Moku:Lab
LabVIEW API Migration Guide

Upgrading Moku:Lab to software version 3.0 unlocks a host of new features. When updating, API users must take extra steps to migrate their scripts to the new Moku API package. This migration guide outlines API changes, new features available in the version 3.0 update, and any backward compatibility limitations.

Table of Contents

Overview

Moku:Lab software version 3.0 is a major update that brings new firmware, user interfaces, and APIs to Moku:Lab hardware. The update brings Moku:Lab in line with Moku:Pro and Moku:Go, making it easy to share scripts across all Moku platforms. The update unlocks a host of new features to many of the existing instruments. It also adds two new features: Multi-instrument Mode and Moku Cloud Compile. There are some subtle behavioral differences as well outlined in the Backward compatibility section.

This update also affects the API architecture, and therefore the new API package will not be backwards compatible with existing API scripts. API users will need to port their scripts to the new Moku API package if they upgrade their Moku:Lab to version 3.0. API users with significant custom software development should carefully consider the level of effort required to port their existing code. Moku:Lab 1.9 is not recommended for new deployments and all customers are encouraged to upgrade. Eventually, Moku:Lab version 1.9 will lose support, in accordance with our End-of-Life Policy. If issues arise after upgrading, users will have the option to downgrade to software version 1.9.

This migration guide outlines advantages of updating and potential complications of updating to Moku:Lab version 3.0. It also outlines the process to upgrade the LabVIEW API and how to downgrade your Moku:Lab if necessary.

Version 3.0 new features

New features

Software version 3.0 brings Multi-Instrument Mode and Moku Cloud Compile to Moku:Lab for the first time, as well as many performance and usability upgrades across the suite of instruments.

Multi-instrument Mode

Multi-instrument Mode on Moku:Lab allows users to deploy two instruments simultaneously to create a custom test station. Each instrument has full access to the analog inputs and outputs, along with interconnections between instrument slots. The interconnections between instruments support high-speed, low-latency, real-time digital communication up to 2 Gb/s, so instruments can run independently or be connected to build advanced signal processing pipelines. Users can dynamically swap Instruments in and out without interrupting the other adjacent. Advanced users can also deploy their own custom algorithms in Multi-instrument Mode using Moku Cloud Compile.

Moku Cloud Compile

Moku Cloud Compile allows you to deploy custom digital signal processing (DSP) directly onto the Moku:Lab FPGA in Multi-instrument Mode. Write code using a web browser and compile it in the cloud; then use Moku Cloud Compile to deploy the bitstream to one or more target Moku devices.

Oscilloscope

- Deep memory mode – save up to 4M samples per channel at the full sampling rate (500 MSa/s)

Spectrum Analyzer

- Improved noise floor
- Logarithmic Vrms and Vpp scale
- Five new window functions (Bartlett, Hamming, Nuttall, Gaussian, Kaiser)

Phasemeter

- Users can now output frequency offset, phase, and amplitude as analog voltage signals
- Users can now add DC offset to output signals
- The phase-locked sine wave output can now be frequency multiplied up to 250x or divided down to 0.125x
- Improved bandwidth (1 Hz to 100 kHz)
- Advanced phase wrapping and auto-reset functions

Waveform Generator

- Noise output
- Pulse width modulation (PWM)

Lock-in Amplifier (LIA)

- Improved performance of low-frequency PLL locking
- The minimum PLL frequency has been decreased to 10 Hz
- The internal PLL signal can now be frequency multiplied up to 250x or divided down to 0.125x for use in demodulation
- 6-digit precision for phase values

Frequency Response Analyzer

- Increased maximum frequency from 120 MHz to 200 MHz
- Increased maximum sweep points from 512 to 8192
- New Dynamic Amplitude feature optimizes output signal automatically for best measurement dynamic range
- New In/In1 measurement mode
- Input saturation warnings
- The math channel now supports arbitrary complex-valued equations involving the channel signals, enabling new types of complex transfer function measurements
- Users can now measure input signals in dBVpp and dBVrms in addition to dBm
- The progress of the sweep is now displayed on the graph
- The frequency axis can now be locked to prevent accidental changes during a long sweep

Laser Lock Box

- Improved block diagram shows scan and modulation signal paths
- New locking stages feature allows users to customize their lock procedure

