

# dSpace Solutions for Control Tutorial

## ***Introduction***

The dSpace systems used for Motor/Control labs combine a data acquisition system with an independent processing system to implement digital control models. An overview of the pertinent system specification includes:

- Four multiplexed inputs to 16-bit analog to digital converter (ADC), Four inputs with independent 12-bit ADCs, and an 8-output digital to analog converter (DAC).
- 2 incremental Encoders
- Onboard independent 64-bit floating point processor
- Onboard Slave DSP
- Onboard memory
- Other digital I/O capabilities

The system includes three main components:

- PCI development Board
- I/O breakout box
- ControlDesk software and software protection dongle

The first thing to understand about the dSpace system is that it is an embedded or *self contained* system. The PCI board installed in the lab computers is its own entity. None of the processing for a system implemented on the board is done by the host PC. As a result the board requires that software be created and downloaded to the board for the system to function.

The ControlDesk software is used to design the system implementation and interface for the DS1104 PCI dSpace board. It is used to download software to the board, start and stop the function of the DS1104 as well as create a layout for interfacing with global variables in C/C++ programs used for implementation.

## ***Hardware Setup***

**Make sure the computer is OFF prior to installation or removal of Hardware.** An I/O breakout box must be connected to the header of the DS1104 PCI card located on the back of the computers (see Figure 2). Connections to the breakout box can now be made based on the requirements of the lab.

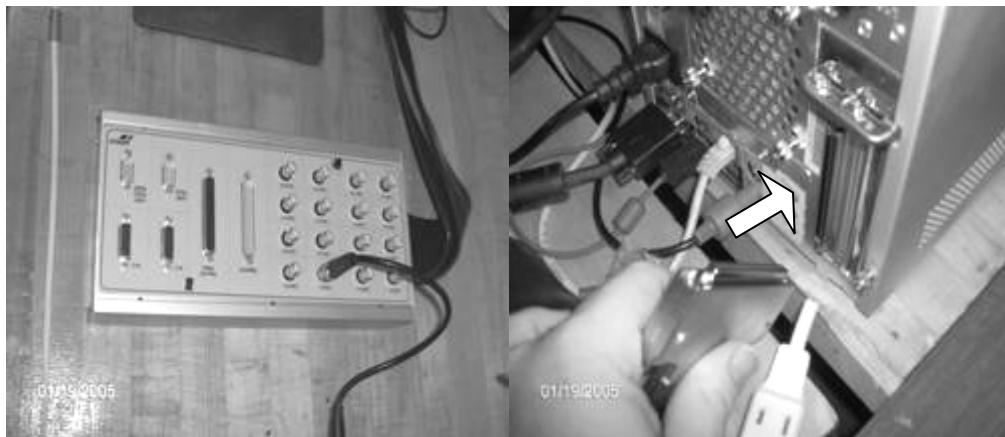


Figure 2 - I/O Breakout

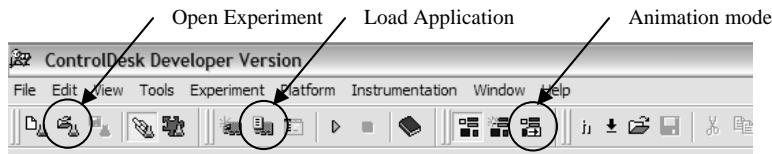
## Software Setup

The Control desk icon should be located on the desktop of the lab computers and can be used to start the application once the hardware is installed and the computer has booted up. The basic design structure in ControlDesk is called an Experiment. An Experiment includes all the C/C++ code files that comprise the software for the DS1104 board, Layout windows that provide a graphical user interface (GUI) to the system variables, and support files that connect the global variables of the software to items in a layout.

To perform an experiment the following steps must be taken:

- Start ControlDesk

Figure 3 – ControlDesk Menu / Toolbar Interface



- Open an existing experiment by selecting **Open Experiment...** from the **File** menu or use the **Open Experiment** button on the Experiment Manager toolbar. Use the dialog box to find the desired experiment file with extension .cdx and open it.
- Now the software for the embedded system must be downloaded to the DS1104 PCI board. Use the function **Load Application** from **Application** on the **Platform** menu or the **Load Application** button on the platform manager toolbar to download the software to the board. The dialog window should open up to the

folder of the experiment and display a file with an .sdf extension. Select the .sdf file to load the board. Click yes to any of the following warning\dialog boxes.

