last update 9-20-17

LL_USB_single_python Win sample app for Windows – overview

Sample app "LL_USB_single_python.exe" makes use of our DLL "LL_USB2k.dll" and driver "LL_USB2k.sys" built for WinXP/2k/Vista/7/8/10. The DLL and driver for your system can be found at:

install file download

Install driver as instructed in the documentation accompanying the download, and place the DLL in the same folder as the LL_USB_single_python.exe application.

The sample makes use of many of the functions mentioned in the documentation for our Windows DLL API which can be found at our website. The link below provides the API as well as much more information that further explains using the functions, and some information concerning interaction with the hardware. Below is the link:

lawsonlabs.com web link

Sample app LL_USB_python.exe makes use of only a handful of the functions available, in order to demonstrate their usage in connecting to the device and performing some basic tasks. The app connects to the device, reads/displays the voltage from the default channel 0, sets channel 6, and then reads/displays the voltage again. Next the app sets DAC 0 to 3.5 volts and changes to channel 3. Please connect DAC 0 to channel 3 (analog out 1 to 4+ and GD to 4-). The app reads the voltage from that channel (should read approx. 3.5 volts). It then demonstrates how to get the "actual rate" the board will run at based on a requested rate, since the board is not capable of always running at the precise rate one might request. It sets a temporary rate variable to 1000Hz and then calls into the library to get what the "actual board rate" would be (approx. 1000.651466Hz) at that requested rate. Note that trying to set the rate, for example with a call to **EX_SendRate(...)** with the rate that was returned by that call, could then set it to a different rate. That returned rate should only be used within an app for timing within the app. Finally, the app does a single-channel 100Hz scan on channel 3, displaying 25 voltages, then exits.

It makes use of the following function calls within the DLL:

EX_ConnectOneDevice(...)
EX_GetOneConversion(...)
EX_SendChan(...)
EX_SendDAC(...)
EX_GetCalculatedRate(...)
EX_SetScanType(...)
EX_SetDataLogOptions(...)
EX_Run(...)
EX_CheckScanStatus(...)
EX_GetScanDataDbl(...)
EX_Stop(...)
EX_StopComplete(...)

The device ID is hard-coded to 5206 at the top of the "main()" function call so that will need to be changed to "your" device ID before running the sample application.

The next page shows how it looks when run. Note that the single-channel scan data is for channel 3, which was the last channel selected, and DAC0 was connected to it with it's output set to 3.5 volts. Only ten scans are read in order to keep the example (and displayed ouput) simple.


C:\WINDOWS\system32\cmd.exe
signing on (may take a while) - WAIT . . .
signed on
getting a conversion - WAIT . . .
got volts chan0: 0.095942
changing to full-scale channel 6 - WAIT . . .
sendChan success, lastDigin: 0
getting a conversion - WAIT . . .
got volts chan6: 4.999999
setting DAC0 to 3.5 volts - WAIT . . .
sendDAC success
changing to channel 3 (should be DAC0 volts)
see pythonSampleAppMini.pdf for how to connect - WAIT . . .
sendChan success, lastDigin: 0
getting a conversion - WAIT . . .
got volts chan3: 3.498849
checking actual rate when 1000Hz is requested - WAIT . . .
1000Hz would be: 1000.651466
==========
====== S C A N N I N G P R O C E S S ======
==========
setting scan type CMND_SINGLE_CHAN_SCAN . . .
setting scan log type SCAN_USE_DATA_ARRAY . . .
calling EX_Run . . .
EX_Run success
scan: 0 volts: 3.488807
scan: 1 volts: 3.489892
scan: 2 volts: 3.490825
scan: 3 volts: 3.491700
scan: 4 volts: 3.492512
scan: 5 volts: 3.493311
scan: 6 volts: 3.494084
scan: 7 volts: 3.494915
scan: 8 volts: 3.495708
scan: 9 volts: 3.496286
scan: 10 volts: 3.496968
scan: 11 volts: 3.497612
scan: 12 volts: 3.498268
scan: 13 volts: 3.498849
scan: 14 volts: 3.499309
scan: 15 volts: 3.499729
scan: 16 volts: 3.500242
scan: 17 volts: 3.500802
scan: 18 volts: 3.501070
scan: 19 volts: 3.501260
calling EX_Stop . . .
EX_Stop success
calling EX_StopComplete . . .
EX_StopComplete success
exiting application - WAIT . . .
C:\pythonAppTest>