- Improved performance of low-frequency PLL locking
- 6-digit precision for phase values
- Improved performance of low-frequency PLL locking
- Minimum PLL frequency decreased to 10 Hz
- The PLL signal can now be frequency multiplied up to 250x or divided down to 0.125x for use in demodulation

Other

- Added support for the sinc function to the equation editor which can be used to generate custom waveforms in the Arbitrary Waveform Generator
- Convert binary LI files to CSV, MATLAB, or NumPy formats when downloading from the device

Upgraded API support

The new Moku API package provides enhanced functionality and stability. It will receive regular updates to improve performance and introduce new features.

Backward compatibility limitations

API

The new Moku LabVIEW API package is not backward compatible with the previous Moku:Lab LabVIEW API. Inputs and outputs are wholly different. If you have done extensive custom software development utilizing the Moku:Lab LabVIEW API, consider the impact of migrating all your software to be compatible with the new API.

While the Moku:Lab 1.9 LabVIEW API package will no longer receive updates, Liquid Instruments will continue to provide support for users who are unable to migrate to the new API package.

Find detailed examples for each instrument in the new Moku LabVIEW API package to serve as a baseline for converting prior API development to the new API package.

Regressions

RAM disk for data logging

Version 1.9 a had 512 MB filesystem in the device's RAM, which could be used to log data at high sampling rates. This is no longer available in version 3.0. To enable data logging, an SD card is required. This limits data logging speed to approximately 250 kSa/s for 1 channel and 125 kSa/s for two channels.

Data logging to CSV

Version 1.9 had the ability to save data directly to a CSV file while logging. This feature is not directly available on version 3.0. Users whose workflow included saving CSV files directly to an SD card or the client will now need to first convert the binary file to CSV, either using the client app or

by installing the standalone Liquid Instruments File Converter onto the computer they use for data processing.

Non-backwards-compatible changes

Data scaling in LIA

In version 1.9, we implemented data scaling such that multiplying two 0.1 V DC signals resulted in a 0.02 V DC output. In version 3.0, we changed this such that the result was 0.01 V DC, which is more in line with customers' intuitive expectations.

Waveform Generator output must be enabled to use as modulation source/trigger

In version 1.9, a different channel's waveform could be used as a modulation or trigger source in the Waveform Generator, even if that channel's output was disabled. This was removed in version 3.0. Users who want to do cross-modulation without needing to unplug the outputs of their device would need to adjust their workflow.

Moku LabVIEW API

The Moku LabVIEW API package is intended to provide LabVIEW developers the resources needed to control any Moku device and, ultimately, the ability to incorporate these controls into larger end-user applications.

The new Moku LabVIEW API package provides the following:

- Fully functional examples for each instrument.
- A block diagram structure that is easy to understand and can serve as an end user's starting point for customization and adaptation.
- A set of VI functions providing full control over the Moku device.

Currently supported instruments

1. Arbitrary Waveform Generator
2. Data Logger
3. Digital Filter Box
4. FIR Filter Builder
5. Frequency Response Analyzer
6. Lock-in Amplifier
7. Laser Lock Box
8. Logic Analyzer
9. Oscilloscope
10. Phasemeter
11. Spectrum Analyzer
12. Waveform Generator
13. PID Controller
14. Multi-instrument Mode
15. Moku Cloud Compile

