

LinearLabTools Step-by-step installation for Matlab users

February, 2017

This document shows all steps for the installation of LinearLabTools for Matlab users, including the installation of PScope.

Step 1: Linear Tech converter evaluation software installation

Install the software required by the demo board being evaluated. This is PScope for Analog to Digital converters, or LTDACgen for high-speed DACs such as the LTC2000. This document will use PScope as an example. The procedure for LTDACgen is similar.

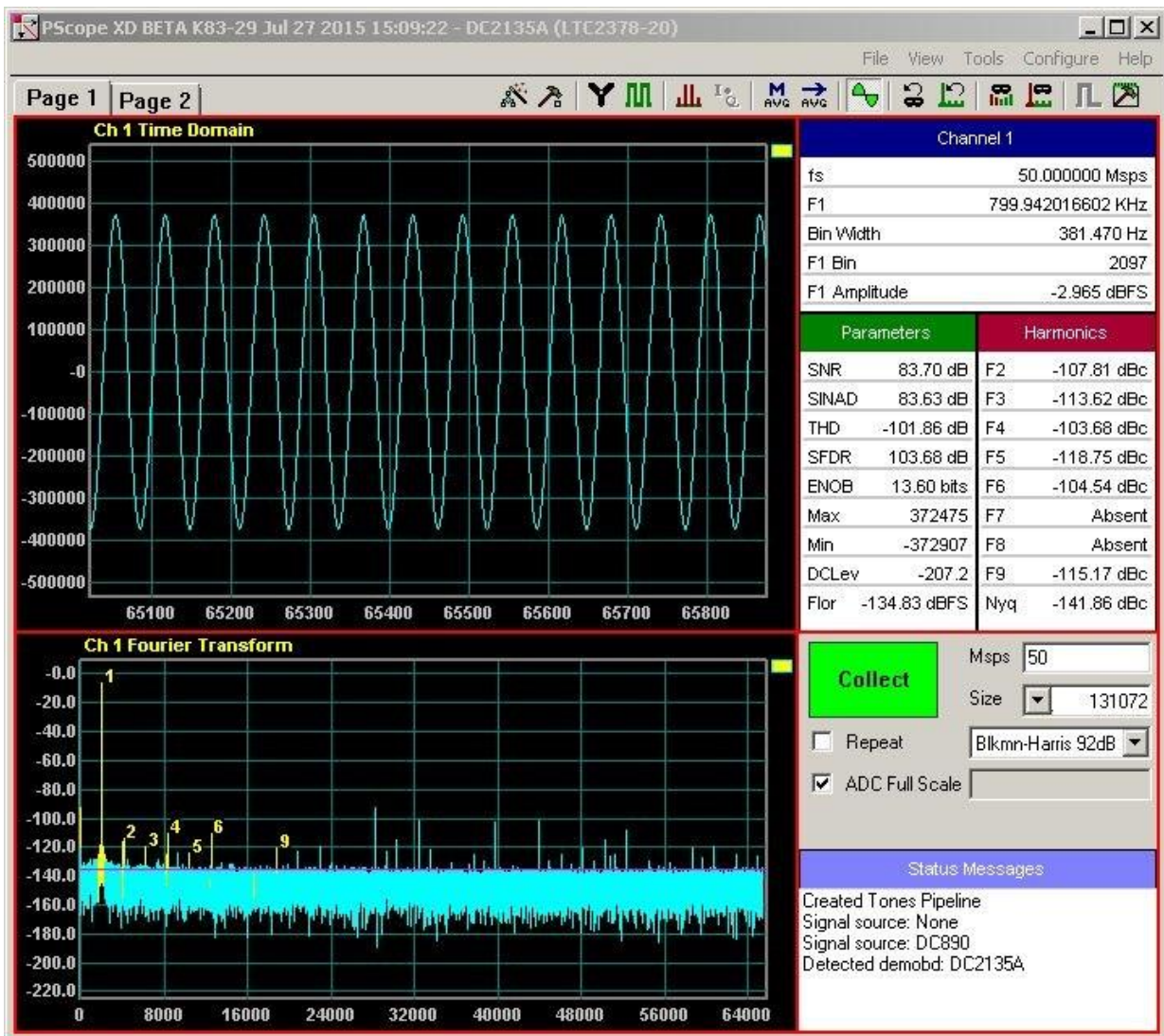
Run the installer and follow the directions:



PScope will then open. Follow the procedure in the data converter evaluation board's Demo Manual. This will include board configuration, connection of power supplies, clock signals, analog signals, and any other requirements.

