Beta LJLogM - Windows Only

Supported devices

- T4
- T7
- T8

Overview

LJLogM - Beta is a Windows application that is capable of reading 20+ registers from a single T-Series device. The collected data can be named, organized, logged, and visualized in several ways. To facilitate collecting data from so many registers and displaying the data on the screen, LJLogM allows users to define "Tags" which can be organized into "Groups" and those groups can be selected from to be displayed on any of the data viewing pages.

Downloads

The beta version (latest v1.095) is currently not released in the LabJack installer. Instead, the application is being distributed through this page in a .zip file. After downloading and extracting the .zip file, the .exe can be installed into the LabJack Applications directory "C:\Program Files (x86)\LabJack\Applications" manually or by double clicking on the "install.bat" file. The "install.bat" file pops up a window to request for administrative permissions and moves the "LJLogM.exe" file into the Program Files directory.

LJLogM_v1.093.zip

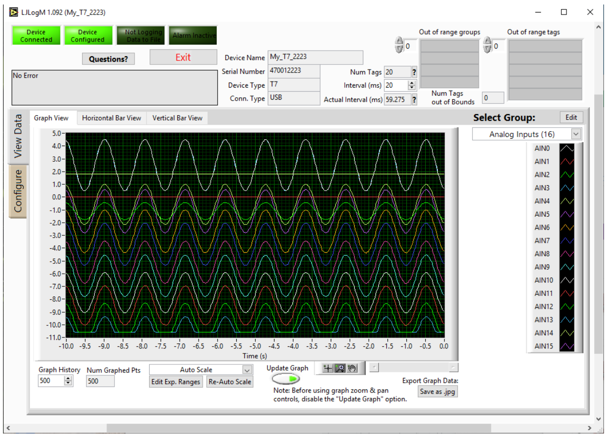
LJLogM_v1.094.zip

LJLogM_v1.095.zip

Subsections

- View Data
- Opening and Searching for Devices
- Device Configuration
- Define Tags, Groups, Ranges, and Alarms
- Data Logging, Viewing, and Reporting
- Device and Program Performance

Program Highlights



LJLogM 1.092 (My_T7_2223)

No Error

Device Name: My_T7_2223
Serial Number: 470012223
Device Type: T7
Conn. Type: USB
Num Tags: 20
Interval (ms): 20
Actual Interval (ms): 59.275
Out of range groups: 0
Out of range tags: 0
Num Tags out of Bounds: 0

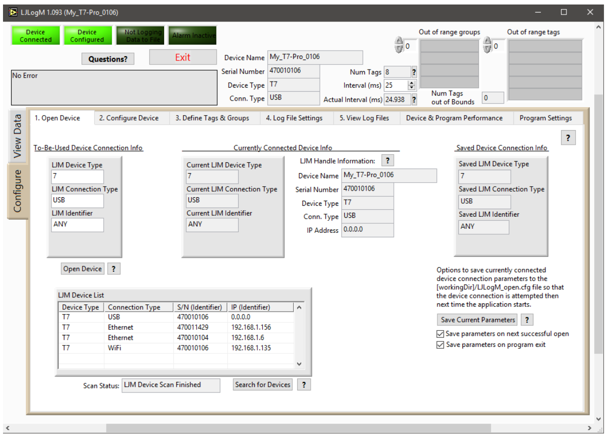
Select Group: Analog Inputs (16)

Graph History: 500
Num Graphed Pts: 500

Note: Before using graph zoom & pan controls, disable the "Update Graph" option.

View Data in Real Time

LJLogM features 3 different data views: graph view, horizontal view, and vertical view. Each view has the ability to display a single data group worth of data at a time. The Graph view has features to take screen shots, as well as the ability to pause the graph and zoom in/out on data during a data logging session. The horizontal and vertical bar views can be used to see singular values within a customizable range. These views can also be used to display visual alarms, and whether or not a value is within a given range.



LJLogM 1.093 (My_T7-Pro_0106)

No Error

Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 25
Actual Interval (ms): 24.938
Out of range groups: 0
Out of range tags: 0
Num Tags out of Bounds: 0

To-Be-Used Device Connection Info
LJM Device Type: 7
LJM Connection Type: USB
LJM Identifier: ANY

Currently Connected Device Info
Current LJM Device Type: 7
Current LJM Connection Type: USB
Current LJM Identifier: ANY
LJM Handle Information:
Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
IP Address: 0.0.0.0

Saved Device Connection Info
Saved LJM Device Type: 7
Saved LJM Connection Type: USB
Saved LJM Identifier: ANY

Options to save currently connected device connection parameters to the [workingDir]/LJLogM_open.cfg file so that the device connection is attempted then next time the application starts.

☑ Save parameters on next successful open
☑ Save parameters on program exit

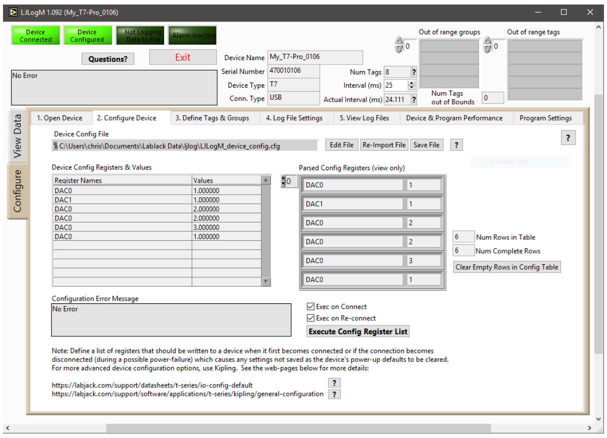
LJM Device List

Device Type	Connection Type	S/N (Identifier)	IP (Identifier)
T7	USB	470010106	0.0.0.0
T7	Ethernet	470011429	192.168.1.156
T7	Ethernet	470010104	192.168.1.6
T7	WiFi	470010106	192.168.1.135

Scan Status: LJM Device Scan Finished

Opening & Searching for Devices

When LJLogM starts, it automatically attempts to connect to a device given the saved LJM Open parameters in the LJLogM_open.cfg file. LJLogM can be re-configured and a new device can be opened through this tab.



No Error

Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 25
Actual Interval (ms): 24.111
Num Tags out of Bounds: 0

Device Config File: C:\Users\chris\Documents\LabJack Data\ljlog\LJLogM_device_config.cfg

Device Config Registers & Values

Register Names	Values
DAC0	1.000000
DAC1	1.000000
DAC0	2.000000
DAC0	2.000000
DAC0	3.000000
DAC0	1.000000

Parsed Config Registers (view only)

DAC0	1
DAC1	1
DAC0	2
DAC0	2
DAC0	3
DAC0	1

6 Num Rows in Table
6 Num Complete Rows

Configuration Error Message: No Error

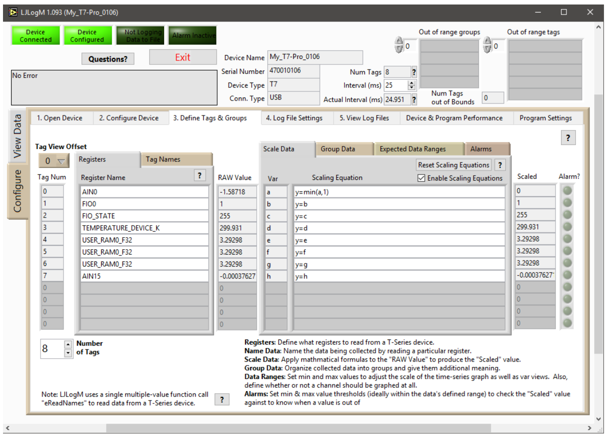
☑ Exec on Connect
☑ Exec on Re-connect

Note: Define a list of registers that should be written to a device when it first becomes connected or if the connection becomes disconnected (during a possible power-failure) which causes any settings not saved as the device's power-up defaults to be cleared. For more advanced device configuration options, use Kipling. See the web-pages below for more details:

https://labjack.com/support/datasheets/t-series/io-config-default
https://labjack.com/support/software/applications/t-series/kipling/general-configuration

Device Configuration

When LJLogM starts and a device becomes successfully connected, LJLogM can perform a list of register write commands to configure a T-Series device. Each register is written one value at a time. This tool also makes it easier to enable AIN_EF and DIO_EF features that aren't made easily accessable by Kipling or would typically require the use of the Register Matrix. The registers that get written can be found in the LJLogM_device_config.cfg file.



No Error

Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 25
Actual Interval (ms): 24.951
Num Tags out of Bounds: 0

Tag View Offset: 0

☑ Enable Scaling Equations

Tag Num	Register Name	RAW Value	Var	Scaling Equation	Scaled
0	AIN0	-1.58718	a	y=min(a,1)	0
1	FIO0	1	b	y=b	1
2	FIO_STATE	255	c	y=c	255
3	TEMPERATURE_DEVICE_K	299.931	d	y=d	299.931
4	USER_RAM0_F32	3.29298	e	y=e	3.29298
5	USER_RAM0_F32	3.29298	f	y=f	3.29298
6	USER_RAM0_F32	3.29298	g	y=g	3.29298
7	AIN15	-0.00037627	h	y=h	-0.00037627
0		0			0
0		0			0
0		0			0
0		0			0

Alarm?

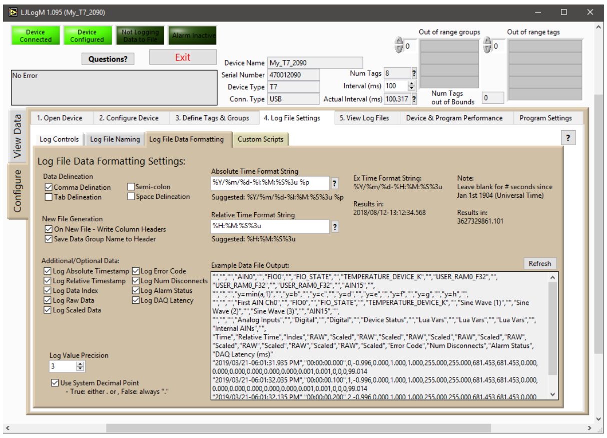
8 Number of Tags

Note: LJLogM uses a single multiple-value function call "eReadNames" to read data from a T-Series device.

Registers: Define what registers to read from a T-Series device.
Name Data: Name the data being collected by reading a particular register.
Scale Data: Apply mathmatical formulas to the "RAW Value" to produce the "Scaled" value.
Group Data: Organize collected data into groups and give them additional meaning.
Data Ranges: Set min and max values to adjust the scale of the time-series graph as well as var views. Also, define whether or not a channel should be graphed at all.
Alarms: Set min & max value thresholds (ideally within the data's defined range) to check the "Scaled" value against to know when a value is out of

Define Tags, Groups, Ranges, and Alarms

To facilitate collecting data from more than 16 channels at a time and providing a method of viewing all of the data being collected, LJLogM uses the terminology "Tags" and "Groups". Tags are consist of a register and a human-readable name as well as any other relevant configuration. The Tags can be organized into 10 different groups which can then be viewed one at a time on each of the data view pages. This is especially useful for customers looking to collect data from a T7 paired with a Mux80 which provides 84 single-ended analog inputs.



No Error

Device Name: My_T7_2090
Serial Number: 470012090
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 100
Actual Interval (ms): 100.317
Out of range groups: 0
Out of range tags: 0
Num Tags out of Bounds: 0

Log File Data Formatting Settings:

Data Delineation
☑ Comma Delination
☐ Tab Delineation
☐ Semi-colon
☐ Space Delineation

New File Generation
☑ On New File - Write Column Headers
☑ Save Data Group Name to Header

Additional/Optional Data:
☑ Log Absolute Timestamp
☑ Log Relative Timestamp
☑ Log Data Index
☑ Log Raw Data
☑ Log Scaled Data
☑ Log Error Code
☑ Log Num Disconnects
☑ Log Alarm Status
☑ Log DAQ Latency

Log Value Precision: 3

☑ Use System Decimal Point
- True: either . or , False: always "."

Absolute Time Format String: %Y/%m/%d-%I:%M:%S%3u %p
Suggested: %Y/%m/%d-%I:%M:%S%3u %p

Relative Time Format String: %H:%M:%S%3u
Suggested: %H:%M:%S%3u

Ex Time Format String:
%Y/%m/%d-%H:%M:%S%3u

Results in:
2018/08/12-13:12:34.568

Note:
Leave blank for # seconds since
Jan 1st 1904 (Universal Time)

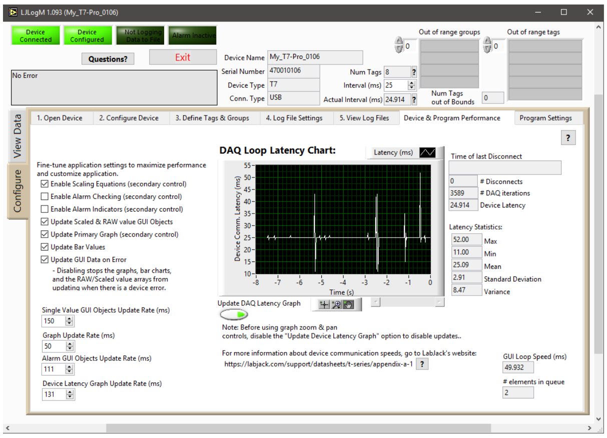
Results in:
3627329861.101

Example Data File Output:

```
"","","","AIN0","","FIO0","","FIO_STATE","","TEMPERATURE_DEVICE_K","","USER_RAM0_F32","",
"USER_RAM0_F32","","USER_RAM0_F32","","AIN15","",
"","","","y=min(a,1)","","y=b","","y=c","","y=d","","y=e","","y=f","","y=g","","y=h","",
"","","","First AIN Ch0","","FIO0","","FIO_STATE","","TEMPERATURE_DEVICE_K","","Sine Wave (1)","","Sine
Wave (2)","","Sine Wave (3)","","AIN15","",
"","","","Analog Inputs","","Digital","","Digital","","Device Status","","Lua Vars","","Lua Vars","","Lua Vars","",
"Internal AINs","",
"Time","Relative Time","Index","RAW","Scaled","RAW","Scaled","RAW","Scaled","RAW","Scaled","RAW",
"Scaled","RAW","Scaled","RAW","Scaled","RAW","Scaled","Error Code","Num Disconnects","Alarm Status",
"DAQ Latency (ms)"
"2019/03/21-06:01:31.935 PM","00:00:00.000",0,-0.996,0.000,1.000,1.000,255.000,255.000,681.453,681.453,0.000,
0.000,0.000,0.000,0.000,0.000,0.001,0.001,0,0,0,99.014
"2019/03/21-06:01:32.035 PM","00:00:00.100",1,-0.996,0.000,1.000,1.000,255.000,255.000,681.453,681.453,0.000,
0.000,0.000,0.000,0.000,0.000,0.001,0.001,0,0,0,99.014
"2019/03/21-06:01:32.135 PM","00:00:00.200",2,-0.996,0.000,1.000,1.000,255.000,255.000,681.453,681.453,0.000
```

Data Logging, Viewing, and Reporting

This version of LJLogM has additional features to enable more customized log files. There are settings to customize the file name as well as what data gets saved to the file. After data has been logged, there is another section that lists what files have been created and has application short cuts to open the files in OpenOffice or Excel.



No Error

Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 25
Actual Interval (ms): 24.914
Out of range groups: 0
Out of range tags: 0
Num Tags out of Bounds: 0

Fine-tune application settings to maximize performance and customize application.
☑ Enable Scaling Equations (secondary control)
☐ Enable Alarm Checking (secondary control)
☐ Enable Alarm Indicators (secondary control)
☑ Update Scaled & RAW value GUI Objects
☑ Update Primary Graph (secondary control)
☑ Update Bar Values
☑ Update GUI Data on Error
- Disabling stops the graphs, bar charts, and the RAW/Scaled value arrays from updating when there is a device error.

Single Value GUI Objects Update Rate (ms): 150
Graph Update Rate (ms): 50
Alarm GUI Objects Update Rate (ms): 111
Device Latency Graph Update Rate (ms): 131

Note: Before using graph zoom & pan
controls, disable the "Update Device Latency Graph" option to disable updates..