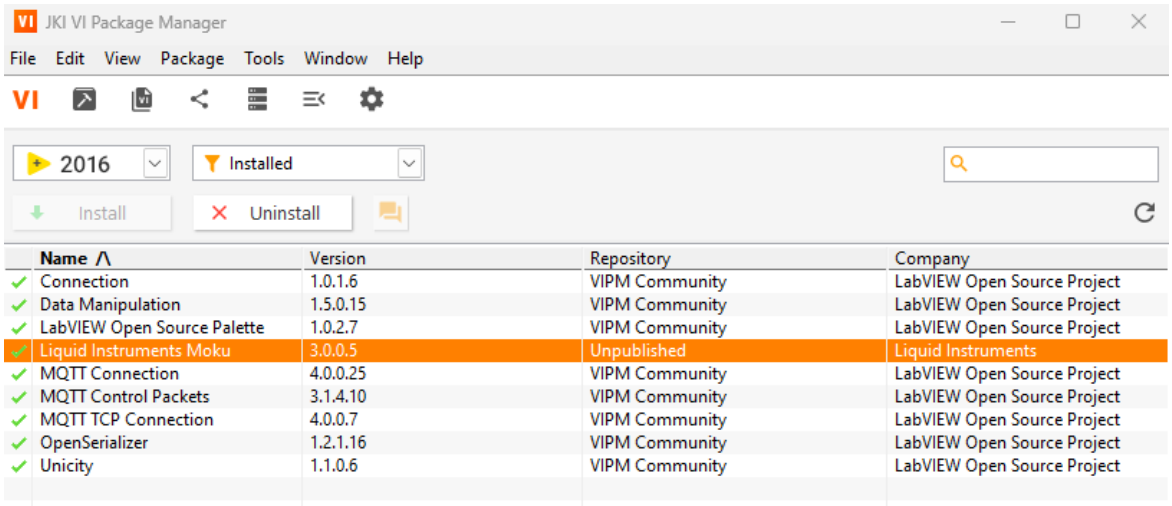
Installation

Requirements

- LabVIEW version 2016 or later
- VI Package Manager (VIPM)

If you already have a previous version of the Moku LabVIEW API installed, please uninstall it before proceeding. You can uninstall the package from the VI package manager by selecting *Uninstall Package*.

1. Download and install the **Moku LabVIEW package** from the Liquid Instruments website at
2. The package will install through the VI packet manager. Once complete, you should be able to see the package listed under “installed” in VI Package Manager.



Name	Version	Repository	Company
Connection	1.0.1.6	VIPM Community	LabVIEW Open Source Project
Data Manipulation	1.5.0.15	VIPM Community	LabVIEW Open Source Project
LabVIEW Open Source Palette	1.0.2.7	VIPM Community	LabVIEW Open Source Project
Liquid Instruments Moku	3.0.0.5	Unpublished	Liquid Instruments
MQTT Connection	4.0.0.25	VIPM Community	LabVIEW Open Source Project
MQTT Control Packets	3.1.4.10	VIPM Community	LabVIEW Open Source Project
MQTT TCP Connection	4.0.0.7	VIPM Community	LabVIEW Open Source Project
OpenSerializer	1.2.1.16	VIPM Community	LabVIEW Open Source Project
Unicity	1.1.0.6	VIPM Community	LabVIEW Open Source Project

Figure 1: JKI VI Package Manager

Note: The other packages listed here are dependencies used for data streaming.

Moku API changes

The new Moku LabVIEW API architecture is sufficiently different from its predecessor and therefore not backwards compatible with existing API scripts. The following simplified Oscilloscope example shows the differences between the legacy and new API packages and serves as a road map for porting existing code.