IMPORTANT: Both PScope and LinearLabTools programs will show errors if the demo boards are not connected properly.

Once the hardware is set up properly, PScope should be able to collect data properly as shown below.



At this point the hardware is working properly and communicating with the host computer. Quit PScope before proceeding to avoid communication conflicts with LinearLabTools programs.

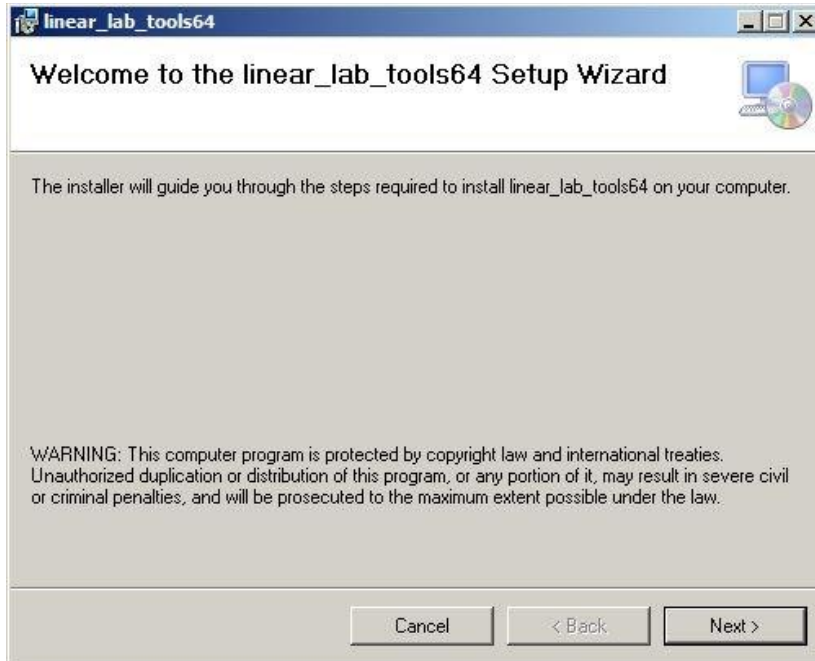
Step 2: Install LinearLabTools

Note that both 32-bit and 64-bit installers are provided. The choice depends on the Matlab environment. If using a 32-bit Matlab on a 64-bit system, use the 32-bit installer.

[32-bit installer](#)

[64-bit installer](#)

Run the installer and follow the directions.



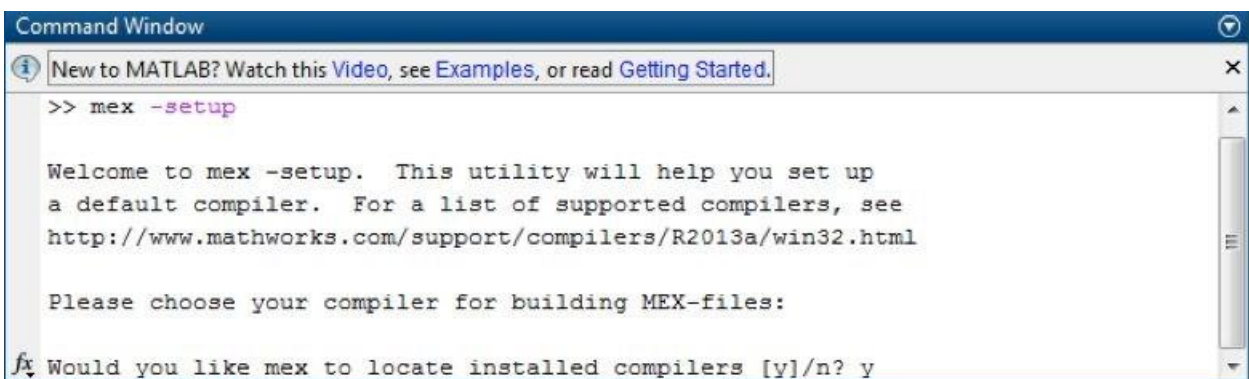
Step 3: Prepare Matlab Mex

Matlab requires mex-setup to be run in order to use external native libraries.

