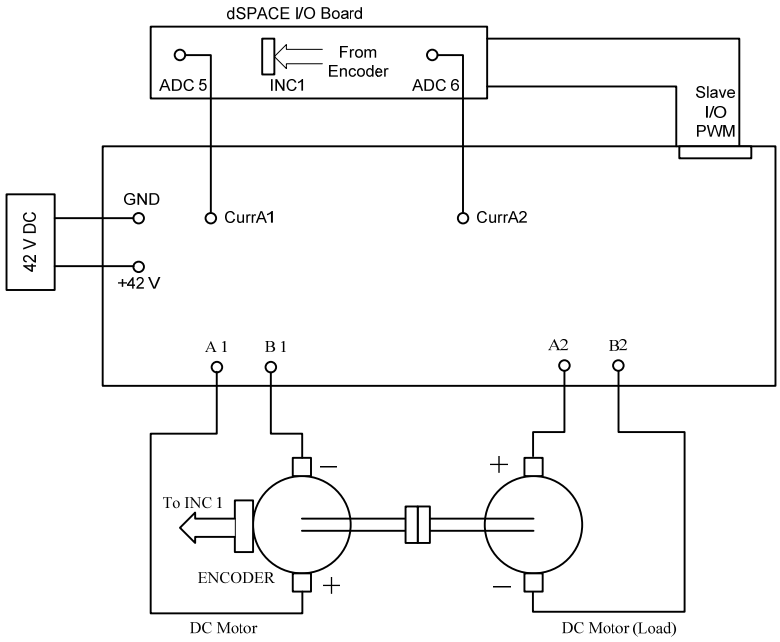


Figure 6.5: Connections for constant torque on DC motor

1. Apply a constant voltage to the dc motor of value **12V.**

2. Note down the speed of motor when the load torque is zero.

3. Give a step change in load torque from **0 to 0.15Nm** (check the box Torque_step) and note the resultant speed.

4. Theoretically estimate the speed value in rad/s.

5. What is the net power supplied by the motor? What is the quadrant of operation?

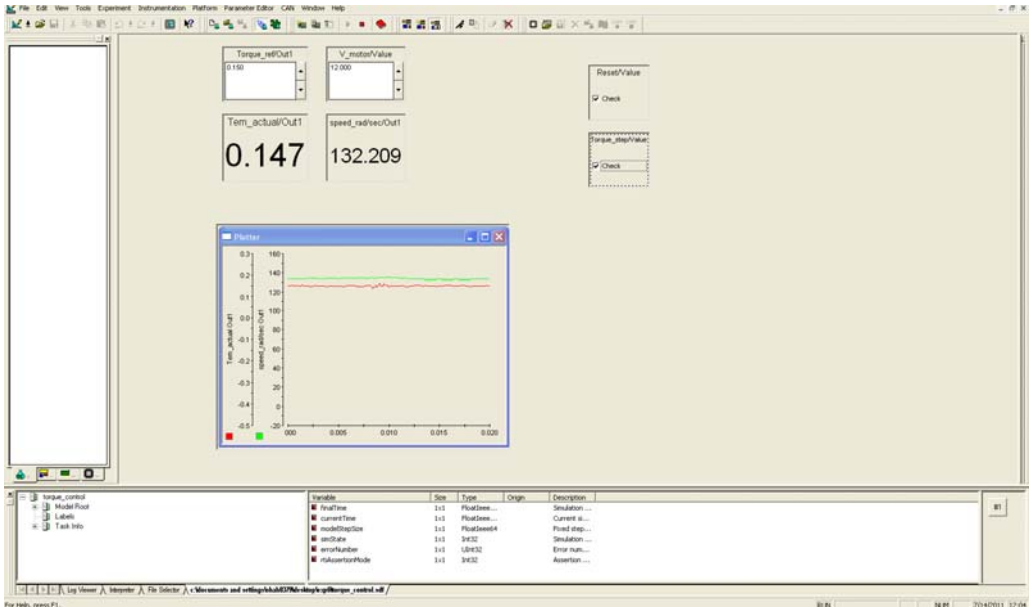


Figure 6.6: Modeling of a constant load

The torque Tem and speed of the motor can be viewed on the controldesk window as in Fig. 6.6.

Part b: Modeling of a crane load

Modeling of a crane load (constant load torque) is done in 'Part a' for lifting action. In this part, the crane is used to lower the heavy load i.e., the load torque retains the same direction but the speed reverses.

1. Set up the experiment as shown in Fig. 6.5. The response of torque and speed can be seen in Fig. 6.7.

2. Use the given simulink model "**torque_control.mdl**" and the layout file "**torque_control.lay**".

3. Apply a constant voltage to the dc motor of value **-12V** to make the motor rotate in opposite direction.

4. Note down the speed of the motor.

5. Give a step change in load torque from **0 to 0.15Nm** and note the resultant speed.

6. Theoretically estimate the speed in rad/s.

7. What is the net power supplied by the motor? What is the quadrant of operation?

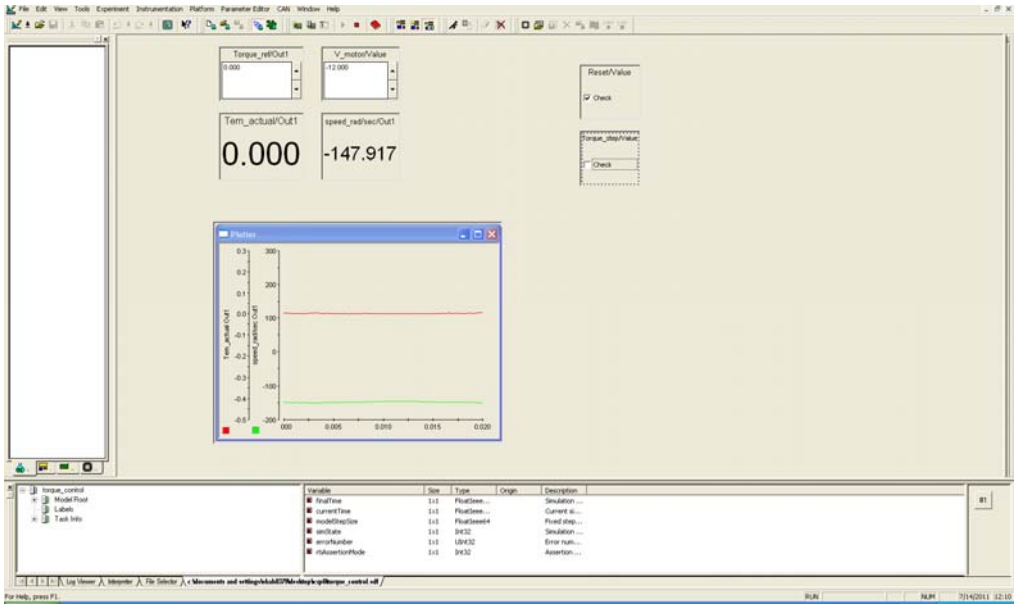


Figure 6.7: Modeling A Crane Load

6.4 Closed loop speed control

In speed control the motor speed is maintained constant irrespective of load torque variations. Let us say you have to maintain the speed of the motor at **150 rad/sec.**

1. Setup the experiment as shown in Fig. 6.5.

2. Open the file **'torque_control_w_control.mdl'.** Your model should look like Fig.6.8. In this model, inverter 1 controls the motor with speed control and inverter 2 controls the second motor (load) with torque control. Note that these motors are identical. Assign the values to all the PI controllers. (run the m file which contains all values, the values of Kp and Ki are obtained from the previous experiment).

3. Build the model (Ctrl+B) and restart dSpace ControlDesk. Open the (.sdf) file and open the layout file **'torque_control_w_control.lay'.**

4. In the capture setting window set the length to 1s and level to 0.5 Drag Torque_step value to the gray box such that Model Root/Torque_step/Value is displayed there. Check On/Off.

5. Ramp up the speed to **150 rad/sec.**

6. Give a step change in load torque from **0 to 0.15Nm** and note the resultant speed. **Capture this waveform** (Fig 6.9.).

7. Theoretically estimate the voltage required to maintain the same speed from the equations (1) and (2).

8. Verify this value displayed in Control Desk. The duty ratio value is displayed in the control desk. Estimate using that.

9. Change the direction of rotation **-150 rad/sec**.

10. Give a step change in load torque from 0 to 0.15 Nm and note the resultant speed.

11. Theoretically estimate the voltage required to maintain the same speed using eq. (1) & (2).

12. Verify this value displayed in Control Desk.

13. **BONUS!** Give a sinusoidal disturbance of 0.05 Nm in load torque over a constant value of 0.15Nm. (use frequency of 1Hz). Observe how the control system responds to the disturbance.

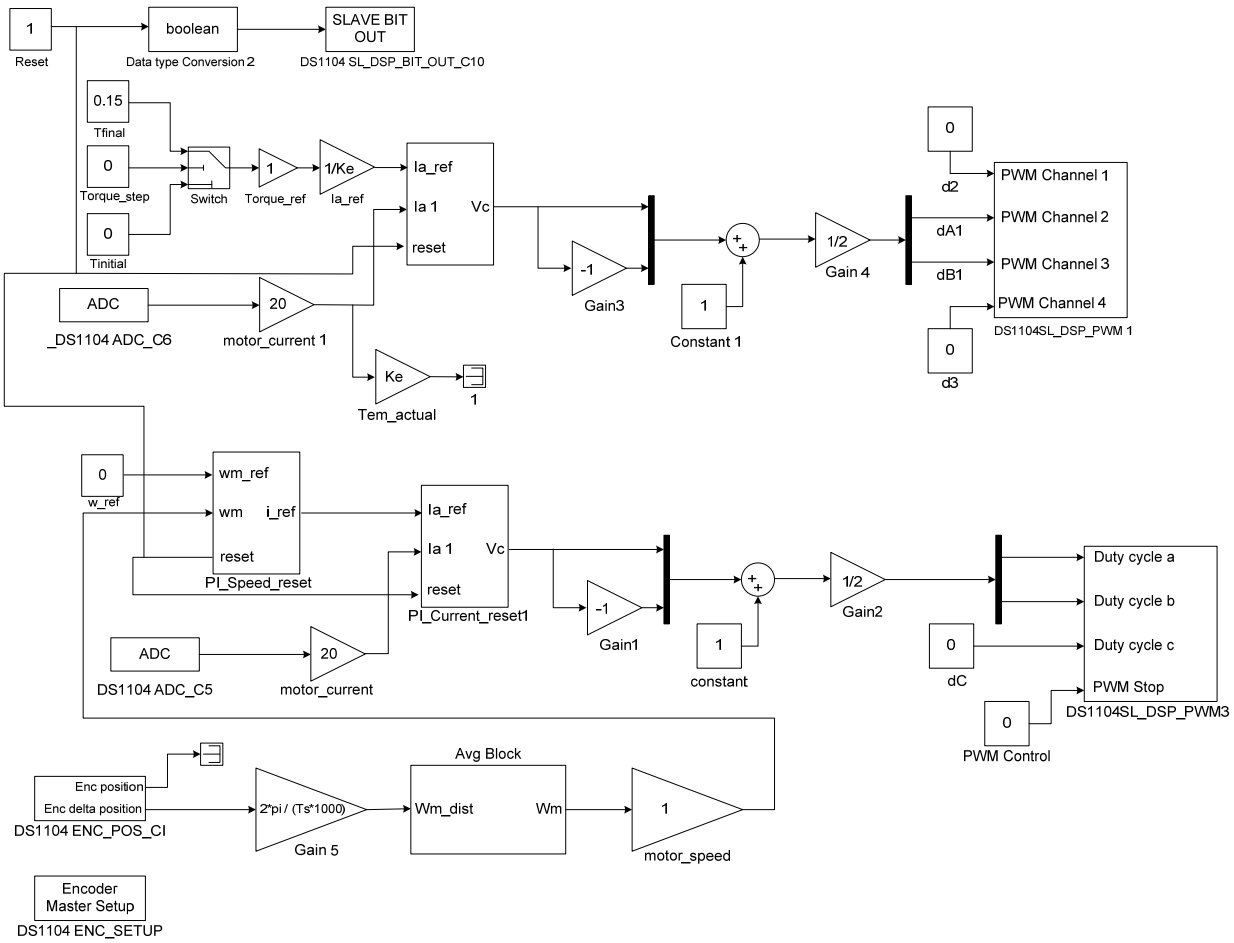


Figure 6.8: Simulink model file for speed controlled DC motor with torque controlled load.