For more information about device communication speeds, go to LabJack's website:
https://labjack.com/support/datasheets/t-series/appendix-a-1

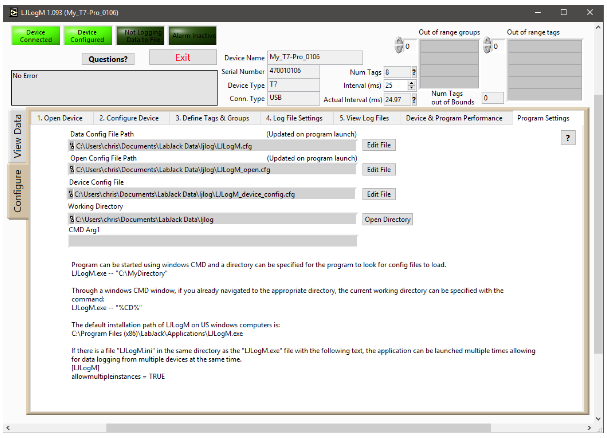
Time of last Disconnect:
0 # Disconnects
3589 # DAQ iterations
24.914 Device Latency

Latency Statistics:
52.00 Max
11.00 Min
25.09 Mean
2.91 Standard Deviation
8.47 Variance

GUI Loop Speed (ms): 49.932
# elements in queue: 2

Program Performance

When collecting data from a LabJack at high data rates or trying to collect data at tightly controlled intervals, there are several factors to consider. The active connection type, how many channels are being logged, what type of data is being collected, and what calculations are being run afterwards are all important. This feature of LJLogM provides a method to detect, diagnose, and understand communication speed issues and limitations.



No Error

Device Name My_T7-Pro_0106
Serial Number 470010106
Device Type T7
Conn. Type USB
Num Tags 8
Interval (ms) 25
Actual Interval (ms) 24.97
Num Tags out of Bounds 0

Data Config File Path (Updated on program launch)
C:\Users\chris\Documents\LabJack Data\ljlog\LJLogM.cfg

Open Config File Path (Updated on program launch)
C:\Users\chris\Documents\LabJack Data\ljlog\LJLogM_open.cfg

Device Config File
C:\Users\chris\Documents\LabJack Data\ljlog\LJLogM_device_config.cfg

Working Directory
C:\Users\chris\Documents\LabJack Data\ljlog

CMD Arg1

Program can be started using windows CMD and a directory can be specified for the program to look for config files to load.

```
LJLogM.exe -- "C:\MyDirectory"
```

Through a windows CMD window, if you already navigated to the appropriate directory, the current working directory can be specified with the command:

```
LJLogM.exe -- "%CD%"
```

The default installation path of LJLogM on US windows computers is:
C:\Program Files (x86)\LabJack\Applications\LJLogM.exe

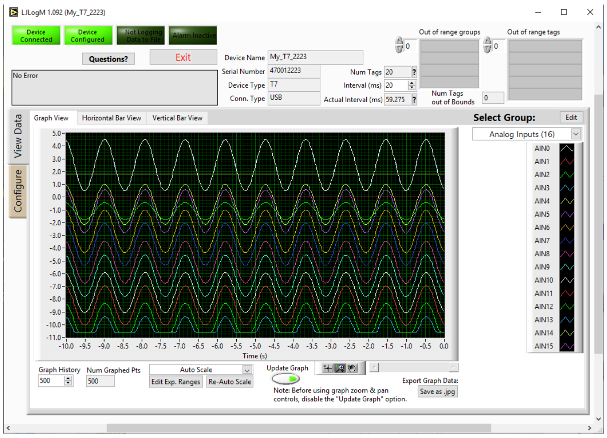
If there is a file "LJLogM.ini" in the same directory as the "LJLogM.exe" file with the following text, the application can be launched multiple times allowing for data logging from multiple devices at the same time.

```
[LJLogM]
allowmultipleinstances = TRUE
```

Program Settings

LJLogM is a highly configurable program and all of its features/settings can be configured by editing .cfg files so that the program starts up in the desired state. The program's configurations are saved when ever the program exits and loaded when it launches.

View Data



No Error

Device Name My_T7_2223

Serial Number 470012223

Device Type T7

Conn. Type USB

Num Tags 20

Interval (ms) 20

Actual Interval (ms) 59.275

Num Tags out of Bounds 0

Select Group: Analog Inputs (16)

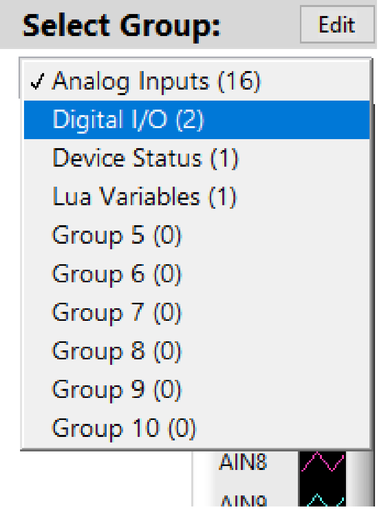
Graph History 500

Num Graphed Pts 500

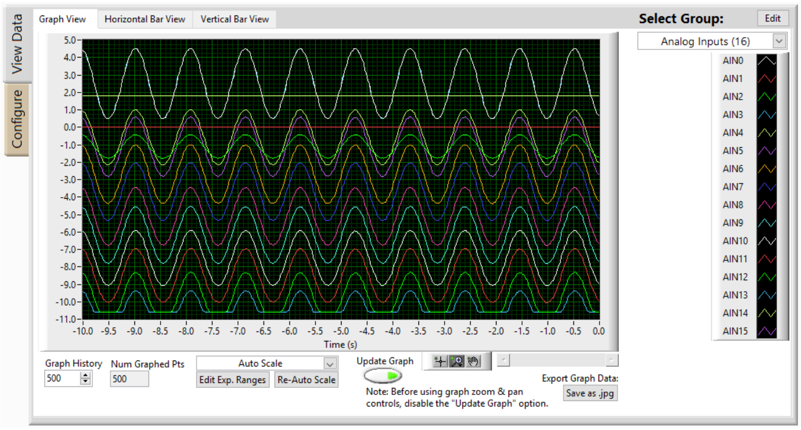
Note: Before using graph zoom & pan controls, disable the "Update Graph" option.

LJLogM features 3 different data views: graph view, horizontal view, and vertical view. Each view has the ability to display a single data group worth of data at a time.

Select Group: Choose one of the 10 available data groups to display on the selected view.



Graph View



Select Group: Analog Inputs (16)

Graph History 500

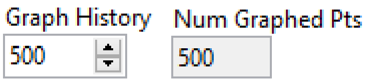
Num Graphed Pts 500

Note: Before using graph zoom & pan controls, disable the "Update Graph" option.

Number of displayed tags: The graph view page can display up to 20 tags at a time and several features are available.

Tag Names: The names of all of the displayed tags by editing the `Tag Name` value under: `Configure` → `3. Define Tags & Groups` → `Tag Name`.

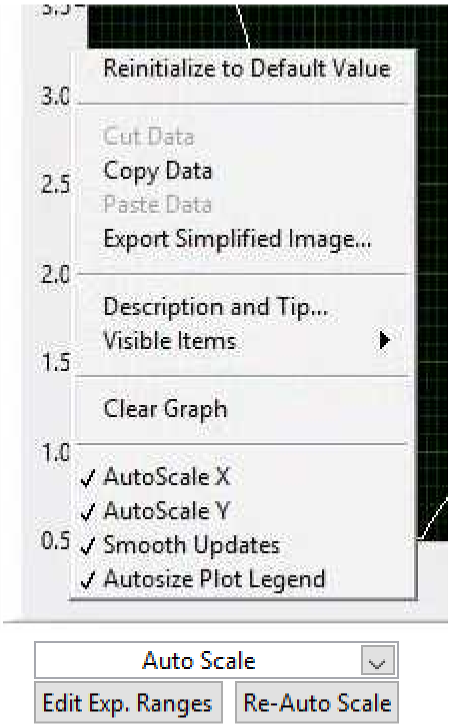
Graph History: Define how many historical data points to store in memory and display on the graph.



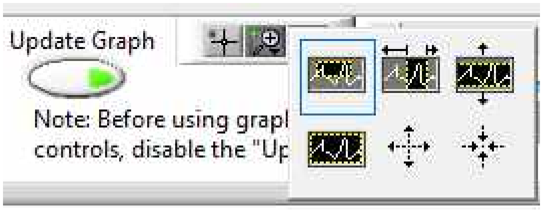
Graph History: 500

Num Graphed Pts: 500

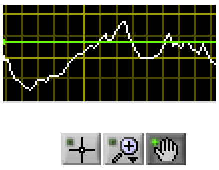
Graph Scaling: Enable/Disable graph Auto-Scaling features by right-clicking on the graph axis or select between one of the pre-defined modes.



Zoom Tools: After pausing the graph, zoom in, zoom out, or selectively highlight and view data by using the zoom tools. It is also possible to disable the "AutoScale X" or "AutoScale Y" modes and have the graph update while zoomed in.

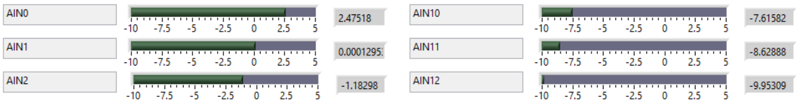


Hand Move: After pausing the screen and zooming into a section of data, use the hand tool or the scroll bar to view data around the zoomed in area.

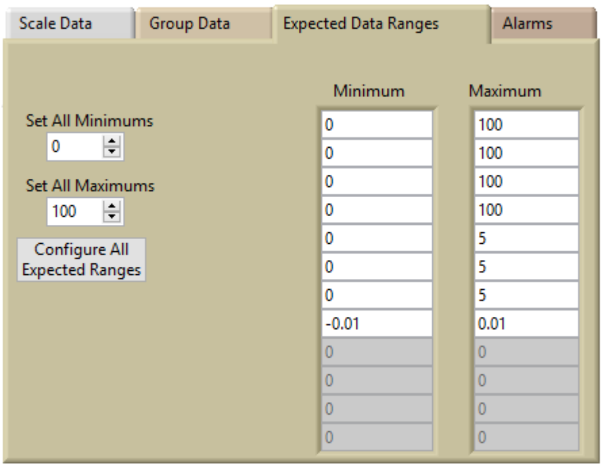


Take Screen Shots: If using the windows print screen or snipping tools aren't easily accessible, there is a button that saves a `.jpg` image of the data currently visible on the graph.

Horizontal Bar View



The horizontal bar view can display up to 20 tags at a time. The range of the horizontal bars can be adjusted by editing the tag's expected data range values: `Configure` → `3. Define Tags & Groups` → `Expected Data Ranges`.

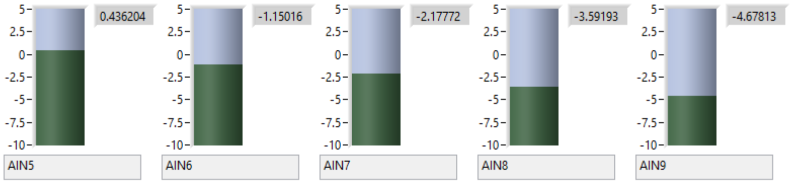


Set All Minimums 0

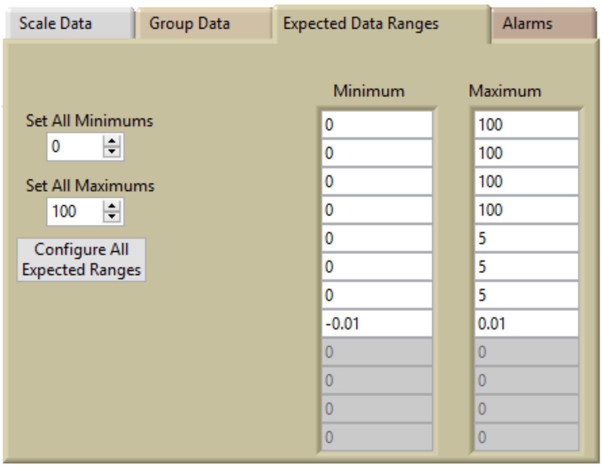
Set All Maximums 100

Minimum	Maximum
0	100
0	100
0	100
0	100
0	5
0	5
0	5
-0.01	0.01
0	0
0	0
0	0
0	0

Vertical Bar View



The vertical bar view can display up to 10 tags at a time. The range of each of the vertical bars can be adjusted by editing the tag's expected data range values: `Configure` → `3. Define Tags & Groups` → `Expected Data Ranges`.



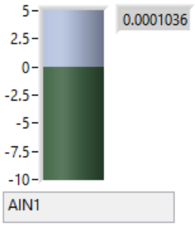
Set All Minimums 0

Set All Maximums 100

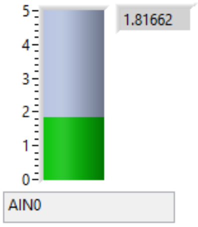
Minimum	Maximum
0	100
0	100
0	100
0	100
0	5
0	5
0	5
-0.01	0.01
0	0
0	0
0	0
0	0

Bar View Alarm Indication

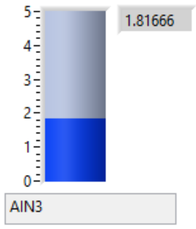
The colors of both the horizontal and vertical bars are capable of indicating when alarm conditions occur. Alarms can be edited in the in the `Configure` → `3. Define Tags & Groups` → `Alarms` section and look like the following:



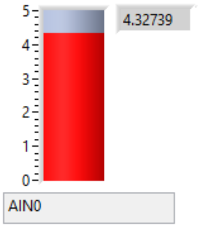
No Alarm Configured



Value within the acceptable range

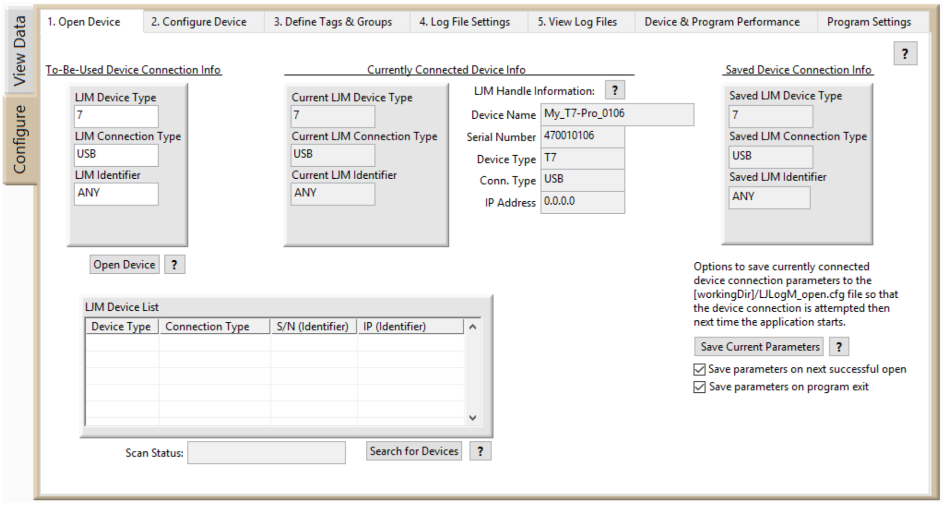


Value below the acceptable range



Value above the acceptable range

Opening and Searching for Devices



To-Be-Used Device Connection Info

LJM Device Type: 7
LJM Connection Type: USB
LJM Identifier: ANY

Currently Connected Device Info

Current LJM Device Type: 7
Current LJM Connection Type: USB
Current LJM Identifier: ANY

LJM Handle Information:
Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
IP Address: 0.0.0.0

Saved Device Connection Info

Saved LJM Device Type: 7
Saved LJM Connection Type: USB
Saved LJM Identifier: ANY

Options to save currently connected device connection parameters to the [workingDir]/LJLogM_open.cfg file so that the device connection is attempted then next time the application starts.

