Cockpit - User guide

PN300 | Posted on April 12, 2022 | Updated on August 25, 2025

Stéphane LOVEJOY
Senior Software Developer
imperix

Mateja ILIĆ
Software Development Engineer
imperix

Table of Contents

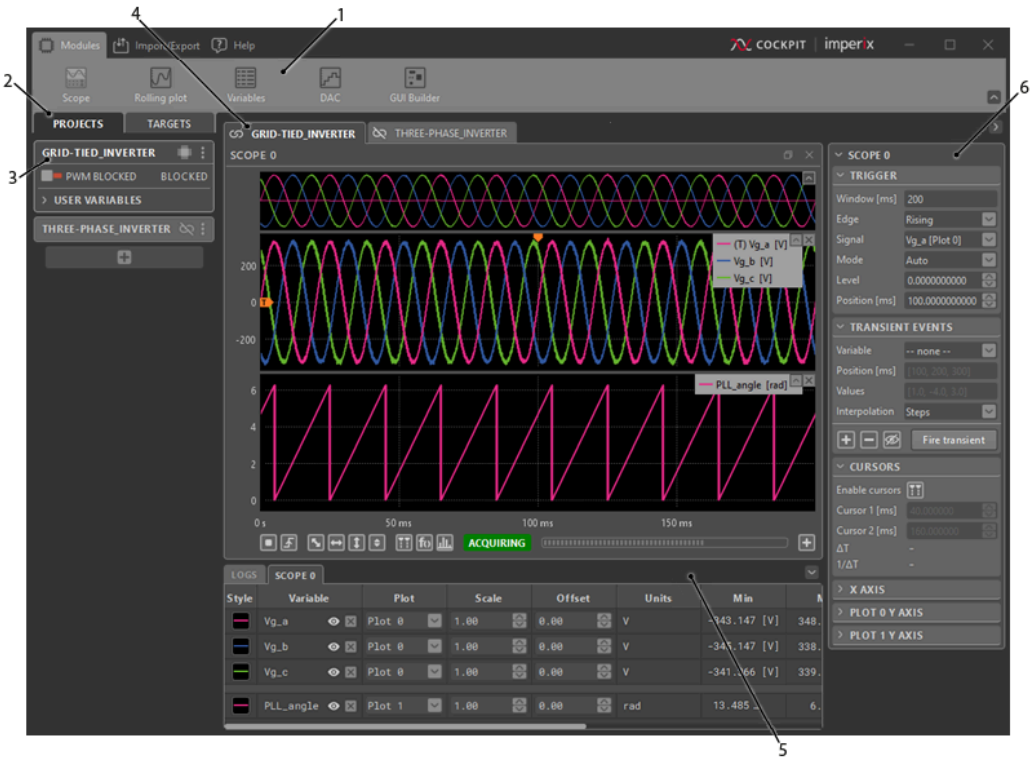
This user guide explains how to use imperix Cockpit to interact with imperix power converter controllers, namely the B-Box RCP, the B-Board PRO, the Programmable Inverter, and the B-Box Micro. The article describes in detail the tools and features provided by Cockpit as well as how to use them.

For new users, it is recommended to read the following article beforehand to get started with the imperix software development kit (SDK): Programming imperix controllers.

What is Cockpit

Cockpit is the monitoring software included with both the ACG SDK and CPP SDK. It is designed to facilitate the experimental testing of power electronics systems by leveraging the hardware capabilities of imperix programmable controllers.

The software provides multiple tools (modules) for both observing and acting on the user code running on imperix controllers.



Overview of Cockpit's interface

1 – Top bar

The top bar offers the following tabs:

- The *Modules* tab contains several modules that can be placed in the project view to monitor and tune the variables of the control code.
- The *Import/Export* tab provides tools to export signals acquired with the Scope and Rolling plot modules in the CSV or MAT file format or directly as a MATLAB figure.
- The *Help* tab contains tools that help interact with the hardware setup, like the target explorer and the device analog channel configurator, along with the global settings menu and various links to the documentation or other information.

2 – The left bar

The left bar allows switching between the two perspectives provided by Cockpit:

- *Projects*, which displays all of the projects created in Cockpit.
- *Targets*, which displays all of the imperix controllers detected by the host PC through Ethernet and the projects they are linked to.