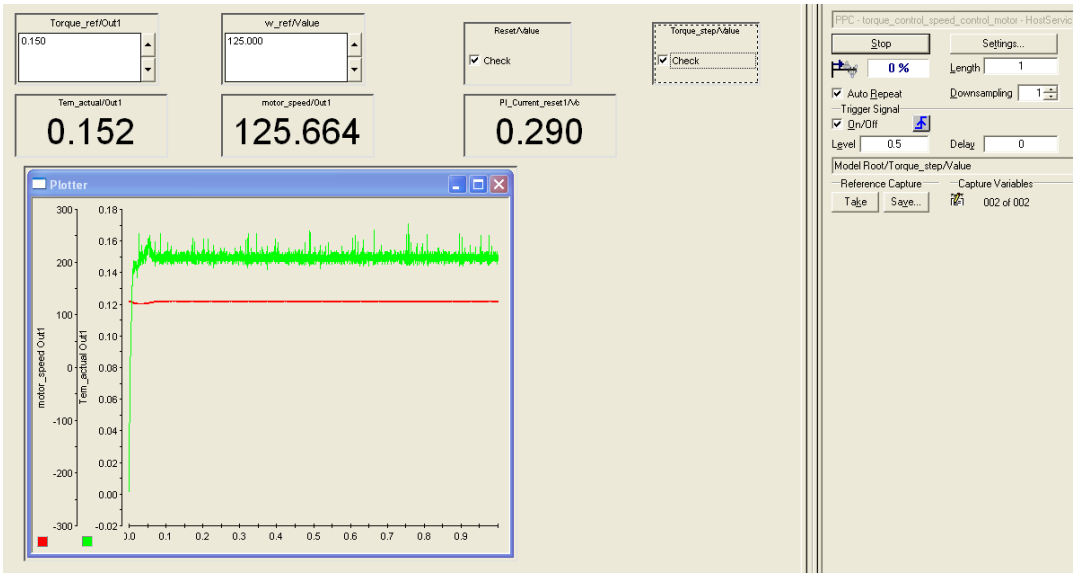


Figure 6.9: dSpace ControlDesk -speed controlled DC motor with torque controlled load

6.5 Lab Report *(10 points)*

1. Section 6.1: Save the waveform for the actual speed and measured speed and label the quadrant of operation. *(2 points)*

2. Section 6.2:

   - Part a → Attach the waveform from dSpaceControlDesk when the torque step is applied to the motor. *(1 points)*

   - Answer Part a → 3 to 6 (including 3 and 6) *(1 points)*

   - Part b → Attach the waveform from dSpaceControlDesk when the torque step is applied to the motor. *(1 points)*

   - Answer Part b → 3 to 6 (including 3 and 6) *(1 points)*

3. Section 6.3:

   - Attach the waveform for 6.3 → 5 and 8 *(2 points)*

   - Answer 6.3→5 to 11(including 5 and 11) *(2 points)*

4. Bonus**!** Keep the speed of the motor constant at 150 rad/sec. Give a sinusoidal disturbance (use frequency of 1Hz) of 0.05 Nm in load torque over a constant value of 0.15Nm. Observe how the control system responds to the disturbance. Attach the waveform of the reference torque, actual torque developed by the motor and the speed of the motor. *(3 points)*

E x p e r i m e n t – 7
Permanent Magnet AC (PMAC) Motor

7.1 Introduction

In this experiment, the vector control of a three-phase Permanent Magnet AC (PMAC) motor will be studied. Real-time Simulink and layout files are given to perform the experiment. The objective would be to understand how maximum electromagnetic torque from the motor is achieved when stator current space vector is maintained perpendicular to the rotor flux vector. This angle is changed in real-time to verify the decrease in electromagnetic torque. This experiment is divided into three parts,

1. Observe the back emf of the motor and calculate its back emf constant.
2. Run the motor with current control by correct placement of the current space vector.
3. Run the motor with speed control.

7.2 Theory - Space Vectors, dq-windings

In a 2-pole motor with sinusoidally distributed stator windings, as shown in Fig. 7.1, the current and voltage space vectors are given by (1) and (2) as in reference [1].

$$\overline{i_s(t)} = i_a(t)\angle 0^o + i_b(t)\angle 120^o + i_c(t)\angle 240^o \quad (1)$$

$$\overline{v_s(t)} = v_a(t)\angle 0^o + v_b(t)\angle 120^o + v_c(t)\angle 240^o \quad (2)$$

For dynamic analysis and control of ac machines, two orthogonal axis (d & q) can be defined such that windings along these two axis can generate the same mmf as the three sinusoidally distributed windings [2]. These dq-windings can be at some arbitrary angle θ_{da} with respect to the phase-a axis. In this case, the d-axis winding is taken to be along the rotor axis. The transformation from abc-frame to dq-frame is given by (3) for the currents. The same equation applies for voltages as well, where θ_{da} is the angle between the rotor and the a-axis.

$$\begin{bmatrix} I_d \\ I_q \end{bmatrix} = \sqrt{\frac{2}{3}} \begin{bmatrix} \cos(\theta_{da}) & \cos\left(\theta_{da} - \frac{2\pi}{3}\right) & \cos\left(\theta_{da} + \frac{2\pi}{3}\right) \\ -\sin(\theta_{da}) & -\sin\left(\theta_{da} - \frac{2\pi}{3}\right) & -\sin\left(\theta_{da} + \frac{2\pi}{3}\right) \end{bmatrix} \begin{bmatrix} I_a \\ I_b \\ I_c \end{bmatrix} \quad (3)$$

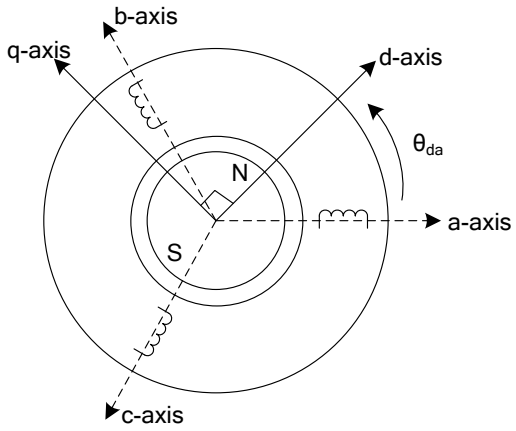


Figure 7.1: Two-pole PMAC Machine

7.3 Observing the Back-emf of the PMAC Motor

In this section, the PMAC motor is run as a generator by coupling it to a DC motor as shown in Fig. 7.2.

- Make connections are as shown in Fig. 7.2.

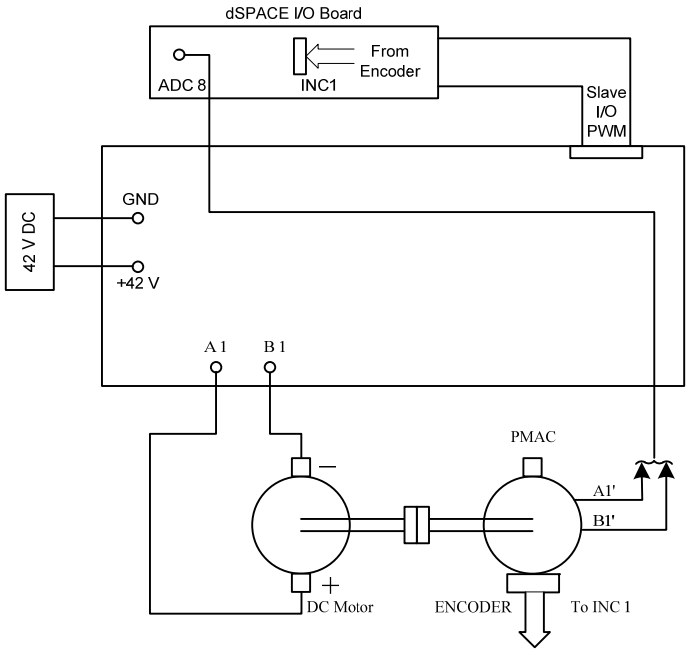


Figure 7.2: Connections for Section 7.3

- Connect A1' and B1' phases of the PMAC motor to ADC_8 (for measuring E_{ab}) and make sure that the third terminal of the PMAC is NOT connected anywhere (**NOTE**: For some motors (A1', B1') correspond to Red and Yellow and in some cases we need to check the phase sequence of the motor.) The phase sequence of the motor can be checked by applying a small voltage (+1V) between phases in order AB, BC and CA, if the speed if positive (counter clock wise), this is the correct sequence (ABC). If the speed is negative (clockwise) then interchange any two phases.

- Download and extract the Lab 7 files to the desktop. Start Matlab and set the working directory to this folder. **Open PMAC_pole_Det_index.mdl**. Set Vd=42, Ts = le-4 in the Matlab prompt. Build this model (Ctrl+B)

- Open dSpace ControlDesk and open the PMAC_pole_Det_index.sdf file and the corresponding pmacpole_indexdet.lay file.

- In capture setting window set the length to 2s.

- Apply a voltage to the DC motor such that the speed is positive. Observe the Speed encoder position and back emf of the PMAC Motor.

- **NOTE**: The ADC can measure a maximum of 10V, the back emf must not exceed this, hence do not run the motor faster than 50 rad/sec (±5V).

- Save the waveform from ControlDesk as motor_emf.mat and use the code in PMAC.m to calculate the value of theta_initial and back_emf constant, Ke.

- For example see the screen shot in Fig 7.3 where the length has been set to 1.5 in the capture setting window.

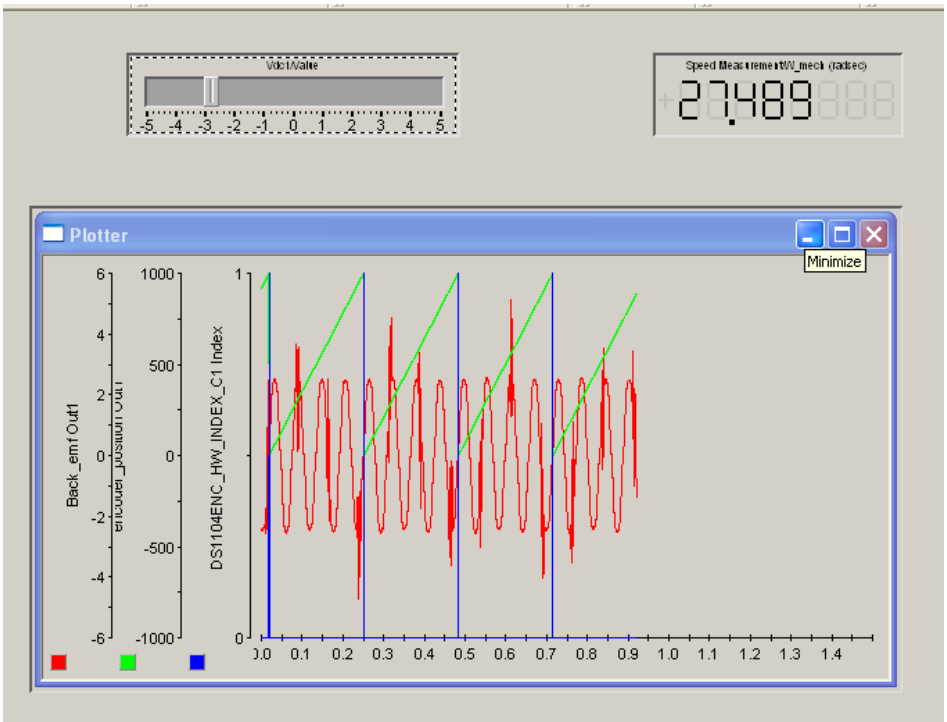


Figure 7.3: Screen Shot of ControlDesk

7.3.1 Observation of the index-pulse

The encoder generates 1000 pulses per rotation, each time the motor completes one rotation an index-pulse is generated that resets the encoder position value back to zero.

7.3.2 Observation of the back-emf

For one complete mechanical rotation of the motor, the number of electrical cycles completed is given by p/2, where p is the number of poles of the motor. In this lab motors have 8 poles. **Count the number of cycles between two index pulses to determine the number of poles of the motor.**

7.3.3 Determine the value of $\theta_{initial}$

The equations to determine the value of $\theta_{initial}$ are given in PMAC.m file from lines 29 to 39. The explanation is given below.

In order to control the PMAC motor, it is essential to know the exact position of the rotor. This position can be calculated using the index pulse of the encoder. Depending on the installation of the encoder, the index pulse may not be aligned with B+. Let the zero-crossing of E_{ab} be

considered to be the reference 0 as shown in Fig. 7.4. The index pulse is observed only at θ_{index}. Hence, when calculating the value of θ_{da}, an initial offset ($\theta_{initial}$) must be added. For a positive direction of rotation, the value of $\theta_{initial}$ is given by θ_{da+} in Fig. 7.4 (the angle between the positive peak of B and the index pulse) and for a negative direction of rotation it is given by θ_{da-} (the angle between the negative peak of B and the index pulse). The values of θ_{da+} and θ_{da-} can be calculated using (4) and (5) respectively. **NOTE**: E_a lags E_{ab} by 30^0 and B lags E_a by 90^0.

$$\theta_{da+} = -(\theta_{B+} - \theta_{index})...\theta_{B+} = \pi + \pi/6 \quad (4)$$

$$\theta_{da-} = -(\theta_{B-} - \theta_{index})...\theta_{B-} = \pi/6 \quad (5)$$

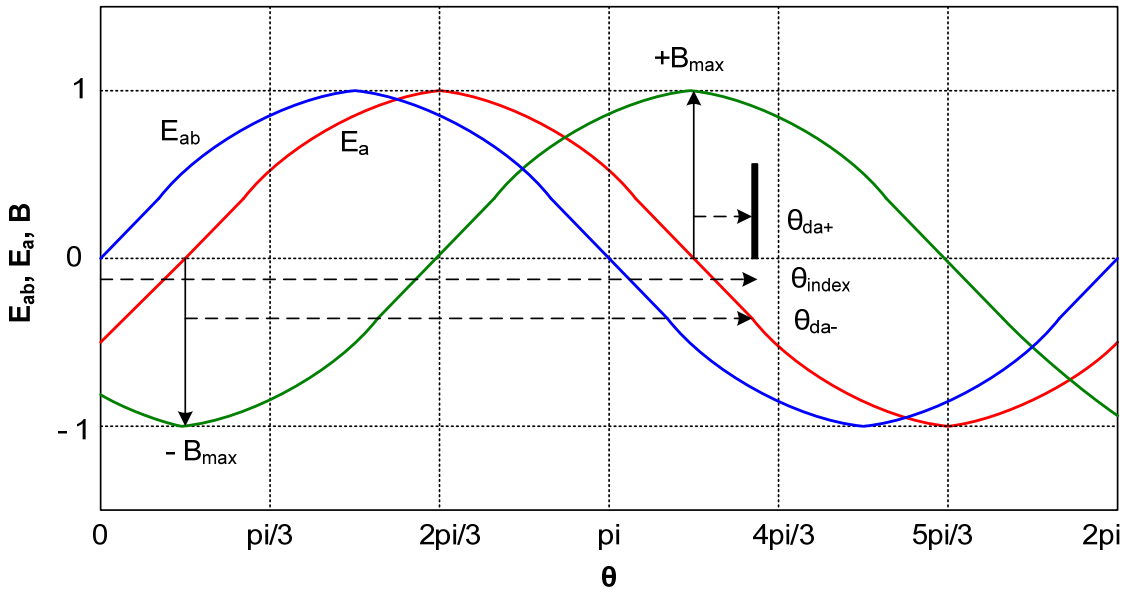


Figure 7.4: Determination of $\theta_{initial}$ when PMAC has positive speed

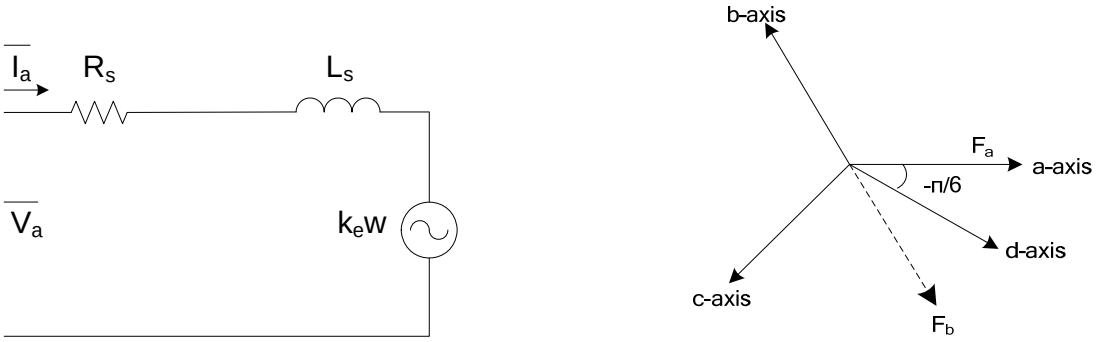
7.4 Current Controlled PMAC Machine

7.4.1 Theory

The equivalent circuit of the PMAC motor after dq transformation is shown in Fig. 7.5(a). The parameters of the motor are listed in Table. 1. The current and speed controllers can be designed using the steps in experiment 5. Or use the controller values given in Table. 2. The simulink model for current control of PMAC motor is Fig 7.7 .The angle theta_da is calculated using the enc_position and theta_initial. If theta_initial is set to theta_da+ or theta_da-, the d axis is aligned along the rotor axis. Before running the PMAC motor, the rotor is initialized to a known