☑ Save parameters on next successful open
☑ Save parameters on program exit

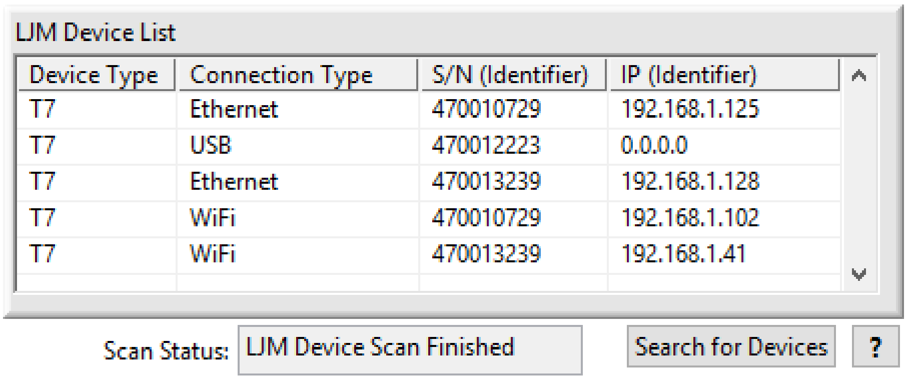
LJM Device List

Device Type	Connection Type	S/N (Identifier)	IP (Identifier)

Scan Status:

When LJLogM starts, it automatically attempts to connect to a device given a set of saved LJM Open parameters that are saved in the `LJLogM_open.cfg` file. This tab allows users to edit these settings, search for new devices, and open a new device.

Searching for a new device: To search for a new device, press the `Search for Devices` button. The search can be around 5-30 seconds depending on a computers configuration, but it performs a LJM_ListAllS function call to find any available T-Series devices over USB, Ethernet or WiFi.



LJM Device List

Device Type	Connection Type	S/N (Identifier)	IP (Identifier)
T7	Ethernet	470010729	192.168.1.125
T7	USB	470012223	0.0.0.0
T7	Ethernet	470013239	192.168.1.128
T7	WiFi	470010729	192.168.1.102
T7	WiFi	470013239	192.168.1.41

Scan Status: LJM Device Scan Finished

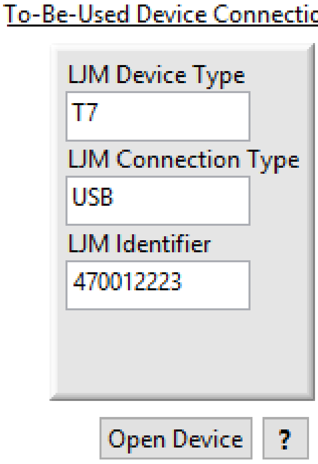
Device Not Found? If a device is not being correctly found, some useful debugging steps can be found in the Device Not Found App-Note. If a network connection needs to be debugged, check out the Setup WiFi and Ethernet App-Note as well.

Opening a new device: To open a new device, type in new connection parameters and then press the `Open Device` button. Any LJM_OpenS parameter can be used, a short list of the valid parameters is as follows:

- Device Type: `ANY`, `T4`, `T7`

- Connection Type: `ANY`, `USB`, `Ethernet`, `WiFi`

- Identifier: `ANY`, `IP Address` (ex: `192.168.1.101`), `Device Name` (ex: `My_T7-Pro_0106`), `Serial Number` (ex: `470010106`)



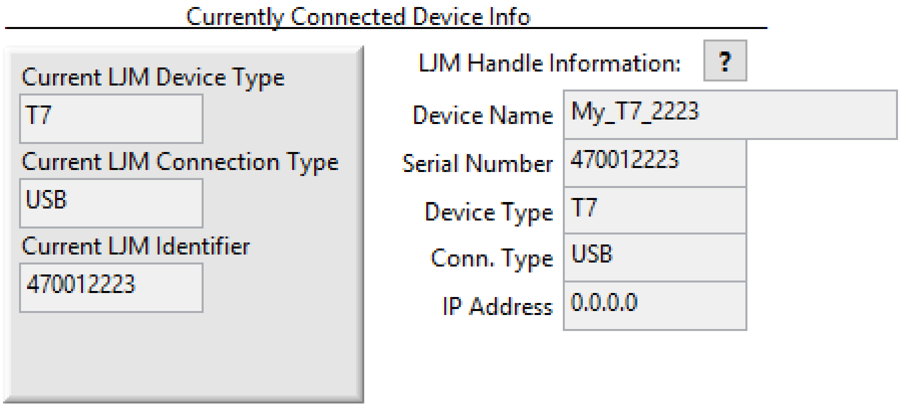
To-Be-Used Device Connection Info

LJM Device Type: T7

LJM Connection Type: USB

LJM Identifier: 470012223

Currently Active Device Information: After a device has been connected, some basic information about the device and its active connection type can be found next to the device opening controls.



Currently Connected Device Info

Current LJM Device Type: T7

Current LJM Connection Type: USB

Current LJM Identifier: 470012223

LJM Handle Information:

Device Name: My_T7_2223

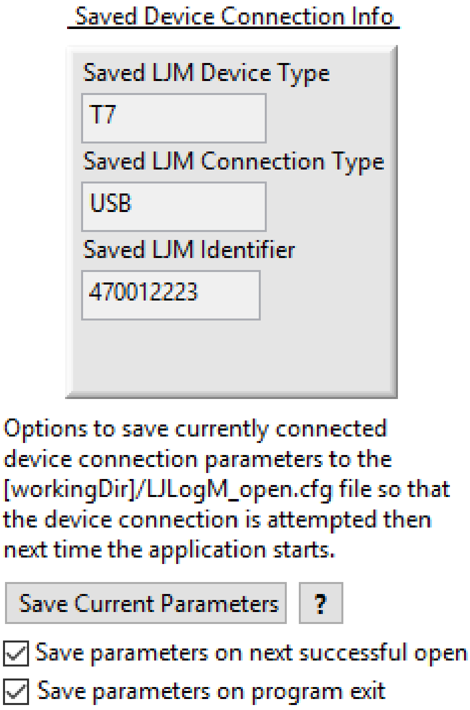
Serial Number: 470012223

Device Type: T7

Conn. Type: USB

IP Address: 0.0.0.0

Saved Device Connection Info: The LJM parameters that are used to open a device when LJLogM first starts are visible in this section and there are options to control how these settings are updated.



Saved Device Connection Info

Saved LJM Device Type: T7

Saved LJM Connection Type: USB

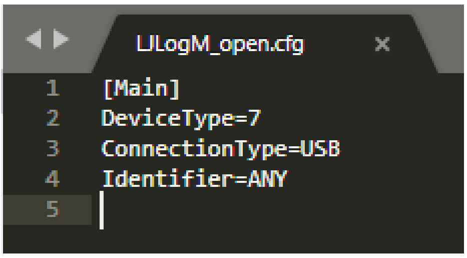
Saved LJM Identifier: 470012223

Options to save currently connected device connection parameters to the [workingDir]/LJLogM_open.cfg file so that the device connection is attempted then next time the application starts.

☑ Save parameters on next successful open

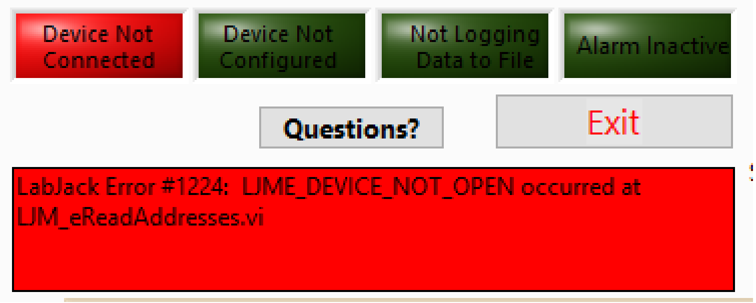
☑ Save parameters on program exit

These settings can be edited manually by editing the `LJLogM_open.cfg` file with any text editing program, the format of this file looks like the following:



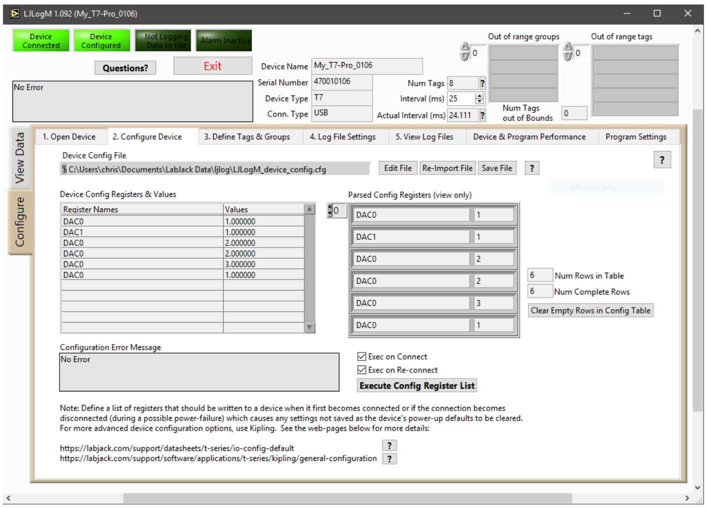
```
[Main]
DeviceType=7
ConnectionType=USB
Identifier=ANY
```

Device Opening Error: If a device doesn't get opened when LJLogM starts, there will be two sections that turn red indicating the error and a device scan will be initiated. It looks like the following:



LabJack Error #1224: LJME_DEVICE_NOT_OPEN occurred at LJM_eReadAddresses.vi

Device Configuration



No Error

Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 25
Actual Interval (ms): 24.111
Num Tags out of Bounds: 0

Device Config File: C:\Users\chris\Documents\LabJack Data\ljlog\LJLogM_device_config.cfg

Device Config Registers & Values

Register Names	Values
DAC0	1.000000
DAC1	1.000000
DAC0	2.000000
DAC0	2.000000
DAC0	3.000000
DAC0	1.000000

Parsed Config Registers (view only)

DAC0	1
DAC1	1
DAC0	2
DAC0	2
DAC0	3
DAC0	1

6 Num Rows in Table
6 Num Complete Rows

Configuration Error Message: No Error

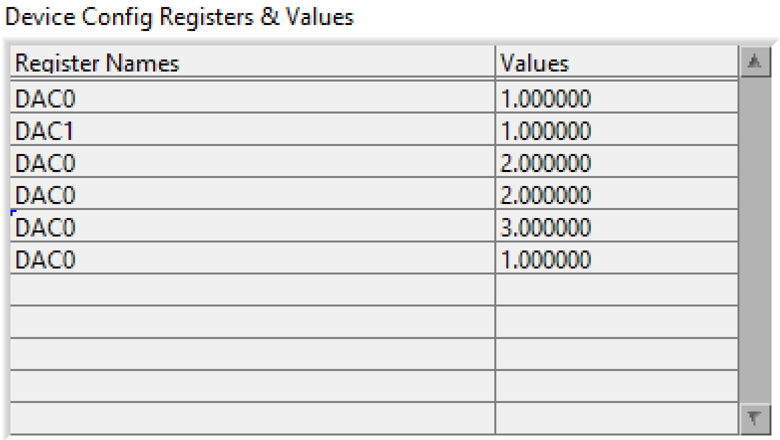
☑ Exec on Connect
☑ Exec on Re-connect

Note: Define a list of registers that should be written to a device when it first becomes connected or if the connection becomes disconnected (during a possible power-failure) which causes any settings not saved as the device's power-up defaults to be cleared. For more advanced device configuration options, use Kipling. See the web-pages below for more details:

https://labjack.com/support/datasheets/t-series/io-config-default
https://labjack.com/support/software/applications/t-series/kipling/general-configuration

When LJLogM starts and a device becomes successfully connected, LJLogM can perform a list of register write commands to configure a T-Series device. Each register is written one value at a time. This tool also makes it easier to enable AIN_EF and DIO_EF features that aren't as easily accessible through Kipling and its Register Matrix. The registers that get written can be found in the `LJLogM_device_config.cfg` file. A full list of registers that can be found on the Modbus Map page

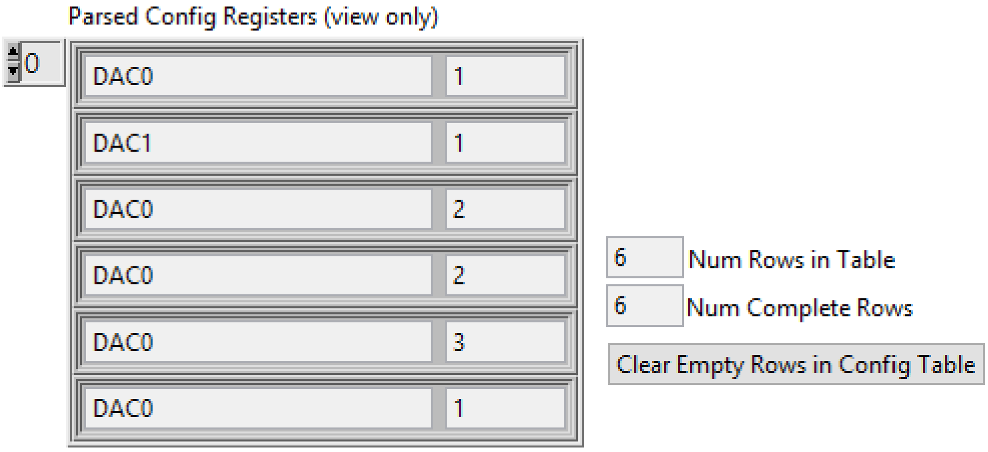
Defining Registers: Enter one register per row in the left column and the value that needs to be written to that register in the right column of the "Device Config Registers & Values" table.



Device Config Registers & Values

Register Names	Values
DAC0	1.000000
DAC1	1.000000
DAC0	2.000000
DAC0	2.000000
DAC0	3.000000
DAC0	1.000000

Parsed List of Config Registers: As new registers get added, a parsed list of registers and values get updated. The LabVIEW 7.1 table GUI element is not very clear/easy to use, so some extra tooling is in place to get things cleaned up if the `.cfg` file itself isn't being edited.



Parsed Config Registers (view only)

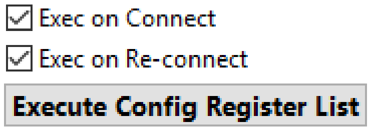
0

DAC0	1
DAC1	1
DAC0	2
DAC0	2
DAC0	3
DAC0	1

6 Num Rows in Table

6 Num Complete Rows

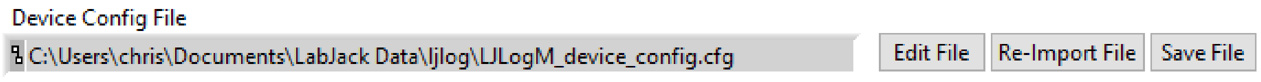
Configure the Device: After all of the required registers have been entered into the table, press the `Execute Config Register List` to instruct LJLogM to write the values to the connected T-Series device. LJLogM can be configured to execute this list of commands whenever a device connects, or when a device disconnects and then re-connects which can happen when the device power cycles. When a device disconnects, error code 1239, "LJME_RECONNECT_FAILED" is returned. When it re-connects there will be no error and the device will get reconfigured if the `Exec on Re-connect` is enable.