Oscilloscope example

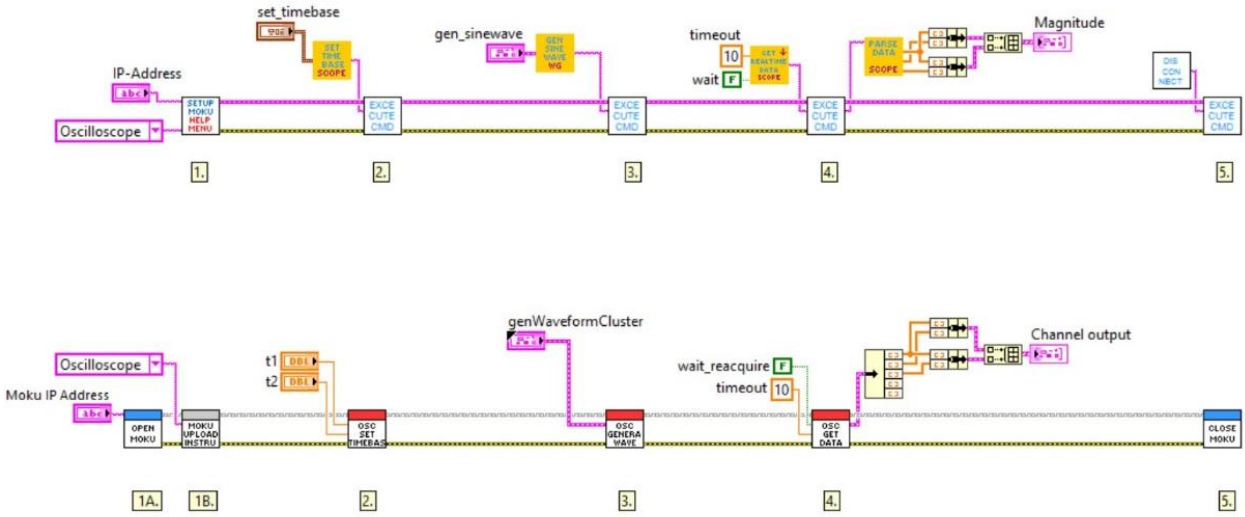


Figure 2: Oscilloscope API comparison

Sequence steps

1. Begin Client session and upload Oscilloscope bitstream to Moku.
2. Set time base and set the left- and right-hand span for the time axis.
3. Generate waveform, configure, and generate a sinewave on channel 1.
4. Get data, acquire a single frame of the data from the Oscilloscope.
5. End Client session.

The sequence described above is a simplified example to illustrate the differences between the legacy and new API packages. Aside from beginning a client session, uploading an instrument bitstream to Moku, and ending the client session, an end user can exercise any number of functions in various order to meet the needs of their application.

Differences

Here, we look at the differences between the two APIs for each step in the sequence.

1. Begin Moku-Client session and upload Oscilloscope bitstream to Moku.

The new API has split the client session connection and uploading the instrument bitstream into separate functions, 1A and 1B. All scripts begin with these 2 functions.

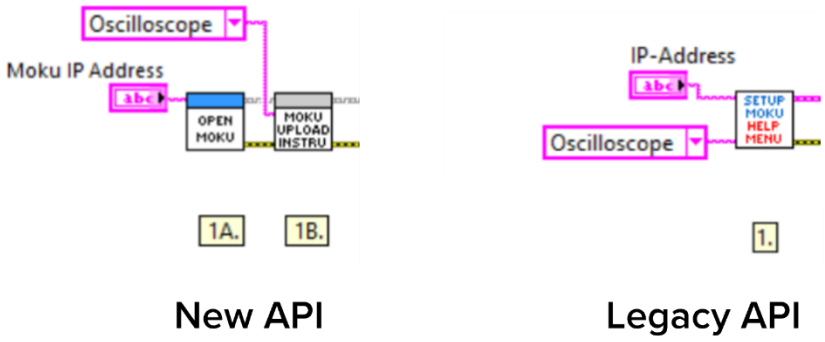


Figure 3: Begin session, upload bitstream

2. Set time base

Instrument functions in the new API are now single functions. Previously, this was a 2-step process in the legacy API. The first function converts the input parameters to a JSON string and the seconds function sends the command to the Moku. Additionally, function parameters in the legacy API were contained in clusters. The majority of function parameters in the new API are individual controls.

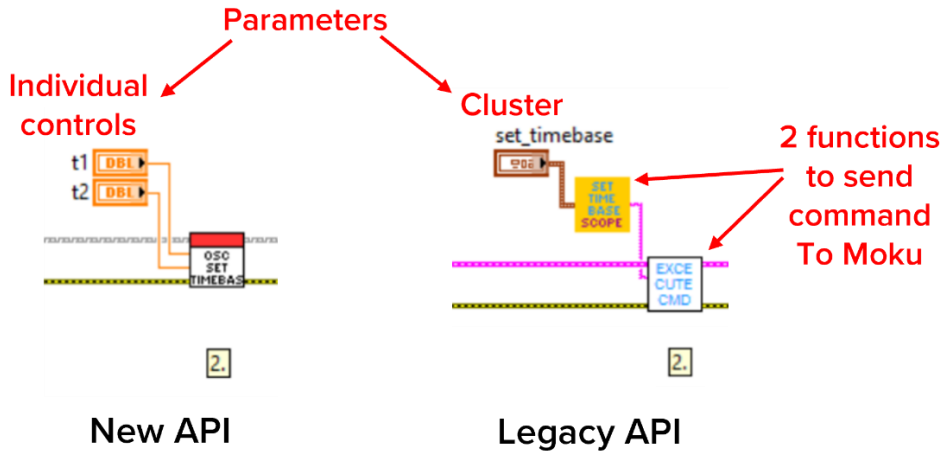


Figure 4 Set time base

3. Generate waveform

The generate_waveform function is a single function in the new API. In this instance, the function parameters are contained in a cluster. There are several functions in the new API that require many input parameters; in those instances, a cluster is used.