In the *Projects* perspective, the user can quickly switch between multiple projects, making it easy to swap one control code for another or quickly test variants of the same control algorithm on a single target. In a multi-controller scenario, the left bar allows users to keep an eye on the status of every controller.

3 – Project pane

The project pane offers a centralized view from which the user can pilot, configure, and maintain the controllers in addition to a quick overview of the controller state and connection status.

4 – Project/target view

The project view is the work area where modules can be placed and rearranged at will to tune and monitor control algorithms. Each project has its own project view, which allows the user to tailor the area according to the application.

In the *Targets* perspective the view is reserved for displaying the info and configuration options related to devices selected through the left bar.

5 – The bottom bar

This is another project-specific area, organized in the form of tabs:

- The *Logs* tab, which is always present in every project, displays messages reported by the connected controller.
- The *Scope* and *Rolling Plot* tabs are generated when their respective modules are added to the project view. They display information related to the signals plotted in them.

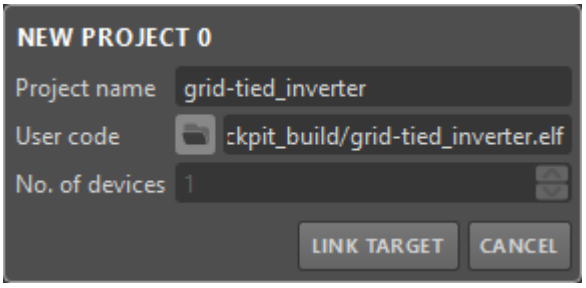
6 – The right bar

Similar to the Scope and Rolling Plot tabs in the bottom bar, the right bar hosts various configuration menus that are created when corresponding modules are added to the project view.

How to create a new project

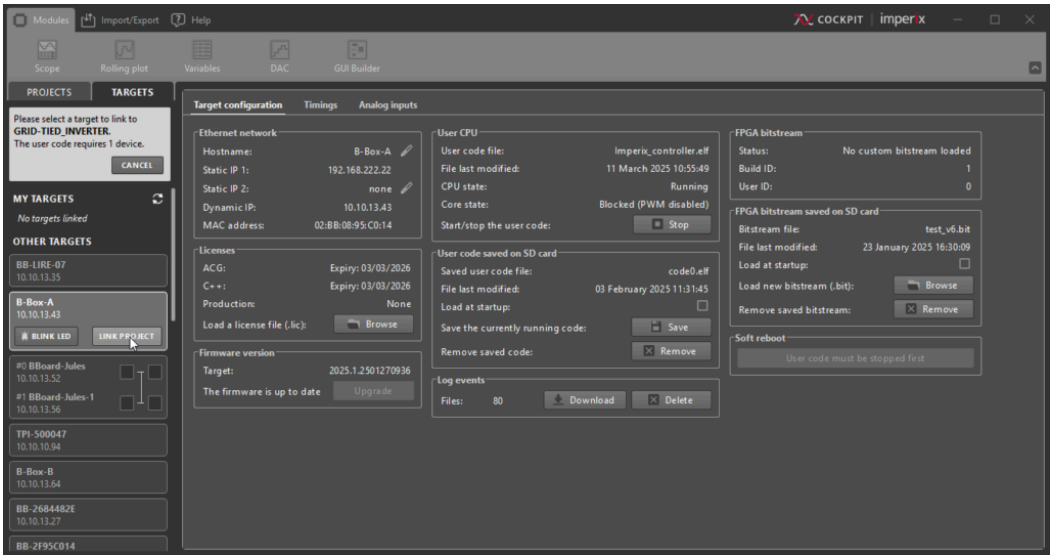
To interact with an imperix controller, creating a new project is required.
After the steps described in How to program the controller are achieved, Cockpit is automatically launched and a new project is created with a pre-filled project name and path to the user code.

Alternatively, it is also possible to manually open Cockpit and create a new project by clicking on the + button at the bottom of the left bar in the *Projects* perspective.



New project pane

To finalize the project creation, the project must be linked to a Target. Clicking on the *LINK TARGET* button in the new project pane will automatically switch Cockpit to the *Targets* perspective.