☑ Exec on Connect

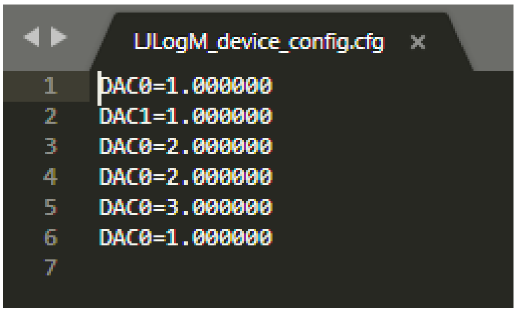
☑ Exec on Re-connect

Config File: Instead of editing the list of configuration registers through LJLogM, users can edit the `LJLogM_device_config.cfg` file in their preferred text editor and then re-import the file into LJLogM with the following controls and format:



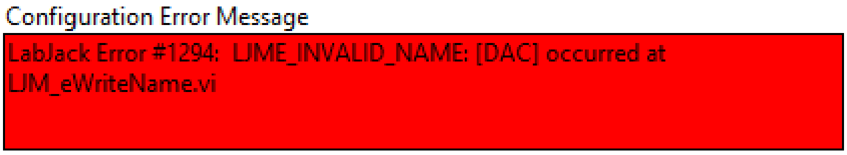
Device Config File

C:\Users\chris\Documents\LabJack Data\ljlog\LJLogM_device_config.cfg



```
DAC0=1.000000
DAC1=1.000000
DAC0=2.000000
DAC0=2.000000
DAC0=3.000000
DAC0=1.000000
```

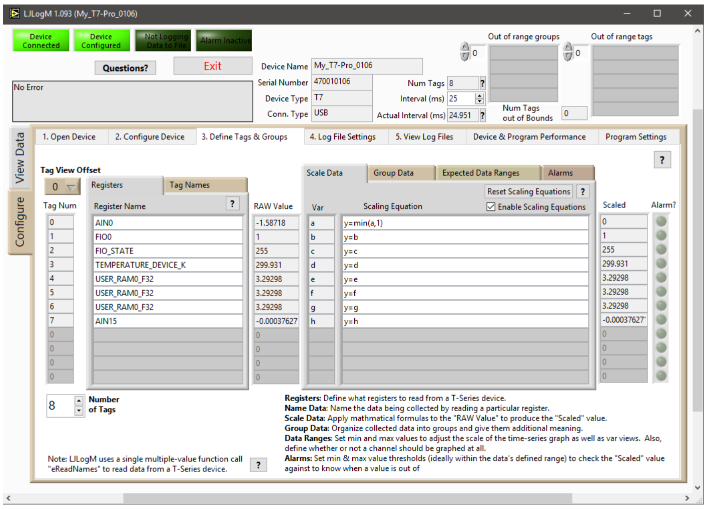
Device Configuration Errors: If device configuration fails, the error window will get populated with an LJM error code and additional information as follows:



Configuration Error Message

LabJack Error #1294: LJME_INVALID_NAME: [DAC] occurred at LJM_eWriteName.vi

Define Tags, Groups, Ranges, and Alarms



No Error

Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 25
Actual Interval (ms): 24.951
Num Tags out of Bounds: 0

Tag View Offset: 0

☑ Enable Scaling Equations

Tag Num	Register Name	RAW Value	Var	Scaling Equation	Scaled	Alarm?
0	AIN0	-1.58718	a	y=min(a,1)	0	●
1	FIO0	1	b	y=b	1	●
2	FIO_STATE	255	c	y=c	255	●
3	TEMPERATURE_DEVICE_K	299.931	d	y=d	299.931	●
4	USER_RAM0_F32	3.29298	e	y=e	3.29298	●
5	USER_RAM0_F32	3.29298	f	y=f	3.29298	●
6	USER_RAM0_F32	3.29298	g	y=g	3.29298	●
7	AIN15	-0.00037627	h	y=h	-0.00037627	●
0		0			0	●
0		0			0	●
0		0			0	●
0		0			0	●

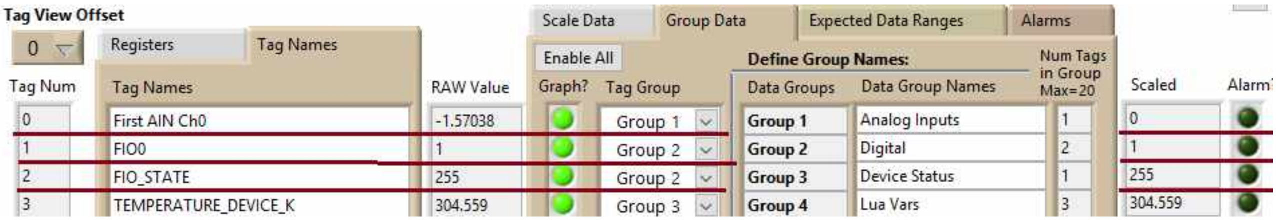
8 Number of Tags

Registers: Define what registers to read from a T-Series device.
Name Data: Name the data being collected by reading a particular register.
Scale Data: Apply mathmatical formulas to the "RAW Value" to produce the "Scaled" value.
Group Data: Organize collected data into groups and give them additional meaning.
Data Ranges: Set min and max values to adjust the scale of the time-series graph as well as var views. Also, define whether or not a channel should be graphed at all.
Alarms: Set min & max value thresholds (ideally within the data's defined range) to check the "Scaled" value against to know when a value is out of

Note: LJLogM uses a single multiple-value function call "eReadNames" to read data from a T-Series device.

To facilitate collecting data from more than 16 channels at a time and providing a method of viewing all data being collected, LJLogM uses the terminology "Tags" and "Groups". Tags consist of a register and a human-readable name as well as any other relevant configuration. Tags can be organized into 10 different Groups which can be viewed one at a time on each of the data view pages. This is especially useful for customers looking to collect data from a T7 paired with a Mux80 which provides 84 single-ended analog inputs. There are several controls on this page, but before continuing, find and understand the following:

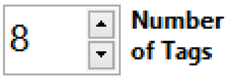
Horizontal Page Organization: Depending on what tabs are selected, data for a given Tag is organized horizontally across the page. Starting with `Tag Num` and finishing with the `Scaled` and `Alarm?` indicators. The one exception to this rule is how data Groups are defined and given names.



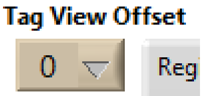
Tag View Offset: 0

Tag Num	Tag Names	RAW Value	Graph?	Tag Group	Data Groups	Data Group Names	Num Tags in Group Max=20	Scaled	Alarm
0	First AIN Ch0	-1.57038	●	Group 1	Group 1	Analog Inputs	1	0	●
1	FIO0	1	●	Group 2	Group 2	Digital	2	1	●
2	FIO_STATE	255	●	Group 2	Group 3	Device Status	1	255	●
3	TEMPERATURE_DEVICE_K	304.559	●	Group 3	Group 4	Lua Vars	3	304.559	●

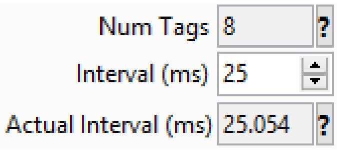
Number of Tags: This defines how many tags or registers LJLogM is reading from the connected T-Series device. To add a new tag, increment the Number of Tags by one.


8 Number of Tags

Tag View Offset: Sets the first Tag number in the display list of 12 Tags, allowing users to scroll through all enabled Tags.


Tag View Offset
0

Data Collection Rate: To configure how fast data is collected from a device, look towards the top of the page and find the following controls. The "Interval (ms)" control lets users define how fast LJLogM should attempt to collect data from a device, and the "Actual Interval (ms)" indicator shows how fast data is actually being collected. For more data collection statistics, press the `?` button or look at the "Program Performance" section of LJLogM.

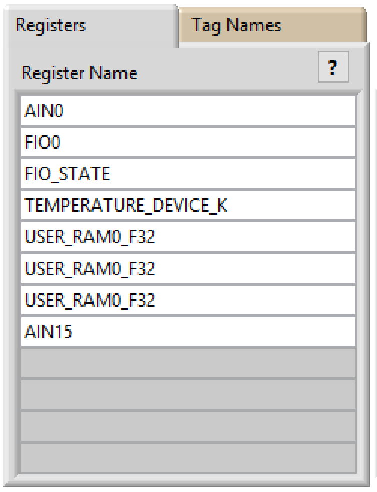


Num Tags 8

Interval (ms) 25

Actual Interval (ms) 25.054

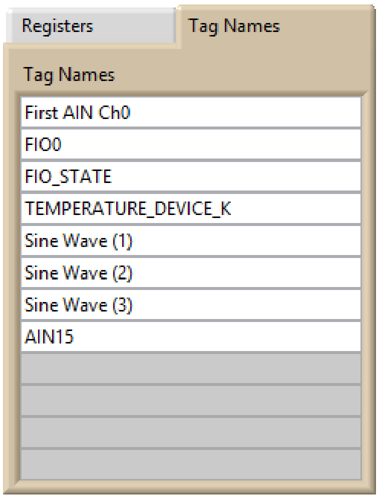
Registers: To edit the register being read by LJLogM from the connected T-Series device, edit the following field. The `?` button is a shortcut to take users to the Modbus Map page where a list of all the available registers can be found. For example: `AIN0`, `AIN9`, `EIO2`, `WIFI_STATUS`, `DIO0_EF_READ_A_F`, or `AIN3_EF_READ_A`, etc.



Register Name

AIN0
FIO0
FIO_STATE
TEMPERATURE_DEVICE_K
USER_RAM0_F32
USER_RAM0_F32
USER_RAM0_F32
AIN15

Tag Names: This control allows customers to define the name that gets displayed in all of the data views and what name is saved to the `.log` file. After editing a register, the associated Tag Name will be automatically changed to become the name of the register. To edit the name of the tag, simply edit the associated field.



Tag Names

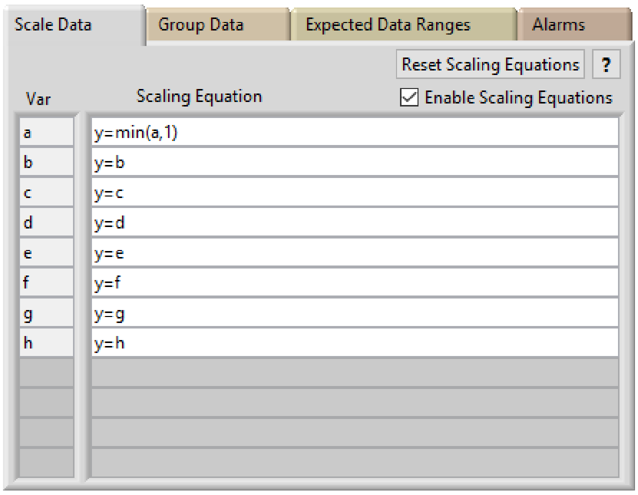
First AIN Ch0
FIO0
FIO_STATE
TEMPERATURE_DEVICE_K
Sine Wave (1)
Sine Wave (2)
Sine Wave (3)
AIN15

RAW Value: After defining a list of valid registers, the `RAW Value` array will begin updating. These are the values directly from the device before scaling.

Scaling Equations: This should be something like `y=a`, where `y` is the scaled output and `a` is the raw value of Tag `0`. `b` through `p` would be Tags `1` to `15`, and so on. Everything after `//` is ignored, so use for comments. A few examples:

```
y=c                         // Scaled value equal to raw value from 3rd row
y=100*c                     // EI-1034/LM34 voltage to deg F
y=c-273.15                  // deg K to deg C
y=((c-273.15)*9/5)+32       // deg K to deg F
y=TCVoltsToTemp[K:c:a]      // Type K, t/c voltage from 3rd row, CJ temp from 1st row
```

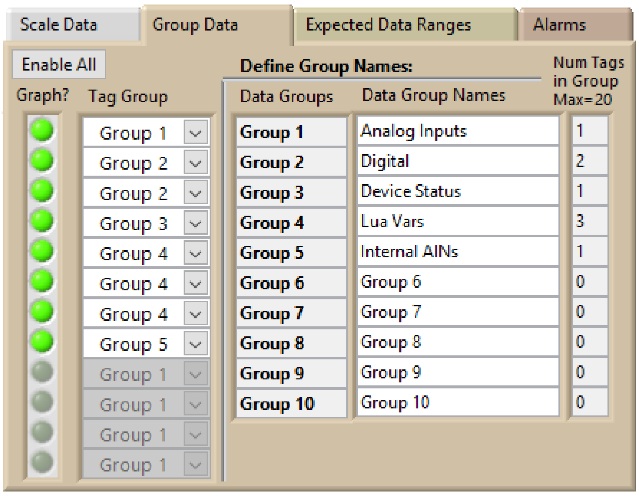
For more details see the LJLog/Stream Scaling Equations page. The defined scaling equations can be reset to default values by pressing the `Reset Scaling Equations` button. If scaling equations do not need to be applied or the maximum data throughput of LJLogM is being reached, uncheck the `Enable Scaling Equations` check box.



☑ Enable Scaling Equations

Var	Scaling Equation
a	y=min(a,1)
b	y=b
c	y=c
d	y=d
e	y=e
f	y=f
g	y=g
h	y=h

Group Data: After a list of Tags have been defined, they can be organized into Groups so that related data can be graphed together or displayed in the bar views. There are 10 groups that data can be organized into. Groups can be given customized human readable names.

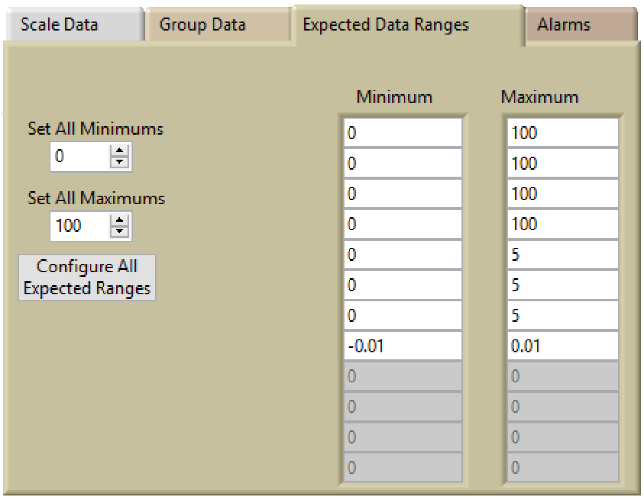


Graph?	Tag Group
●	Group 1
●	Group 2
●	Group 2
●	Group 3
●	Group 4
●	Group 4
●	Group 4
●	Group 5
○	Group 1
○	Group 1
○	Group 1
○	Group 1

Define Group Names:

Data Groups	Data Group Names	Num Tags in Group Max=20
Group 1	Analog Inputs	1
Group 2	Digital	2
Group 3	Device Status	1
Group 4	Lua Vars	3
Group 5	Internal AINs	1
Group 6	Group 6	0
Group 7	Group 7	0
Group 8	Group 8	0
Group 9	Group 9	0
Group 10	Group 10	0

Expected Data Ranges: If the data being read have explicit ranges or if users wish to customize the scales of the various views, edit the expected data ranges for each of the tags. All of the tags have the same ranges, edit the set all minimums and maximums controls and press the `Configure All Expected Ranges` button.

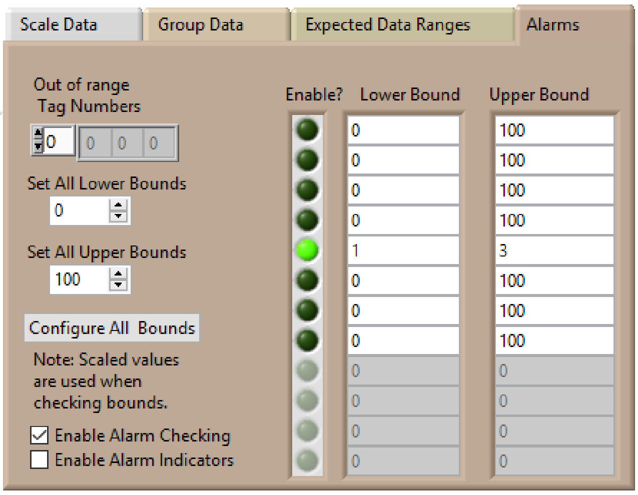


Set All Minimums 0

Set All Maximums 100

Minimum	Maximum
0	100
0	100
0	100
0	100
0	5
0	5
0	5
-0.01	0.01
0	0
0	0
0	0
0	0

Alarms: Alarms can be configured, enabled, and disabled on a tag-by-tag basis. This tab is where they are edited. If all of the tags need to have the same alarm bounds, edit the `Set All Lower Bounds` and `Set All Upper Bounds` controls and press the `Configure All Bounds` button. Once configured, the `Enable?` indicator should be pressed to enable the alarm. When enabled it will light up bright green. The `Enable Alarm Checking` option enables LJLogM to perform the necessary math operations to calculate whether the values are in or out of bounds. The `Enable Alarm Indicators` option is a final catch-all that enables or disables the GUI from being updated to show the various alarm conditions.



Out of range Tag Numbers 0 | 0 0 0

Set All Lower Bounds 0

Set All Upper Bounds 100

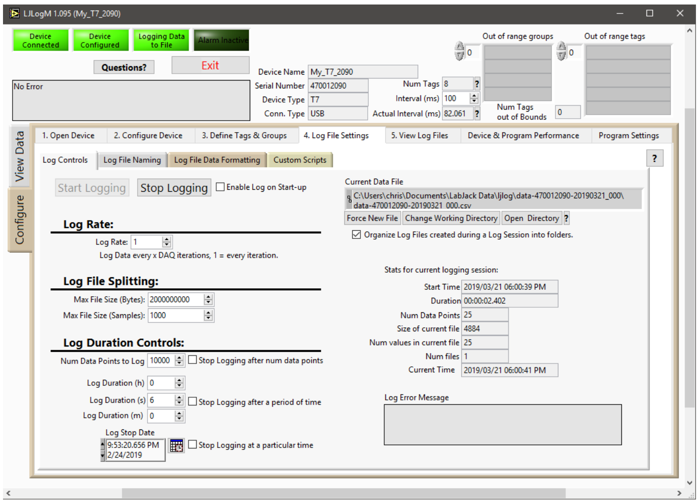
Note: Scaled values are used when checking bounds.

