Moku PID Controller

User Manual

LIQUID INSTRUMENTS

Table of Contents

Introduction

The Moku PID (Proportional-Integral-Derivative) Controller features real-time configurable feedback controllers with a closed-loop bandwidth of >100 kHz. This enables each controller to be used in applications requiring both low and high feedback bandwidths such as temperature and laser frequency stabilization. The PID Controller also comes with embedded Oscilloscope and Data Logger to observe short- and long-term behavior of the controller.

Below we provide a guide to the underlying architecture of the instrument. We also include a general example in the quick start guide and a small number of in-depth examples to showcase different ways to use the Moku's PID Controller.

These user manuals are tailored to the graphical interfaces available on macOS, Windows, iPadOS, and visionOS. If you'd prefer to automate your application, you can use Moku API; available for Python, MATLAB, LabVIEW, and more. Refer to the API Reference to get started.

AI-powered help is available to aid both workflows. AI help is built into the Moku application, and provides fast, intelligent answers to your questions, whether you're configuring instruments or troubleshooting setups. It draws from Moku manuals, the Liquid Instruments Knowledge Base, and more, so you can skip the datasheets and get straight to the solution.

Access AI help from the main menu

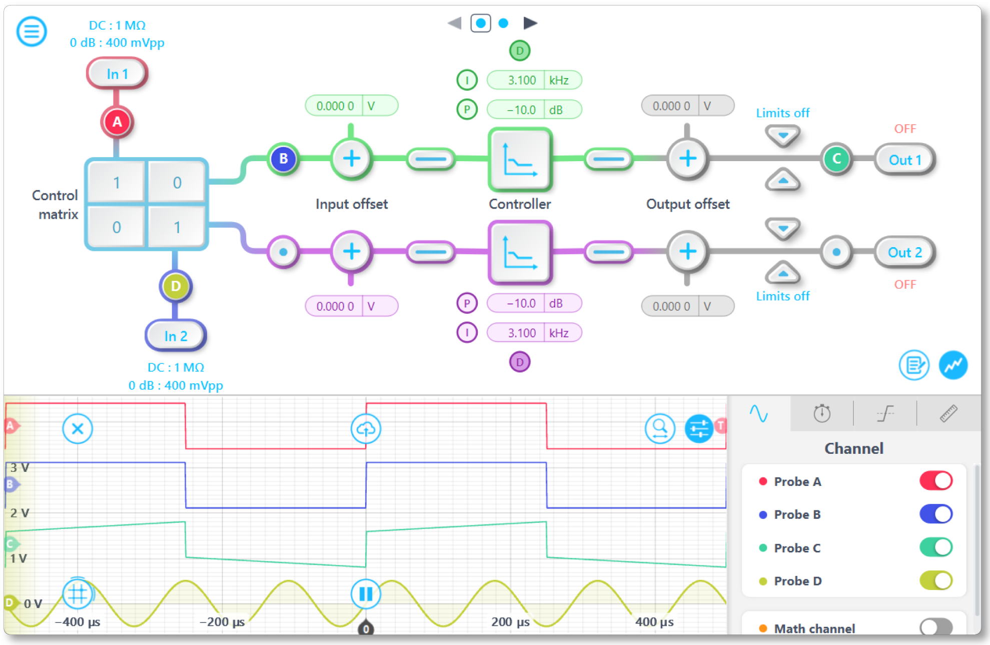


Figure 1. PID Controller user interface showing the instrument block diagram (top), embedded Oscilloscope panel (bottom) and the Oscilloscope settings panels (bottom right)

For more information on the specifications for each Moku device, please refer to our Product Documentation, where you can find the Specifications and the PID Controller Datasheets.

Quick start guide

Here we outline how to set up the Moku PID Controller and highlight a typical use case for the instrument.

In this example we incorporate the PID Controller into a feedback system. The measured signal is provided as Input1 with a reference signal provided as Input2. The output is sent to the actuator in the feedback system from Output1. In this case the PID Controller is used as a simple proportional-integral (PI) controller, with no derivative term.

Step 1: Configure the analog front end settings for the signal inputs

- Set the analog front end settings for the input. In this case, both Input1 and Input2 have a 50 Ω input impedance, 0 dB attenuation and use DC coupling.

Step 2: Configure the Control matrix

- In this example, the matrix is chosen to be [1,-1;0,0]. This indicates the matrix takes the difference between the two inputs, the sensed and reference signal, and then gives it to the controller.

Step 3: Configure the input/output offset

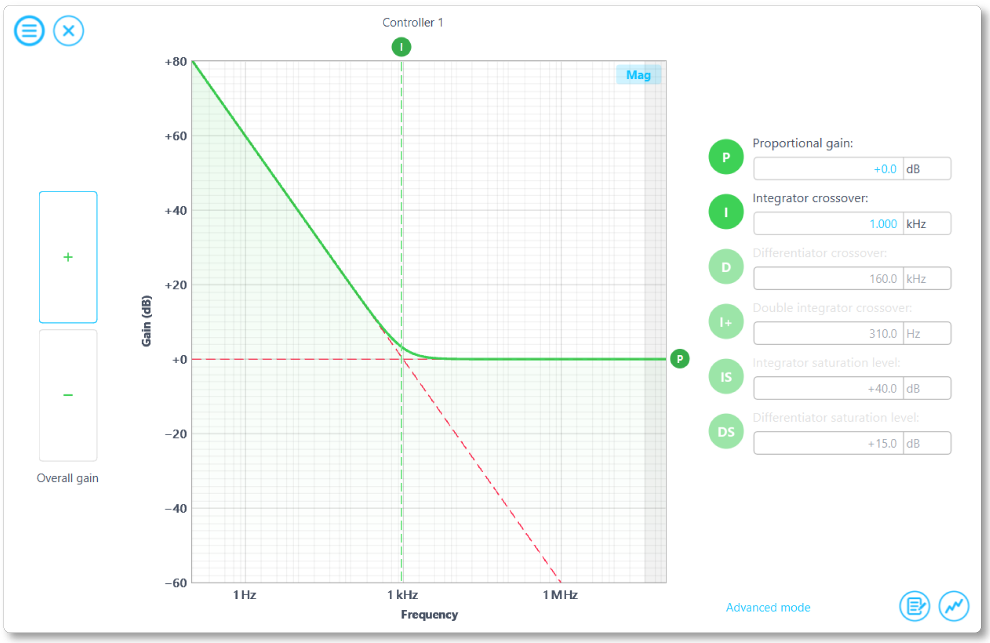
- Depending on the control loop settings, it is sometimes desirable to introduce a DC offset in the error signal calculation. For example, if the error signal at Input 1 has a DC offset of 10 mV, setting the input offset to –10 mV would compensate for it. Similar adjustments can be made by adding output offsets after the controller block.

Step 4 Configure the voltage limits

- In addition to the offsets, the user can also put voltage limits on each of the output ports. These limits ensure that excessive voltages are not applied to any component in the control system. For this example, the offsets are set to 0 with no limits on the output port.

Step 5: Configure the PID Controller

- Now configure the response by selecting the PID block. Doing so opens an interactive window that displays the PID response as a function of frequency. The behavior of the PID Controller can then be changed by enabling/disabling the different terms and putting in the gain value for each term. This can be done by dragging the markers on the interactive graph and changing it as desired. For this example, the Derivative and Double Integrator are disabled with only the Integrator and Proportional gain active. The Proportional gain is at 0 dB, with the Integrator crossover frequency at 1 kHz. Note: This step can be repeated multiple times to change the PID Controller behavior as required.



Proportional gain: +0.0 dB
Integrator crossover: 1.000 kHz
Differentiator crossover: 160.0 kHz
Double integrator crossover: 310.0 Hz
Integrator saturation level: +40.0 dB
Differentiator saturation level: +15.0 dB

Figure 2. Interface to change the gains of the PID Controller

Step 6: Observe signals on Oscilloscope

- After the PID Controller is set, probe points can be used to observe the signals. Enable the probe points before the controller and at the controller output. Clicking on these probe points opens up the embedded Oscilloscope menu and displays the signal at that point in the chain. Please see the Oscilloscope manual for more details on its operation.

Step 7: Enable the outputs.

- Once the Oscilloscope is setup to observe the signals, the output can be enabled. Click on the output icon to select between Off, 0 dB gain and 14 dB gain. For this example, 0 dB is selected as the smallest range.