Overview of the Cockpit interface in the *Targets* perspective

In the *Targets* perspective the left bar shows a list of imperix controllers detected by the host PC through Ethernet. Please refer to the chapter on How to connect the controller to the host PC to ensure your target is properly connected to the host computer.

The controllers are grouped based on their RealSync connections. The RealSync networks are then separated into *MY TARGETS* and *OTHER TARGETS* based on whether at least one device in the network is already linked to a Cockpit project.

Clicking on a controller will bring up the related Target view and show the button that allows linking the project to it. If the selected target is already linked, relinking will unlink the previous project and link the current one.

While linking to any target is allowed, to avoid configuration errors please mind the warnings given by Cockpit and BBOS and ensure the device configuration in your

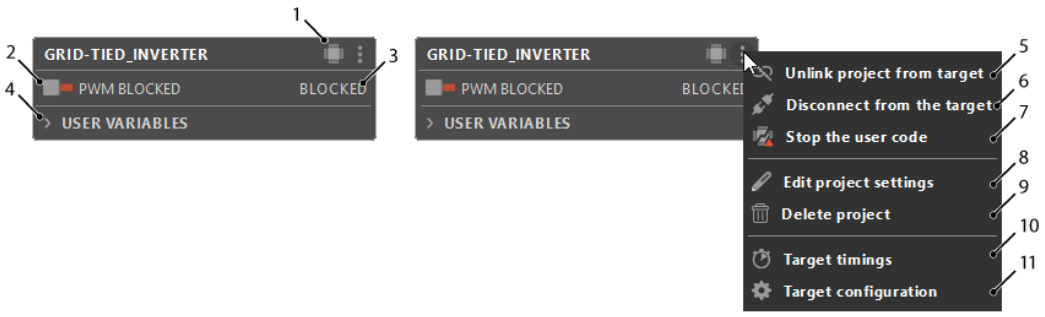
user codes matches your setup. To learn more about multi-device setups, refer to the chapter on Multi-device systems.

Once the project is linked, Cockpit will automatically shift back to the *Projects* perspective, where the newly created project pane will then appear in the left bar and a blank project view will be created.

How to interact with the project pane

The project pane offers a centralized view from which the user can pilot his imperix controller. It also allows quick access and an overview of all of the control variables defined in the user code.

Upon linking, the project pane will automatically connect to the specified target, load, and start the user code (.elf file). The following image shows the project pane once it is connected to the target.



Project pane when connected to a target

1 – Target status indicators

This space is reserved for status icons indicating if the project is disconnected from the device or, in the case it is connected, if there's any notifications from BBOS, user code running status etc.

2 – **Enable/Disable the PWM outputs**

Enables/disables the target's gating signals. Further information on this mechanism is available in the chapter Enabling/disabling PWM signals.

3 – **Target operating state**

Displays the current operating state of the controller.

4 – **User variables section**

This section displays all the user variables defined in the user code. These user variables are signals that are either connected to a Probe block or a Tunable parameter block.

Signals connected to Tunable parameter blocks can be modified directly from this section but also in the Project View, where various Cockpit modules make it easier to visualize and track these user variables in run-time.

5 – Unlink project from target

Clicking on this entry will unlink this project from its target.

Note that when the project is unlinked, the user code will continue to run, and if the PWM outputs are enabled, they will not be automatically disabled.

6 – **Disconnect from the target**

Clicking on this entry will disconnect the target from the host computer.

Note that when the project is disconnected, the user code will continue to run, and if the PWM outputs are enabled, they will not be automatically disabled.

7 – Stop the user code

Clicking on this entry will stop the user code on the target linked to the project.

8 – **Edit project settings**

Clicking on this entry allows the user to change the user code or delete this project.

9 – Delete project

This entry allows the user to delete the project.

Note that when the project is deleted, the user code will continue to run, and if the PWM outputs are enabled, they will not be automatically disabled.