☑ Enable Alarm Checking
☐ Enable Alarm Indicators

Enable?	Lower Bound	Upper Bound
○	0	100
○	0	100
○	0	100
○	0	100
●	1	3
○	0	100
○	0	100
○	0	100
○	0	0
○	0	0
○	0	0
○	0	0

Data Logging, Viewing, and Reporting

LJLogM is capable of collecting data from T-Series devices in a flexible and configurable manner. LJLogM is also capable of executing custom scripts that can execute command line scripts, custom Python programs, etc. that save collected data to databases, custom servers, or to the cloud. Once data has been collected and saved locally (after a logging session or an experiment), generated data files can be opened in local programs like Excel, Open Office, or have custom scripts be executed to expedite data analysis in programs like Matlab or Python.



No Error

Device Name: My_T7_2090
Serial Number: 470012090
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 100
Actual Interval (ms): 82.061
Num Tags out of Bounds: 0

□ Enable Log on Start-up

Log Rate:
Log Rate: 1
Log Data every x DAQ iterations, 1 = every iteration.

Log File Splitting:
Max File Size (Bytes): 2000000000
Max File Size (Samples): 1000

Log Duration Controls:
Num Data Points to Log: 10000 □ Stop Logging after num data points
Log Duration (h): 0
Log Duration (s): 6 □ Stop Logging after a period of time
Log Duration (m): 0
Log Stop Date: 9:53:20.656 PM 2/24/2019 □ Stop Logging at a particular time

Current Data File: C:\Users\chris\Documents\LabJack Data\ljlog\data-470012090-20190321_000\data-470012090-20190321_000.csv
☑ Organize Log Files created during a Log Session into folders.

Stats for current logging session:
Start Time: 2019/03/21 06:00:39 PM
Duration: 00:00:02.402
Num Data Points: 25
Size of current file: 4884
Num values in current file: 25
Num files: 1
Current Time: 2019/03/21 06:00:41 PM

Log Error Message

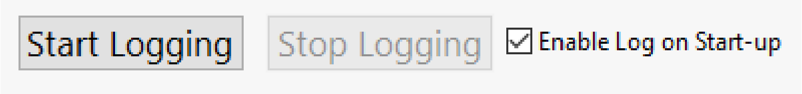
LJLogM is capable of generating customized log files that can be easily imported into spreadsheet tools such as OpenOffice, Excel, or any other post-processing tools. There are settings to customize the file name, data formatting and optional data. After data has been logged, users can go to the tab `5. View Log Files`, where they can find a list of the generated files and there are application short cuts to open the files in OpenOffice or Excel.

Subsections

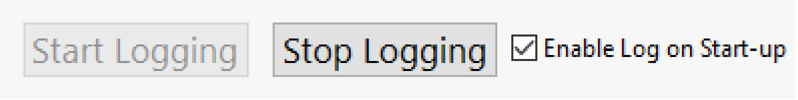
- Custom Scripts
- timestamps

Log Controls

Start and Stop Logging: These controls start and stop data from being logged to a file. When logging, the program-wide logging indicator as well as the logging statistics will be updated. The `Enable Log on Start-up` checkbox allows users to configure LJLogM to begin logging data when the application starts and a device becomes connected/configured.

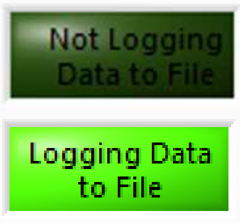


☑ Enable Log on Start-up

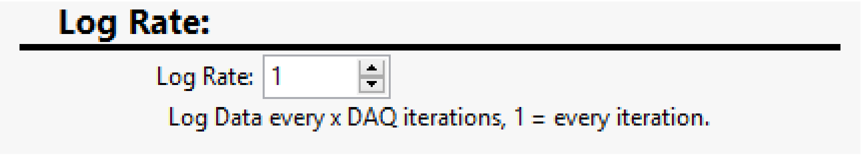


☑ Enable Log on Start-up

Logging Indicators: This program-wide indicator informs users whether or not data is being logged to a file. When it is illuminated light green, data is being logged to a file.



Log Rate: LJLogM can be configured to save data at a different rate than data is collected. When `Log Rate` is `1`, data is logged every data collection interval. When `Log Rate` is `2`, every other value gets saved, etc.

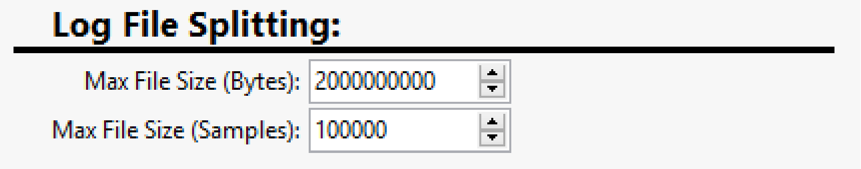


Log Rate:

Log Rate: 1

Log Data every x DAQ iterations, 1 = every iteration.

Log File Splitting:



Log File Splitting:

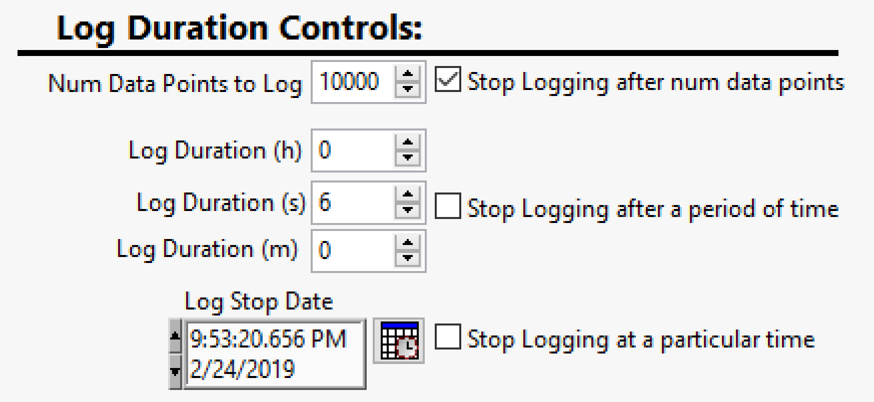
Max File Size (Bytes): 2000000000

Max File Size (Samples): 100000

Maximum file sizes can be set using either of the following two controls. When either of them is reached, a new file will be created.

- **Max File Size (Bytes):** Starts a new file after the current data file becomes larger than "x" number of Bytes in size.
- **Max File Size (Samples):** Starts a new file after the current data file has "x" number of samples saved to it.

Log Duration:



Log Duration Controls:

Num Data Points to Log 10000 ☑ Stop Logging after num data points

Log Duration (h) 0

Log Duration (s) 6 □ Stop Logging after a period of time

Log Duration (m) 0

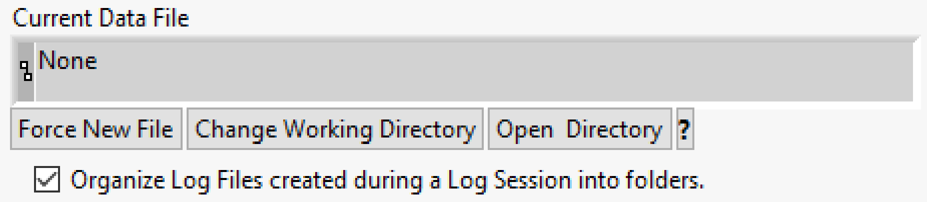
Log Stop Date 9:53:20.656 PM 2/24/2019 □ Stop Logging at a particular time

The log duration controls allow LJLogM to stop logging after either:

- A number of samples
- A duration (hours, minutes, seconds)
- A specific time.

Current Data File: When data is being logged to a file, this indicator displays the current file path and name of the file being logged to. There are buttons here that let users force a new file to be created, re-configure

LJLogM to change directories where data is being saved, and a button that lets users open Windows Explorer at the directory where data is being logged.

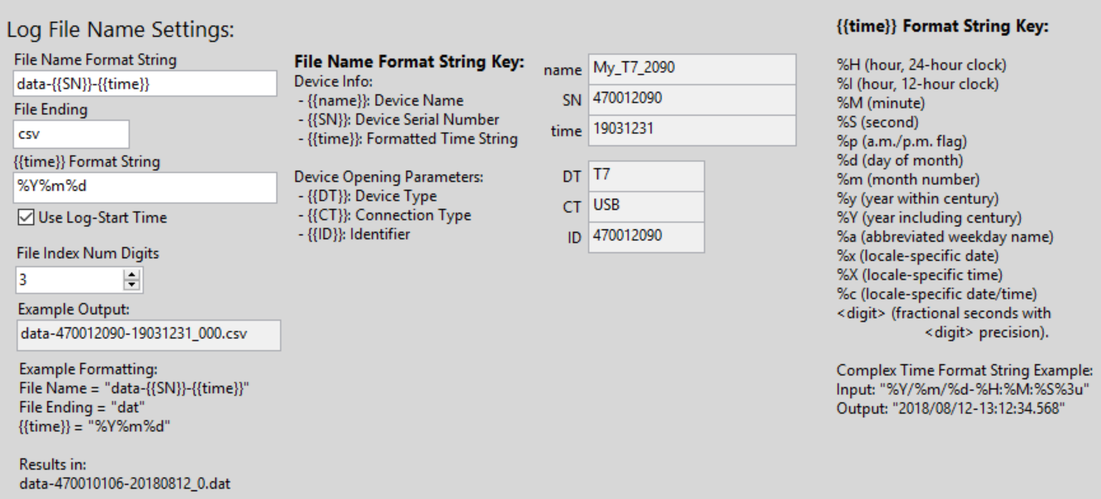


Current Data File

None

☑ Organize Log Files created during a Log Session into folders.

Log File Naming Settings



Log File Name Settings:

File Name Format String: data-{{SN}}-{{time}}

File Ending: csv

{{time}} Format String: %Y%m%d

☑ Use Log-Start Time

File Index Num Digits: 3

Example Output: data-470012090-19031231_000.csv

Example Formatting:
File Name = "data-{{SN}}-{{time}}"
File Ending = "dat"
{{time}} = "%Y%m%d"

Results in:
data-470010106-20180812_0.dat

File Name Format String Key:
Device Info:
- {{name}}: Device Name
- {{SN}}: Device Serial Number
- {{time}}: Formatted Time String

Device Opening Parameters:
- {{DT}}: Device Type
- {{CT}}: Connection Type
- {{ID}}: Identifier

name: My_T7_2090
SN: 470012090
time: 19031231

DT: T7
CT: USB
ID: 470012090

{{time}} Format String Key:

%H (hour, 24-hour clock)
%I (hour, 12-hour clock)
%M (minute)
%S (second)
%p (a.m./p.m. flag)
%d (day of month)
%m (month number)
%y (year within century)
%Y (year including century)
%a (abbreviated weekday name)
%x (locale-specific date)
%X (locale-specific time)
%c (locale-specific date/time)
<digit> (fractional seconds with <digit> precision).

Complex Time Format String Example:
Input: "%Y/%m/%d-%H:%M:%S%3u"
Output: "2018/08/12-13:12:34.568"

The file that gets generated during a logging session by LJLogM can be customized by defining a `File Name Format String` which can include bracketed strings/keys that get replaced by the applicable referenced device settings or a time string. At the end of the file, a file counter string `_0` is added which gets incremented each time a new file is created due to its size getting to large (`_1`, `_2`, etc).

To create the list of files:

`data_0.dat`
`data_1.dat`

Use the settings:

`File Name Format String: "data"`
`File Ending: "dat"`
`{{time}} Format String: N/A`

To create the list of files:

`data-20180812_0.csv`
`data-20180812_1.csv`
`data-20180812_2.csv`

Use the settings:

`File Name Format String: "data-{{time}}"`

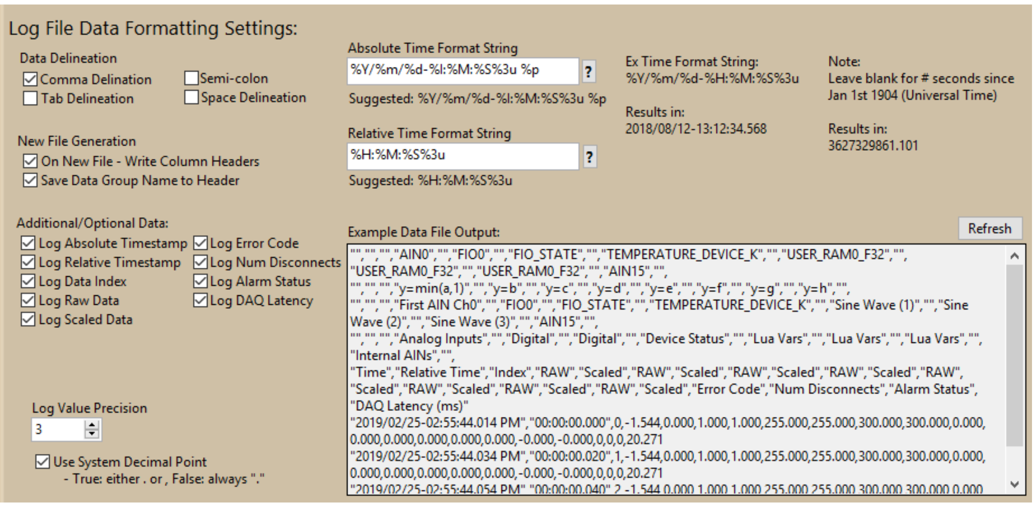
`File Ending: "csv"`

`{{time}} Format String: "%Y%m%d"`

Known Issues with Log File Naming

The `{{time}}` format string may not contain a colon (`:`).

Log File Data Formatting Settings



Log File Data Formatting Settings:

Data Delineation
☑ Comma Delination
□ Semi-colon
□ Tab Delineation
□ Space Delineation

New File Generation
☑ On New File - Write Column Headers
☑ Save Data Group Name to Header

Additional/Optional Data:
☑ Log Absolute Timestamp
☑ Log Error Code
☑ Log Relative Timestamp
☑ Log Num Disconnects
☑ Log Data Index
☑ Log Alarm Status
☑ Log Raw Data
☑ Log DAQ Latency
☑ Log Scaled Data

Log Value Precision: 3

☑ Use System Decimal Point
- True: either . or , False: always "."