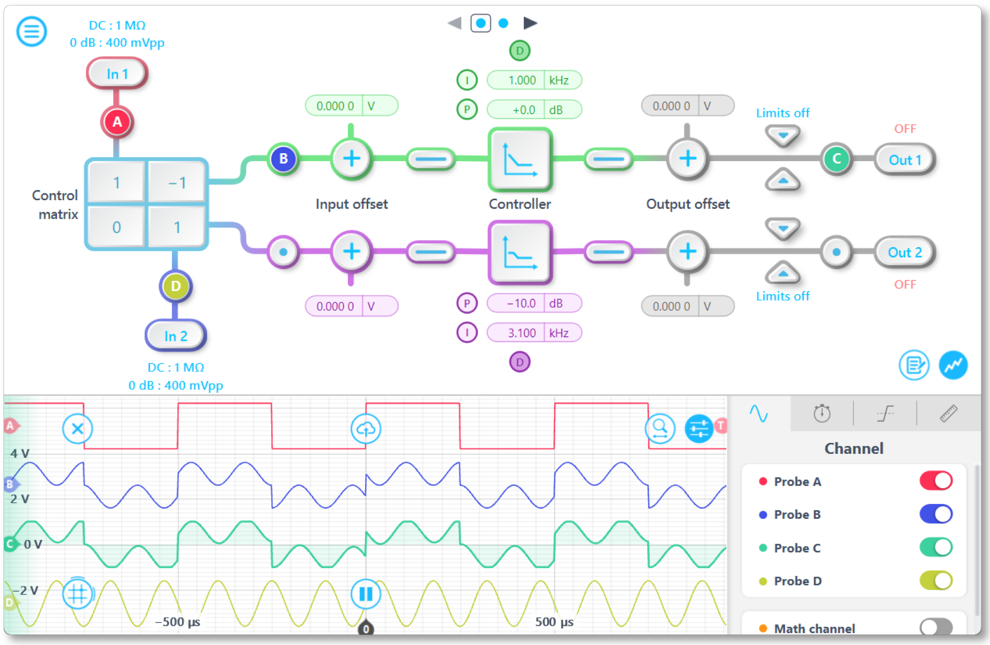
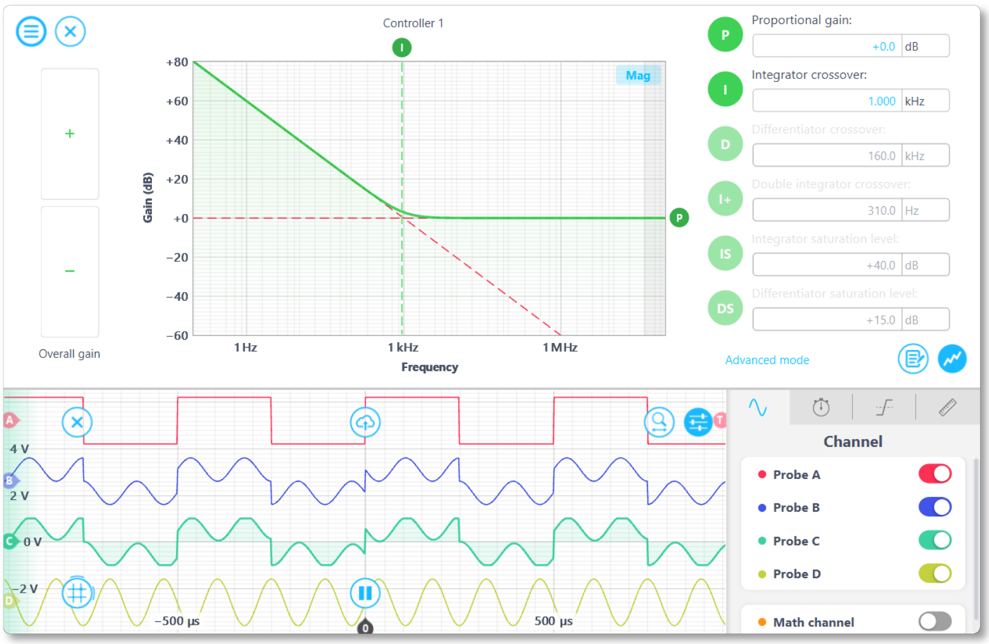


Figure 3. Using the embedded Oscilloscope to monitor signals before and after controller.

Step 8: Updating the PID Controller

- With the output enabled, the feedback system becomes closed. The embedded Oscilloscope is useful to observe the error and control signal. Using these probe points to monitor changes, the PID Controller can be tuned to optimize loop performance or maximize noise suppression. Note: other Moku instruments such as the Phasemeter and Time & Frequency Analyzer can offer additional metrics to help quantify performance.



Proportional gain: +0.0 dB
Integrator crossover: 1.000 kHz
Differentiator crossover: 160.0 kHz
Double integrator crossover: 310.0 Hz
Integrator saturation level: +40.0 dB
Differentiator saturation level: +15.0 dB

Probe A
Probe B
Probe C
Probe D

Figure 4. Tuning the PID Controller gains by observing the signals on the Oscilloscope.

Principle of operation

Moku's PID Controller instrument provides an easy-to-use interface for tuning proportional, integral, and derivative gains in a feedback loop. The PID is implemented by cascading two PID controllers to yield the final output. This architecture enables features such as a double integrator or multiple-section frequency response in Advanced mode. The basic control structure is shown in the block diagram below.

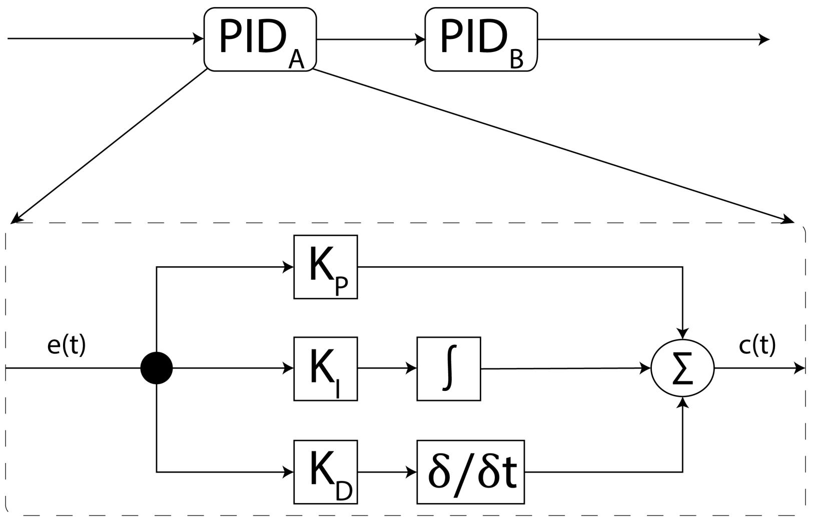


Figure 5. Block diagram of the Moku PID Controller.

Both PID$_A$ and PID$_B$ have identical structure. The behavior of the PID controller can be encapsulated by the time domain expression as

$$c(t) = K_p e(t) + K_I \int e(t)dt + K_D \frac{dx(t)}{dt} \tag{1}$$

Using a Laplace transform, this can be converted to the frequency domain as:

$$C(s) = K_P E(s) + \frac{K_I E(s)}{s} + K_D E(s)s \tag{2}$$

PID controllers are commonly used in feedback systems as they are easy to use and implement. Conceptually, each path contributes a correction to the measured error between the input and the reference signal. The proportional term applies a correction based on the current error but cannot eliminate steady-state error. The integral term addresses this by accumulating the error signal over time, which helps stability by driving the steady-state error toward zero. To further improve performance, the derivative term responds to the rate of change of the error, which dampens rapid fluctuations that the proportional and integral terms might otherwise amplify. In practice, the PI configuration is widely used, as it offers low steady-state error while being simple to implement.

The Moku PID Controller also provides the ability to set saturation on the integrator and derivative terms. These saturation levels allow the systems to have a finite gain at very low and very high frequencies. Limiting the integrator gain at low frequencies prevents long-term noise accumulation that could otherwise drive the system to its voltage limits. Similarly, setting saturation limits can avoid infinite gain for high frequency noise in differentiators, and thereby improve performance. While saturation limits improve stability and help during tuning, setting them too low can restrict the controller's ability to correct errors, leading to poor steady-state performance.

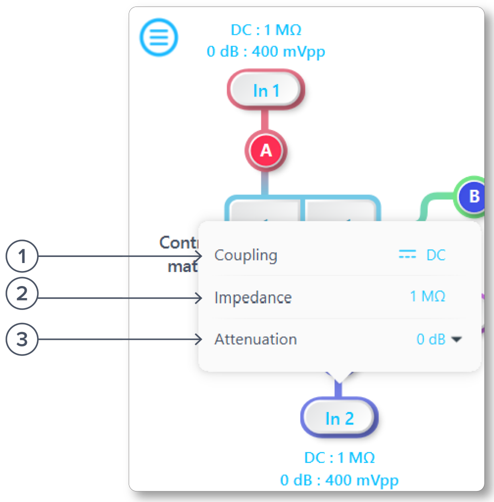
Please refer to the six-part app series for a deeper understanding on feedback systems and PID controllers.