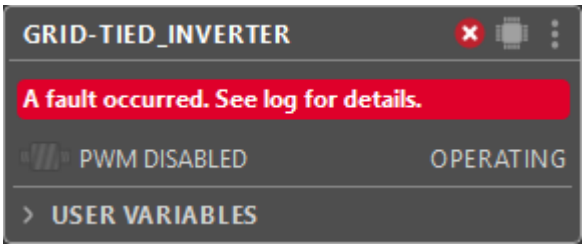
10 – Target timings

This entry switches Cockpit to the *Targets* perspective and shows the Timings tab of the linked target in the central view. This tab provides a graphical representation of the various computation and communication delays involved in the imperix controllers during run-time.

11 – Target configuration

This entry switches Cockpit to the *Targets* perspective and shows the Target Configuration tab of the linked target in the central view. From this tab, it is possible to configure the target. In the other tabs the user can see the visualization of the various computation and communication delays of the control, and generate an analog front-end configuration.

If a fault is triggered on the controller, the project pane interface will be changed to the one displayed in the following image. Further information on how faults work and how to troubleshoot them are detailed in The most common types of faults chapter.



If Cockpit unsuccessfully tries to connect to the target (image below), the project pane will indicate that the connection with the target cannot be established or is lost. In this case, check that the Ethernet cable is correctly connected and the target is powered on. Subsequently, verify that the specified IP address is correct and validate that the target can be pinged.

If Cockpit unsuccessfully tries to connect to the target (image below), the project pane will indicate that the connection with the target cannot be established or is lost. In this case, check that the Ethernet cable is correctly connected and the target is powered on. Subsequently, verify that the specified IP address is correct and validate that the target can be pinged.

Using the modules to interact with the controller

Once a project is created, the user has access to various control and monitoring modules that can be dragged and dropped from the top bar to the project view. The user can then freely rearrange and resize these modules inside the project view.

Scope module

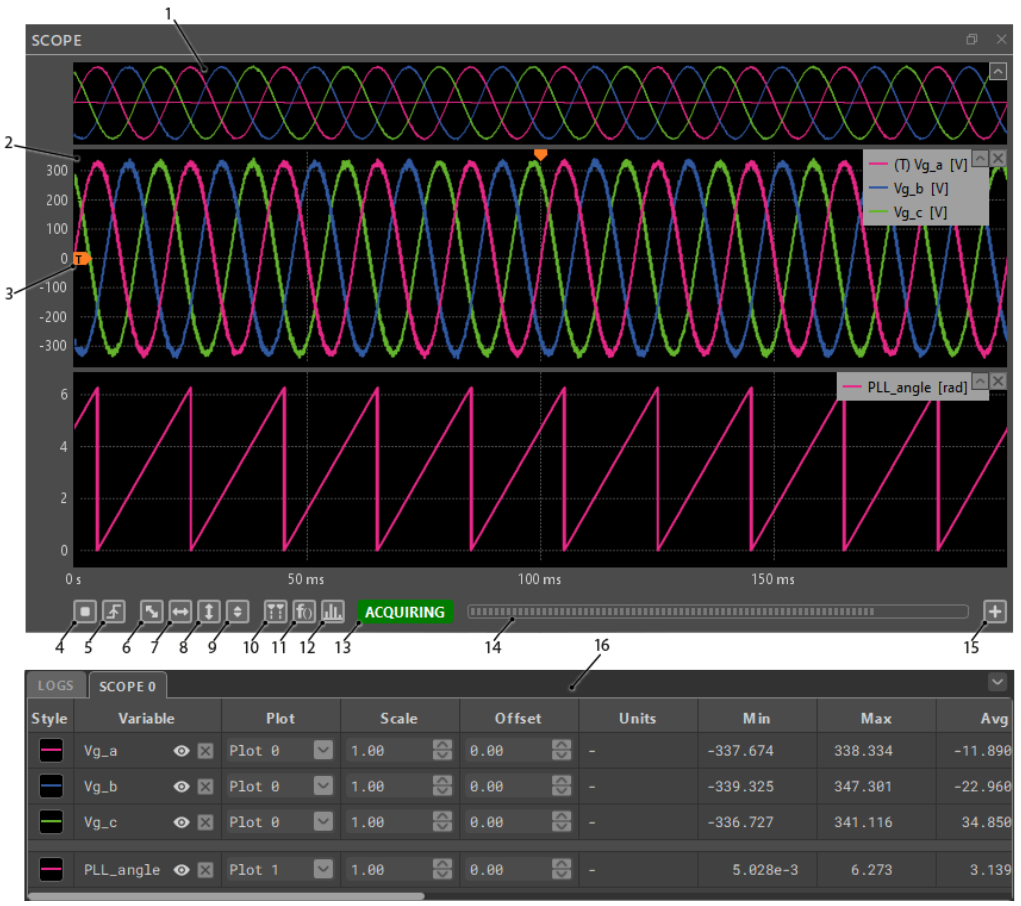
This module allows the user to display control signals on an oscilloscope-like interface by capturing and plotting every sample of the scoped user variables. The acquisition is done at the control task rate (i.e. the main interrupt frequency of the controller), ensuring that each and every sample is scoped.