Absolute Time Format String: %Y/%m/%d-%I:%M:%S%3u %p
Suggested: %Y/%m/%d-%I:%M:%S%3u %p

Relative Time Format String: %H:%M:%S%3u
Suggested: %H:%M:%S%3u

Ex Time Format String:
%Y/%m/%d-%H:%M:%S%3u

Results in:
2018/08/12-13:12:34.568

Note:
Leave blank for # seconds since
Jan 1st 1904 (Universal Time)

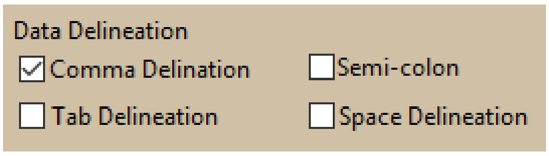
Results in:
3627329861.101

Example Data File Output:

```
"","","","AIN0","","FIO0","","FIO_STATE","","TEMPERATURE_DEVICE_K","","USER_RAM0_F32","",
"USER_RAM0_F32","","USER_RAM0_F32","","AIN15","",
"","","","y=min(a,1)","","y=b","","y=c","","y=d","","y=e","","y=f","","y=g","","y=h","",
"","","","First AIN Ch0","","FIO0","","FIO_STATE","","TEMPERATURE_DEVICE_K","","Sine Wave (1)","","Sine
Wave (2)","","Sine Wave (3)","","AIN15","",
"","","","Analog Inputs","","Digital","","Digital","","Device Status","","Lua Vars","","Lua Vars","","Lua Vars","",
"Internal AINs","",
"Time","Relative Time","Index","RAW","Scaled","RAW","Scaled","RAW","Scaled","RAW","Scaled","RAW",
"Scaled","RAW","Scaled","RAW","Scaled","RAW","Scaled","Error Code","Num Disconnects","Alarm Status",
"DAQ Latency (ms)"
"2019/02/25-02:55:44.014 PM","00:00:00.000",0,-1.544,0.000,1.000,1.000,255.000,255.000,300.000,300.000,0.000,
0.000,0.000,0.000,0.000,0.000,0.000,-0.000,-0.000,0,0,0,20.271
"2019/02/25-02:55:44.034 PM","00:00:00.020",1,-1.544,0.000,1.000,1.000,255.000,255.000,300.000,300.000,0.000,
0.000,0.000,0.000,0.000,0.000,0.000,-0.000,-0.000,0,0,0,20.271
"2019/02/25-02:55:44.054 PM","00:00:00.040",2,-1.544,0.000,1.000,1.000,255.000,255.000,300.000,300.000,0.000
```

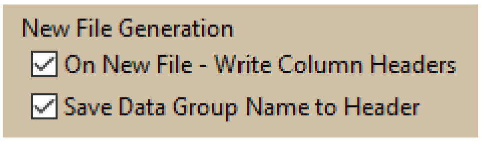
LJLogM also has a variety of settings to customize the data that gets logged to a file. As settings get enabled and disabled, a file-preview window is updated to reflect what data is going to be saved to the file. There are these options to customize:

Data Delineation: Data is saved into a file with standard CSV (comma-separated values) formatting. Rows are terminated with a newline (`\r\n`), and columns are separated by comma (`,`), tab (`\t`), semi-colon (`;`) , and/or space (`" "`).



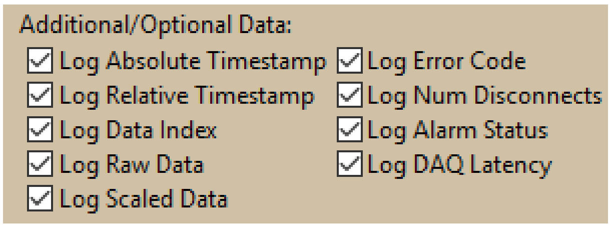
Data Delineation
☑ Comma Delination
□ Semi-colon
□ Tab Delineation
□ Space Delineation

Spreadsheet Header: A header section can be optionally included or excluded from the generated file to save space and make it easier to parse data with post-processing tools. The `On New File` option lets users choose whether or not the header data is added. The `Save Data Group Name` is an option to include or exclude what data group each of the values were associated to.



New File Generation
☑ On New File - Write Column Headers
☑ Save Data Group Name to Header

Additional/Optional Data:

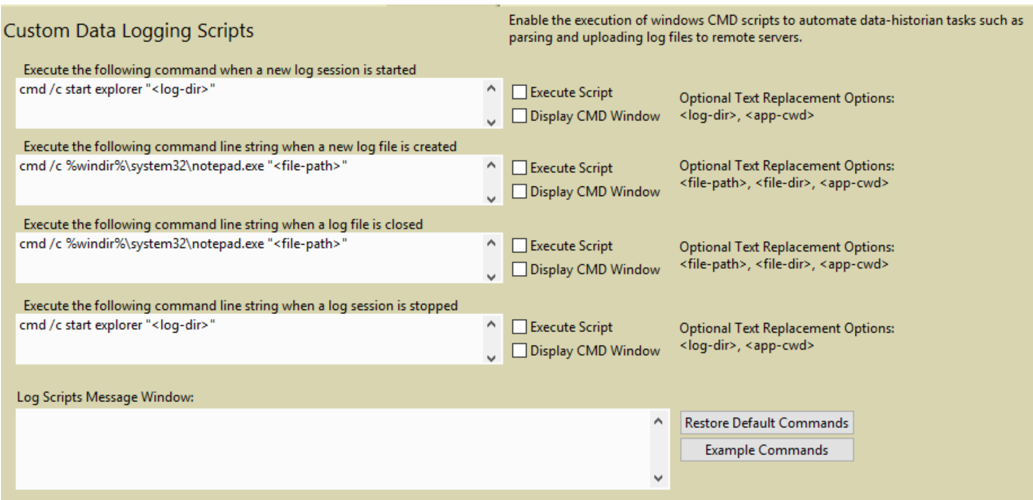


Additional/Optional Data:

☑ Log Absolute Timestamp
☑ Log Relative Timestamp
☑ Log Data Index
☑ Log Raw Data
☑ Log Scaled Data
☑ Log Error Code
☑ Log Num Disconnects
☑ Log Alarm Status
☑ Log DAQ Latency

- Log Absolute Timestamp: When enabled, a timestamp is logged for each interval that reflects the "computer time". The timestamp can be customized by editing the `Absolute Time Format String` field. For more information about what special characters can be used, see the timestamp section.
- Log Relative Timestamp: When enabled, a timestamp is logged for each interval that reflects how long it has been since the log started. The timestamp can be customized by editing the `Relative Time Format Strin` " field. For more information about what special characters can be used, see the timestamp section.
- Log Data Index: When enabled, an "index" value will be saved for each value that gets logged to the file. This can make some spreadsheet calculations easier and allows users to keep better track of data when multiple files are created.
- Log Raw Data: When enabled, the raw value for each register is logged to the file.
- Log Scaled Data: When enabled, the scaled value for each data point is logged to the file.
- Log Error Code: When enabled, the LJM error code returned by the `LJM_eReadNames` function is logged.
- Log Num Disconnects: LJLogM keeps track of how many times a device has become disconnected and re-connected. When enabled, LJLogM will log this value.
- Log Alarm Status: When enabled, LJLogM will save a boolean `Yes` or `No` value (`1` or `0`) to indicate whether or not there was an overall alarm condition triggered due to one of the scaled values being to high or low.
- Log DAQ Latency: When enabled, LJLogM will save the time between log intervals to the file which makes some post-processing calculations easier.

Custom Scripts



Custom Data Logging Scripts

Enable the execution of windows CMD scripts to automate data-historian tasks such as parsing and uploading log files to remote servers.

Execute the following command when a new log session is started

```
cmd /c start explorer "<log-dir>"
```

□ Execute Script
□ Display CMD Window

Optional Text Replacement Options:
<log-dir>, <app-cwd>

Execute the following command line string when a new log file is created

```
cmd /c %windir%\system32\notepad.exe "<file-path>"
```

□ Execute Script
□ Display CMD Window

Optional Text Replacement Options:
<file-path>, <file-dir>, <app-cwd>

Execute the following command line string when a log file is closed

```
cmd /c %windir%\system32\notepad.exe "<file-path>"
```

□ Execute Script
□ Display CMD Window

Optional Text Replacement Options:
<file-path>, <file-dir>, <app-cwd>

Execute the following command line string when a log session is stopped

```
cmd /c start explorer "<log-dir>"
```

□ Execute Script
□ Display CMD Window

Optional Text Replacement Options:
<log-dir>, <app-cwd>

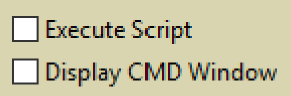
Log Scripts Message Window:

Configure LJLogM to run applications with configurable arguments at specific times during a logging session to automate repetitive tasks that are common when logging data. Scripts can be executed:

- At the start of a logging session.
- When a new file is generated and is starting to be written to.
- When a file finishes being written to. This happens when LJLogM is switching to a new file during a logging session or when a logging session finishes.
- At the end of a logging session.

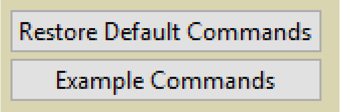
For some ideas on what scripts to write and how this feature can be used, see the custom scripts sub-section.

Script Execution Options: Scripts can be individually enabled and can optionally be configured to display a CMD window when being launched have them launch silently.

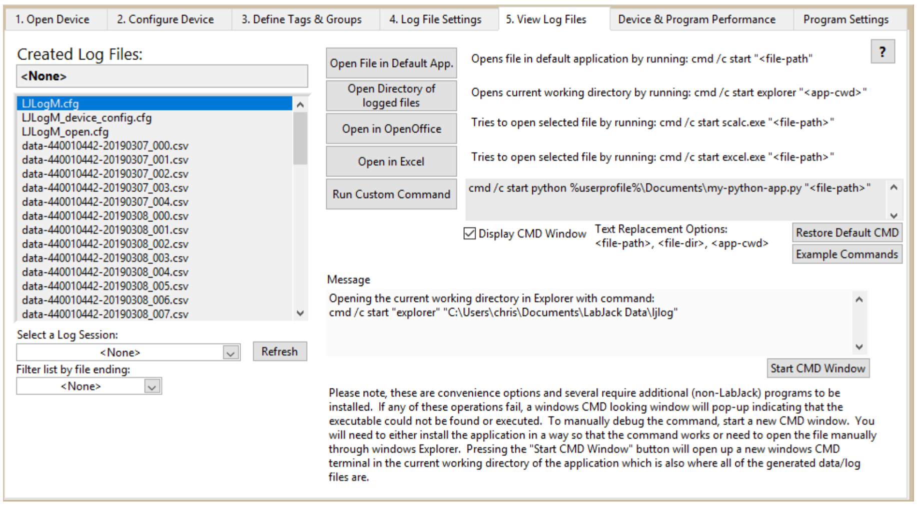


□ Execute Script
□ Display CMD Window

Restore Defaults & see Examples: If there is a concern that a script input field isn't working properly restore the scripts to default "safe" options. Also, view a few script ideas.



View Logged Data



Created Log Files:

<None>

LJLogM.cfg
LJLogM_device_config.cfg
LJLogM_open.cfg
data-440010442-20190307_000.csv
data-440010442-20190307_001.csv
data-440010442-20190307_002.csv
data-440010442-20190307_003.csv
data-440010442-20190307_004.csv
data-440010442-20190308_000.csv
data-440010442-20190308_001.csv
data-440010442-20190308_002.csv
data-440010442-20190308_003.csv
data-440010442-20190308_004.csv
data-440010442-20190308_005.csv
data-440010442-20190308_006.csv
data-440010442-20190308_007.csv

Select a Log Session: <None>

Filter list by file ending: <None>

Opens file in default application by running: cmd /c start "<file-path>"

Opens current working directory by running: cmd /c start explorer "<app-cwd>"

Tries to open selected file by running: cmd /c start scalc.exe "<file-path>"

Tries to open selected file by running: cmd /c start excel.exe "<file-path>"

```
cmd /c start python %userprofile%\Documents\my-python-app.py "<file-path>"
```

☑ Display CMD Window

Text Replacement Options:
<file-path>, <file-dir>, <app-cwd>

Message

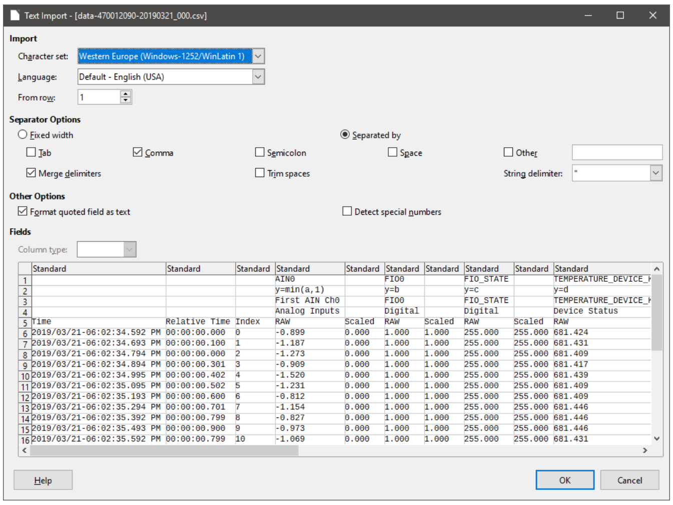
```
Opening the current working directory in Explorer with command:
cmd /c start "explorer" "C:\Users\chris\Documents\LabJack Data\ljlog"
```

Please note, these are convenience options and several require additional (non-LabJack) programs to be installed. If any of these operations fail, a windows CMD looking window will pop-up indicating that the executable could not be found or executed. To manually debug the command, start a new CMD window. You will need to either install the application in a way so that the command works or need to open the file manually through windows Explorer. Pressing the "Start CMD Window" button will open up a new windows CMD terminal in the current working directory of the application which is also where all of the generated data/log files are.

After generating data log files, LJLogM can help users find and view the generated files by running some Windows command prompt commands. The list of log files that have been created can be filtered by the drop-down menu. After selecting a file, users have the option to:

- Open the file in its default application. This is set in Windows settings.
- Open the directory of where all of the logged files have been created.
- (If installed) Open the file in OpenOffice.
- (If installed) Open the file in Excel.

Importing data into a spreadsheet application is typically a pain-free process. OpenOffice, LibreOffice, and Excel all have a data importing wizard that pops up and looks something like the following:



Text Import - [data-470012090-20190321_000.csv]

Import

Character set: Western Europe (Windows-1252/WinLatin 1)

Language: Default - English (USA)

From row: 1

Separator Options

○ Fixed width ● Separated by

☐ Tab ☑ Comma ☐ Semicolon ☐ Space ☐ Other

☑ Merge delimiters ☐ Trim spaces String delimiter: "

Other Options

☑ Format quoted field as text ☐ Detect special numbers

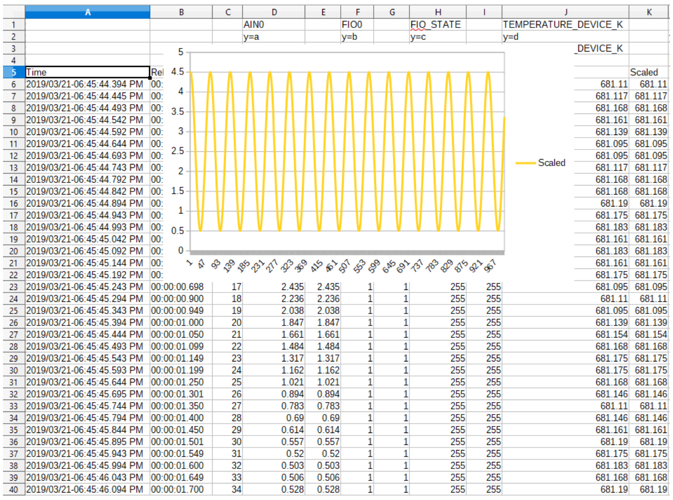
Fields

Column type:

	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard	Standard
1				AIN0		FIO0		FIO_STATE		TEMPERATURE_DEVICE_K
2				y=min(a,1)		y=b		y=c		y=d
3				First AIN Ch0		FIO0		FIO_STATE		TEMPERATURE_DEVICE_K
4				Analog Inputs		Digital		Digital		Device Status
5	Time	Relative Time	Index	RAW	Scaled	RAW	Scaled	RAW	Scaled	RAW
6	2019/03/21-06:02:34.592 PM	00:00:00.000	0	-0.899	0.000	1.000	1.000	255.000	255.000	681.424
7	2019/03/21-06:02:34.693 PM	00:00:00.100	1	-1.187	0.000	1.000	1.000	255.000	255.000	681.431
8	2019/03/21-06:02:34.794 PM	00:00:00.000	2	-1.273	0.000	1.000	1.000	255.000	255.000	681.409
9	2019/03/21-06:02:34.894 PM	00:00:00.301	3	-0.909	0.000	1.000	1.000	255.000	255.000	681.417
10	2019/03/21-06:02:34.995 PM	00:00:00.402	4	-1.520	0.000	1.000	1.000	255.000	255.000	681.439
11	2019/03/21-06:02:35.095 PM	00:00:00.502	5	-1.231	0.000	1.000	1.000	255.000	255.000	681.409
12	2019/03/21-06:02:35.193 PM	00:00:00.600	6	-0.812	0.000	1.000	1.000	255.000	255.000	681.409
13	2019/03/21-06:02:35.294 PM	00:00:00.701	7	-1.154	0.000	1.000	1.000	255.000	255.000	681.446
14	2019/03/21-06:02:35.392 PM	00:00:00.799	8	-0.827	0.000	1.000	1.000	255.000	255.000	681.446
15	2019/03/21-06:02:35.493 PM	00:00:00.900	9	-0.973	0.000	1.000	1.000	255.000	255.000	681.446
16	2019/03/21-06:02:35.592 PM	00:00:00.799	10	-1.069	0.000	1.000	1.000	255.000	255.000	681.431

LJLogM has configurable data delineation settings so select the appropriate options for how data was saved into the file. String data in LJLogM is saved as double-quoted text to allow strings to contain separator text options so make sure that option is selected as well and select the " string delimiter.