- Part 1: Frequency-domain control: defining a transfer function
- Part 2: Feedback control: constructing feedback control loops
- Part 3: Stability and delays: assessing stability in feedback control loops
- Part 4: Loop shaping: frequency domain tuning
- Part 5: Understanding actuator saturation in control systems
- Part 6: PID Controllers: Frequency-Domain Models & Applications

Using the instrument

Signal inputs

The analog frontend settings for each input channel of the PID Controller can be individually configured. Click the In 1 icon to configure the input settings for the signal input.



Coupling DC
Impedance 1 MΩ
Attenuation 0 dB

Figure 6. Configuration of analog inputs on the PID Controller.

① Select between AC and DC input coupling.

② Select between 50 Ω and 1 MΩ input impedance (hardware dependent).

③ Select an input attenuation.

Control matrix

The control matrix combines, re-scales, and re-distributes the input signal to the two independent PID Controllers. The output vector is the product of the control matrix multiplied by the input vector.

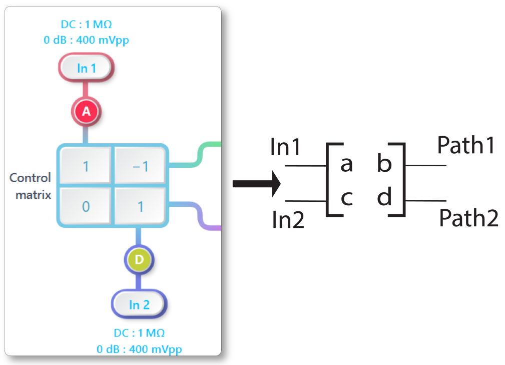


Figure 7. Control matrix in the block diagram and path schematic.

where $\mathrm{Path1} = a \times \mathrm{In1} + b \times \mathrm{In2}$ and $\mathrm{Path2} = c \times \mathrm{In1} + d \times \mathrm{In2}$.

The value of each element in the control matrix can be set between -20 to +20. The gain can be incremented by 0.1 when the absolute value is less than 10 and by 1 when the absolute value is between 10 and 20. Thus the matrix can be used to add or subtract two input signals to instead utilize a differential or common mode input for the PID Controller..

PID controller

Each channel is equipped with an independent PID Controller, positioned after the Control matrix that combines inputs from a pair of channels. This configuration allows precise control over each channel's feedback path following signal blending. If more than two channels are available, you can access the other channels by clicking the arrow at the top. Each Control matrix feeds two PID blocks, each of which in turn are connected to an output. The signal path is shown as block diagram in the PID instrument. To configure the PID gains, the PID block can be selected and then operated either in Basic or Advanced Mode.

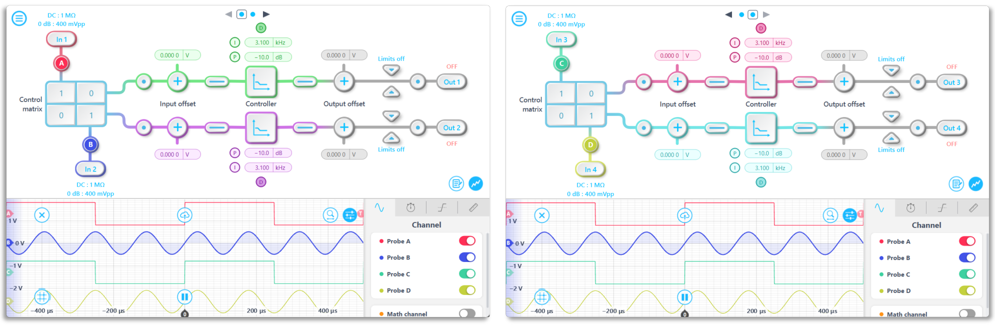


Figure 8. Accessing multiple PIDs on Moku:Pro.

Basic mode

The Basic mode of the PID Controller provides a simplistic way to change the PID gains.

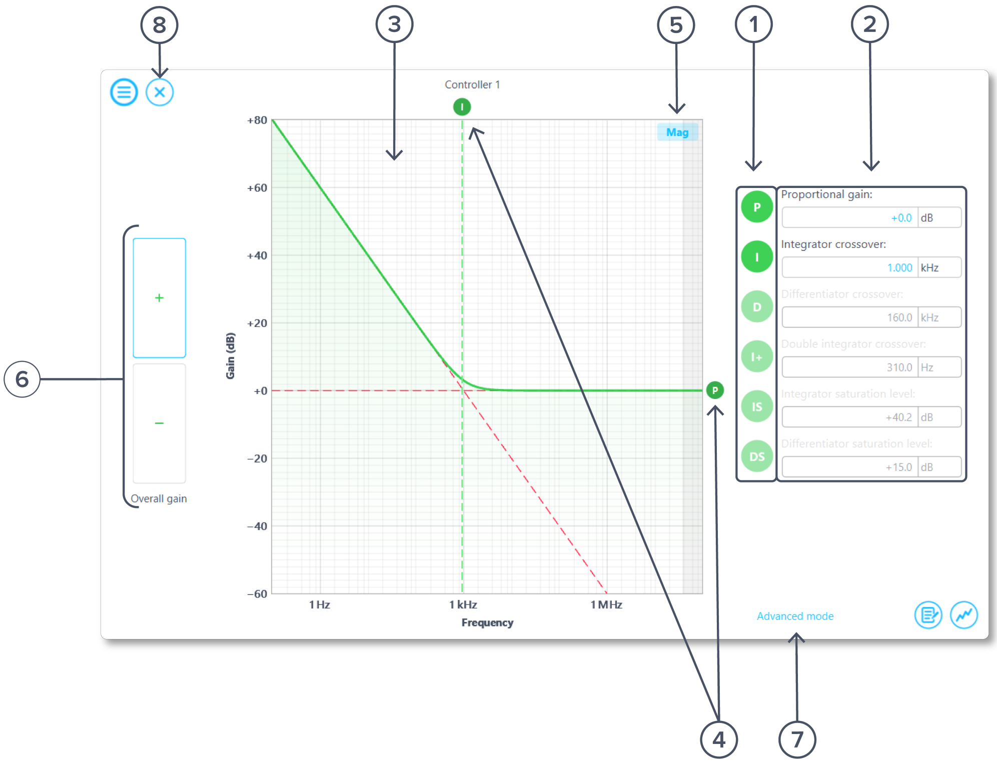


Figure 9. Interface to access Basic mode of the PID block.

① Enable/Disable button for the corresponding gain parameter.

② Field to observe or type in the numbers for each gain parameter.

③ Corresponding interactive PID response plot.

④ Markers on the plot indicate the enabled gain parameters.

⑤ Toggle between magnitude and phase graphs.

⑥ Increase/Decrease the Overall gain of the PID Controller.

⑦ Toggle between Basic and Advanced mode.

⑧ Close the PID block.

The gain fields of the different parameters are described as below:

Table 1. Parameters of the PID block

Parameter	Description
Proportional gain [dB]	Adjust the Proportional gain of the PID Controller.
Integrator crossover/ unity gain frequency [Hz]	Adjust the Integrator gain. If the Proportional gain is enabled, the gain changes the crossover frequency, otherwise the Integrator gain changes its unity gain frequency.
Differentiator crossover/unity gain frequency [Hz]	Adjust the Differentiator gain. If the Proportional gain is enabled, the gain changes the crossover frequency, otherwise the Differentiator gain changes its unity gain frequency.
Double integrator crossover [Hz]	Adjust the Double integrator crossover with the Integrator. Note that the Double integrator can only operate with the Integrator enabled. The crossover of the Double integrator is less than or equal to the Integrator crossover/unity gain frequency.
Integrator saturation level [dB]	Adjust the saturation level on the Integrators (both single and Double Integrators) at low frequencies. Note that the Integrator saturation can only operate with the Integrator enabled. The Integrator saturation level cannot be below the Proportional gain level.
Differentiator saturation level [dB]	Adjust the saturation level on the Differentiator at high frequencies. Note that the Differentiator saturation can only operate with the Differentiator enabled. The Differentiator saturation level cannot be below the Proportional gain level.
Overall gain [dB and/or Hz in other parameters]	Adjust the overall gain of the PID Controller. The Overall gain applies the effect to all paths of the PID Controller at the same time. If any of the paths hits the maximum or minimum allowable limits, the Overall gain adjusts the other paths.