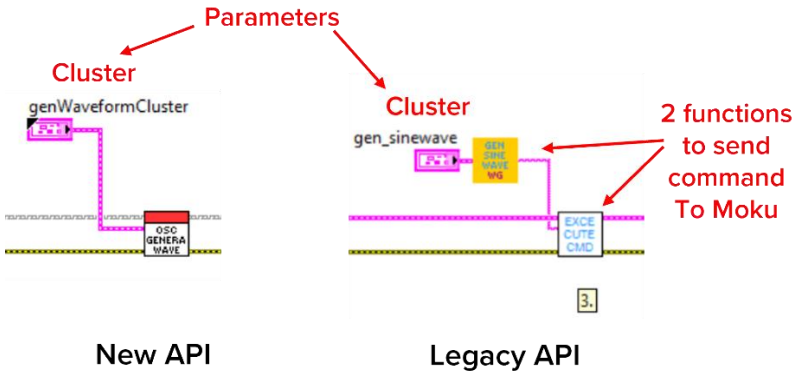


Figure 5: Generate waveform

4. Get data

The get_data function is also a single function in the new API. In this instance, the function parameters are individual controls for both APIs. The legacy API requires an additional function to convert the output data from the JSON string format into numerical arrays for each channel.

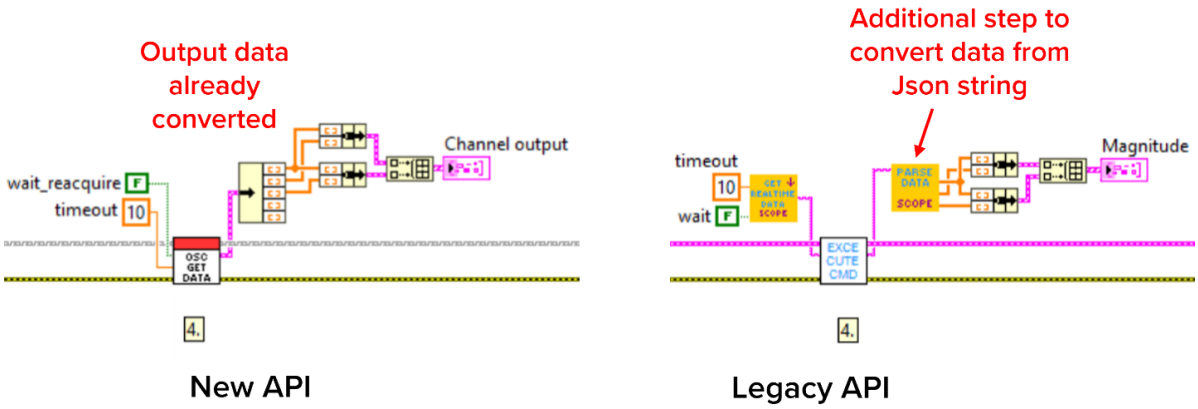


Figure 6 Get data

5. End Moku-Client session

The Close API function is a single function in the new API. All scripts end with this function.

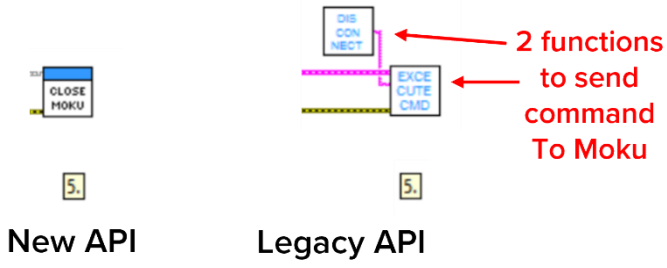


Figure 7: Close API

Palette Comparison

You can find equivalent instrument folders in the Liquid Instruments Moku main palette, seen in Figure 8. Then, in each instrument folder you'll find the equivalent instrument functions, seen in Figure 9 and Figure 10.