Once data has been imported, it is relatively easy to graph the data and perform additional calculations:



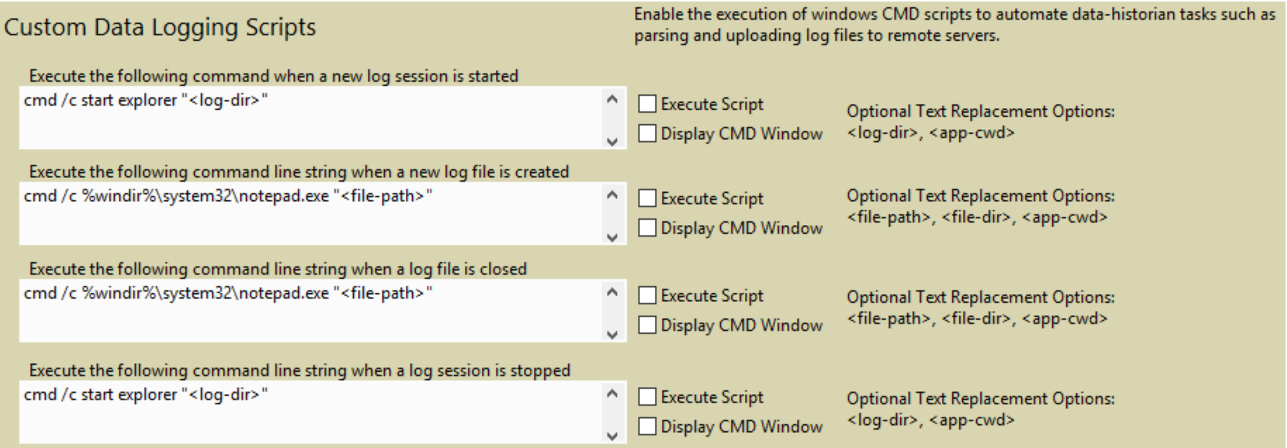
	A	B	C	D	E	F	G	H	I	J	K
1				AIN0		FIO0		FIO_STATE		TEMPERATURE_DEVICE_K	
2				y=a		y=b		y=c		y=d	
3										_DEVICE_K	
4											
5	Time	Rel									Scaled
6	2019/03/21-06:45:44.394 PM	00:								681.11	681.11
7	2019/03/21-06:45:44.445 PM	00:								681.117	681.117
8	2019/03/21-06:45:44.493 PM	00:								681.168	681.168
9	2019/03/21-06:45:44.542 PM	00:								681.161	681.161
10	2019/03/21-06:45:44.592 PM	00:								681.139	681.139
11	2019/03/21-06:45:44.644 PM	00:								681.095	681.095
12	2019/03/21-06:45:44.693 PM	00:								681.095	681.095
13	2019/03/21-06:45:44.743 PM	00:								681.117	681.117
14	2019/03/21-06:45:44.792 PM	00:								681.168	681.168
15	2019/03/21-06:45:44.842 PM	00:								681.168	681.168
16	2019/03/21-06:45:44.894 PM	00:								681.19	681.19
17	2019/03/21-06:45:44.943 PM	00:								681.175	681.175
18	2019/03/21-06:45:44.993 PM	00:								681.183	681.183
19	2019/03/21-06:45:45.042 PM	00:								681.161	681.161
20	2019/03/21-06:45:45.092 PM	00:								681.183	681.183
21	2019/03/21-06:45:45.144 PM	00:								681.161	681.161
22	2019/03/21-06:45:45.192 PM	00:								681.175	681.175
23	2019/03/21-06:45:45.243 PM	00:00:00.698	17	2.435	2.435	1	1	255	255	681.095	681.095
24	2019/03/21-06:45:45.294 PM	00:00:00.900	18	2.236	2.236	1	1	255	255	681.11	681.11
25	2019/03/21-06:45:45.343 PM	00:00:00.949	19	2.038	2.038	1	1	255	255	681.095	681.095
26	2019/03/21-06:45:45.394 PM	00:00:01.000	20	1.847	1.847	1	1	255	255	681.139	681.139
27	2019/03/21-06:45:45.444 PM	00:00:01.050	21	1.661	1.661	1	1	255	255	681.154	681.154
28	2019/03/21-06:45:45.493 PM	00:00:01.099	22	1.484	1.484	1	1	255	255	681.168	681.168
29	2019/03/21-06:45:45.543 PM	00:00:01.149	23	1.317	1.317	1	1	255	255	681.175	681.175
30	2019/03/21-06:45:45.593 PM	00:00:01.199	24	1.162	1.162	1	1	255	255	681.175	681.175
31	2019/03/21-06:45:45.644 PM	00:00:01.250	25	1.021	1.021	1	1	255	255	681.168	681.168
32	2019/03/21-06:45:45.695 PM	00:00:01.301	26	0.894	0.894	1	1	255	255	681.146	681.146
33	2019/03/21-06:45:45.744 PM	00:00:01.350	27	0.783	0.783	1	1	255	255	681.11	681.11
34	2019/03/21-06:45:45.794 PM	00:00:01.400	28	0.69	0.69	1	1	255	255	681.146	681.146
35	2019/03/21-06:45:45.844 PM	00:00:01.450	29	0.614	0.614	1	1	255	255	681.161	681.161
36	2019/03/21-06:45:45.895 PM	00:00:01.501	30	0.557	0.557	1	1	255	255	681.19	681.19
37	2019/03/21-06:45:45.943 PM	00:00:01.549	31	0.52	0.52	1	1	255	255	681.175	681.175
38	2019/03/21-06:45:45.994 PM	00:00:01.600	32	0.503	0.503	1	1	255	255	681.183	681.183
39	2019/03/21-06:45:46.043 PM	00:00:01.649	33	0.506	0.506	1	1	255	255	681.168	681.168
40	2019/03/21-06:45:46.094 PM	00:00:01.700	34	0.528	0.528	1	1	255	255	681.19	681.19

Custom Scripts

The possibilities for what to do in a custom script are almost endless. Some options:

- After a log file has been created, it can be copied to alternate locations using the `cp` command or copied to remote servers using the `scp` command.
- Data can also be uploaded to public cloud storage locations such as AWS, Dropbox, or Google Drive by calling custom python programs.

Data Logging Scripts



Custom Data Logging Scripts

Enable the execution of windows CMD scripts to automate data-historian tasks such as parsing and uploading log files to remote servers.

Execute the following command when a new log session is started

```
cmd /c start explorer "<log-dir>"
```

□ Execute Script
□ Display CMD Window

Optional Text Replacement Options:
<log-dir>, <app-cwd>

Execute the following command line string when a new log file is created

```
cmd /c %windir%\system32\notepad.exe "<file-path>"
```

□ Execute Script
□ Display CMD Window

Optional Text Replacement Options:
<file-path>, <file-dir>, <app-cwd>

Execute the following command line string when a log file is closed

```
cmd /c %windir%\system32\notepad.exe "<file-path>"
```

□ Execute Script
□ Display CMD Window

Optional Text Replacement Options:
<file-path>, <file-dir>, <app-cwd>

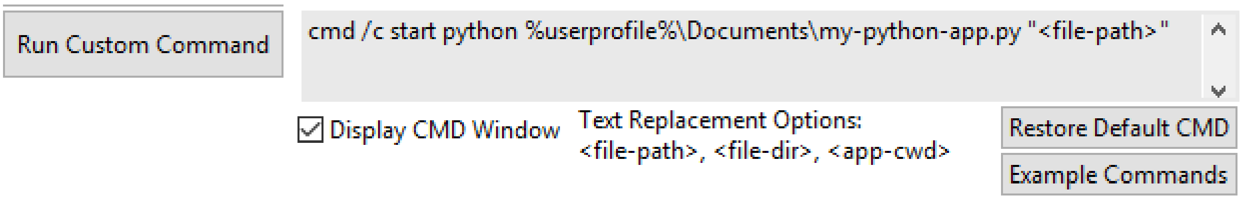
Execute the following command line string when a log session is stopped

```
cmd /c start explorer "<log-dir>"
```

□ Execute Script
□ Display CMD Window

Optional Text Replacement Options:
<log-dir>, <app-cwd>

Data Viewing Scripts



```
cmd /c start python %userprofile%\Documents\my-python-app.py "<file-path>"
```

☑ Display CMD Window

Text Replacement Options:
<file-path>, <file-dir>, <app-cwd>

Windows Batch Scripts & Commands

There are a plethora of online resources tailored to scripting for Windows machines. Here are a few good resources:

- Batch Files & Batch Commands: (DOS) Commands and their usage in batch files
- Batch Script Commands: Hosted by Tutorials Point.
- Windows Commands: Documentation by Microsoft.

Before running any scripts, it is recommend to look into the "CMD" command. At the start of each defined script, we recommend adding either:

```
// Terminate the CMD shell after executing the specified command.
cmd /c [your command.........]
```

```
// Leave the CMD window open after executing the specified command.
cmd /k [your command.........]
```

SCP & Linux commands on Windows

After installing a variety of Linux command line utilities onto a Windows machine, users can configure LJLogM to perform a lot of powerful features. Internally, we use and suggest others to download the tools provided by Cygwin and GnuWin32. Once installed and properly configured, users can copy files created by LJLogM to a remote host with SCP by following the following syntax:

```
scp "<file-path>" username@host:/remote/directory
```

There are many examples for how to use SCP online, here is a link to one: Linux - How to Securely Copy Files using SCP examples as well as a Stack Exchange topic: Use scp to transfer a file from local directory X to remote directory Y.

Going Further, Python and Node.js

If there aren't any existing Windows batch commands or Linux bash commands that accomplish the task at hand, look into installing tools like Python or Node.js to greatly enhance what can be done with LJLogM's scripting capabilities. Both Python and Node.js have a plethora of online libraries that can be leveraged to perform tasks that need to be automated. After installing python or installing node.js head back to Google and start searching for cool places to upload files. There are more tutorials out there than you can shake a stick at.

- How to upload files automatically to Google Drive with Python

Scripting with MATLAB

If the logged data needs to be analyzed and a variety of complex calculations need to be performed, consider writing a MATLAB program that reads in a log file and have it perform calculations either during an experiment (after each log file is created) or when the log session completes.

- MathWorks Documentation: Start MATLAB program from Windows system prompt
- Stack Overflow: Matlab: Running a m-file from command-line

Quickly read a `.csv` file using the "readtable" function and generate XY Plots.

Additional Potential Integrations

- How to Write Points from CSV to InfluxDB
- Getting started with Telegraf
- Telegraf Downloads

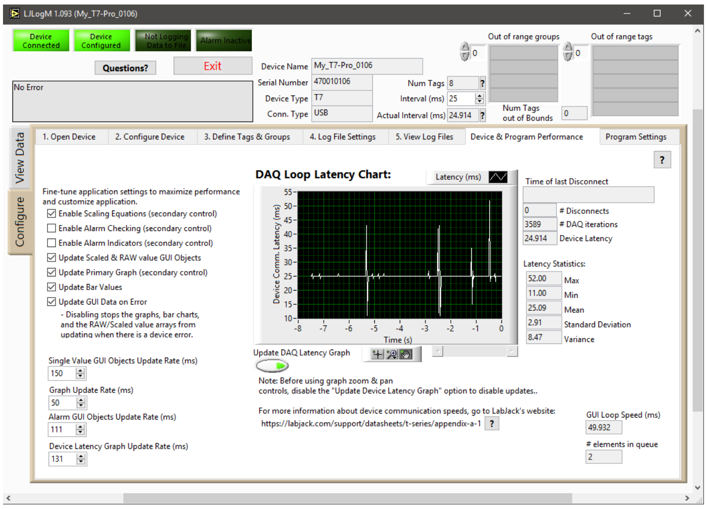
timestamps

Both absolute (current computer time) relative (time compared to a starting point) time stamps that get saved to a log file can be customized using the following special characters.

Relative Time Format Codes:
Weeks: %W
Days: %D
Hours: %H
Minutes: %M
Seconds: %S
Fractional seconds with <digit> %<digit>u
A single percent character: %

Absolute Time Format Codes:
Abbreviated weekday name: %a
Weekday name: %A
Abbreviated month name: %b
Month name: %B
Default date and time %c
Day of month (01-31): %d
Hour (24-hour clock) (00-23): %H
Hour (12-hour clock) (01-12): %I
Day of year (001-366): %j
Month number (01-12): %m
Minute (0-59): %M
AM or PM flag: %p
Seconds (00-59): %S
Fractional seconds with <digit> precision: %<digit>u
Week number (0-53) with Sunday as the first day of week 1: %U
Weekday number (0-6): %w
Week number (00-53) with Monday as the first day of week 1: %W
Locale date: %x
Locale time: %X
Year within century (00-99): %y
Year including century: %Y
Time zone: %Z

Device and Program Performance



No Error

Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 25
Actual Interval (ms): 24.914
Out of range groups: 0
Out of range tags: 0
Num Tags out of Bounds: 0

Fine-tune application settings to maximize performance and customize application.

- ☑ Enable Scaling Equations (secondary control)
- ☐ Enable Alarm Checking (secondary control)
- ☐ Enable Alarm Indicators (secondary control)
- ☑ Update Scaled & RAW value GUI Objects
- ☑ Update Primary Graph (secondary control)
- ☑ Update Bar Values
- ☑ Update GUI Data on Error
  - Disabling stops the graphs, bar charts, and the RAW/Scaled value arrays from updating when there is a device error.

Single Value GUI Objects Update Rate (ms): 150
Graph Update Rate (ms): 50
Alarm GUI Objects Update Rate (ms): 111
Device Latency Graph Update Rate (ms): 131

Time of last Disconnect:
0 # Disconnects
3589 # DAQ iterations
24.914 Device Latency

Latency Statistics:

52.00	Max
11.00	Min
25.09	Mean
2.91	Standard Deviation
8.47	Variance

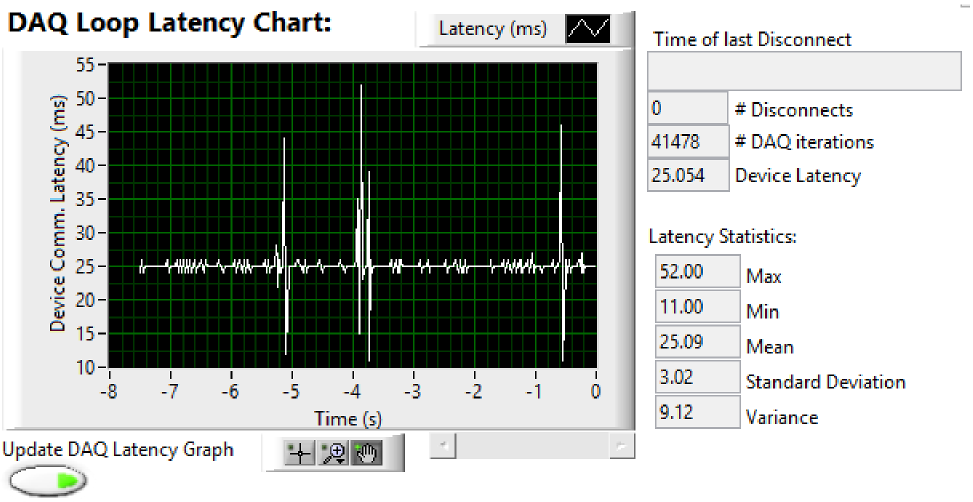
Note: Before using graph zoom & pan controls, disable the "Update Device Latency Graph" option to disable updates..

For more information about device communication speeds, go to LabJack's website: https://labjack.com/support/datasheets/t-series/appendix-a-1 ?

GUI Loop Speed (ms): 49.932
# elements in queue: 2

When collecting data from a LabJack at high data rates or trying to collect data at tightly controlled intervals, there are several factors to consider. The active connection type, how many channels are being logged, what type of data is being collected, and what calculations are being ran afterwards are all important. This feature of LJLogM provides a method to detect, diagnose, and understand communication speed issues and limitations.