Quick PID configuration

In the Basic mode of the PID controller, users can change the Proportional, Integrator and Differentiator without the need to open the block as shown in the screenshot.

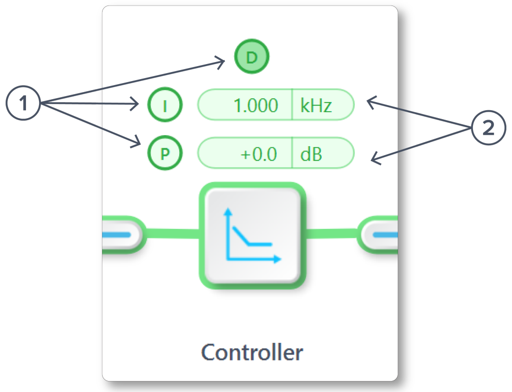


Figure 10. Accessing quick control on the PID block.

① Enable/Disable button for Proportional (P), Integrator (I) and Derivative (D).

② Field to observe and/or type in the numbers for each gain parameter.

Advanced mode

The Advanced mode in the PID Controller provides users the flexibility to manually adjust the gain settings of the PID Controller. The user can access each gain parameter from two PID cascaded blocks - Section A and Section B. The combined response of the two sections is shown in the PID response plot.

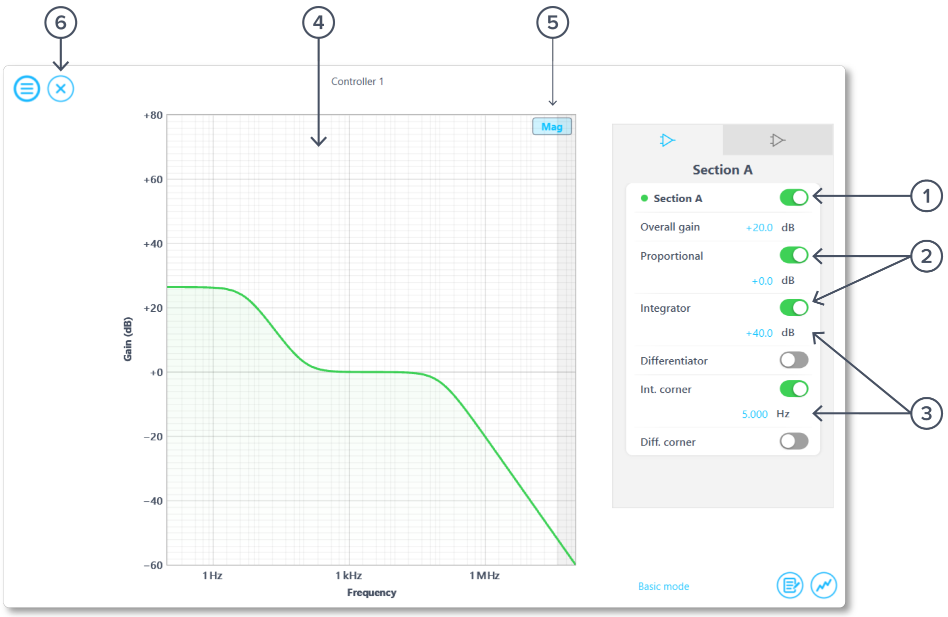


Figure 11. Accessing interface for Advanced mode on the PID Block.

① Enable/Disable button to select the corresponding Section. Disabling any Section would ensure only the other Section is active. Disabling both Sections would result in a pass through/ signal relay logic.

② Enable/Disable the corresponding gain parameter in each Section.

③ Field to observe or type in the numbers for each gain parameter in dB or Hz.

④ Corresponding PID response plot.

⑤ Toggle between magnitude and phase graphs.

⑥ Close the PID block.

The gains of the different parameters are shown below:

Table 2. Different Parameters of the PID section

Parameter	Description
Proportional [dB]	Adjust the Proportional gain of the PID Controller.
Integrator [dB]	Adjust the Integrator gain. The Integrator gain changes the unity gain frequency of the Integrator. If the Proportional gain is enabled, the crossover would depend on both the gain parameters. Note the Integrator cannot go below 0 dB.
Differentiator [dB]	Adjust the Differentiator gain. The Differentiator gain changes the unity gain frequency of the Differentiator. If the Proportional gain is enabled, the crossover would depend on both the gain parameters. Note the Differentiator cannot go above 0 dB.
Integrator corner [Hz]	Adjust the saturation level on the Integrator at low frequencies. The Integrator corner applies a flat passband from DC till the specified corner frequency. The gain of the passband is determined by the gain of the Integrator at the specified corner frequency. Note that the Integrator corner is only relevant when the Integrator is enabled.
Differentiator corner [Hz]	Adjust the saturation level on the Differentiator at high frequencies. The Differentiator corner applies a flat passband from the specified corner frequency till the maximum sampling rate of the device. The gain of the passband is determined by the gain of the Differentiator at the specified corner frequency. Note that the Differentiator corner is only relevant if the Differentiator is enabled.
Overall gain [dB]	Adjust the overall gain of the PID controller. The Overall gain applies the effect to all paths of the PID Controller at the same time.

Note: Double integrators can be implemented in the Advanced mode by the cascade of Integrator in Section A and Section B.

Controller path settings

Other block diagram elements in the PID Controller includes switches to enable/disable the signal in the processing path, offsets that can be applied to the input signal or the control signal and applying voltage limits on the output channels.

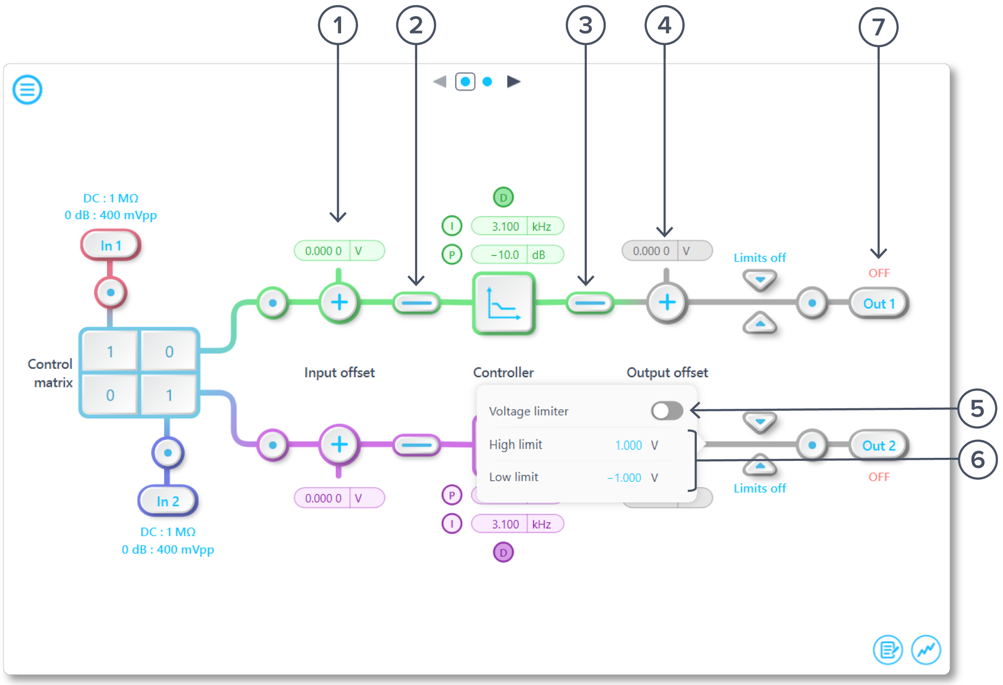


Figure 12. PID Controller path settings.

① Type in the Input offset before the Controller.

② Open/close the input switch from input signal to Controller.

③ Open/close the output switch from Controller to the output.

④ Type in the Output offset before being generated as output.

⑤ Enable/Disable the Voltage limiter.

⑥ Type in the high and low voltage limits.

⑦ Enable/Disable the output and set the output gain (if applicable).

Offsets

A DC offset can be applied to the signal both before and after the controller. Input offsets can be added or subtracted from the measured process variable before it is fed to the PID block. These are used to correct for any sensor calibration errors or to handle known deviations from the error point. Output offsets are added to the output of the PID block before it's sent to the actuator or system. These offsets are used to maintain operation in the system around a known nominal value, or when the actuator needs a default bias to operate.