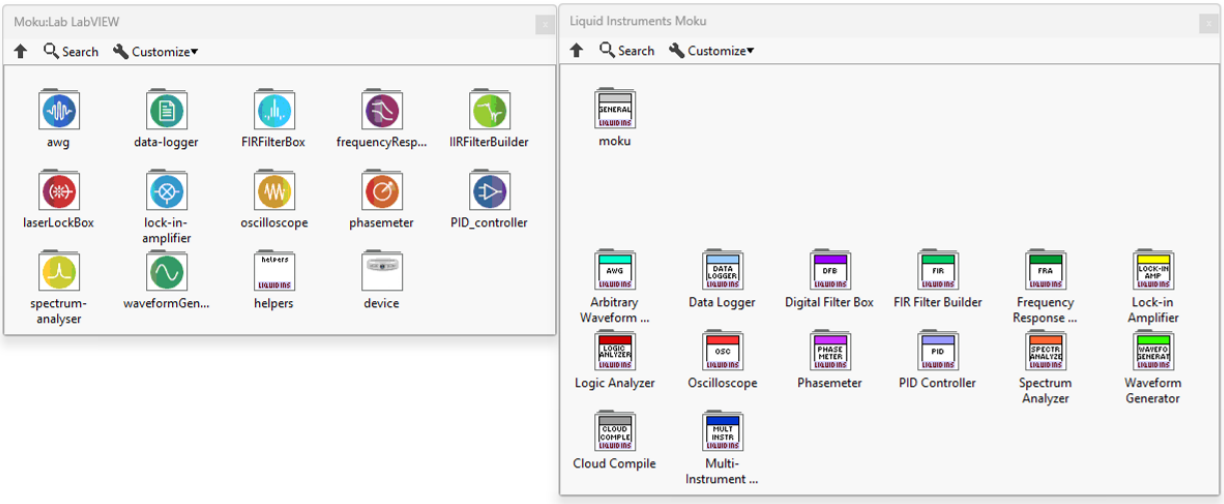


Figure 8: Main palette, legacy API palette left, new API palette right.

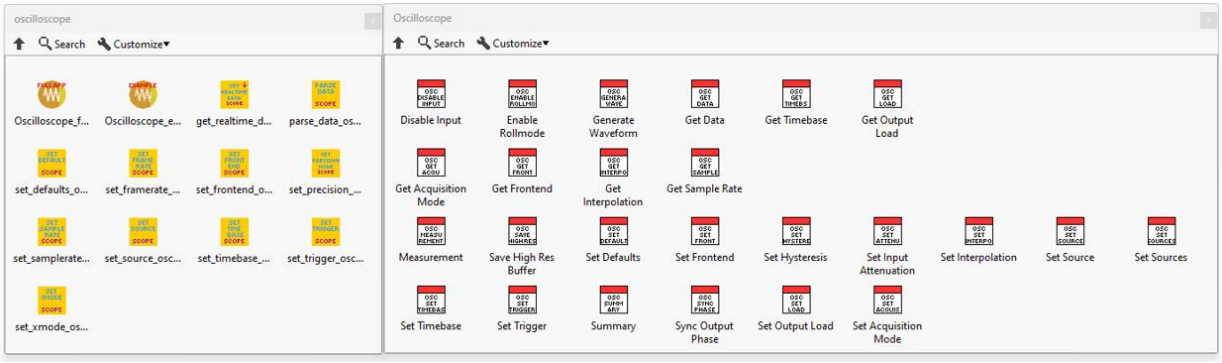


Figure 9: Legacy Oscilloscope instrument folder left, new Oscilloscope instrument folder right.

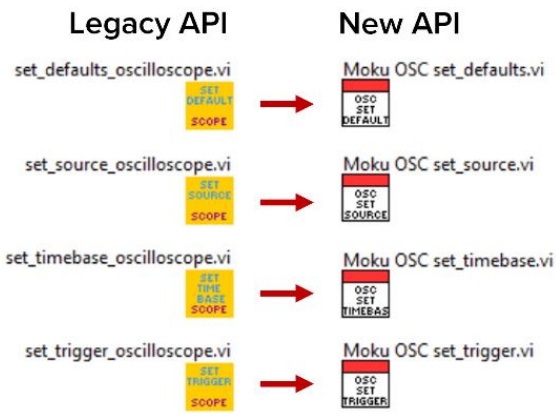


Figure 10: Instrument functions

The Moku LabVIEW API is based upon Moku API. For full Moku API documentation, refer to the Moku API Reference found here https://apis.liquidinstruments.com/reference/.