The scope module can monitor up to 100 MBytes of data. The window length and the control task rate will affect how many user variables can be scoped.

If the control task is set to a standard frequency of 20kHz, it represents scoping one variable for 21 minutes or 32 variables (maximum number of scoped variables) for 40 seconds.

To start the acquisition, simply drag and drop a variable from the project pane directly to a plot of the scope module.
Activating the scope will affect the CPU load. This might be of concern if the CPU load is already close to 100% prior to the activation of the scope module.

Scope module interface



Scope module

1 – **Plot preview**

The plot preview shows an overview of all the scoped signals. The vertical axis of the preview is automatically scaled so all of the signals fit the plot. The plot preview can be collapsed to optimize space when using Cockpit on a small monitor.

2 – **Plot**

This is the area where the scoped signals are displayed. User variables can be dragged and dropped from the project pane into a plot, which will add them to the acquisition and display the corresponding samples in the area.
The x-axis range (when no zoom is applied) is defined by the *Window [ms]* field of

the Trigger pane (located in the right bar). If multiple plots are created, their x-axis will always remain synchronized, including when zooming.
The user can remove a variable from a plot from the *Scope* tab of the bottom bar or through the variable plot context menu accessed by right-clicking on the plotted signal.

3 – **Trigger position and level**

The plot containing the trigger signal will have two extra cursors added to it. The user can move these cursors to adjust the trigger level and position to set the trigger point (more information in the trigger configuration chapter).

4 – **Start/stop** button

Starts or stops the acquisition. Note that at least one variable must be added to a plot for the acquisition to start.

5 – **Force trigger** button

Force the acquisition of one window, bypassing the trigger mechanism. This is useful when the trigger is configured in Normal or Single mode.

6 – **Vertical and horizontal autoscale** button

Perform an autoscale on the horizontal and vertical axis, ensuring that the acquired signals fit in the plots.

7 – **Horizontal autoscale** button

Perform an autoscale on the horizontal axis of every plot while keeping the vertical axis unchanged.

8 – **Vertical autoscale** button

Perform an autoscale on the vertical axis of every plot while keeping the horizontal axis unchanged.

9 – **Continuous vertical autoscale** toggle

When switched on, this feature performs a vertical autoscale on every plot after each new acquisition window.

10 – **Scope vertical cursor toggle**

When switched on, a pair of slideable vertical cursors are shown. The cursors allow the user to mark an interval inside which several metrics are calculated on the scoped signals and displayed in the bottom bar.

11 – **Formula builder button**

Clicking this button opens a window that allows the user to create a new plotted variable based on a mathematical formula. The formula can combine any of the variables currently displayed inside the Scope module.

12 – Spectral analyzer button

Clicking this button opens a window where the user can examine the signals acquired by the scope in the frequency domain. The displayed spectra correspond one-to-one with the user and math variables displayed in the scope.

13 – Acquisition state

Displays the current acquisition state. The possible acquisition states are:

- *Stopped*: the acquisition is stopped. The displayed data is from the last acquisition.
- *Waiting:* the acquisition has started and is now waiting for a trigger event to occur.
- *Acquiring:* the trigger event has occurred and the signals are being captured.
- *Offline*: the target is disconnected from the host computer or the user code is not running on the target.

14 – Acquisition loading bar

The loading bar displays the acquisition and transmission status of the acquired window.
Note that it is normal to see the loading bar partially filled and blocked when the acquisition is in *Waiting* state.