Switches

The switches can be used to engage or disengage the control loop. When the switches are open, the input switch feeds zeros to the controller while the output switch gives zeros to the output. Upon clicking the input switch and closing it, the input signal is again fed to the controller. Similarly, upon clicking the output switch, the controller signal is passed to the output signal path. **Every time the switches are opened and closed, the Integrator and Differentiator registers in the PID Controller are cleared.**

Voltage limits

Voltage limits can be applied before the signals are generated from the output ports. These limits ensures the output is maintained at these voltage levels whenever the signal crosses the specified threshold.

For example, consider a system that only works with positive voltages. An input offset would be useful to generate a zero-crossing error signal with an output offset to return it to a positive level. The voltage limits would be useful to ensure that the minimum voltage is always greater than zero.

Observing the data

Embedded Oscilloscope

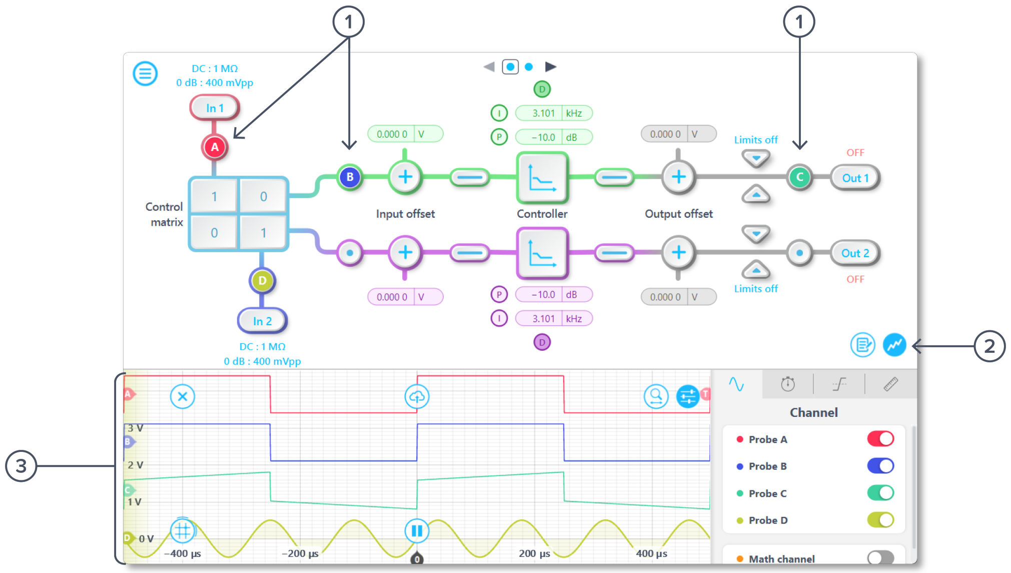
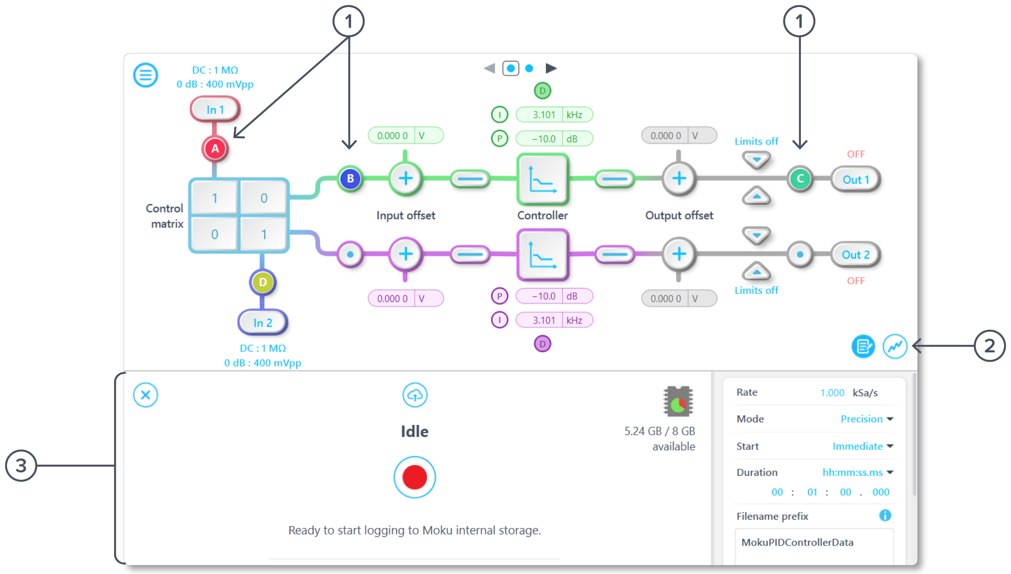


Figure 13. Probe point signals viewed in the embedded Oscilloscope.

ID	Parameter	Description
①	Probe points	Click to place the probe point, the number available is device dependent.
②	Open embedded Oscilloscope and Data Logger	Open and close the embedded Oscilloscope and Data Logger.
③	Oscilloscope	Refer to the Oscilloscope user manual for the details.

Data logging



Ready to start logging to Moku internal storage.

Figure 14. Embedded Data Logger in PID Controller.

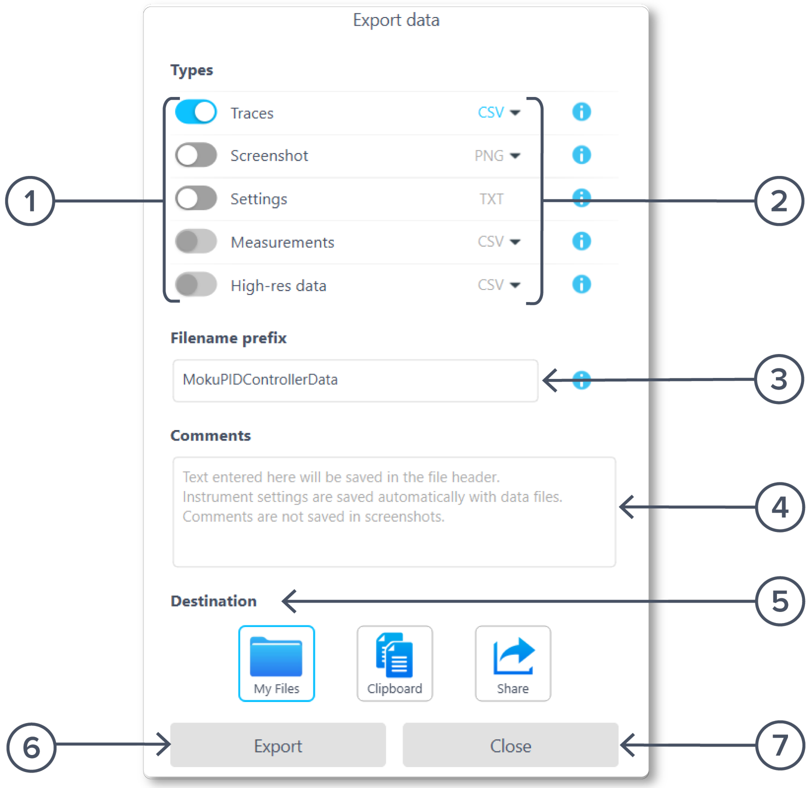
ID	Parameter	Description
①	Probe points	Click to place the probe point. You can enable up to four probe points at a time.
②	Open the embedded Oscilloscope or Data Logger	Open and close the embedded Oscilloscope and Data Logger.
③	Data Logger	Refer to the Data Logger user manual for the details.

The embedded Data Logger can stream over a network or save data to the onboard storage of your Moku. For details, refer to the Data Logger user manual. More streaming information is in our API Reference.

Exporting data

Export data by clicking the share icon. Any active probe points will be captured in the live data export or logging. Open the embedded Oscilloscope or Data Logger to export live and logged data, respectively.

Live data


Export data

Types

ON	Traces	CSV ▾	i
OFF	Screenshot	PNG ▾	i
OFF	Settings	TXT	i
OFF	Measurements	CSV ▾	i
OFF	High-res data	CSV ▾	i

Filename prefix

MokuPIDControllerData

Comments

Text entered here will be saved in the file header. Instrument settings are saved automatically with data files. Comments are not saved in screenshots.

Destination

Figure 15. Data exporting user interface and settings.

To save live data:

① Select the type of data to export:

- **Traces** Saves the trace data for all visible signal traces, in either a CSV or MATLAB format.
- **Screenshots** Save the app window as an image, in either a PNG or JPG format.
- **Settings** Saves the current instrument settings to a TXT file.
- **Measurements** Saves the active measurement values, in either a CSV or MATLAB format.
- **High-res data** Saves the full memory depth of statistic values for all visible channels, in LI, CSV, HDF5, MAT or NPY format.