position (done by checking lock). To initialize the motor, a small voltage of 1V is applied between phases A1' and B1'. The dc currents in phase-a and phase-b produce mmf vectors as marked in Fig. 7.5(b). This causes the rotor (thus the d-axis) to align $-\pi/6$ away from the physical a-axis. When lock is unchecked the a and b phase currents are sensed and the c phase current is calculated from it, an abc-dq transformation is applied (eq 3) to obtain i_{sd} and i_{sq}. The error between the actual currents and the ref currents are passed through a PI controller that dictates the necessary phase voltages.

The initial voltage can now be removed, and a reference current that is aligned with the q-axis (i_{sq}) is applied to the motor to generate the maximum torque.



(a) Per-phase equivalent circuit of PMAC (b) Rotor Locking Position Machine

Figure 7.5: Current Controlled PMAC Machine

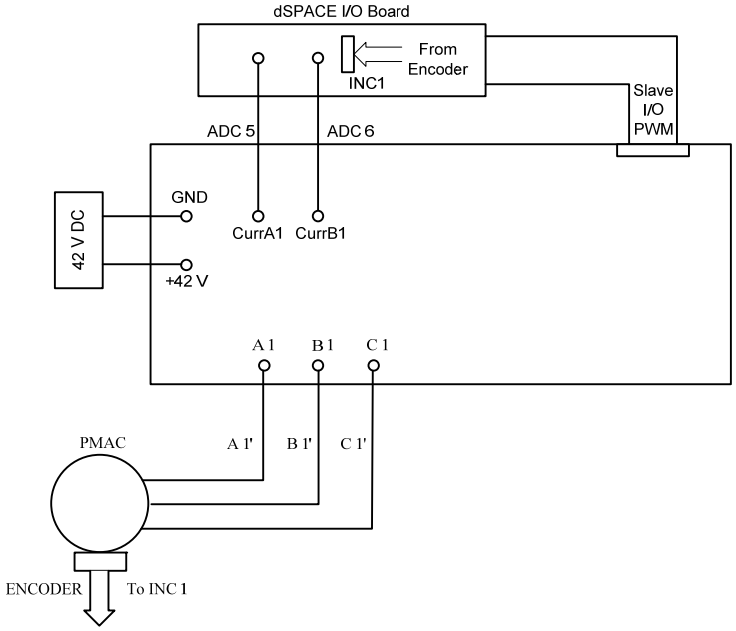


Figure 7.6: Connections for Section 7.4 and 7.5

7.4.2 Real-time implementation of current controlled motor

- Make the connections as shown in Fig. 7.6.

- Open the file **PMAC_current.mdl.** Obtain the values for Kp_i and Ki_i from Table. 2. of this experiment. Enter the values of Kp_i, Ki_i, theta_intital , Vd=42, poles=4 and Ts=le-4 at the MATLAB command prompt. The model can be seen in Fig 7.7.

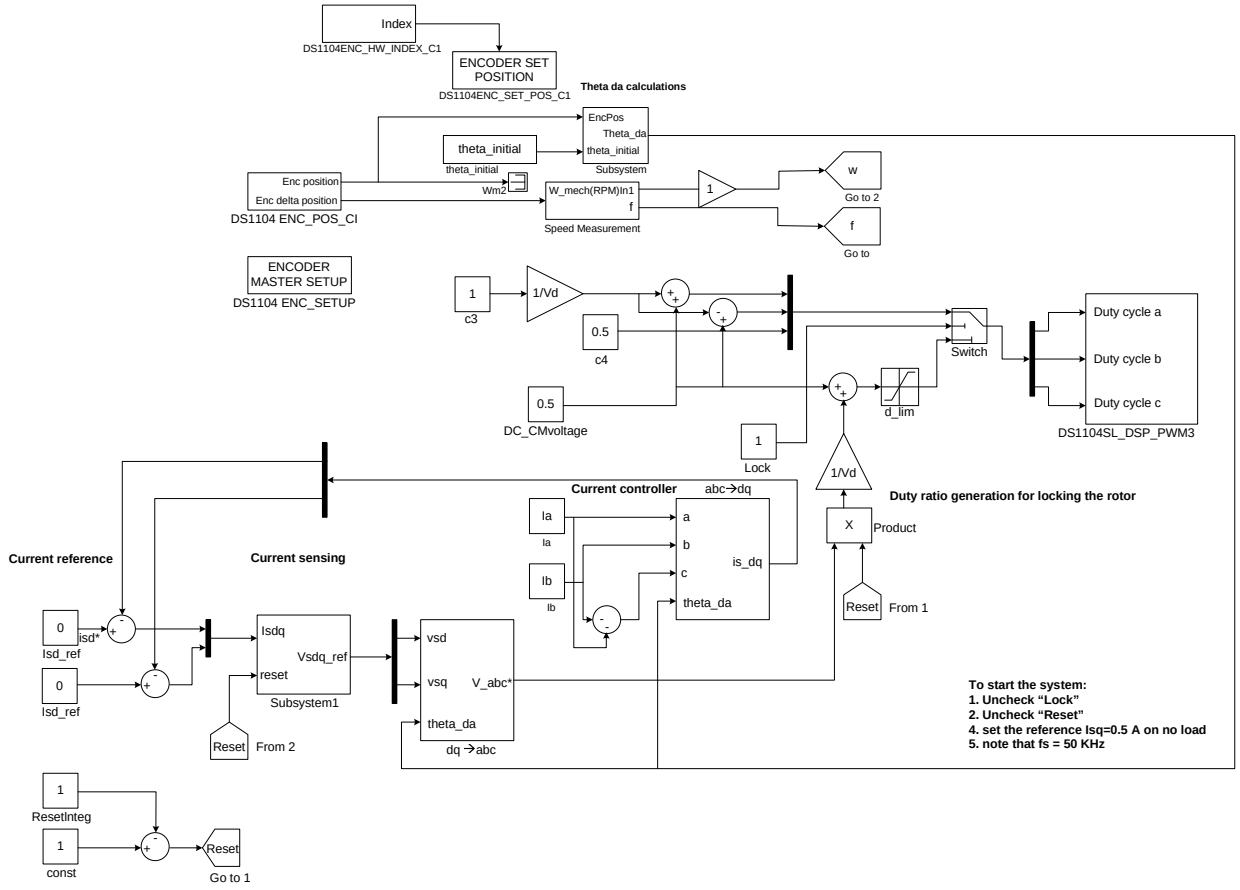


Figure 7.7: Simulation model PMAC_current.mdl

- Build (Ctrl+B) the .mdl file.

- Open dSpace ContrlDesk and open the variable file **PMAC_current.sdf.** Open the file **PMAC_current.lay.**

- Uncheck "Lock".

- Uncheck “Reset”.
- Increase the value i_{sq}, the ‘q’ component of the stator current to 0.5.
- Observe how the speed changes with change in $i_{sq.}$
- Vary the value of theta_initial by 10% and observe the change in speed. Does it increase or decrease?
- Check if it’s possible to reverse the speed of the motor without changing the value of theta_initial?

7.5 Run the motor with speed control (Optional)

In this section, the speed controller generates a reference value for $i_{sq.}$ The speed controller can be designed using the parameters in Table. 1 or just the values of Kp_w and Ki_w listed in Table. 2. In the provided model file, the direction of rotation is taken positive, the value of theta_initial must be the one that corresponds to positive speed.

Table1: PMAC Motor Parameters

Parameter	Value
Ra	0.6253
L	4.4797e-4
Ke	0.0924
J	0.5e-3
B	0.0002

Table 2: PMAC Motor – PI Controller Values

Parameter	Value
Ki_i	37.4179
Kp_i	0.0268
Ki_w	10.6814
Kp_w	0.2944

1. The connections for the real-time implementation are in Fig. 7.6.

2. Open the file **PMAC_current.mdl.** Enter the values for Kp_i, Ki_i, Kp_w, Ki_w, theta_intital Vd=42, poles=4 and Ts=le-4 at the MATLAB command prompt. The model is shown in Fig 7.8.

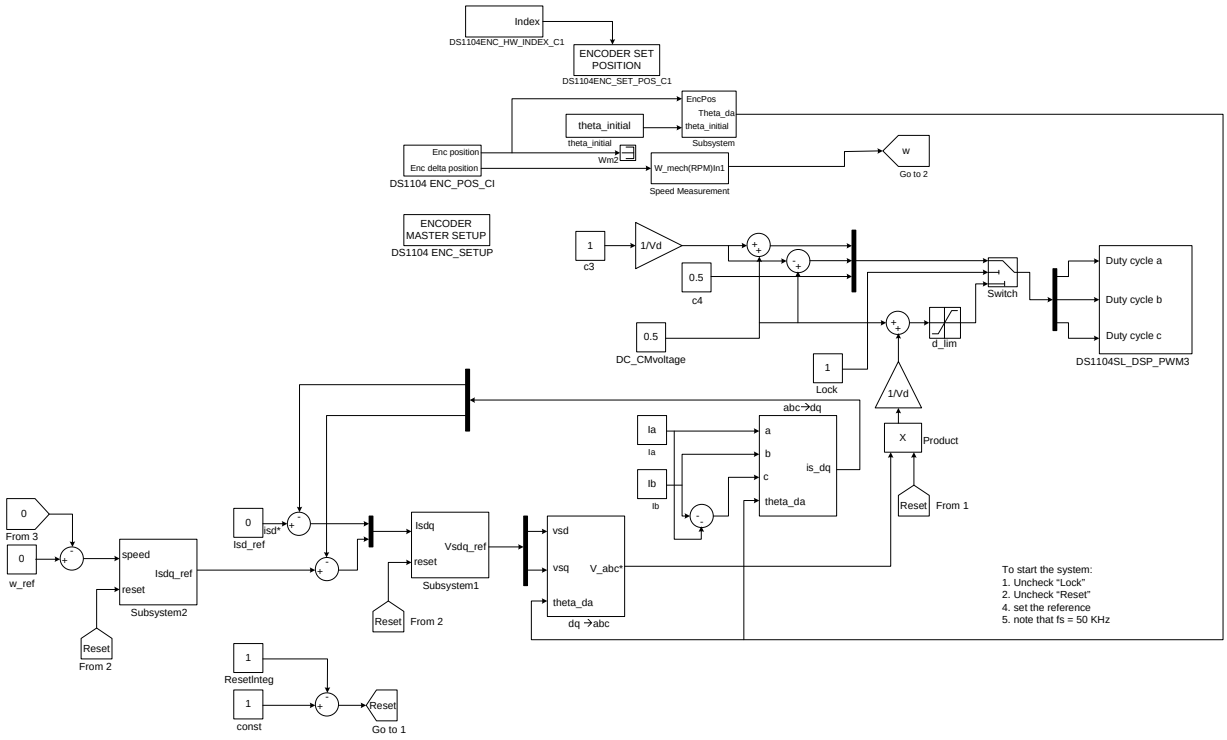


Figure 7.8: Simulation Model

The model is shown in Fig 7.8. In this model, the speed of PMAC can be controlled. The error between the measured speed and actual speed is passed through a controller. This controller generates a reference value for i_{sq}. The PI controller for speed is designed with a cut-off frequency of 10Hz, which is much slower than the current control loop.

