LabJack Corporation
3232 S Vance Street, Suite 100
Lakewood, CO 80227

Phone: (303) 942-0228
Fax: (303) 951-2916
info@labjack.com

MATLAB – LabJackUD Drivers and Samples
04/24/2007 Revision 1.32

Contained within the MATLAB_LJUD.zip are the necessary functions and samples to get you started using your LabJack in the MATLAB environment. This document applies to the LabJack UE9, U3 and other UD compatible LabJacks. The LabJackUD functions and samples were all tested using MATLAB version 7.0.0.19920 (R14). We expect that these functions and samples will work with versions 6.5 and higher as this is when MATLAB made direct dll calls possible. Use of these functions with a version of MATLAB that is older than 6.5 may not be possible. If difficulties arise when using these functions, please contact LabJack (support@labjack.com).

This download contains MATLAB functions that make direct calls to the UD Driver and MATLAB samples to demonstrate these functions. In order for it to work properly the full UD download must be installed first. The full UD download can be found by going to either of the following URLs:

http://www.labjack.com/labjack_ue9_downloads.html
http://www.labjack.com/labjack_u3_downloads.html

For MATLAB simplicity the UD functions have been written into M-Files that can be called from the main MATLAB command window or any other custom M-File.

Getting Started:
To use the LabJack functions within MATLAB, start by extracting the files in MATLAB_LJUD.zip to a folder located somewhere on your hard drive. Open up MATLAB, choose *File* from the menu bar and select *Set Path* from the drop down menu. A window will open that allows the user to set the path of the files containing the LabJackUD function calls and samples. In the *Set Path* window select the *Add with Subfolders...* button and locate the folder MATLAB_LJUD. Once the file has been selected press *OK*. This should add the directory path for both LJUD_Functions and LJUD_Samples to the top of the list. If they are both present, select the *Save* button. The MATLAB search path has now been modified to include the file containing all the LabJackUD functions and samples. Close the *Set Path* window by pressing the *Close* button.

Using the LabJackUD Functions:
Calling the LabJackUD functions within MATLAB is done just as any standard MATLAB function. Each function has a set of input arguments and return values. The general calling syntax for the MATLAB function is

[Output Values] = Function_Name(Input Arguments)

See the '*LabJack UE9 User's Guide*' document for a list of all the available UE9 UD functions and their parameters and see the '*LabJack U3 User's Guide*' for the available U3 UD functions. For each function, the document lists the required inputs and outputs for the associated function with a description for each. In most of the functions the left-most Output Value will be an error code. An error code of zero means the function was executed properly. While using these functions, it is recommended that you always check the error code associated with each function call. Both of the user guides mentioned above have lists of LabJackUD error codes.

Two of the functions included do not have Output Values or Input Arguments. They are ljud_LoadDriver and ljud_Constants. These two M-Files both need to be ran before using any of the other UD functions. To run them, simply type their names in the MATLAB command window. The ljud_LoadDriver function loads the standard LabJackUD library into MATLAB. This library comes from the standard UD installation available from the URLs listed above and includes all of the LabJack UD functions found in the user guide documents. The ljud_LoadDriver function also loads a second library from the header file that was included in this download. The file LabJackUD_doublePtr.h is a special header file that was created to handle array passing.

If you look at the LabJack UD eGet function in section 4.3 of either of the user guides, you will notice that the last Input Argument is defined as a *long* data type. This can not handle arrays so the special header file was created with this last parameter changed from a *long* to a *doublePtr* data type. This allows the passing of arrays. When using the standard ljud_eGet function in MATLAB, it uses the standard LabJack UD library. When using the ljud_eGet_array function in MATLAB it uses this special library to accommodate the arrays. The ljud_eGet_array function is particularly handy when streaming (Streaming is only supported by the UE9). See the sample Simple_Stream.m for more information on using the ljud_eGet_array function. If you would like the ljud_LoadDriver function to display separate windows of the available LabJack UD functions with their parameters, open up the ljud_LoadDriver M-File and uncomment out the appropriate lines of code.

The M-File ljud_Constants is a file that simply contains all Device Types, Connection Types, IO Types and Channel Types and special channel types with their associated numeric values. This allows you to use the string as one of the Input Arguments when calling the function rather than just the numeric value.