15 – Add plot button

Creates an empty plot area at the bottom of the scope module.

16 – Bottom bar

The scoped variables are displayed in the *Scope* tab of the bottom bar. The bottom bar *Scope* tab menus allow the user to:

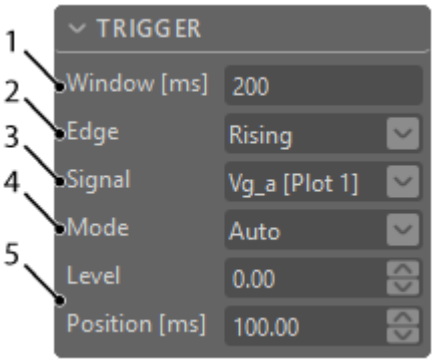
- Modify each variable's display style, scale multiplier, offset value, and measurement unit
- Remove a variable from scope
- Toggle a variable's visibility without removing it
- Move variables between scope plot areas

In addition to this, the position and the distance between vertical cursors are displayed here, along with the metrics calculated inside the interval defined by the

cursors if they are enabled. If the cursors are not enabled, the metrics are calculated over the entire acquisition window.

Trigger configuration

The trigger mechanism of the scope module behaves the same way as the trigger on a regular oscilloscope. The Tigger pane located in the right bar allows the configuration of the scope trigger.



Trigger configuration pane

1 – **Window[ms]**

Defines the acquisition window length and changes the x-axis length. This value affects the total number of acquired points.

2 – **Edge**

Defines whether to trigger on rising, falling, or both edges of the trigger signal.

3 – **Signal**

This field defines on which signal the trigger will be applied. This signal will be marked with (T) in the plot legend and can also be changed by right-clicking on a variable and setting it as a trigger through the context menu.
Note that only variables added to the scope module are present in this drop-down list.

4 – **Mode**

Defines the trigger acquisition mode.
In *Normal* mode, a window is acquired if the input signal reaches the defined trigger point.
In *Auto* mode (which is set by default), the trigger acts like in normal mode. However, the window acquisition is forced by a timer set to 100 ms, which ensures acquisition even if no trigger event occurs.

In *Single* mode, a single window is acquired if the input signal reaches the defined trigger point. After that, the acquisition is stopped.

5 – **Level and Position**

Change these values to set the trigger point. Alternatively, the trigger position and level can be adjusted directly in the plot area by moving the trigger level and position icons.

Transient generator configuration

The transient generator allows the user to apply a predefined signal on multiple user variables, provided they are connected to the Tunable parameter block. The signal is defined as a sequence of transient events at different positions within the scope acquisition window.

In the following example, three transient events are applied to `Ig_d_ref`. It can be seen that these events are generated simultaneously with the acquisition of `Ig_a`, `Ig_b`, `Ig_c`, and `Ig_d`.

Scope module containing the result of a transient sequence (on the left).
Transient configuration pane (on the right)

1 – Variable

This field defines on which variable the defined signal will be applied. In this example, the variable `Ig_d_ref` is used.

2 – **Position [ms]**

This field defines a vector of positions at which the transient signal values will be applied. In the example, the events are set at 50ms, 100ms, and 150ms.

3 – **Values**

This field is a vector of values that will be applied on the transient variable. In the above example, the values [10, 5, 15] are applied at [50, 100, 150] ms. The initial value of the tunable variable `Ig_d_ref` was set to 2.5 and that value is held until the first event at 50ms.

Please note that, since Cockpit version 2024.3, this field defines the exact values at the corresponding positions, and not how much the signal should jump at those positions. The initial value of the variable is relevant only in terms of determining the applied values between the beginning of the acquisition window (0ms) and the first defined transient event.

4 – **Interpolation**

This field defines whether the set values are held until the next event (the Steps option) or used to define linear ramps between the event points (the Linear option). After the last event, the set value is held until the end of the acquisition window regardless of the picked interpolation option.

5 – **Second transient** sequence

The transient generator allows entering multiple transient sequences on different variables.
In this example, the second transient sequence is unused.

6 – Add/remove a transient sequence

These two buttons allow the removal or the addition of transient events. There is no limit to the number of transient events.