3. Build (Ctrl+B) the .mdl file.

4. Open dSpace ContrlDesk and open the variable file **PMAC_current.sdf.** Open the file **PMAC_current.lay.**

5. Uncheck "Lock".

6. Uncheck “Reset”.

7. Change the value of W_ref and check if motor is tracking the reference speed.

8. Change the reference speed to go from 500 RPM to -500 RPM. Capture this waveform for the report.

9. For example see the screenshot of dSpace control desk in figure 7.9.

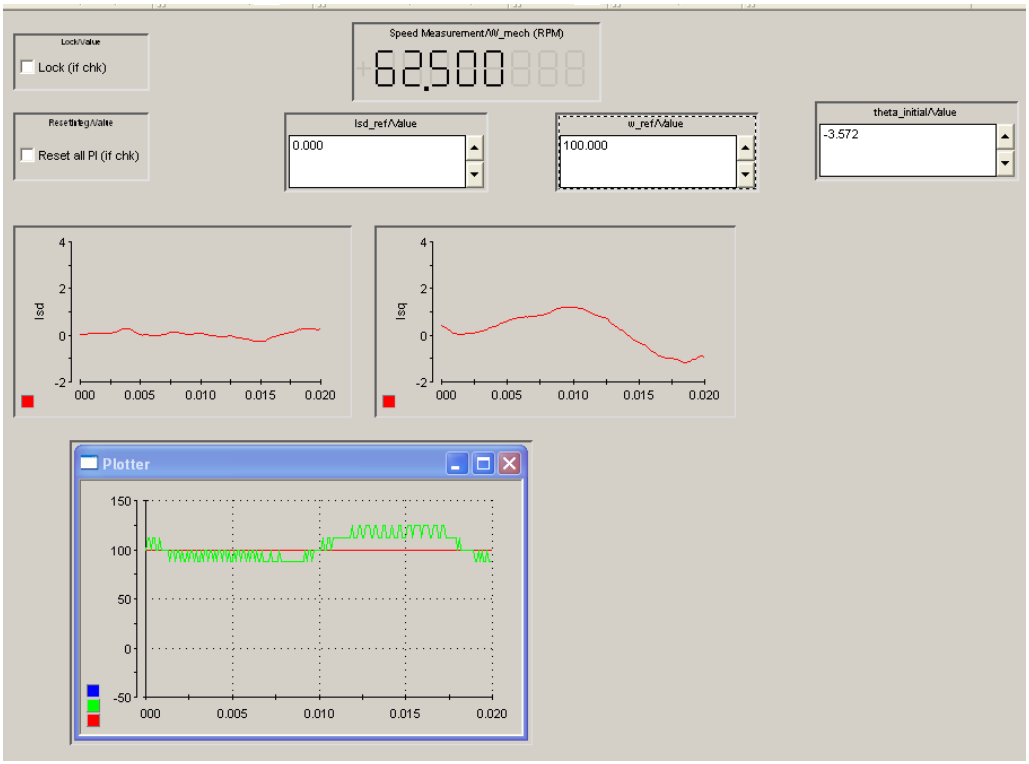


Figure 7.9: Screen Shot of dSpace control desk.

7.6 Lab Report (10 points)

1. Section

   (a) Attach the waveform of the back emf and index pulse *(1 point)*

   (b) Count the number of cycles between two index pulses to determining the number of poles of the motor. *(1 point)*

   (c) What is the value of theta_initial for the rotor to move in the positive direction? *(1.5 point)*

(d) What is the value of theta_initial for the rotor to move in the negative direction? *(1.5 point)*

2. Section 2:

   (a) What is the speed of the motor for a current reference i_{sq} of 0.5? Attach the waveform. *(1 point)*

   (b) Change the value of theta_inital by 10% does the speed increase or decrease? *(1 point)*

3. Section 5:

   (a) Give a varying speed reference from 500 RPM to -500 RPM. Attach the waveform. (*1 point)*

4. List two advantages of PMAC motors. *(2 points)*

7.7 References

[1] Ned Mohan. Electric Drives, An Integrative Approach. MNPERE, 2000.

[2] Ned Mohan. Advanced Electric Drives, Analysis, Control and Modeling using Simulink. MNPERE, 2001.

E x p e r i m e n t – 8
Determination of Induction Machine Parameters

8.1 Introduction

In this experiment, a three-phase induction motor will be characterized to determine the various parameters used in its per-phase equivalent circuit. The circuit diagram for this experiment is shown in Fig. 8.1, where a DC motor is coupled to the induction motor under test. DC resistance test will be done to determine the value of Rs. The magnetizing inductance ($L_m >> L_{ls}$) will be calculated by running the induction motor at synchronous speed, at rated-voltage and rated-frequency. Speed of the DC motor (coupled to the induction motor) will be controlled, to run the induction motor at synchronous speed. The rotor circuit parameters i.e. L_{lr} and R'_r will be calculated by blocked-rotor test while injecting slip frequency at the stator terminals.

Note: 1. The induction motor can get hot especially in the blocked rotor test. Please turn off the power when you are not taking any readings.

Note: 2. The speed is measured from DC motor.

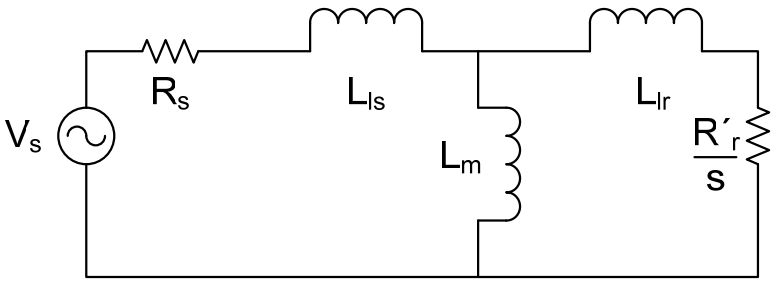


Figure 8.1: Per-phase equivalent circuit of a three-phase induction motor

Download the file 'Lab 8 Summer 2011.zip' and extract its contents to a folder on the desktop. Start MATLAB and change the working directory to the folder containing all the extracted files.

8.2 Determine Rs

Measure the resistance between any two terminals of the induction machine and the resistance of the multi-meter leads. Subtract the resistance of the leads to get the resistance between the two terminals which is twice of Rs. Calculate the value of Rs and enter it in the 'm-file' **(im parameters.m).**

8.3 Determine L_m

8.3.1 *Check the phase sequence of the induction motor*

- Connect the circuit as shown in Fig.8.3. The machines terminals have to be connected as: RED-A1, YELLOW-B1, BLUE-C1.

- Enter Vd=42, Ts=1e-4 at the Matlab command prompt.

- Open the file **ParameterDetermination.mdl** and build (Ctrl+B) this file as in Fig 8.2.

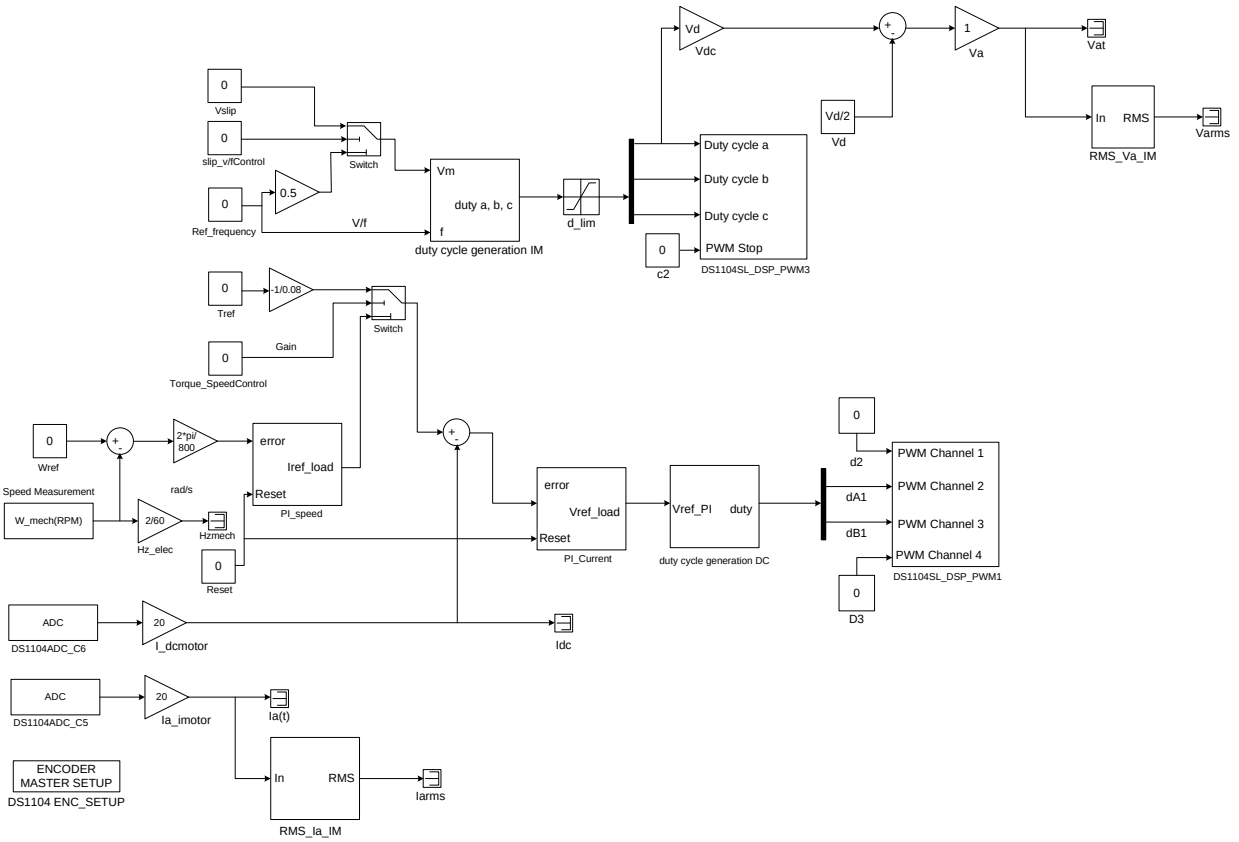


Figure: 8.2 Simulation Model ParameterDetermination.mdl

The simulink model is in Fig 8.2. The experiment can be configured to operate the motors in the modes given below:

Induction motor: a) V/f control b) Independent command for V and f. DC Motor: a) Torque Control b) Speed Control.

- Start dSpace ControlDesk. Open the variable file **ParameterDetermination.sdf** and open the layout file **ParameterDetermination.lay**. Check 'Torque_SpeedControl' so that the DC motor runs under torque control.

Reset	Slip_v / f Control	Torque_SpeedControl
☐	☐	☑

- Increase the value of ref_frequency slowly upto 30 Hz. If the dc-motor speed (w_mech(RPM)) is negative, the phase sequence is correct. If the speed is positive, interchange any two terminals (after bringing the speed back to zero) of the induction motor.

- **Note down the speed of the motor. Can you estimate the number of poles of the induction motor?**

- Reduce ref_frequency back to zero and stop the experiment in edit mode.

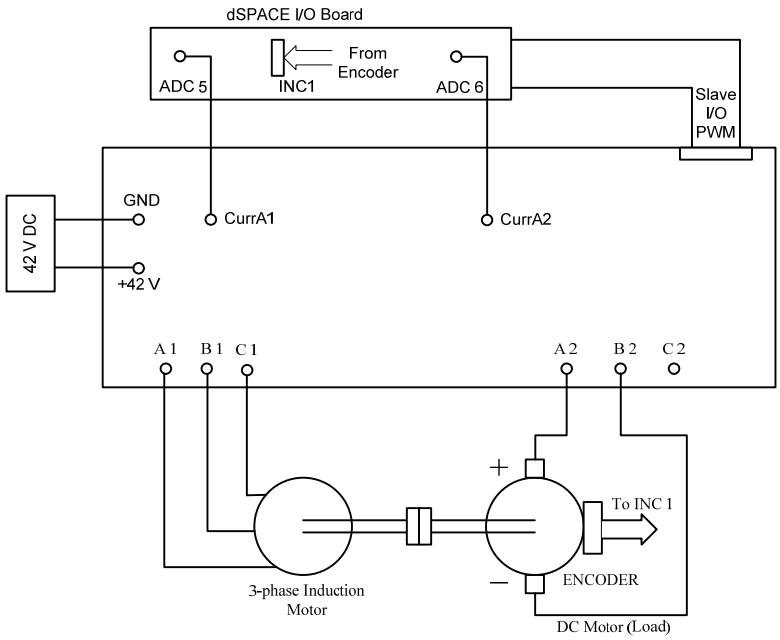


Figure 8.3: Connections for characterization of three phase induction motor

8.3.2 Lm

In this section, the DC-motor is run under speed control at the rated speed and the induction motor is supplied voltages at rated frequency so that the motor slip is zero.

- Go to animation mode,

Reset	Slip_v / f Control	Torque_SpeedControl
□	□	□

- Increase the speed of the dc-motor **Wref** to -1800RPM = (60 Hz electrical).

- Increase the value in ref_frequency numerical input box to 60 Hz. Observe induction motor phase-a voltage and current in controldesk (set length to 0.2 in capture setting window).

- Save this waveform as **vi_unloaded.mat**. Using these waveforms, the value of L_m can be calculated. The steps used in the m file are given below,

$$I_{rms} = \hat{I}/\sqrt{2}$$
$$V_{rms} = \hat{V}/\sqrt{2}$$
$$\theta = \Delta t \times 2\pi f$$
$$Z = \frac{V_{rms}}{I_{rms}} \angle\theta$$
$$X_m = \text{imag}(Z)$$
$$L_m = \frac{X_m}{2\pi f}$$

The simplification in the expressions above is possible under the assumption that $L_m >> L_{ls}$ and therefore $L_m + L_{ls} \approx L_m$, and that the branch containing L_{lr}, R'_r is open.

- Get a screen shot of ControlDesk. Now set the frequency reference of induction motor to zero. Set the speed of the dc-motor to zero so that the machines come to rest. Stop the experiment in the edit mode.

8.4 Rated slip

In this section, the induction motor is run at rated frequency. The DC motor is run under torque control. The load torque is increased until the induction motor draws rated load current.

- Go to Animation mode and check 'Torque_SpeedControl' so that the DC motor is torque-controlled.

Reset	Slip_v / fControl	Torque_SpeedControl
☐	☐	☑

- Increase the value in the ref frequency numerical input box to 60 Hz in small steps. Now, increase the DC motor torque reference while observing the i_{a_imotor} waveform. Increase the torque reference (so that absolute the speed reduces) till the rms current reaches 4.54 A. This is the rated current of the machine and it is now operating at rated conditions. Take a screen shot of dSpace ControlDesk.