Run "mex -setup" and chose a compiler from the options listed as shown below.



```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.
fx >> mex -setup
```

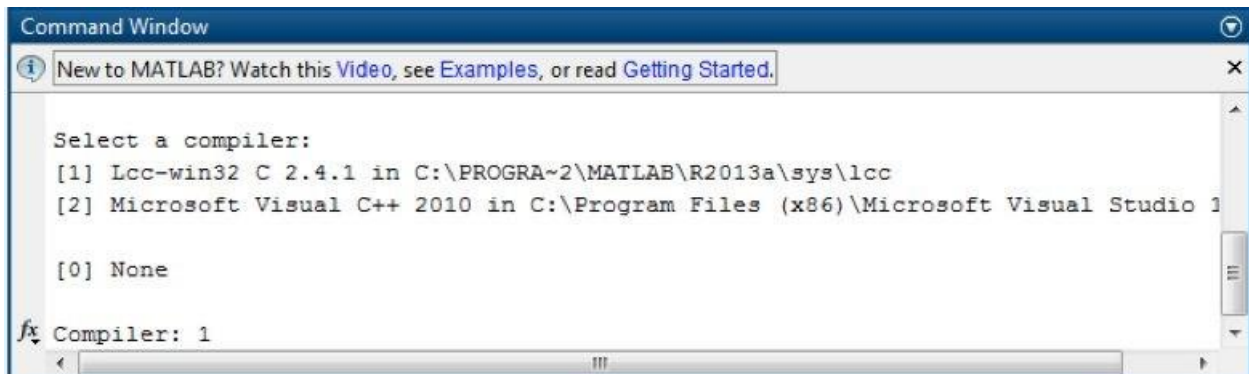


```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.
>> mex -setup

Welcome to mex -setup. This utility will help you set up
a default compiler. For a list of supported compilers, see
http://www.mathworks.com/support/compilers/R2013a/win32.html

Please choose your compiler for building MEX-files:

fx Would you like mex to locate installed compilers [y]/n? y
```

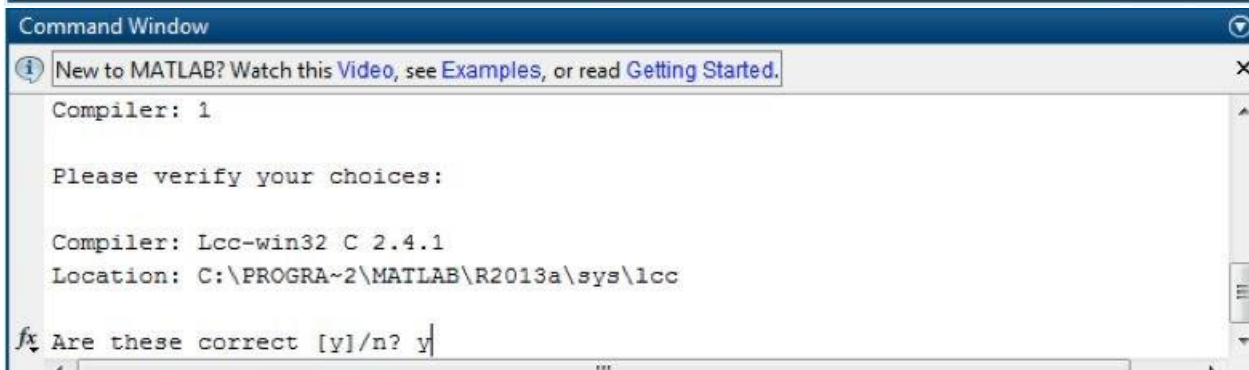


```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

Select a compiler:
[1] Lcc-win32 C 2.4.1 in C:\PROGRA~2\MATLAB\R2013a\sys\lcc
[2] Microsoft Visual C++ 2010 in C:\Program Files (x86)\Microsoft Visual Studio 1

[0] None

fx Compiler: 1
```



```
Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

Compiler: 1

Please verify your choices:

Compiler: Lcc-win32 C 2.4.1
Location: C:\PROGRA~2\MATLAB\R2013a\sys\lcc

fx Are these correct [y]/n? y
```

32-bit:

Matlab includes the Lcc-Win32 compiler so this option can always be used.

64-bit:

64-bit Matlab does not provide a compiler. If a compiler is not installed or listed in the available options, follow the instructions in the following link to install a compatible compiler:

<https://www.mathworks.com/matlabcentral/answers/101105-how-do-i-install-microsoft-windows-sdk-7-1>