Example LabJackUD Functions:
The following will walk through the commands for reading an analog input and controlling an analog output on a LabJack using MATLAB. Make sure you have gone through the above steps and added the drivers to the MATLAB search path. Plug your LabJack into your PC over USB and type the following commands into the MATLAB command window.

For more information about LabJack UD functions and the functions parameters refer to section 4.0 of the user guides.

The first two commands load the UD library and the Constants M-File.

```
>> ljud_LoadDriver
>> ljud_Constants
```

The commands that are exemplified below use ljud_ePut and ljud_eGet functions. These functions do AddRequest() and GoOne() all at once. The ePut function has the same functionality as the AddRequest-GoOne sequence, but the eGet function in addition does GetResult all in one step.

The next command will return the unique handle of your LabJack device.

```
>> [Error ljHandle] = ljud_OpenLabJack( LJ_dtU3, LJ_ctUSB, '1', 1 ) % (Note that LJ_dtUE9 would be used for a UE9 instead of LJ_dtU3)

Error =

     0
ljHandle =

   231572328
```

The next command will return the analog input data. This command can be used with the UE9 and the U3, but the U3's multifunctional IO's need to be configured as analog inputs before the analog input data can be retrieved.

```
>>% [Error] = ljud_ePut( ljHandle, LJ_ioPUT_ANALOG_ENABLE_BIT, 0, 1, 0 ) %  Need to configure the U3's IO to an analog input before
%  taking a measurement.

>> [Error AIN0] = ljud_eGet( ljHandle, LJ_ioGET_AIN, 0, 0, 0 )

Error =

     0

AIN0 =

    2.497
```

The next command will enable and set an analog output on DAC0.

```
>> [Error DAC0] = ljud_ePut ( ljHandle, LJ_ioPUT_DAC, 0, 2.5, 0 )

Error =

     0
```

Using the LabJackUD Samples:
The samples that are included in the download file and they demonstrate many of the different LabJackUD function calls with the LabJack UE9 and U3. Feel free to use these examples as a starting point for your own custom applications. If you have not already used the ljud_LoadDriver function, running one of these samples will load the driver.

Simple_Analog: Call this sample from the MATLAB command window by just typing the name of the sample. It has no parameters. It will return a read from AIN0 on your UE9.

Simple_Digital_Bit: Call this sample from the MATLAB command window by just typing the name of the sample. It has no parameters. The sample will prompt you for a value to set FIO0 to and it will return a value from FIO0 and FIO1. If you jumper a wire from FIO0 to FIO1 you can see FIO1 change as you set FIO0.

Simple_Logger: Call this sample from the MATLAB command window with the following notation: Simple_logger(time,dt). The parameter *time* will determine how long the sample runs, and the parameter *dt* determines the time between reads from the UE9. Both parameters have units of seconds. The sample will graph AIN0-AIN3 and return a table of AIN0-AIN3 and FIO0-FIO1.

Simple_SHT: Call this sample from the MATLAB command window by just typing the name of the sample. It has no parameters. This sample enables an EI-1050 Temperature and Humidity probe, reads the probes measurements and then disables the probe. This sample will display the temperature in Kelvin and Fahrenheit and it will display the Humidity as relative humidity.

Simple_Stream: Call this sample from the MATLAB command window by just typing the name of the sample. It has no parameters. This sample streams for a brief period of time from AIN0-AIN3 and returns the number of actual scans it made with a list of the data from each channel. This sample uses ljud_eGet_array function to return the array of streamed data. This is a single column array of doubles. In this example, since we are streaming from four channels the data has to be parsed from the returned array into four separate arrays for each channel.

Simple_Timer_Counter: Call this sample from the MATLAB command window by just typing the name of the sample. It has no parameters. This sample will configure a PWM output on FIO0 with a frequency of approximately 977 Hz and a 50% duty cycle. The sample will also enable Counter0 on FIO1. Jumper a wire from FIO0 to FIO1 and run the program. Counter will return approximately 977.

Revision History:
07/11/2006: Finished U3 and UE9 examples and constructed all functions.

02/14/2007: Fixed an error with the U3_Simple_Stream.m program which was causing the local variables to be cleared before the drivers were loaded.

04/24/2007: Updated ljud_constants.m with the U3 hardware version 1.21 constants. Also modified ljud_loaddriver.m to work with older versions of windows.