- In im_parameters.m enter the speed at which the machines are running.

$$\%\text{slip} = \frac{\omega_{syn} - \omega_{act}}{\omega_{syn}} \times 100$$

- **Reduce the torque back to zero. Now set ref frequency to zero so that the machines come to rest. Stop the experiment in the edit mode.**

8.5 Determine L_{lr}, L_{ls}, R'_r

If we run Induction machine at slip=1 and assume that $L_m >> L_{lr}$, then

$$j\omega L_m \left\| \left(j\omega L_{lr} + R'_r\right) \approx \left(j\omega L_{lr} + R'_r\right). \right.$$

The equivalent impedance seen at machine terminals becomes

$$j\omega\left(L_{lr} + L_{ls}\right) + R'_r + R_s$$

Further, assuming that L_{ls} = 2/3 L_{lr}, and knowing the value of Rs, we can calculate L_{lr}, L_{ls}, R'_r.

- In the capture setting window the length is set to 0.8s.

- Go to Animation mode and check slip_v/f Control so that the frequency and magnitude of the IM voltages can be controlled independently. The DC machine will run under speed control mode with reference speed equal to zero, which is in same as blocked rotor.

Reset	Slip_v / fControl	Torque_SpeedControl
☐	☑	☐

- Increase the value in the ref frequency numerical input box to f_{slip} Hz in small steps. Now, increase V_{slip} while observing the $_{ia\_imotor}$ waveform till the rms current reaches the rated current of 4.54.

- Save the waveform from ControlDesk as **vi_blocked.mat**. Enter the peak value of the induction motor phase-a voltage and current, and the time difference between the zero crossings of the voltage and current waveforms in the Matlab file. The computations carried out are:

$$I_{rms} = \hat{I}/\sqrt{2}$$
$$V_{rms} = \hat{V}/\sqrt{2}$$
$$\theta = \Delta t \times 2\pi f_{slip}$$
$$Z = \frac{V_{rms}}{I_{rms}} \angle\theta$$
$$X_m = \text{imag}(Z)$$
$$L_l = \frac{X_m}{2\pi f_{slip}}$$
$$L_{lr} = 3/5 L_l$$
$$L_{ls} = 2/5 L_l$$
$$R_r^{'} = \text{real}(Z) - R_s$$

- R_s was measured in section 8.2.

- Reduce V_{slip} back to zero. Stop the experiment in the edit mode.

8.6 Lab Report *(10 points)*

1. Induction Motor parameters *(6 points)*. Please mention the motor number.

Parameter	Value (include the units!)
R_s L_m L_{ls} L_{lr} R'_r f_{slip}	

2. How many poles does the induction motor have? *(0.5 point)*

3. Attach the screen shots of dSpace ControlDesk for section 2.2, 3 and 4. *(1 point)*

4. In the blocked rotor test, why do we apply a reduced voltage to the stator windings? *(0.5 point)*

5. In the no-load test, how much power is supplied to the motor? How much reactive power is supplied? Is the power associated with core loss (P_{core}), copper loss (P_{cu}) or both? What is the value of P_{core}? *(2 point)*

E x p e r i m e n t – 9
Torque-Speed Characteristics and Speed-Control of Three-Phase Induction Motor

9.1 Introduction

This experiment is divided into three sections,

1. Study the torque speed characteristics of a three phase induction motor.

2. Induction motor in generation mode (super synchronous speed) and motoring mode (sub synchronous speed).

3. Speed control of three phase induction motor by two methods:
   a) Using slip compensation without speed feedback [2].
   b) Slip compensation with speed feedback [1].

9.2 Torque -Speed Characteristics

To derive the torque-speed characteristics of the induction motor (IM), it will be operated at a fixed input frequency and a proportional input voltage (defined by the V/f ratio). The DC-motor will act as a variable load and the value of the load-torque will be varied by 'Tref'. In addition to the DC-motor torque, the induction motor will also experience a coulomb friction torque and viscous friction torque. **Note**: The speed is measured from the DC motor; hence the speed of the induction motor is the negative of the measured speed.

$$T_{IM} = T_{DC} + T_{friction} + B_{\omega} \quad (1)$$

$$T_{friction} = 0.1\text{Nm}, B = 0.0002\,\text{Nmsec/rad}...\text{from Lab 4}$$

- Download the files 'Exp 9' and extract its contents to a folder on the desktop. Start Matlab and set the work directory to this folder.

- Set Vd=42, Ts=1e-4 at the Matlab Command prompt. Open the Simulink file **Torque_ Speed.mdl** and build it (Ctrl+B) as shown in Fig 9.1. Here the induction motor is V/f controlled. The induction motor is coupled to a DC motor and this DC motor is run under Torque control or Speed Control.

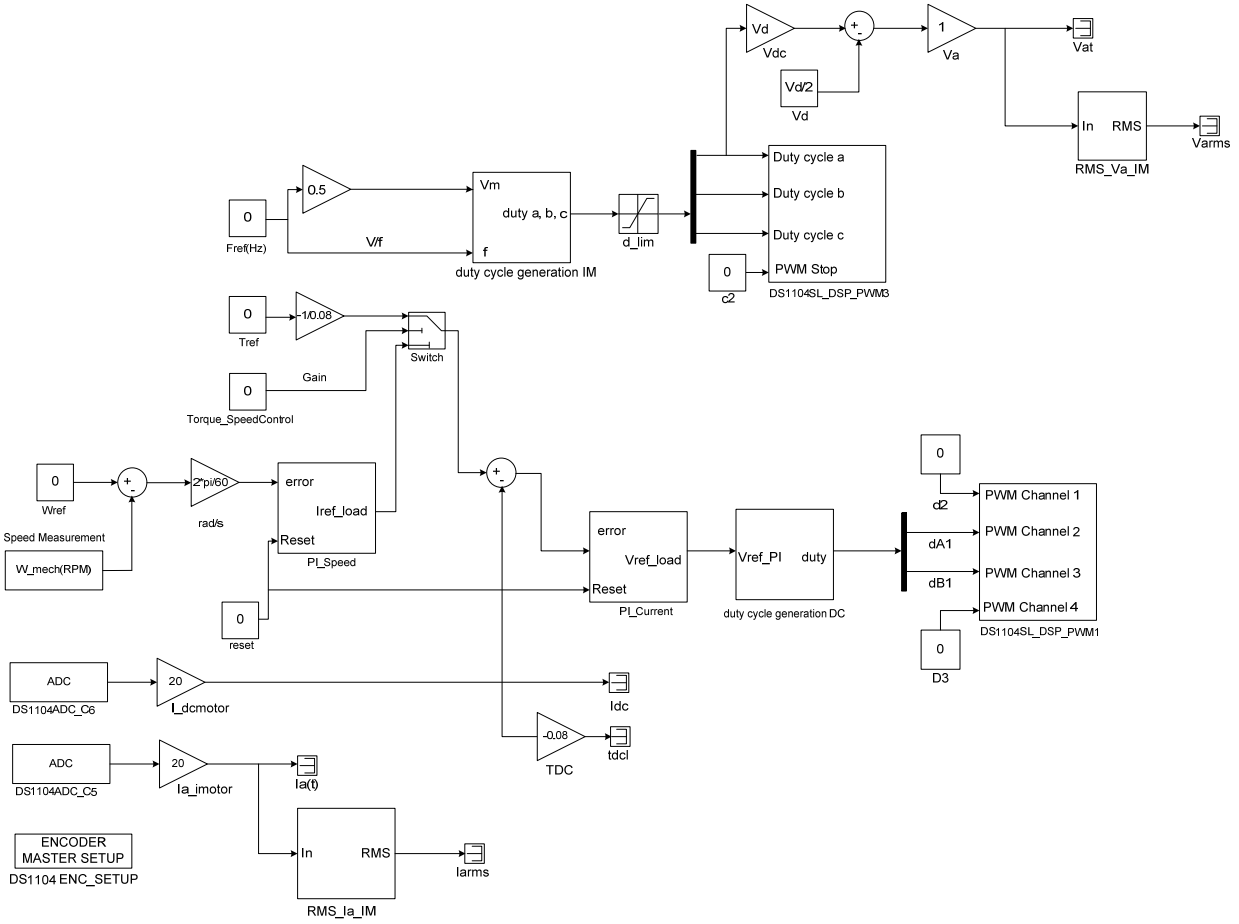


Figure 9.1: Torque Speed Simulink File

- Start Control Desk and open the variable file **Torque_Speed.sdf,** and then the layout **Torque_Speed.lay.**

- Go to Animation mode and check 'Torque_SpeedControl' so that the DC motor is torque-controlled.

Reset	Torque_SpeedControl
☐	☑

- Make connections as shown in Fig. 9.2. Make sure the phase sequence of the induction motor is correct.

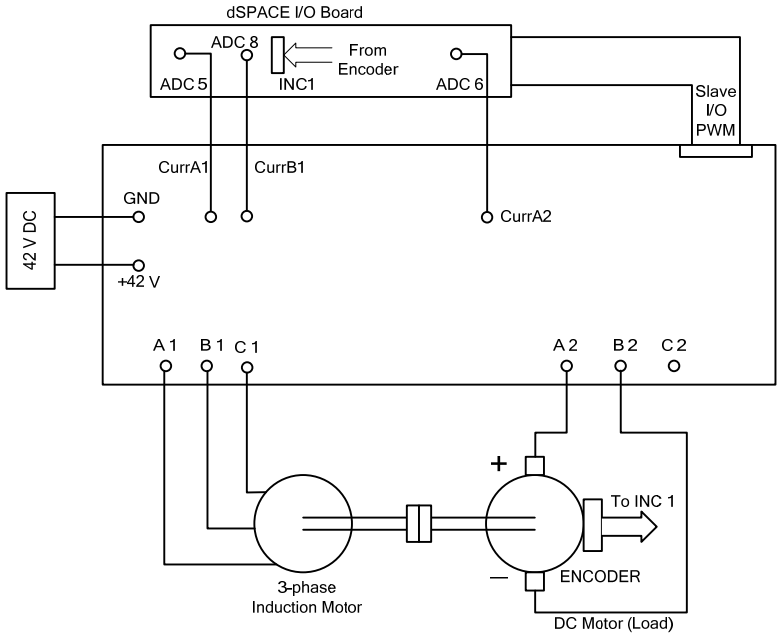


Figure 9.2: Connections for torque-speed characteristics, speed control of three phase induction motor

- Increase the value in the Fref(Hz) numerical input box to 30 Hz in small steps. Change the torque of the dc-motor in the direction to increase the slip (decrease the speed of the motor). Ignore the sign of torque and measure the speed of the motor. Fill out the Table 1 as you increase the load torque. Do the same at 45 Hz and 60 Hz. **Note:** Always, bring 'Tref' back to zero and then change 'Fref(Hz)' slowly. Make sure the RMS value of IM current does not exceed 4.5A. **Think why do we increase the applied frequency slowly? If the motor is at 0 speed and 60Hz voltages are applied to the IM what will happen?**

- Open the file 'im_torquespeed.m' and enter the value of speeds in lines 10, 11, and 12 corresponding to frequencies 30, 45 and 60 Hz. Enter the load torque of the dc-motor in line 13. The electromagnetic torque applied by the machine is found in lines 14 through 16 of the script. Run this section of the script (till line 24). The values of Tem V/s per slip speed (RPM) are plotted. **Calculate the value of $K_{T\omega}$ using equation (2).**

$$T_{em} = K_{T\omega}\omega_{slip} \quad (2)$$

Lines 28-33 plot the plot T_{em} V / s per unit speed $\left(\frac{\omega_m}{\omega_{sync}}\right)$.

- Lines 34-40 plot the plot T_{em} V / s speed in RPM (ω_m).

Table 1

T_{DC} (Nm)	$\omega_{(f=30)}$	$\omega_{(f=45)}$	$\omega_{(f=60)}$
0.000			
0.020			
0.040			
0.060			
0.080			
0.100			
0.120			
0.150			

9.3 Generating and Motoring Mode of Induction Motor

In this section, the generating and motoring modes of the Induction Motor can be studied by coupling the Induction motor to a DC motor. The DC-motor is run under speed control.

- Go to Animation mode and set it up so that the DC motor is speed-controlled.

Torque speed characteristics

Reset	Torque_SpeedControl
☐	☑

- Increase the DC motor speed 'Wref' to -900 RPM. Set the induction motor frequency 'Fref(Hz)' to be 30 Hz. This case is for zero slip. Set the length of capture to be 0.2s. Observe the induction motor phase-a current and voltage. Save the waveform as **atsync.mat**.

- Keep the induction motor frequency set at 30 Hz. Vary the speed of 'Wref' to be at + 10% higher than 900 rpm and save the voltage and current waveforms to **supersync.mat**. Now, change 'Wref' to be at 10% lower than 900 rpm. Save the voltage and current waveforms as **subsync.mat**.

- Set “Fref(Hz)’ back to zero and lower the value of ‘Wref” back to zero. Stop the experiment in edit mode. Close dSpace ControlDesk.

- The power supplied by the induction motor can be calculated using lines 43-87 file ‘im_torquespeed.m’.

9.4 Speed Control of Three-Phase Induction Motor

9.4.1 *Without Speed Feedback*

The control of a three phase induction motor without direct speed measurement is described in [2]. In this experiment, the current limiter circuit is ignored and the applied voltage is V/f controlled. The block diagram for this control is given in Fig. 9.3. The dc current (I_d) is calculated on an average sense using (3). The power supplied to the induction motor is P_{IM} (4). Neglecting losses, the supplied power is related to T_{em} by (5). In the linear region of operation, T_{em} is proportional to ω_{slip} (6). The required ω_{slip} to support this torque can be calculated using (7). By (8), the frequency of voltages (ω_{sync}) that are applied to the motor can be calculated. The voltage magnitude is calculated such that V/f is a constant (Voltage boost is ignored). **Note:** This control strategy works only when the motor is already running, hence the motor is started and stopped using simple V/f control.

$$I_d = i_a d_a + i_b d_b + i_c d_c \tag{3}$$

$$P_{IM} = V_d \times I_d \tag{4}$$

$$P_{IM} = T_{em} \times \omega_{sync} \tag{5}$$

$$T_{em} = K_{T\omega} \times \omega_{slip} \tag{6}$$

$$\omega_{slip} = \frac{P_{IM}}{K_{T\omega}\omega_{sync}} \tag{7}$$

$$\omega_{sync} = \omega_{ref} + \omega_{slip} \tag{8}$$

- Open file **im_speed_control.mdl** as shown in Fig 9.4. Build this file (Ctrl+B). Open dSpace ControlDesk and open variable file **im_speed_control.sdf** and then open layout file **im_speed_control.lay**.