② Select the **export format**.

③ Select the **Filename prefix** for your export. This is defaulted to "MokuPIDControllerData" and can be changed to any filename of alphanumeric characters and underscores. A timestamp and the data format will be appended to the prefix to ensure the filename is unique.

For example: "MokuPIDControllerData_YYYYMMDD_HHMMSS_Traces.csv"

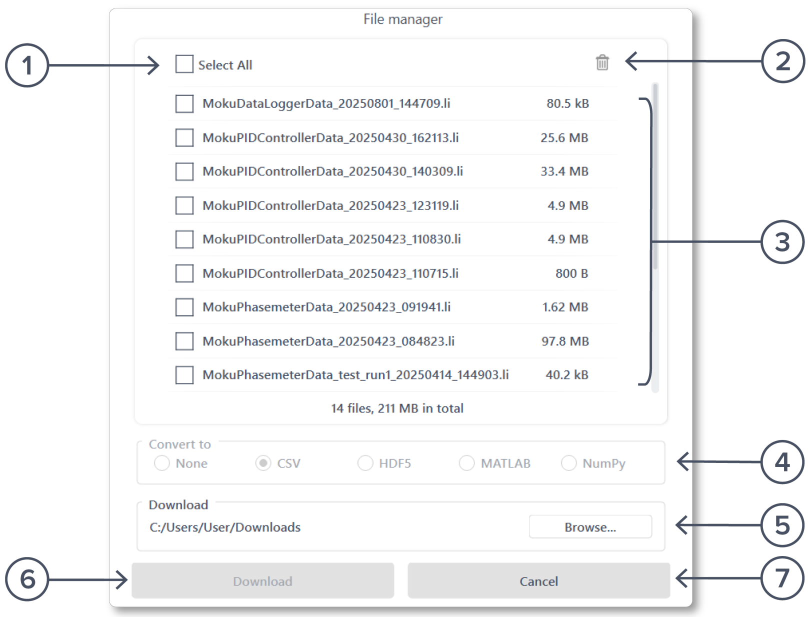
④ Enter additional **Comments** to be saved in any text-based file header.

⑤ Select the export **Destination** on your local computer. If "My files" or "Share" is selected, the exact location is selected when the Export button is clicked. Multiple export types can be exported simultaneously using My Files and Share, but only one export type can be exported to the clipboard at a time.

⑥ **Export** the data, or

⑦ **Close** the export data window, without exporting.

Logged data



File manager

Select All

MokuDataLoggerData_20250801_144709.li	80.5 kB
MokuPIDControllerData_20250430_162113.li	25.6 MB
MokuPIDControllerData_20250430_140309.li	33.4 MB
MokuPIDControllerData_20250423_123119.li	4.9 MB
MokuPIDControllerData_20250423_110830.li	4.9 MB
MokuPIDControllerData_20250423_110715.li	800 B
MokuPhasemeterData_20250423_091941.li	1.62 MB
MokuPhasemeterData_20250423_084823.li	97.8 MB
MokuPhasemeterData_test_run1_20250414_144903.li	40.2 kB

14 files, 211 MB in total

Convert to: None [✓ CSV] HDF5 MATLAB NumPy

Download: C:/Users/User/Downloads

Browse...

Download

Cancel

Figure 16. File exporting user interface and settings.

To save logged data:

① **Select all** files logged to the device's memory, to download or convert.

② **Delete** the selected file/s.

③ Browse and **select file/s** to download or convert.

④ Select an optional **file conversion format**.

⑤ Select a **location** to export your selected files to.

⑥ **Export** the data.

⑦ **Close** the export data window, without exporting.

Examples

Using PID in a feedback system

The Moku PID Controller can be directly incorporated into different feedback systems. A simple example involves using PID controller to control the flow of fluid in a tank.

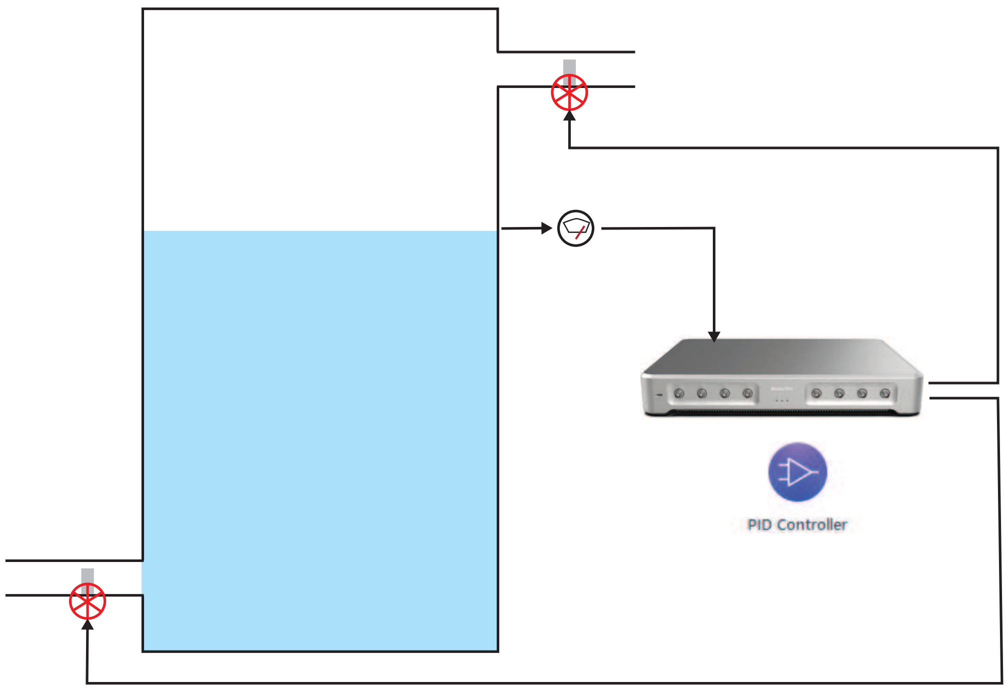


Figure 17. Block diagram of the water tank system.

Consider a simple block diagram of a tank system. The tank uses two valves to control the inflow and outflow of a fluid into the tank. A sensor is used to measure the fluid level in the tank and is given to the Moku as a voltage signal. The Moku PID Controller would then produce a signal to control the valves.

Step 1: Configure the analog front end settings for the signal inputs

- Set the analog front end settings for the input. In this case both inputs have a 50 Ω input impedance to match its source, -20 dB attenuation and use DC coupling.

Step 2: Configure the Control matrix

- Configure the Control matrix to take Input1 in control path 1, and Input1 in control path 2. As the same water level information is required for both the systems, both the control paths would use the same information. The matrix will take the values [1, 0; 1, 0].

Step 3: Configure the input and output offsets

- The input offsets provide the reference set point. Depending on the valve, the height can be translated to a voltage using a scaling factor. This can then be used to generate the reference DC offset and thus create an error signal.

- Since the valves operate in unipolar mode, the output offsets need to ensure the signal is positive at all times. This can be reinforced by enabling the voltage limits to have a minimum of 0 V.

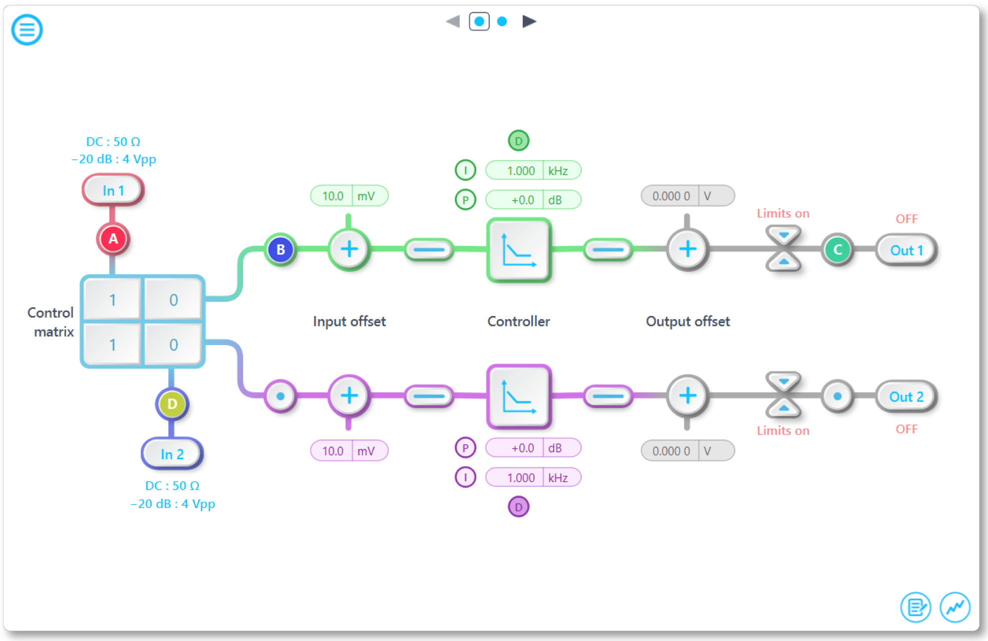


Figure 18. PID Controller interface for implementing feedback in the tank system.