Step 4: Matlab Path

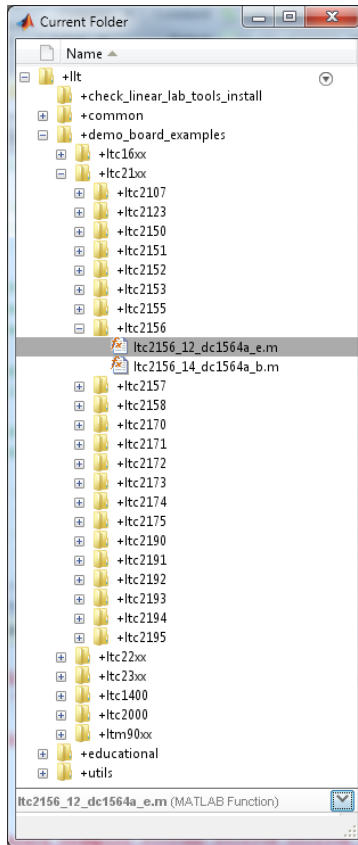
Add the absolute path to the "linear_lab_tools\matlab" or "linear_lab_tools64\matlab" folder before running any scripts. This can be added to the startup.m:

<https://www.mathworks.com/help/matlab/ref/startup.html?requestedDomain=www.mathworks.com>

Or use the Set Path icon in the environment section of the home tab of the top ribbon menu.