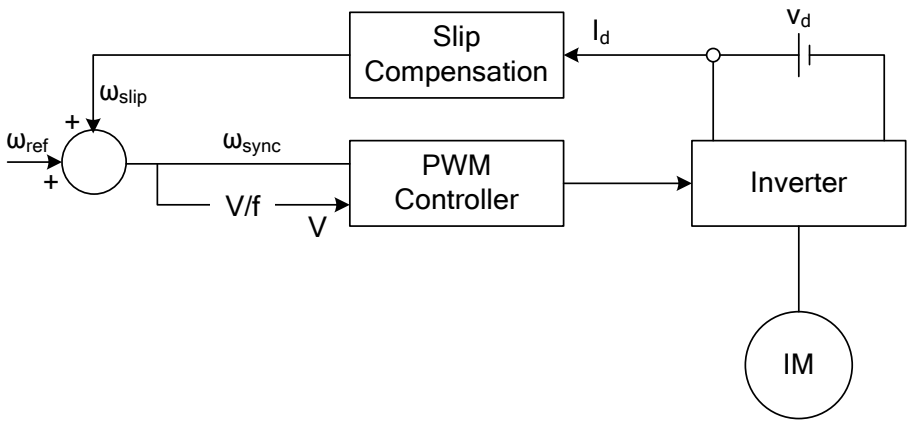


Figure 9.3: IM speed control by slip compensation – without speed feedback

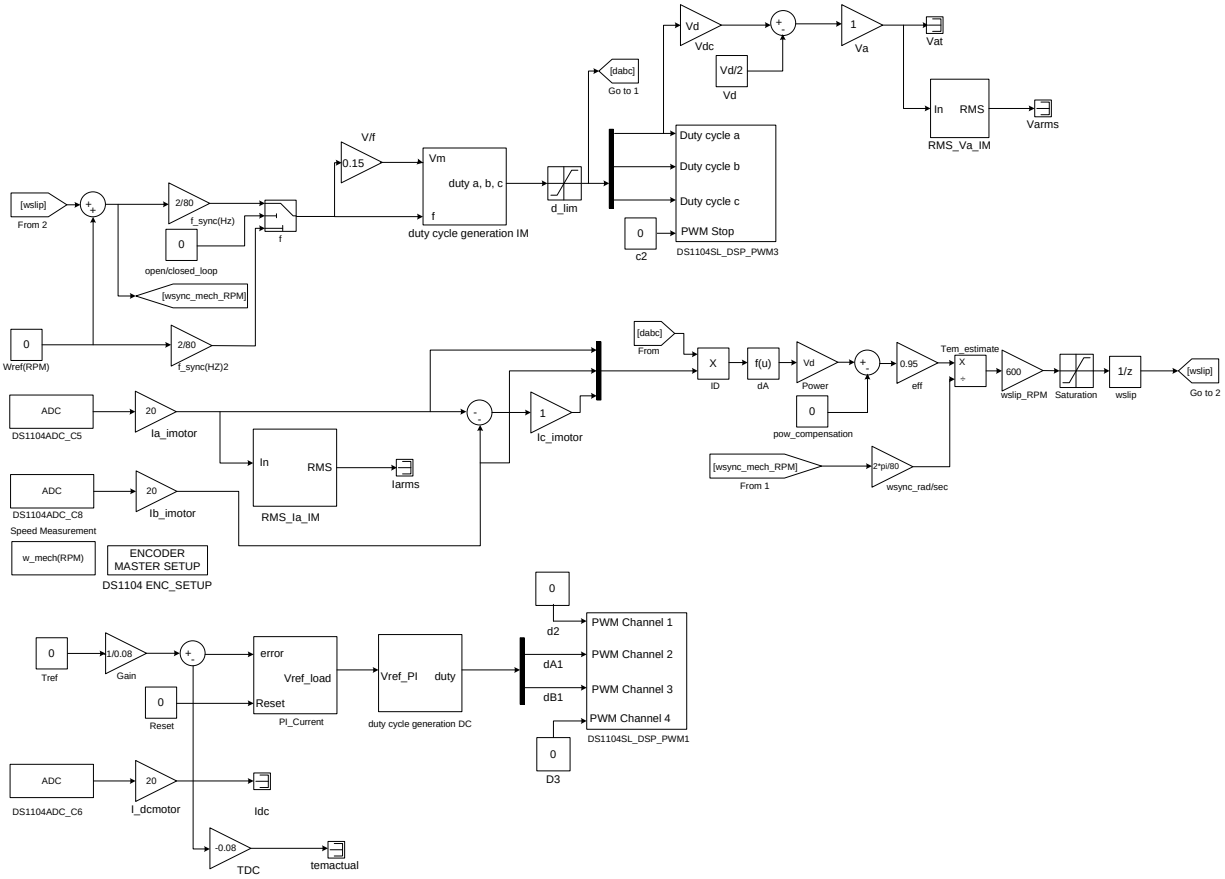


Figure 9.4: IM Speed Control Simulink Block

- Increase ‘Wref (RPM)’while keeping the motor in open-loop speed control (‘open/closed loop’ is unchecked) upto 300 rpm.

- Check ‘open/closed loop’. Now the motor is speed controlled. Increase the value of ‘Wref (RPM)’ till 900 and see if the speed is being controlled. Save a screen shot of the speed control and attach it in your report.

- Change the load torque ‘Tref’ (Do not go above ± 0.1Nm). Check if the speed is being controlled.

- Reduce the speed back to 300 rpm, uncheck ‘open/closed loop’ and bring ‘Wref’ down to zero. Stop the experiment in edit mode and close dSpace ControlDesk.

9.4.2 *With Speed Feedback*

In this control strategy, the actual speed of the motor (ω_m) is measured and compared with a reference speed. The error is speed is fed to a PI controller. This controller generates a value for slip (ω_{slip}), which is added to the measured speed to generate the frequency reference to the PWM controller. The voltage reference is proportional (V/f) to this reference frequency.

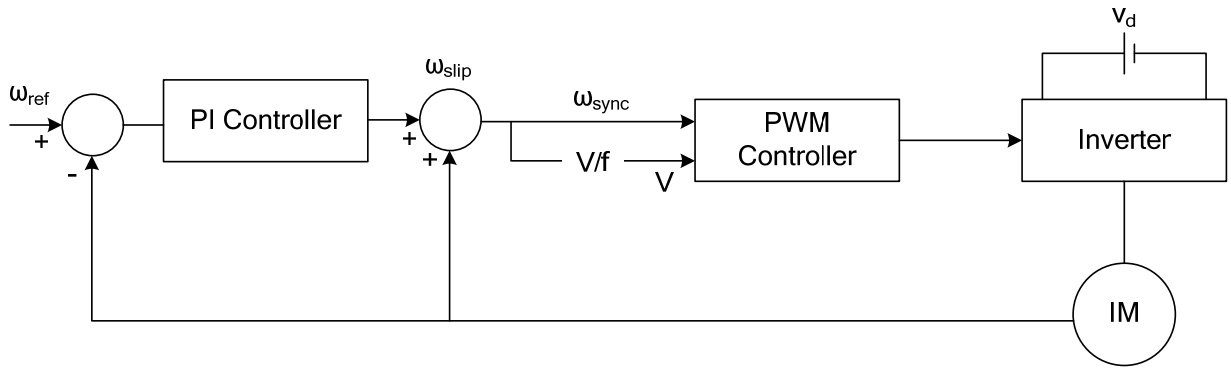


Figure 9.5: IM speed control (PI controller) with speed feedback

- Open file **im_speed _control_pi.mdl** as shown in Fig 9.6. Build this file (Ctrl+B).Open dSpaceControlDesk and open variable file **im_speed_control_pi.sdf** and then open layout file **im_speed** control **pi_lay**.

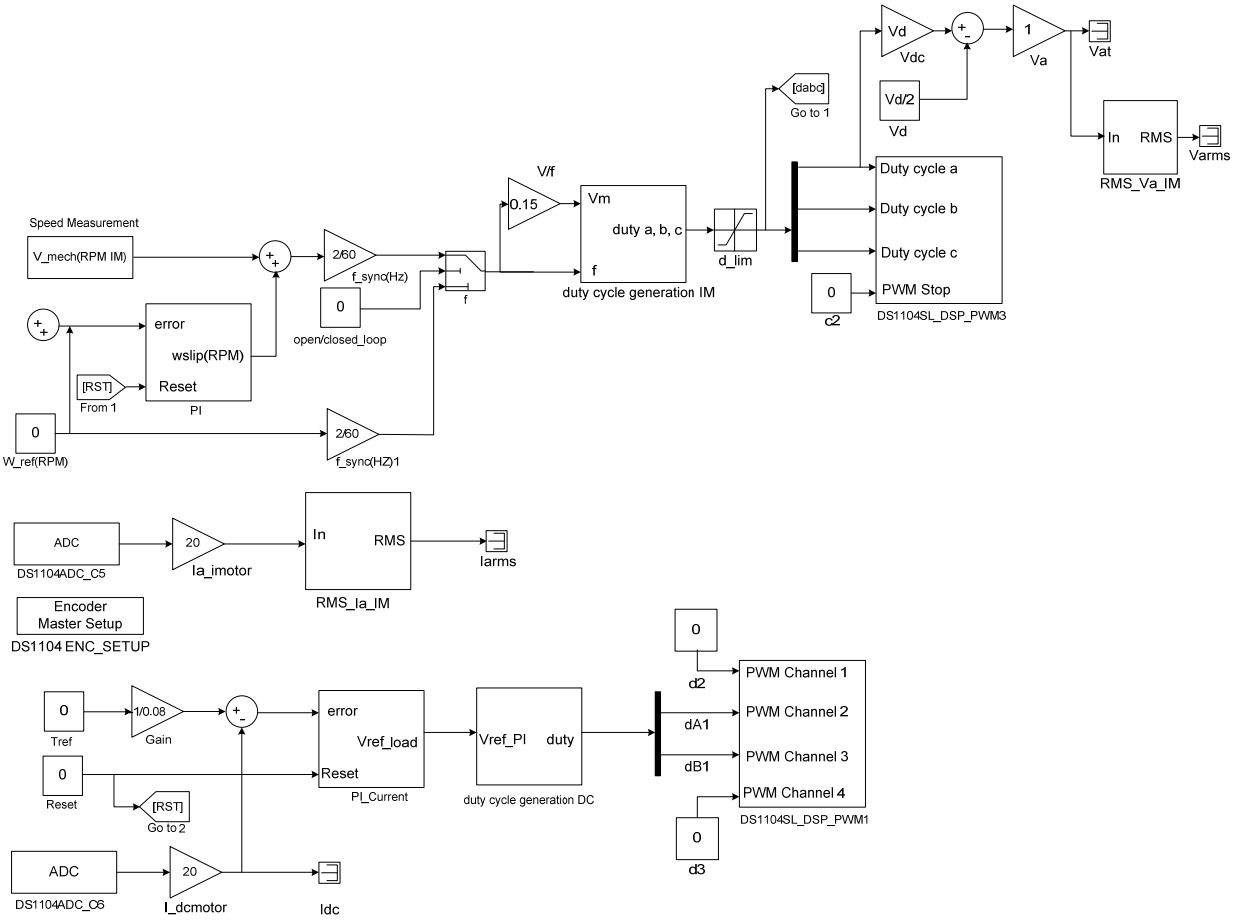


Figure 9.6: IM Speed Control PI

- Increase ‘Wref (RPM)’ while keeping the motor in open-loop speed control (‘open/closed loop’ is unchecked) upto 300 rpm.

- Check ‘open/closed loop’. Now the motor is speed controlled. Increase the value of ‘Wref (RPM)’ till 900 and see if the speed is being controlled. Save a screen shot of the speed control and attach it in your report.

- Change the load torque ‘Tref’ (Do not go above ±0.1Nm). Check if the speed is being controlled.

- Reduce the speed back to 300 rpm, uncheck ‘open/closed loop’ and bring ‘Wref’ down to zero. Stop the experiment in edit mode and close dSpace ControlDesk.

9.5 Lab Report *10 points*

1. Section 1
   (a) Attach the torque-speed characteristics generated in Matlab (3 plots) along with Table 1. (*3 points)*
   (b) Comment on the graph - how the torque-speed characteristics change with frequency? Slip? *(1 points)*
   (c) What is the value of $K_{T\omega}$? *(1 point)*

2. Section 2
   (a) Attach the voltage-current-power waveforms corresponding to sub-synchronous, super-synchronous and synchronous speeds. *(1 point)*
   (b) What is the power supplied to the induction motor in all three modes? *(1 point)*
   (c) Comment on why there power at +10% slip is not equal in magnitude to that at -10% slip. *(1 point)*

3. Section 3
   (a) Attach the screen shots of the speed controlled IM from section 3.1 and 3.2. *(1 point)*
   (b) Give one advantage of each speed controller in section 3.1 and 3.2. *(1 point)*

9.6 References

[1] B.K. Bose. Modern Power Electronics and AC Drives. Pearson Education, Prentice Hall.

[2] Ned Mohan. Electric Drives, An Integrative Approach. MNPERE, 2000.

Appendix – A
Safety Precautions and Power-Electronics-Drives-Board, CP1104 I/O Board, DS 1104 Control Board and Motor Coupling Unit, familiarization

1.1 Why is safety important?

Attention and adherence to safety considerations is even more important in a power electronics laboratory than it's required in any other undergraduate electrical engineering laboratories. Power electronic circuits can involve voltages of several hundred volts and currents of several tens of amperes. By comparison the voltages in all other teaching laboratories rarely exceed 20V and the currents hardly ever exceed a few hundred milliamps.