Step 4: Configure the PID block

- The PID Controller can be set to the desired configuration for operation. The optimal values can be analytically calculated by doing an open loop analysis on the tank system. Alternatively, the control loop can be enabled at very low gains and slowly increase it until it becomes unstable.

Step 5: Enable the outputs

- Once the PID blocks are configured, the outputs can be enabled. These outputs would be used to control the valve operation.

Step 6 Observe the controller inputs and outputs

- Put probes on the input channels and at the outputs of the PID Controller.

Additional tools

Main menu

The main menu can be accessed by clicking the ☰ icon on the top-left corner.

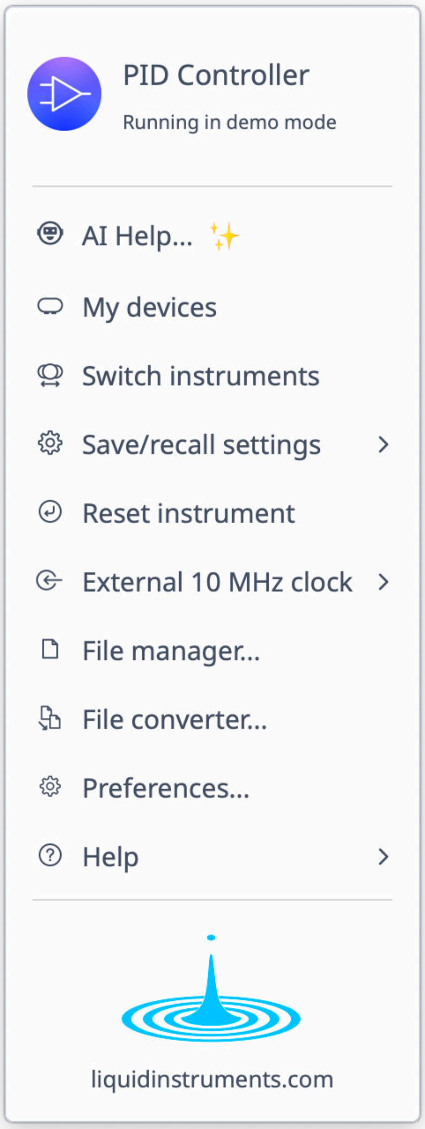


Figure 19. Main menu

AI Help... Opens a window to chat to an AI trained to provide Moku-specific help (Ctrl/Cmd+F1)
My Devices returns to device selection screen
Switch instrument to another instrument
Save/recall settings

- Save current instrument state (Ctrl/Cmd+S)
- Load last saved instrument state (Ctrl/Cmd+O)
- Show the current instrument settings, with the option to export the settings

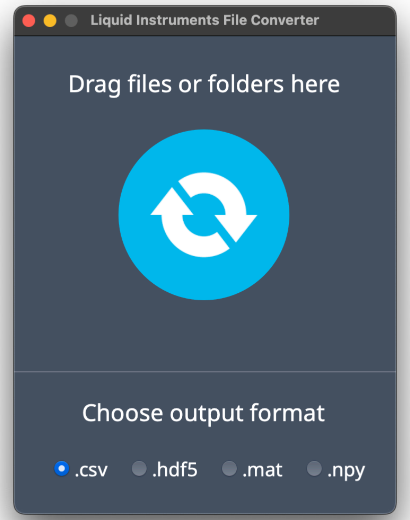
Reset instrument to its default state (Ctrl/Cmd+R)
Sync Instrument slots in Multi-Instrument Mode*
External 10 MHz clock selection determines whether the internal 10 MHz clock is used.
Clock blending configuration opens the clock blending configuration pop-up *
Power Supply access panel*
File Manager access tool
File Converter access tool
Preferences access tool
* If available using the current settings or device.
Help

- **Liquid Instruments website** opens in default browser
- **Shortcuts list** (Crtl/Cmd+H)
- **Manual** Open the user manual in your default browser (F1)
- **Report an issue** to the Liquid Instruments team
- **Privacy Policy** opens in default browser
- **Export diagnostics** exports a diagnostics file you can send to the Liquid Instruments team for support
- **About** Show app version, check for updates or licence information

File converter

The File converter can be accessed from the main menu ☰.

The File converter converts a Moku binary (.li) format on the local computer to either .csv, .mat, .hdf5 or .npy format. The converted file is saved in the same folder as the original file.



Liquid Instruments File Converter

Drag files or folders here

Choose output format

[✓ .csv] .hdf5 .mat .npy

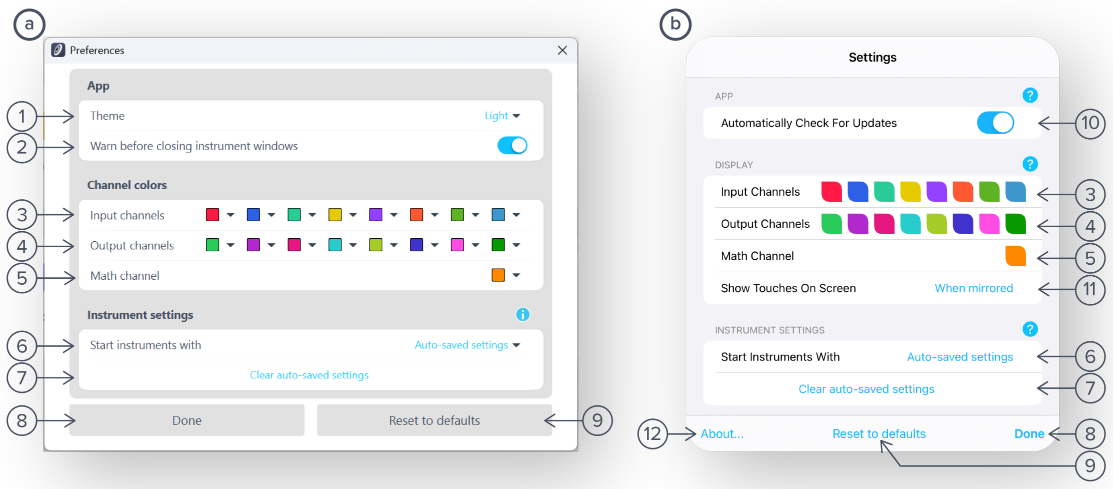
Figure 20. File Converter user interface.

To convert a file:

1. Select a file type.
2. Open a file (Ctrl/Cmd+O) or folder (Ctrl/Cmd+Shift+O) or drag and drop into the File converter to convert the file.

Preferences and settings

The preferences panel can be accessed via the Main Menu ☰. In here, you can reassign the color representations for each channel, switch between light and dark mode, etc. Throughout the manual, the default colors are used to present instrument features.



(a) Preferences

App
- Theme: Light
- Warn before closing instrument windows: on

Channel colors
- Input channels: 🟥 🟦 🟩 🟨 🟪 🟧 🟩 🟦
- Output channels: 🟩 🟪 🟥 🟦 🟩 🟦 🟪 🟩
- Math channel: 🟧

Instrument settings
- Start instruments with: Auto-saved settings
- Clear auto-saved settings

Done | Reset to defaults

(b) Settings

APP
- Automatically Check For Updates: on

DISPLAY
- Input Channels: 🟥 🟦 🟩 🟨 🟪 🟧 🟩 🟦
- Output Channels: 🟩 🟪 🟥 🟦 🟩 🟦 🟪 🟩
- Math Channel: 🟧
- Show Touches On Screen: When mirrored

INSTRUMENT SETTINGS
- Start Instruments With: Auto-saved settings
- Clear auto-saved settings

About... | Reset to defaults | Done

Figure 21. Preferences and settings for the Desktop (a) and for the iPad (b) App.