Additional details for getting started with Moku LabVIEW API can be found at https://apis.liquidinstruments.com/starting-labview.html.

Downgrade process

If the upgrade to version 3.0 has proven to limit, or otherwise adversely affect, something critical to your application, you can downgrade to the previous version 1.9. This can be done through a web browser.

Steps

1. Contact Liquid Instruments and obtain the file for firmware version 1.9.
2. Type your Moku:Lab IP address into a web browser (see screen shot).
3. Under Update Firmware, browse and select the firmware file provided by Liquid Instruments.
4. Select Upload & Update. The update process can take more than 10 minutes to complete.

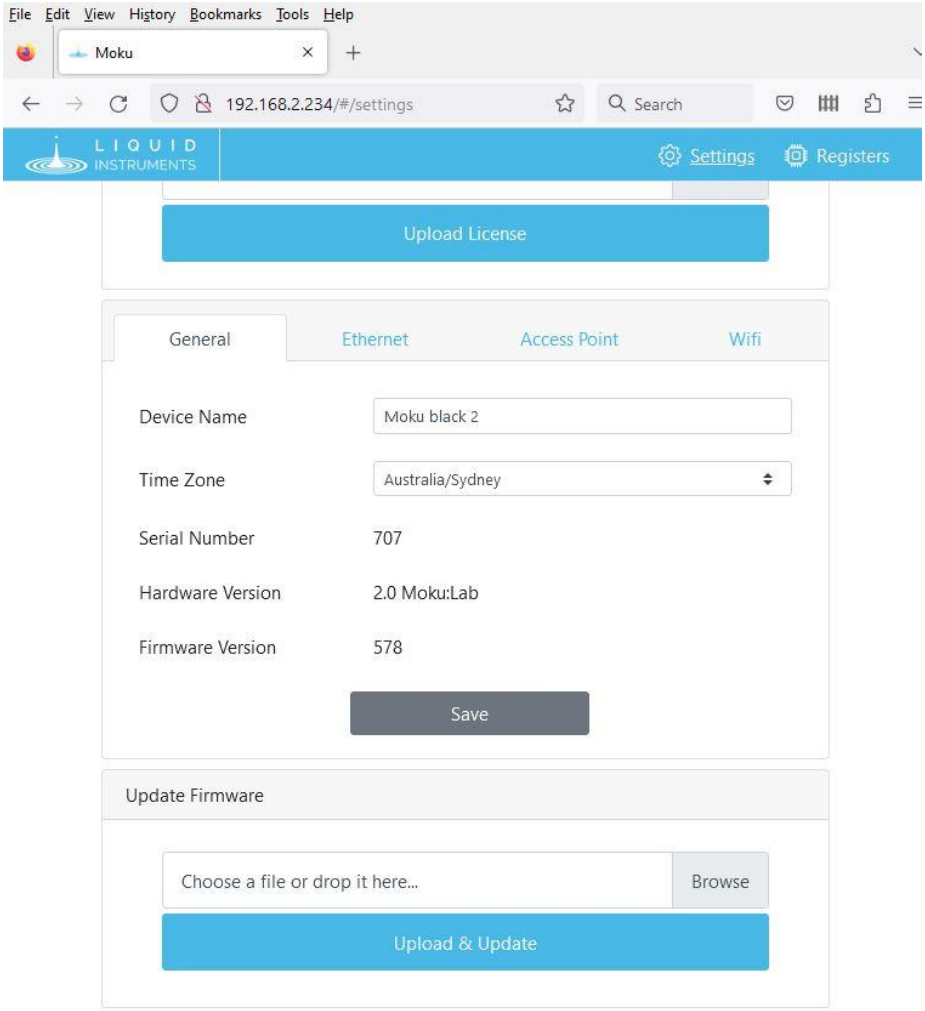


Figure 11: Downgrade procedure