In order to minimize the potential hazards, we will use dc power supplies that never exceed voltages above 40-50V and will have maximum current ratings of 20A or less. We use a dc supply of 42V. However in spite of this precaution, power electronics circuits on which the student will work may involve substantially larger voltages (up to hundreds of volts) due to the presence of large inductances in the circuits and the rapid switching on and off of amperes of current in the inductances. For example in Power Electronics laboratory a boost converter can have an output voltage that can theoretically go to infinite values if it is operating without load. Moreover the currents in portions of some converter circuits may be many times larger than the currents supplied by the dc supplies powering the converter circuits. A simple buck converter is an example of a power electronics circuit in which the output current may be much larger than the dc supply current.

1.2 Potential problems presented by Power Electronic circuits

- Electrical shock may take a life.

- Exploding components (especially electrolytic capacitors) and arcing circuits can cause blindness and severe burns.

- Burning components and arcing can lead to fire.

1.3 Safety precautions to minimize these hazards

1.3.1 *General Precautions*

- Be calm and relaxed, while working in Lab.

- When working with voltages over 40V or with currents over 10A, there must be at least two people in the lab at all times.

- Keep the work area neat and clean.

- No paper lying on table or nearby circuits.

- Always wear safety glasses when working with the circuit at high power or high voltage.

- Use rubber floor mats (if available) to insulate yourself from ground, when working in the Lab.

- Be sure about the locations of fire extinguishers and first aid kits in lab.

- A switch should be included in each supply circuit so that when opened, these switches will de-energize the entire setup. Place these switches so that you can reach them quickly in case of emergency, and without reaching across hot or high voltage components.

1.3.2 *Precautions to be taken when preparing a circuit*

- Use only isolated power sources (either isolated power supplies or AC power through isolation power transformers). This helps using a grounded oscilloscope and reduces the possibility of risk of completing a circuit through your body or destroying the test equipment.

1.3.3 *Precautions to be taken before powering the circuit*

- Check for all the connections of the circuit and scope connections before powering the circuit, to avoid shorting or any ground looping that may lead to electrical shocks or damage of equipment.

- Check any connections for shorting two different voltage levels.

- Check if you have connected load at the output.

- Double-check your wiring and circuit connections. It is a good idea to use a point-to-point wiring diagram to review when making these checks.

1.3.4 *Precautions while switching ON the circuit*

- Apply low voltages or low power to check proper functionality of circuits.

- Once functionality is proven, increase voltages or power, stopping at frequent levels to check for proper functioning of circuit or for any components is hot or for any electrical noise that can affect the circuit's operation.

1.3.5 *Precautions while switching off or shutting down the circuit*

- Reduce the voltage or power slowly till it comes to zero.

- Switch of all the power supplies and remove the power supply connections.

- Let the load be connected at the output for some time, so that it helps to discharge capacitor or inductor if any, completely.

1.3.6 *Precautions while modifying the circuit*

- Switch Off the circuit as per the steps in section 3.5.

- Modify the connections as per your requirement.

- Again check the circuit as per steps in section 1.3.3, and switch ON as per steps in section 1.3.4.

1.3.7 *Other Precautions*

- No loose wires or metal pieces should be lying on table or near the circuit, to cause shorts and sparking.

- Avoid using long wires that may get in your way while making adjustments or changing leads.

- Keep high voltage parts and connections out of the way from accidental touching and from any contacts to test equipment or any parts, connected to other voltage levels

- When working with inductive circuits, reduce voltages or currents to near zero before switching open the circuits.

- BEWARE of bracelets, rings, metal watch bands, and loose necklace (if you are wearing any of them), they conduct electricity and can cause burns. Do not wear them near an energized circuit.

- Learn CPR and keep up to date. You can save a life.

- When working with energized circuits (while operating switches, adjusting controls, adjusting test equipment), use only one hand while keeping the rest of your body away from conducting surfaces.

1.4 Power-Electronics-Drives-Board familiarization

The electric machine drives board, which we use in the Electric Drives Laboratory has been designed to enable us to perform a variety of experiments on AC/DC machines. The main features of the board are:

- Two completely independent 3-phase PWM inverters for complete simultaneous control of two machines

- 42 V dc-bus voltage to reduce electrical hazards

- Digital PWM input channels for real-time digital control

- Complete digital/analog interface with dSPACE board

The basic block diagram of drives board is shown in Fig. 1 and the actual drives board is shown in Fig. 2. Please note that various components on this board are indicated in Table. 1.

1.4.1 *Description of Power Electronic Drive Board.*

The board of Fig 2 is shown with some additional labels in Fig 3. The two independent three phase PWM voltage sources labeled phases A1, B1, C1 and phases A2, B2, C2 in Fig 1 are obtained from a constant DC voltage source (as shown in Circuit in Fig 1). Hence two machines can be controlled independently for independent control variables, at the same time. A single phase motor needs only Phase A1 B1 for powering while three phase motor needs three phases A1, B1, C1.

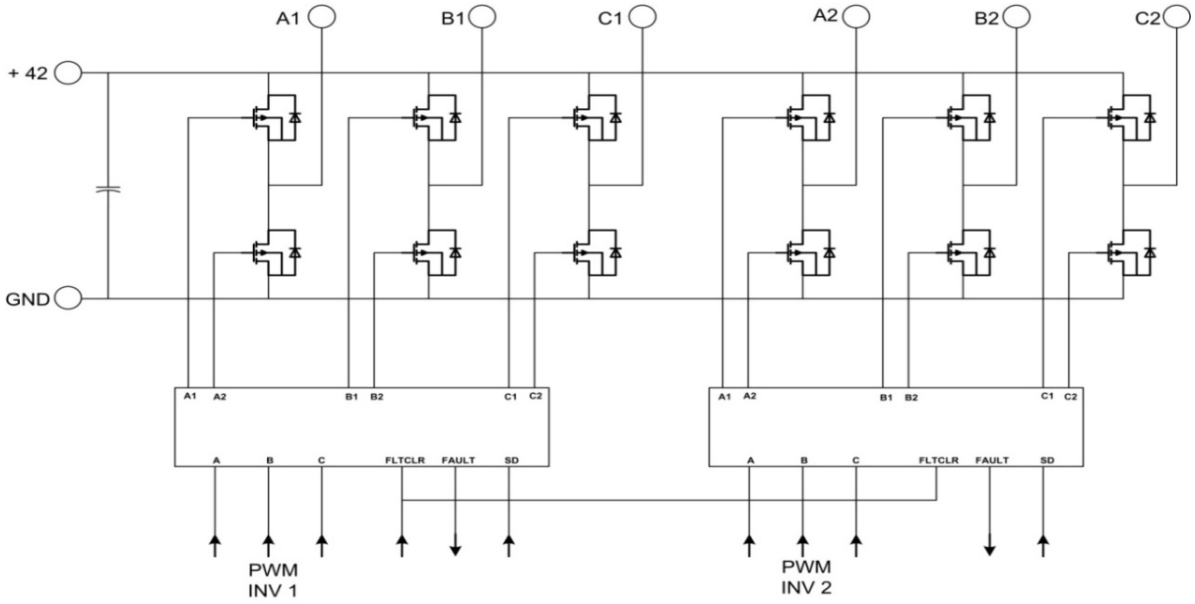


Figure 1: Block Diagram of Electric Drives Board

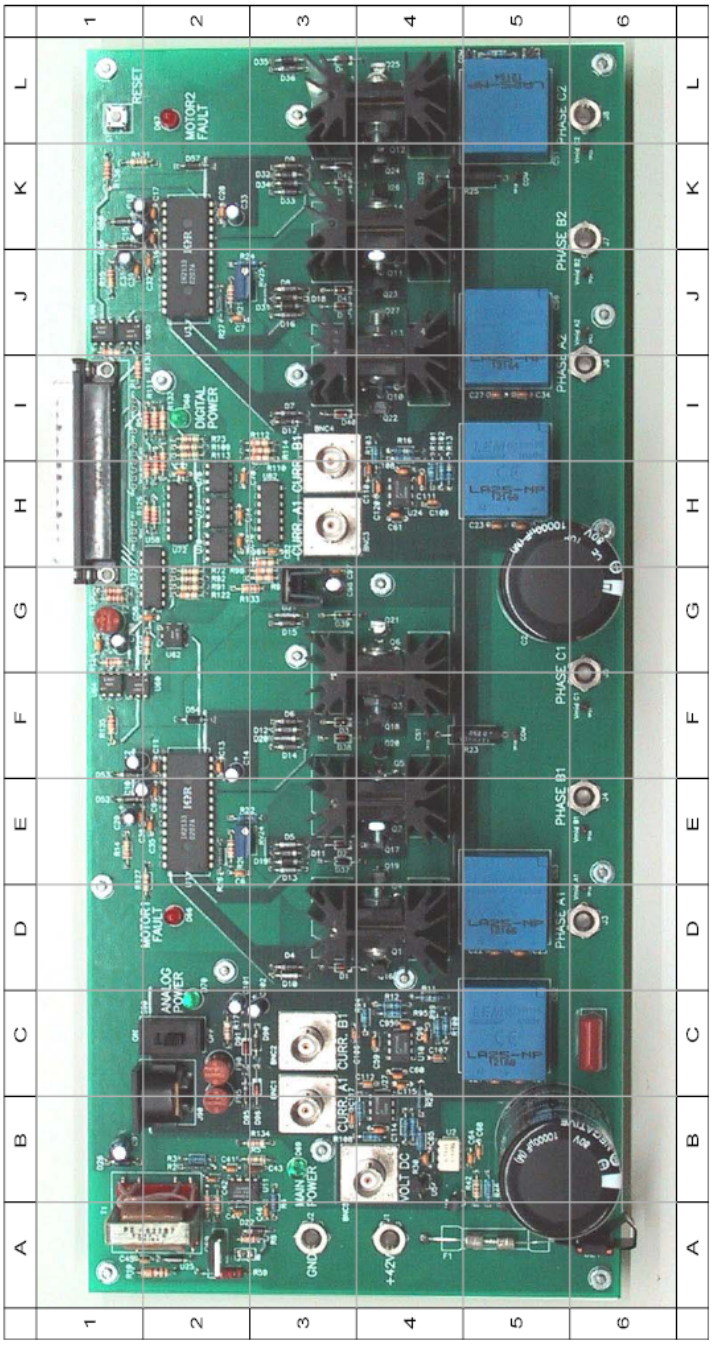


Figure 2: Power Electronics Drives Board

Table 1: Locations of components on Power Electronics Drives board

No.	Component	Ref. Des.	Location in Fig. 2
1	Terminal +42	J1	A-4
2	Terminal GND	J2	A-3
3	Terminal PHASE A1	J3	D-6
4	Terminal PHASE B1	J4	E-6
5	Terminal PHASE C1	J5	G-6
6	Terminal PHASE A2	J6	J-6
7	Terminal PHASE B2	J7	K-6
8	Terminal PHASE C2	J8	L-6
9	DIN connector for ±12 V signal supply	J90	B-2
10	Signal supply switch	S90	C-2
11	Signal supply +12 V fuse	F90	C-2
12	Signal supply-12 V fuse	F95	B-2
13	Signal supply LED	D70	C-2
14	MOTOR1 FAULT LED	D66	D-2
15	MOTOR2 FAULT LED	D67	L-2
16	DIGITAL POWER LED	D68	I-2
17	MAIN POWER LED	D69	B-3
18	Inverter 1		D-3 to G-4
19	Inverter 1		I-3toL-4
20	DC Link capacitor of Inverter 1	C1	B-5
21	DC Link capacitor of Inverter 2	C2	G-5
22	Driver IC IR2133 for Inverter 1	U1	E-2
23	Driver IC IR2133 for Inverter 2	U3	J-2
24	Digital Supply Fuse	F2	G-1
25	dSPACE Input Connector	P1	H-1 and I-1
26	RESET switch	S1	L-1
27	Phase A1 current sensor (LEM)	CS2	C-5
28	Phase B1 current sensor (LEM)	CS3	D-5
29	Phase A2 current sensor (LEM)	CS5	H-5
30	Phase B2 current sensor (LEM)	CS6	J-5
31	DC link current sensor (LEM)	CS1	L-5
32	VOLT DC	BNC5	B-4
33	CURR A1	BNC1	B-3
34	CURR B1	BNC2	C-3
35	CURR A2	BNC3	H-3
36	CURR B2	BNC4	I-3

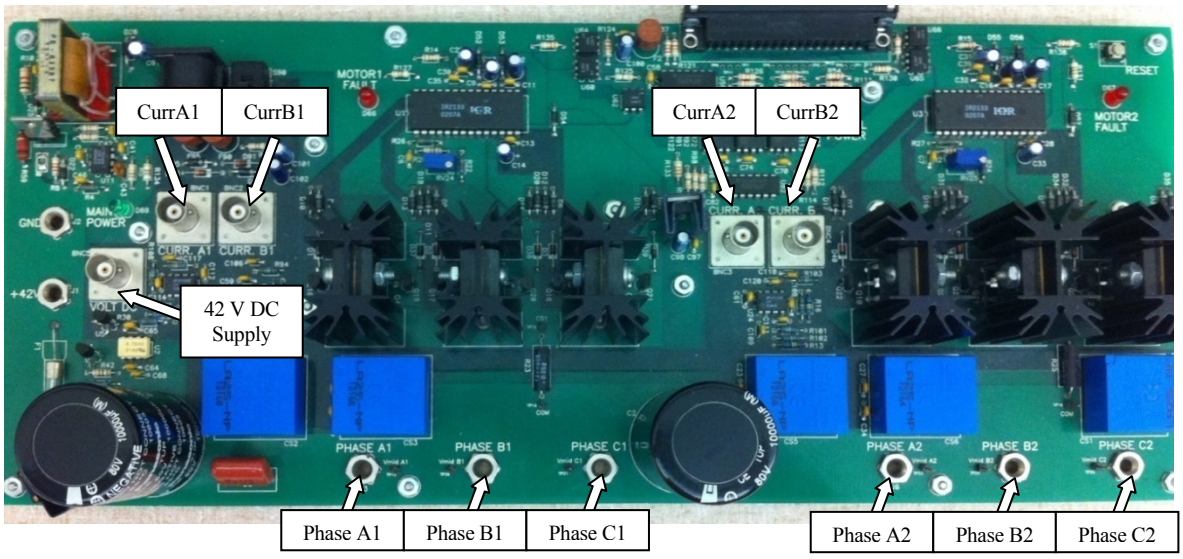


Figure 3: Power Electronics Drive Board (with key lables)

The location of some key supply and measurement points are shown with labeling in Fig 3. The board provides the motor phase currents (currA1, currB1, currA2, currB2), dc-bus voltage (shown to left of Curr A1) as in Fig 4 to control the motor for a desired speed or torque.