This section of LJLogM directly correlates to section 3.0 Communication and section A-1 Data Rates of the T-Series datasheet. It is important to understand that the USB, Ethernet, and WiFi communication interfaces of T-Series devices perform differently, where some are inherently less reliable than others. WiFi connections, for example, can easily become temporarily disconnected due to excess network traffic or 2.4Ghz electrical noise in the surrounding environment. To assist with understanding communication interface performance LJLogM includes a chart that keeps track of how regular data is being polled from the connected device and calculates a few statistical metrics.

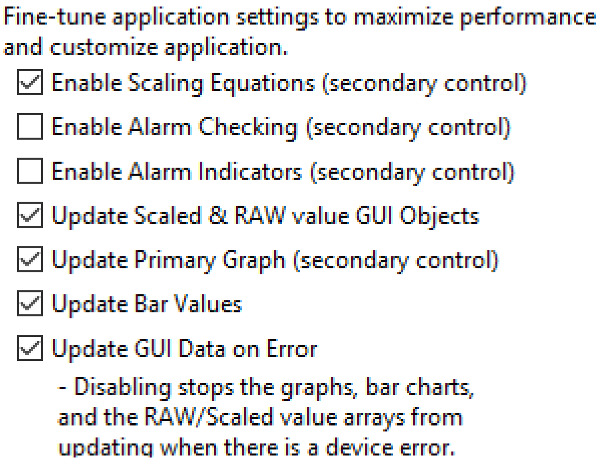


Time of last Disconnect:
0 # Disconnects
41478 # DAQ iterations
25.054 Device Latency

Latency Statistics:

52.00	Max
11.00	Min
25.09	Mean
3.02	Standard Deviation
9.12	Variance

In addition to the communication medium's performance characteristics, users have to remember that Windows is not a real time operating system. For most customers, LJLogM won't be the only application running on the used computer and the CPU or associated memory buses may become seemingly "randomly" busy which can cause additional latency spikes. When trying to collect data as fast as possible (1-10ms intervals), these limitations are sometimes encountered. Therefore, LJLogM includes options to selectively disable a variety of calculations that are performed on the incoming data as well as controls that adjust how often data is rendered to the various graphs and display indicators.



Fine-tune application settings to maximize performance and customize application.

☑ Enable Scaling Equations (secondary control)

□ Enable Alarm Checking (secondary control)

□ Enable Alarm Indicators (secondary control)

☑ Update Scaled & RAW value GUI Objects

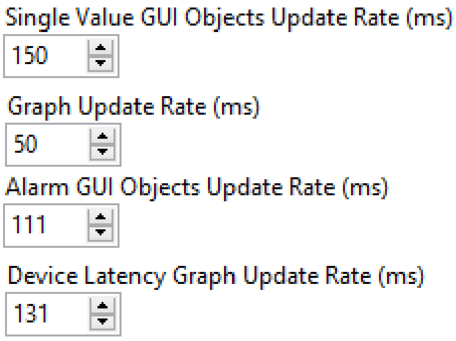
☑ Update Primary Graph (secondary control)

☑ Update Bar Values

☑ Update GUI Data on Error

- Disabling stops the graphs, bar charts, and the RAW/Scaled value arrays from updating when there is a device error.

Users can enable or disable some calculations that are performed such as scaling equation and alarm checking from this tab. Users can also choose to disable the updating of indicators to increase data logging performance.



Single Value GUI Objects Update Rate (ms)
150

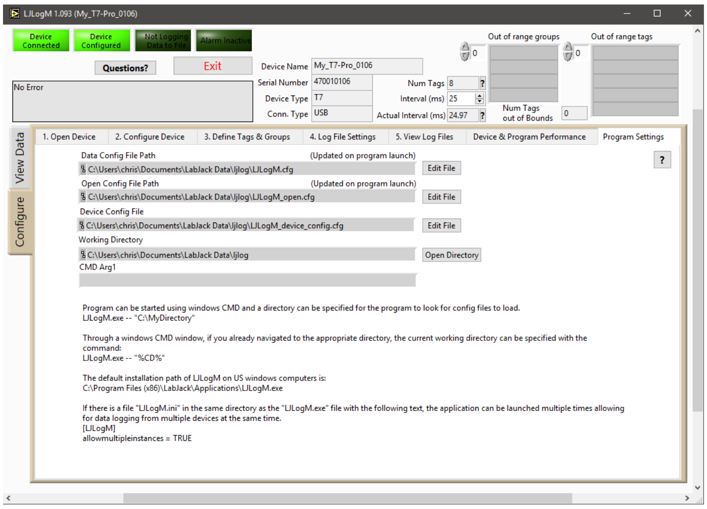
Graph Update Rate (ms)
50

Alarm GUI Objects Update Rate (ms)
111

Device Latency Graph Update Rate (ms)
131

Several of the GUI elements can also have their update rates be individually controlled. Understanding the affects of graph refresh rates is also important if users wish to later design their own logging application starting with the provided example code.

Advanced: Program Settings & Start-up



No Error

Device Name: My_T7-Pro_0106
Serial Number: 470010106
Device Type: T7
Conn. Type: USB
Num Tags: 8
Interval (ms): 25
Actual Interval (ms): 24.97
Num Tags out of Bounds: 0

Data Config File Path (Updated on program launch)
C:\Users\chris\Documents\LabJack Data\ljlog\LJLogM.cfg

Open Config File Path (Updated on program launch)
C:\Users\chris\Documents\LabJack Data\ljlog\LJLogM_open.cfg

Device Config File
C:\Users\chris\Documents\LabJack Data\ljlog\LJLogM_device_config.cfg

Working Directory
C:\Users\chris\Documents\LabJack Data\ljlog

CMD Arg1

Program can be started using windows CMD and a directory can be specified for the program to look for config files to load.
LJLogM.exe -- "C:\MyDirectory"

Through a windows CMD window, if you already navigated to the appropriate directory, the current working directory can be specified with the command:
LJLogM.exe -- "%CD%"

The default installation path of LJLogM on US windows computers is:
C:\Program Files (x86)\LabJack\Applications\LJLogM.exe

If there is a file "LJLogM.ini" in the same directory as the "LJLogM.exe" file with the following text, the application can be launched multiple times allowing for data logging from multiple devices at the same time.
[LJLogM]
allowmultipleinstances = TRUE

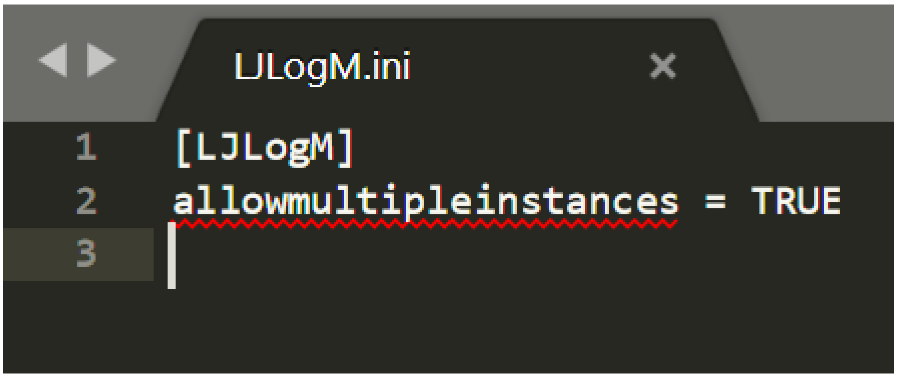
LJLogM is a highly configurable program where all of its features and settings can be configured by editing its `.cfg` files for the desired start up state. The program's configurations are saved whenever the program exits and loads during LJLogM launch.

Multiple Instances

Method A

Due to the application being highly configurable, some customers may wish to use it in some advanced ways. One example of this is to launch multiple instances of the application in order to collect data from several T-Series devices at the same time. Data collection can not be synchronized across instances of LJLogM but each instance can be configured to collect data at the same rate. The instances of LJLogM can be configured to save data to separate files and directories, and at high data rates this provides a fairly robust way to collect a large amount of data. This feature of LJLogM can be enabled by creating a `.ini` file in the same directory as the `.exe` (LabVIEW instructions):

LJLogM.exe	1/9/2019 12:05 PM	Application	3,900 KB
LJLogM.ini	10/18/2018 2:22 AM	Configuration settings	1 KB



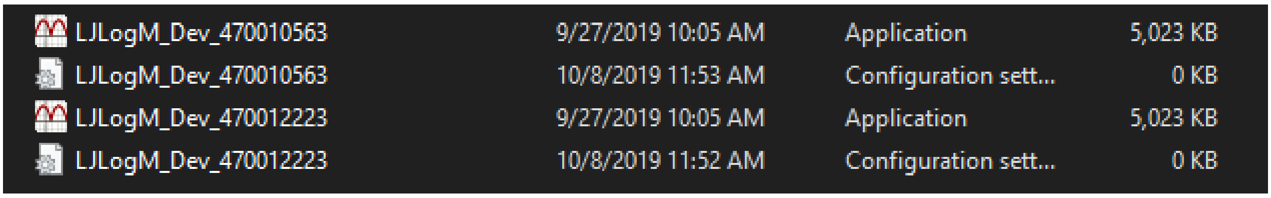
```
[LJLogM]
allowmultipleinstances = TRUE

```

Note: In order to run multiple instances of `LJLogM.exe`, the `.exe` can not be started from the default `C:\Program Files (x86)\LabJack\Applications` directory without administrative privileges. The best way to run multiple instances of the application is to run it from a different directory where the application can create a `.ini` file that can be properly edited.

Method B

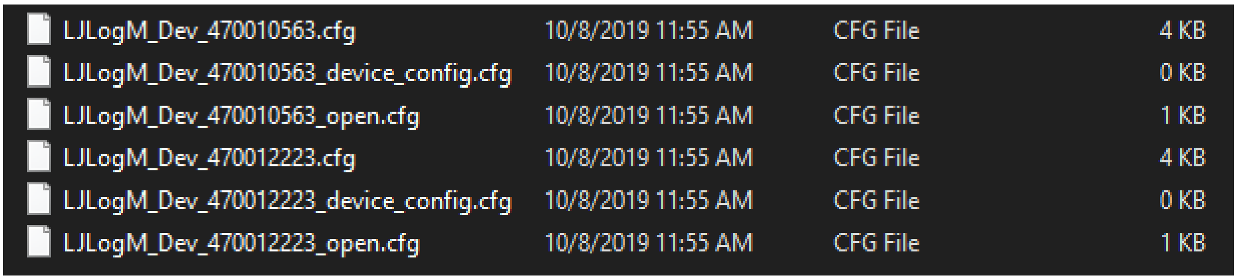
Another method for launching multiple instances of LJLogM in order to acquire data from multiple devices at the same time is to make multiple copies of the `LJLogM.exe` file. The first time each program copy is started an "active" directory will need to be configured so that the program's configurations will be saved correctly. This will allow each application to be started and automatically connect to a specified device with a specific configuration. In short,

1. Make two copies of the `LJLogM.exe`:



LJLogM_Dev_470010563	9/27/2019 10:05 AM	Application	5,023 KB
LJLogM_Dev_470010563	10/8/2019 11:53 AM	Configuration sett...	0 KB
LJLogM_Dev_470012223	9/27/2019 10:05 AM	Application	5,023 KB
LJLogM_Dev_470012223	10/8/2019 11:52 AM	Configuration sett...	0 KB

2. Make two copies of the LJLogM config files:



LJLogM_Dev_470010563.cfg	10/8/2019 11:55 AM	CFG File	4 KB
LJLogM_Dev_470010563_device_config.cfg	10/8/2019 11:55 AM	CFG File	0 KB
LJLogM_Dev_470010563_open.cfg	10/8/2019 11:55 AM	CFG File	1 KB
LJLogM_Dev_470012223.cfg	10/8/2019 11:55 AM	CFG File	4 KB
LJLogM_Dev_470012223_device_config.cfg	10/8/2019 11:55 AM	CFG File	0 KB
LJLogM_Dev_470012223_open.cfg	10/8/2019 11:55 AM	CFG File	1 KB

Command Line Arguments

LJLogM can be configured to start with a specific working directory by passing a single command line parameter like the following:

```
> LJLogM.exe -- "C:\MyDirectory"
```

or

```
> LJLogM.exe -- "%CD%"
```

or

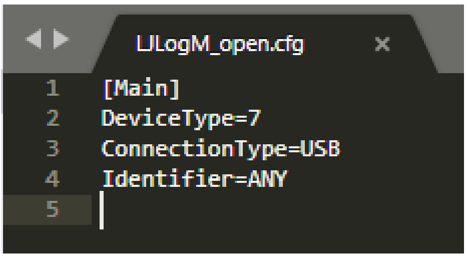
```
> "C:\Program Files (x86)\LabJack\Applications\LJLogM.exe" -- "%CD%"
```

After starting LJLogM with the command line argument, the "Program Settings" tab can be viewed again to see if the argument got properly parsed into the `CMD Arg1` field.

Default Working Directory

LJLogM has three primary configuration files, `LJLogM.cfg`, `LJLogM_open.cfg`, and `LJLogM_device_config.cfg` that can all be edited before the application starts. LJLogM also starts up and keeps track of a "Working Directory" which is where the `.cfg` files get generated and is also where log files are created. LJLogM saves this path to the Windows registry (viewable using Registry Editor) under: `HKEY_CURRENT_USER` → `Software` → `LabJack` → `LJLogM` → `workdir`.

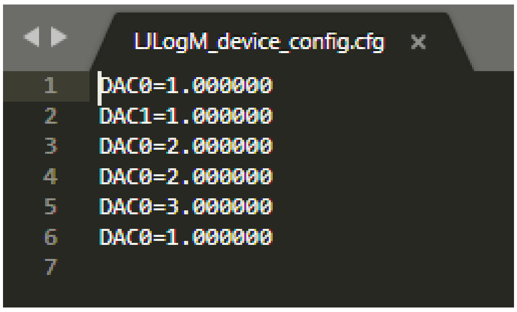
`LJLogM_open.cfg`



```
[Main]
DeviceType=7
ConnectionType=USB
Identifier=ANY

```

This is a fairly simple configuration file that defines the LJM open parameters.

`LJLogM_device_config.cfg`



```
DAC0=1.000000
DAC1=1.000000
DAC0=2.000000
DAC0=2.000000
DAC0=3.000000
DAC0=1.000000

```

This is another fairly simple configuration file that defines a list of registers to be written to the device upon connection.

`LJLogM.cfg`

This is a complex configuration file for LJLogM customization. If multiple logging sessions need to be utilized, it is best to use LJLogM to generate and save these configuration files. Once saved, move these files to their own individual directories, and then use the Command Prompt to launch LJLogM specifying the working directory argument, as mentioned in the Command Lines Arguments section. A few notes about the `LJLogM.cfg` file are as follows.

1. Values are organized into key-value pairs. `[key]=[value]`

2. Most of the [key]'s are easy to interpret as they relate directly to an associated GUI element, but not all.
3. Keys that require arrays of data have values separated by the string `#!#` and end with a `#!#`.
4. Boolean values are `FALSE` or `TRUE`.

LJLogM Beta Change Log

v1.095:

- Fixed the feature that allows data to be logged at a different rate than data is collected.
- Fixed the feature that allows users to generate a new log file.
- Added a feature that allows users to sort logged data into folders. This feature also saves a snap-shot of the LJLogM .cfg files for later use.
- Added a feature that allows users to open created log files by running a custom windows "CMD" prompt script.
- Added a feature enabling LJLogM to run CMD scripts when a log starts and stops as well as when log files are first created and when the file is closed.

v1.094:

- Added abilities to log both absolute (computer time) and relative (from log start) time stamp values to log files as well as the value index.
- Added ability to stop logging after: a number of samples, a period of time, or an absolute time.
- Added the ability to start a new file after a certain number of samples have been collected instead of just the # bytes.
- Added indicators to display log stats.
- Changed from "write to file" button to "Start" and "Stop" buttons.
- Added quotes around strings that get logged to .dat files so that spreadsheet applications can properly load/parse files.
- Added the ability to check for & notify users of new "release" and "beta" versions of LJLogM.
- Other small fixes.

v1.093:

Initial public release of LJLogM-Beta