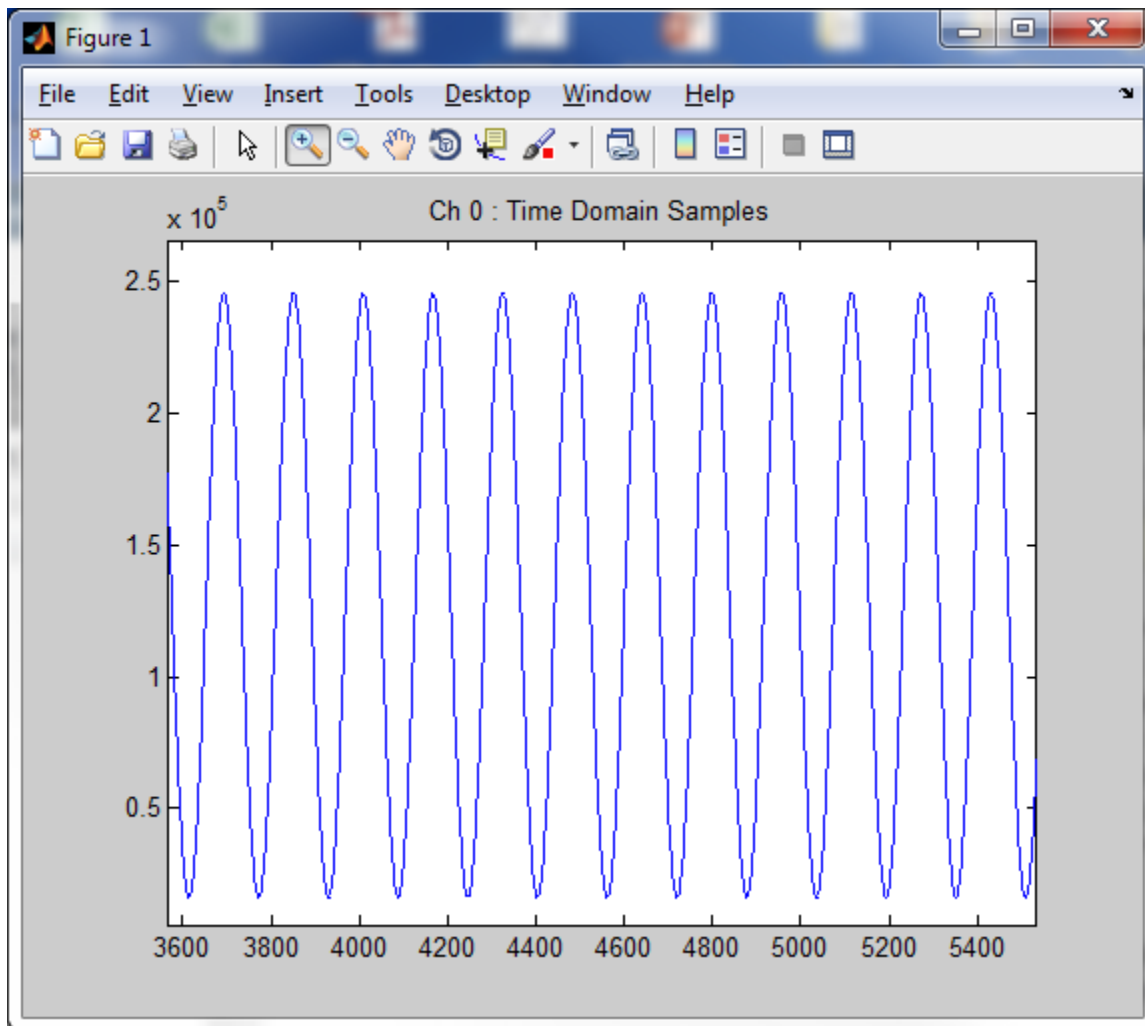
Step 5: Communicating with the Hardware

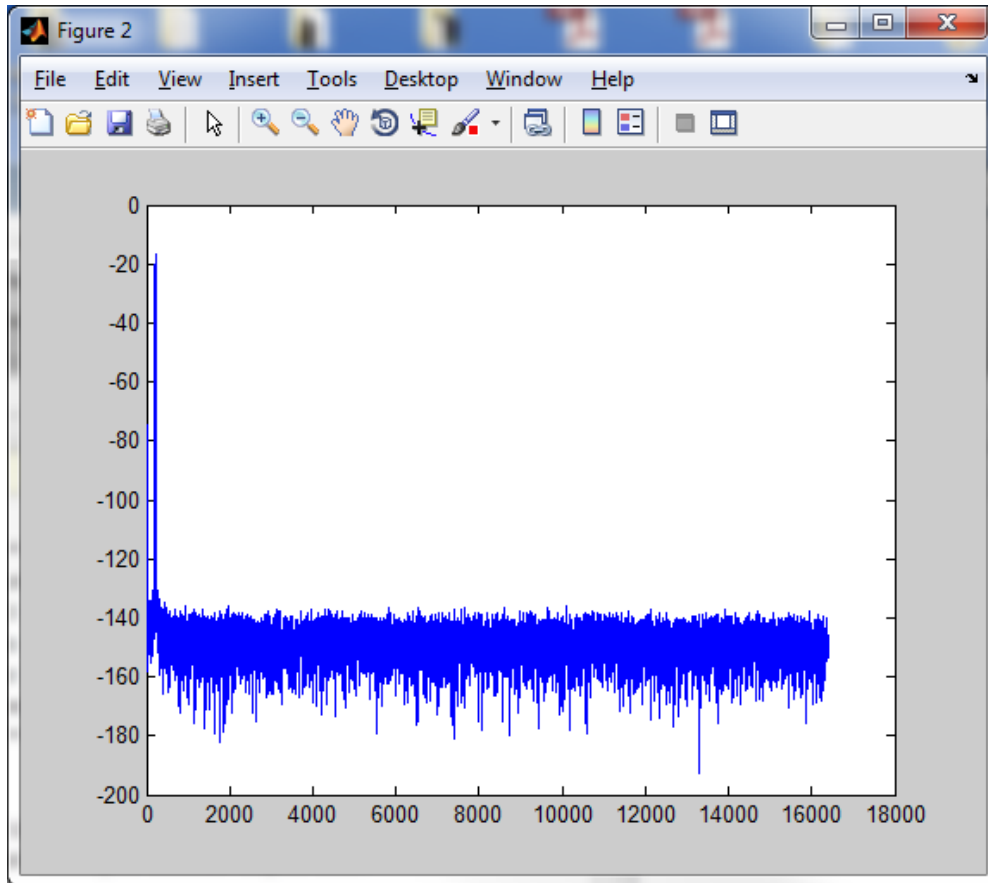
The figure below shows the organization of LinearLabTools:



To run the example Matlab script for your demo board type:
lIt.demo_board_examples.(ltc##xx).(part number).(example file) e.g.
lIt.demo_board_examples.ltc23xx.ltc2378.ltc2378_20_dc2135a
Press the enter key.

The script will go through the basic operations of capturing data from the board, then display time and frequency domain plots. Exact operations may vary from board to board. You should see plots similar to those below:





When run with no outputs as above, each demo-board example makes a time domain plot and a frequency domain plot for each channel and writes the data to a text file. You can also call the function passing it several parameters and returning the data for each channel.

For example:

```
num_samples = 16*1024;  
spi_registers = [];  
is_verbose = false;  
do_plot = true;  
do_write_to_file = false;
```

```
data = ltc2378_20_dc2135a(num_samples, spi_registers, is_verbose, do_plot,  
do_write_to_file);
```

Most functions have a signature similar to the one above. See the code for additional information. Many parts do not have SPI configuration, for these pass [] for the SPI registers. For other parts, look at the code for an example of correct SPI register format. For parts with multiple channels replace data with something like [ch0, ch1, ... chn] for the function output.

At this point, data from the demo board is stored in an array in the program. You can then extend the functionality of the program as required for your evaluation, incorporate other test hardware such as signal generators, etc.