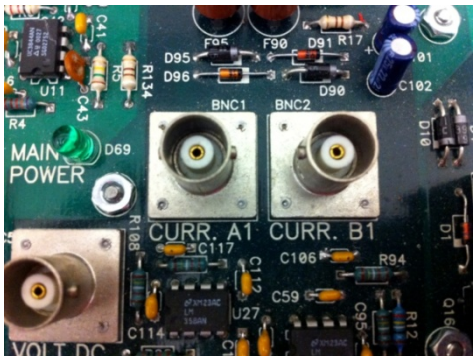


Figure 4(a): DC Bus CURR A1 CURR B1

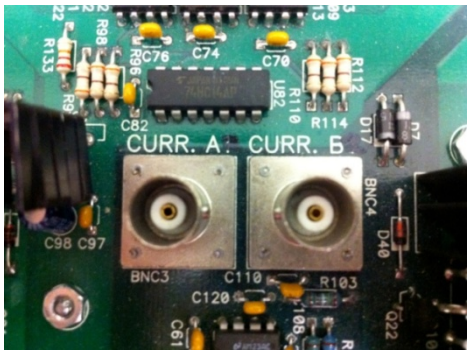


Figure 4(b): CURR A2 CURR B2

To generate the controlled PWM voltage source, this board requires various digital control signals. These control signals dictates the magnitude and phase of the PWM voltage source. The DS1104 R&D Controller board inside the computer generates them.

1.4.2 *Inverters*

Each 3-phase inverter uses MOSFETs as switching devices. The 3-phase outputs of the first inverter are marked A1 (D-6 in Fig. 2), B1 (E-6 in Fig. 2), C1 (F-6 in Fig. 2) shown with labels at bottom of Fig 3, and those of the second inverter are marked A2 (I-6 in Fig. 2), B2 (K-6 in Fig. 2), C2 (L-6 in Fig. 2).

1.4.3 *Signal Supply*

±12 volts signal supply is required for the isolated analog signals output from the drives board. This is obtained from a wall-mounted isolated power supply, which plugs into the DIN connector J90 (B-2 in Fig. 2). Switch S90 (C-2 in Fig. 2) controls the signal power to the board. The green LED D70 (C-2 in Fig. 2) indicates if the signal supply is available to the board. Fuses F90 (C-2 in Fig. 2) and F95 (B-2 in Fig. 2) provide protection for the +12 V and −12 V supplies respectively. Please note that the green LED indicates the presence of only the +12 V supply. Please note that turning off S90 will not stop the PWM signals from being gated to the inverters. The power supply for the 3-phase bridge drivers for the inverters is derived from the DC Bus through a flyback converter (A-2 in Fig. 2).

1.4.4 *Voltage Measurement*

Test points are provided to observe the inverter output voltages. BNC connector VOLT DC Fig 5 and (located at B-4 in Fig. 2) has been provided to sense the DC bus voltage. To measure the DC bus voltage,

- Connect a BNC cable to VOLT DC BNC connector.
- The scaling factor of input voltage is 1/10.

1.4.5 *Current Measurement*

LEM sensors are used to measure the output current of the inverters. Only A and B phase currents (CURR A1 and CURR B1) are sensed. The C phase current can then be calculated using the current relationship $I_a + I_b + I_c = 0$, assuming that there is no neutral connection for the machines. The calibration of the current sensor is such that for 1A current flowing through the current sensor, output is 0.5 V. The current measurement points are shown in Fig. 4.

To measure the output current of phase A of inverter 1

- Connect BNC connector to CURR A1 (B-3 in Fig. 2).

To measure the output current of phase B of inverter 1

- Connect BNC connector to CURR B1 (C-3 in Fig. 2).

To measure the output current of phase A of inverter 2

- Connect BNC connector to CURR A2 (H-3 in Fig. 2).

To measure the output current of phase B of inverter 2

- Connect BNC connector to CURR B2 (I-3 in Fig. 2).

1.4.6 *Inverter Drive Circuit*

The inverters (Fig 1) are driven by 3-phase bridge drivers (IR2133). The PWM inputs are isolated before being fed to the drivers.

1.4.7 *PWM / Digital Signals*

PWM and other digital signals for the board are to be given to the 37-pin DSUB connector in Fig 6 (and located at H-1 in Fig. 2). For pin-out description of this connector, see Table 2.

1.4.8 *Fault Protection*

Shown in Fig 7, the Drives Board has over-current protection for each inverter. An over-current fault occurring on inverter 1 is indicated by red LED “MOTOR FAULT 1” (D-2 in Fig. 2), while that of inverter 2 is indicated by red LED “MOTOR FAULT 2” (L-2 in Fig. 2). Each time a fault occurs, reset the fault using the “RESET” switch (L-1 in Fig. 2) on the board. The switch resets all the faults.

Figure 5: Main DC power input and main power indicator

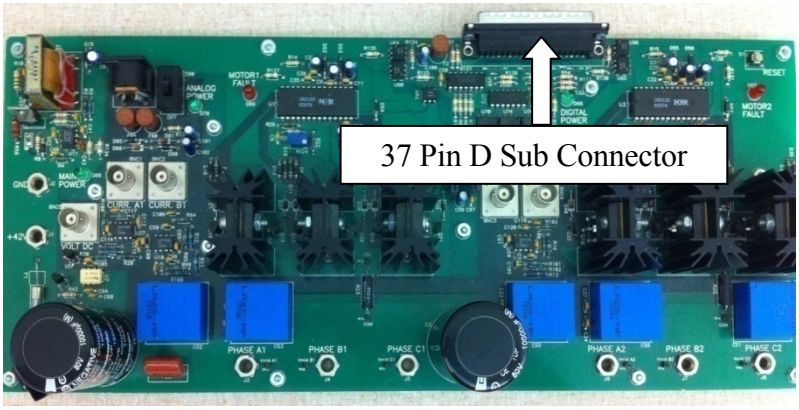


Figure 6: D SUB Connector

Figure 7(a): Motor-1 Fault Indicator

Figure 7(b): Motor -2 Fault and Reset Switch

Table 2: 37-pin DSUB Connector (located at H-1)

No.	Pin Number	Description
1	GND Digital	Digital ground
2	FAULT 1	Inverter 1 Fault output. Fault Signal high
3	NC	Not Connected
4	GND Digital	Digital ground
5	NC	Not Connected
6	GND Digital	Digital ground
7	PWM A1	A phase PWM signal of Inverter 1
8	PWM B1	B phase PWM signal of Inverter 1
9	PWM C1	CphasePWMsignalofInverter1
10	PWM A2	A phase PWM signal of Inverter 2
11	PWM C2	C phase PWM signal of Inverter 2
12	GND Digital	Digital ground
13	GND Digital	Digital ground
14	GND Digital	Digital ground
15	GND Digital	Digital ground
16	$\overline{SD1}$	Shutdown signal for Inverter 1
17	$\overline{FLTCLR\text{-}IN}$	Clear Fault signal
18	VCC	
19	VCC	
20	GND Digital	Digital ground
21	FAULT 2	Inverter 2 Fault output. Fault Signal high
22	NC	Not Connected
23	NC	Not Connected
24	NC	Not Connected
25	GND Digital	Digital ground
26	NC	Not Connected
27	NC	Not Connected
28	NC	Not Connected
29	PWM B2	CphasePWMsignalforInverter2
30	GND Digital	Digital ground
31	GND Digital	Digital ground
32	GND Digital	Digital ground
33	GND Digital	Digital ground
34	NC	Not Connected
35	$\overline{SD2}$	Shutdown signal for Inverter 2
36	GND Digital	Digital ground
37	GND Digital	Digital ground

1.5 DS1104 R&D controller Board and CP 1104 I/O board

In each discrete-time-step, the DS1104 R&D controller board (fitted as an add-on card in computer) takes some action to generate the digital control signals. The type of action is governed by what we have programmed in this board with the help of MATLAB-Simulink real-time interface. This board monitors the input (i.e. motor current, speed, voltage etc) with the help of CP1104 I/O board in each discrete-time step. Based on the inputs and the variables that need to be controlled (i.e. motor speed or torque); it takes the programmed action to generate the controlled digital signals.

The CP1104 I/O board (Fig 8) is an input-output interface board between the Power Electronics Drive Board and DS1104 controller board. It takes the analog signal motor current, dc-voltage etc. from the Power Electronics Drive Board in its ADC ports and also, speed signal (from encoder) at its INC ports from motor coupling system, to the DS1104 controller board. In turn, the controlled digital signals supplied by DS1104 controller board are taken to the Power Electronics Drive Board by CP1104. It also has 8 ADC ports to accept analog signals, 8 DAC ports to output analog signals and other digital I/O connectors as in Fig 9.

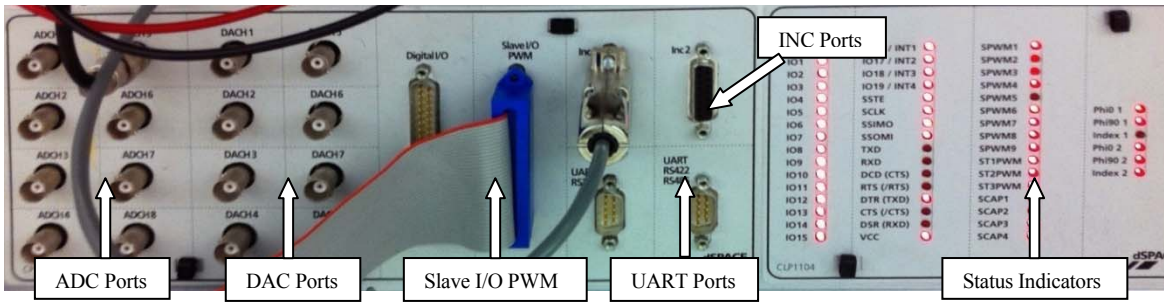


Figure 8: CP 1104 Board

Figure 9(a): CP1104 ADC and DAC Ports

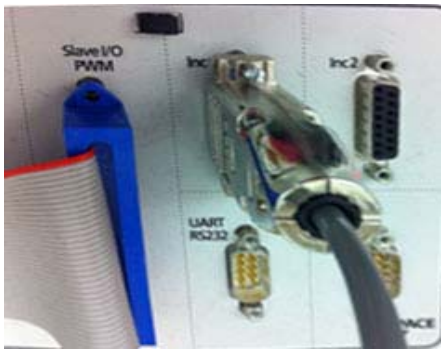


Figure 9(b): PWM port, INC for Encoder

1.6 MATLAB Simulink and Control-desk (Programming DS1104 and control in real-time)

Simulink is a software program with which one can do model-based design such as designing a control system for a DC motor speed-control. The I/O ports of CP 1104 are accessible from inside the Simulink library browser. Creating a program in Simulink and procedure to use the I/O port of CP 1104 will be detailed in future experiments. At this stage, let us assume that we have created a **control-system** inside the Simulink that can control the speed of a DC motor. When you build the Simulink **control-system** (CTRL+B) by using real-time option, it implements the whole system inside the DSP of DS1104 board, i.e. the **control-system** that was earlier in software (Simulink) gets converted into a real-time system on hardware (DS1104). Simulink generates a *.sdf file when you build (CTRL+B) the **control-system**. This file gives access to the variables of **control-system** (like reference speed, gain, tuning the controller etc) to separate software called Control-desk. In this software a control panel can be created that can change the variables of **control-system** in real time to communicate with DS1104 and hence change the reference quantities such as the speed or torque of the motor.

1.7 Motor coupling system

This system has a mechanical coupling arrangement to *couple two electric machines as* shown in Fig. 10. It also has an *encoder* mounted on the machine that is used to measure the speed and for close loop feedback speed-control of the motor. The motor that needs to be characterized or controlled is mounted in this system. The motor under test MUT whose speed/torque needs to be controlled, could be either a DC motor or three-phase induction motor or three-phase permanent-magnet AC (PMAC) motor. The motor demands a controlled pulse-width-modulated (PWM) voltage generated by Power Electronics Drive Board.

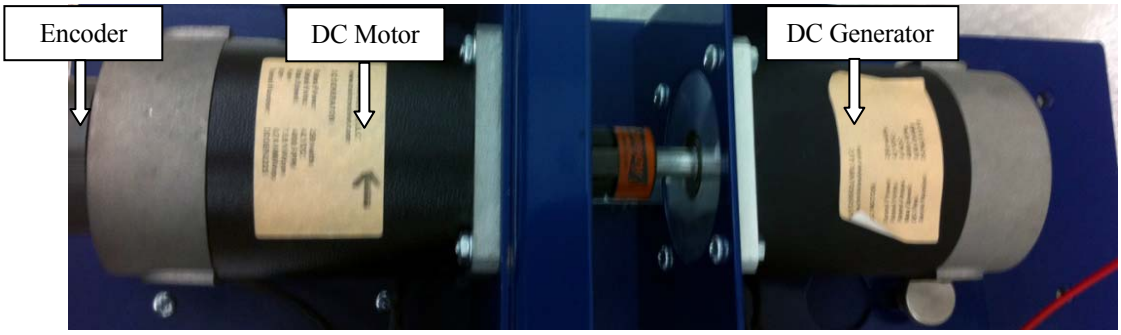


Figure 10: Motor Coupling System showing DC Motor, DC Generator and Encoder