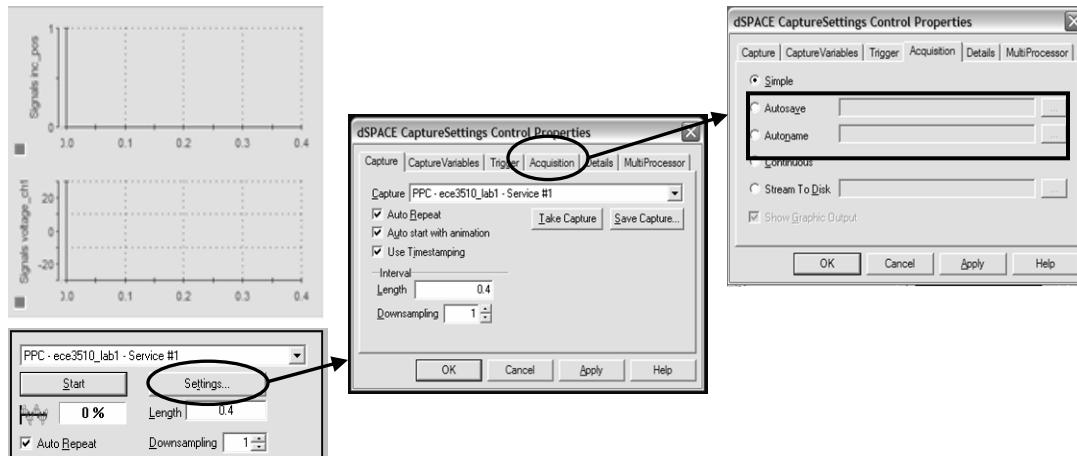
- Last, the animation mode must be started. This mode allows interaction between layout window items such as buttons, plot windows, and sliders with the system variables available on the board. Select **Animation Mode** from the **Instrumentation** menu or click the **Animation mode** button on the instrumentation toolbar.

## Working with data

### Using data capture in ControlDesk

There are many way to get data from the dSpace system but the most straight forward way is to capture the data that has been linked to a plot window in a layout. A plot window has an associated capture setting window that controls the saving of data. A plot window and the capture settings are presented in the first picture of Figure 4.

Figure 4 – Capture settings dialog progression



To save data outside dSpace the dialog progression of **Settings** button then **Acquisition** tab from Figure 4 should be followed. The ultimate goal is to store the data graphed in the plot window to a .mat file for analysis in Matlab®.

The two methods that will apply in the labs are Autosave and Autoname. Autosave creates a file that is overwritten each time the plot window is redrawn. With Autoname a file is created for each refresh of the plot window using a base name like “vars” and then added a suffix with each capture starting from 001 (ie... vars001, vars002, .... , vars999 is max ). **NOTE: You can not rename the file after saving or mat\_unpack won't work**

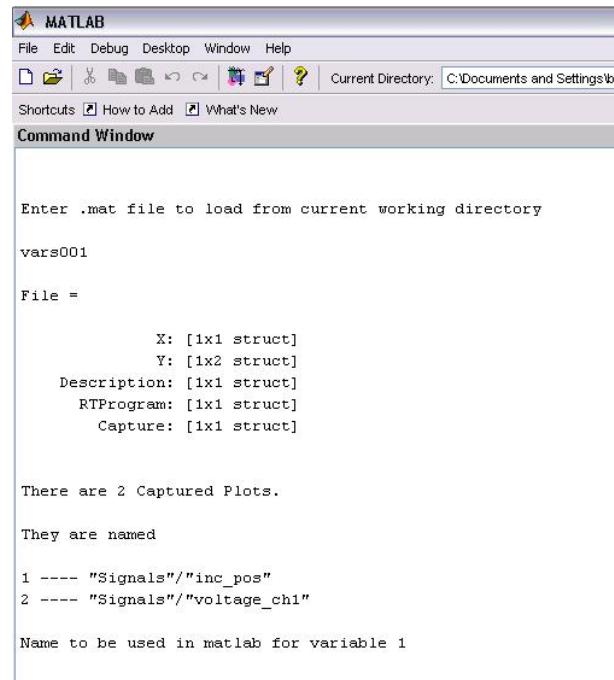
## Using mat\_unpack

The custom Matlab® script *mat\_unpack.m* is used to unpack the .mat data structure that dSpace generates when data is saved to a file. The structure uses the dot convention to access elements in the structure (ie.. name.field.data). This can be somewhat tedious to

work with when you want to perform analysis and manipulation in Matlab®, so *mat\_unpack* is designed to list the contents of the structure and unpack the data to variables you name for use in the workspace.

To use *mat\_unpack*:

- Make sure the file your working with is in the current working directory for Matlab®.
- Make sure *mat\_unpack.m* is in the same directory as the file or in the */work* directory on the machine you're using.
- Type *mat\_unpack* to initialize the script in the command window of Matlab®
- Type the name of the file but do not include the .mat extension. You will be presented with a prompt that gives information about the data in the structure. ***NOTE: if your changed the filename after saving the script will fail.***
- The script will ask for a workspace name for the first variable or *variable 1* from the list of names. You can use any valid Matlab® variable name just remember where you stored what. The script will have you name all of the data signals
- The program will ask you to give a name for the time data.
- Finally all the variables in the workspace are listed. You will notice that along with the variable you created is a struct with the name of your file. That struct is the unpack data.
- Now you can plot the data by using standard Matlab methods ( ie.. *plot(time,data,'properties')*)



The screenshot shows the MATLAB interface with the Command Window active. The window title is "MATLAB". The menu bar includes File, Edit, Debug, Desktop, Window, and Help. The toolbar has icons for New, Open, Save, Run, Stop, and Help. The status bar shows "Current Directory: C:\Documents and Settings\l...". The Command Window displays the following text:

```
Enter .mat file to load from current working directory
vars001
File =
    X: [1x1 struct]
    Y: [1x2 struct]
    Description: [1x1 struct]
    RTProgram: [1x1 struct]
    Capture: [1x1 struct]

There are 2 Captured Plots.

They are named
1 ---- "Signals"/"inc_pos"
2 ---- "Signals"/"voltage_chi"

Name to be used in matlab for variable 1
```