① Change the App theme, between dark and light mode.
② Choose if a warning opens before closing any instrument windows.
③ Tap to change the color associated with the input channels.
④ Tap to change the color associated with the output channels.
⑤ Tap to change the color associated with the math channel.
⑥ Select if instruments open with the last used settings, or default values each time.
⑦ Clear all auto-saved settings and reset them to their defaults.
⑧ Save and apply settings.
⑨ Reset all application preferences to their default state.
⑩ Notify when a new version of the app is available. Your device must be connected to the internet to check for updates.
⑪ Indicate touch points on the screen with circles. This can be useful for demonstrations.
⑫ Open information about the installed Moku application and license.

External reference clock

Your Moku may support the use of an external reference clock, which allows Moku to synchronize with multiple Moku devices, other lab equipment, lock to a more stable timing reference, or integrate with laboratory standards. The reference clock input and output are on the rear panel of the device. Each external reference option is hardware dependent, review the available external reference options for your Moku.

Reference Input: Accepts a clock signal from an external source, such as another Moku, a laboratory frequency standard, or an atomic reference (for example, a rubidium clock or a GPS-disciplined oscillator).

Reference Output: Supplies the Moku internal reference clock to other equipment that require synchronization.

If your signal is lost, or is out of frequency, your Moku will revert to using its own internal clock until the reference signal returns. If this occurs, check the source is enabled, and that the correct impedance, amplitude, tolerance, frequency, and modulation are attached to the reference. Check the required specifications in the device specsheets.

When the reference returns within range, status changes to "validating" and then "valid" once lock is re-established.

10 MHz external reference

To use the 10 MHz external reference function, ensure "always use internal" is disabled in the Moku application, found in the main menu under "External 10 MHz clock". Then, when an external signal is applied to your Moku reference input and your Moku has locked to it, a pop up will show in the app. On some devices, the external reference information will be shown in the LED status as well, more information can be found in your Moku Quick Start Guide.

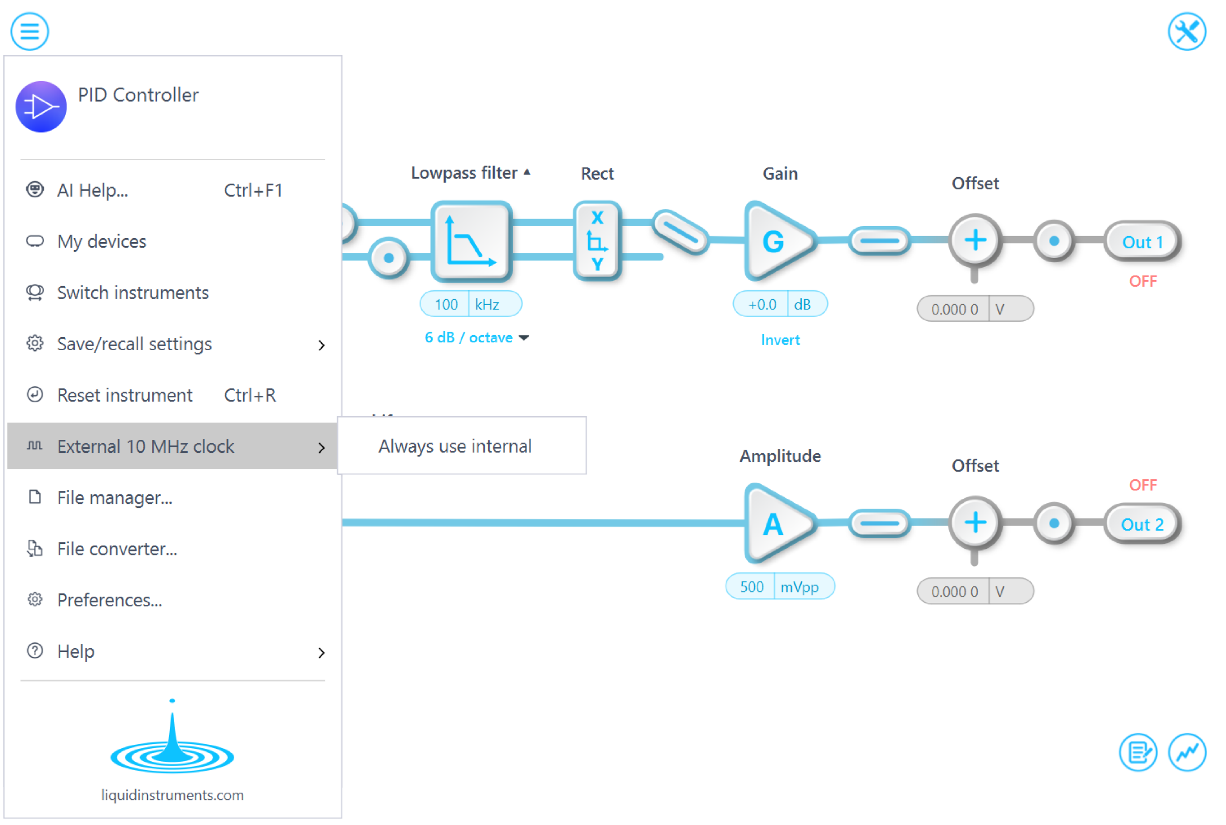


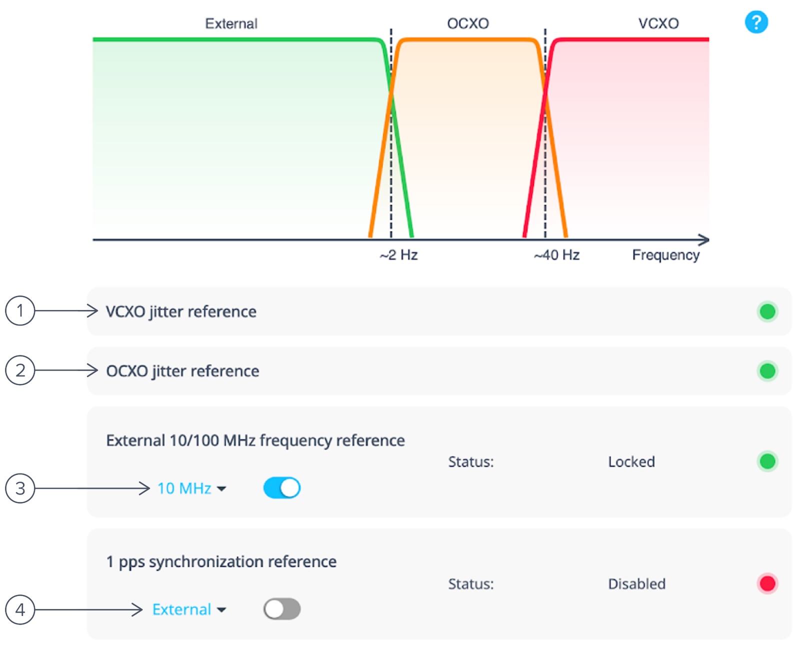
Figure 22. Moku main menu with "Always use internal" reference disabled and using an external reference.

Clock blending configuration

If available, Moku blends up to four clock sources simultaneously for more accurate phase, frequency, and interval measurements across all time scales. A low phase-noise Voltage-Controlled Crystal Oscillator (VCXO) is blended with a 1 ppb Oven-Controlled Crystal Oscillator (OCXO) for optimal wide-band phase noise and stability, which can be blended further with an external frequency reference and GPS disciplining to synchronize Moku with your lab and UTC.

The VCXO and OCXO will always be used for the clock generation signal. The external and 1 pps references are optional and can be enabled or disabled in the “Clock blending configuration...” settings from the main menu ☰. The loop bands are adjusted based on the different possible clock source configurations, shown in Figure 23, where the frequencies of the bands represent where each oscillator's phase noise dominates.

Read how the clock blending works on Moku:Delta for more details.



VCXO jitter reference ●

OCXO jitter reference ●

External 10/100 MHz frequency reference — 10 MHz — Status: Locked ●

1 pps synchronization reference — External — Status: Disabled ●

Figure 23. Moku clock blending configuration dialog with an external 10 MHz frequency reference and GNSS enabled.

① **VCXO jitter reference** is always used for clock generation, handling high frequency jitter with the lowest noise.

② **OCXO jitter reference** is always used for clock generation, ensuring moderate term stability.

③ **External 10/100 MHz frequency reference** uses a "10 MHz" or "100 MHz" external reference to correct drift in the local oscillator, noting your Moku will have to be restarted after each change between a 10 MHz and 100 MHz source.

④ **1 pps synchronization reference** uses an "External" or "GNSS" reference to sync with UTC and correct drift in the local oscillator. The estimated clock stability is a measure of how much the reference performance deviates relative to the local OCXO/VCXO timebase (as currently blended and, if enabled, steered by the external 10 / 100 MHz External reference).