7 – Preview Transient

Clicking the preview button toggles the transient preview in the scope. The transient preview of a given variable shows up as an editable signal in the same plot area as the original acquired variable. Toggling the preview and editing it does not fire the transient or affect the scope acquisition in any way.

8 – Fire transient

Clicking on this button will initiate the transient sequence on all transient variables. Once the sequence is finished, the acquisition will be in *Stopped* state while the transient variables will go back to the value that was set before the transient was fired.

Defining transients graphically

Starting from Cockpit version 2024.3, the transient input signal can be set using an interactive graphical interface. Clicking the preview button or any of the transient

event fields causes the transient preview to show up in the scope, provided that:

- The transient events and transient event definitions are valid
- The variable set in the Variable field is added to the Scope

Transient preview from the previous example after selecting the Linear interpolation option and dragging one transient event point

Once the preview shows up, the transient event points, if any are set, will show up as large dots. Using the mouse, the dots can be dragged to set the transient event position and value in one move. To **add new points**, double click near the editable transient signal and a new dot will show up. To **remove existing points**, right click on the point and select the 'Remove point' option from its context menu.

All of these actions immediately update the corresponding transient event in the right bar. Editing the transient preview **does NOT affect** the actual Tunable variable or the scope acquisition in any way, unless the 'Fire transient' button is pressed. To exit the transient preview mode without firing the transient, simply press the Transient preview button in the right bar again.

Cockpit Formula Builder

The Formula Builder allows the user to create math variables that are then added to the Scope module. Math variables are created based on a mathematical formula that can include any of the currently Scoped variables.

The math variables are calculated on the PC side after the acquisition of a scope window. This alleviates the need to define every variable to monitor in the user code, saving on hardware resources while still allowing the user to track them live through Cockpit. This also saves time, as new math variables can be added or their formula edited without having to recompile the user code.

With the exception of constant expressions and scalar-valued functions, all of the mathematical operators and functions provided by the Formula Builder are applied element-wise. This means that the formula is applied to each and every sample of the variables included in the expression, creating a new signal to be plotted in the scope.

Layout of the Cockpit Formula Builder

1 – Math Variable Name

To create a Math Variable, it needs a valid name. The name has to be unique and cannot be edited after creation.

2 – Math Variable Unit

While not necessary for Math Variable creation, the unit of the new variable can be set while building its formula. If not set here, the unit can always be edited, like with any other variable, from the bottom bar.

3 – Target Plot Area

This option allows the user to choose where the created Math Variable should be plotted or even create a new area to plot it in. After creation, the Math Variable can be moved around between plots like any other scoped variable.

4 – Show/hide history button

This button controls the appearance of the list of formulae of previously created Math Variables.

5 – Math Formula Field

Here, the formula is defined either by typing it or clicking on the options in the menus below.

6 – Formula History List

The list contains all of the formulae of the previously successfully created Math Variables.

The star next to each of the formulae allows us to favorite it, making it available in all projects. Non-favorited formulae disappear when the scope they were used in is deleted.

7 – Math Operator Menu

This menu contains all of the binary operators usable for building a math formula.

8 – Math Function List

This list contains all of the functions usable while building a math formula. The functions are grouped into three categories:

1. General math is a collection of elementary (abs, exp, log, sgn, sqrt) and rounding functions (ceil, floor, round, trunc)
2. Trigonometry includes all of the typical trigonometric and hyperbolic functions, along with conversion functions between degrees, radians, and gradians
3. Scalar-valued functions are grouped based on the fact that they all evaluate into one number.

9 – Available variables list

This list contains all the variables currently displayed in the scope, including acquired user variables and other math variables. The mathematical formula can use all the variables in this list as operands.

10 – Formula parser error message

As the math formula is being typed, the errors provided by the formula parser are displayed here. None of these errors are critical to Cockpit operation; they serve as guidance to creating a math expression that can be properly interpreted and used to evaluate the new Math Variable.

11 – Save variable button

Clicking on *Save variable* will attempt to save the Math Variable with the currently entered parameters. If successful, the window will close, and the saved variable will be displayed in the selected Scope plot area.

Pressing the *Enter* key while the focus is on the formula field is a shortcut to clicking *Save variable*.

12 – Cancel button

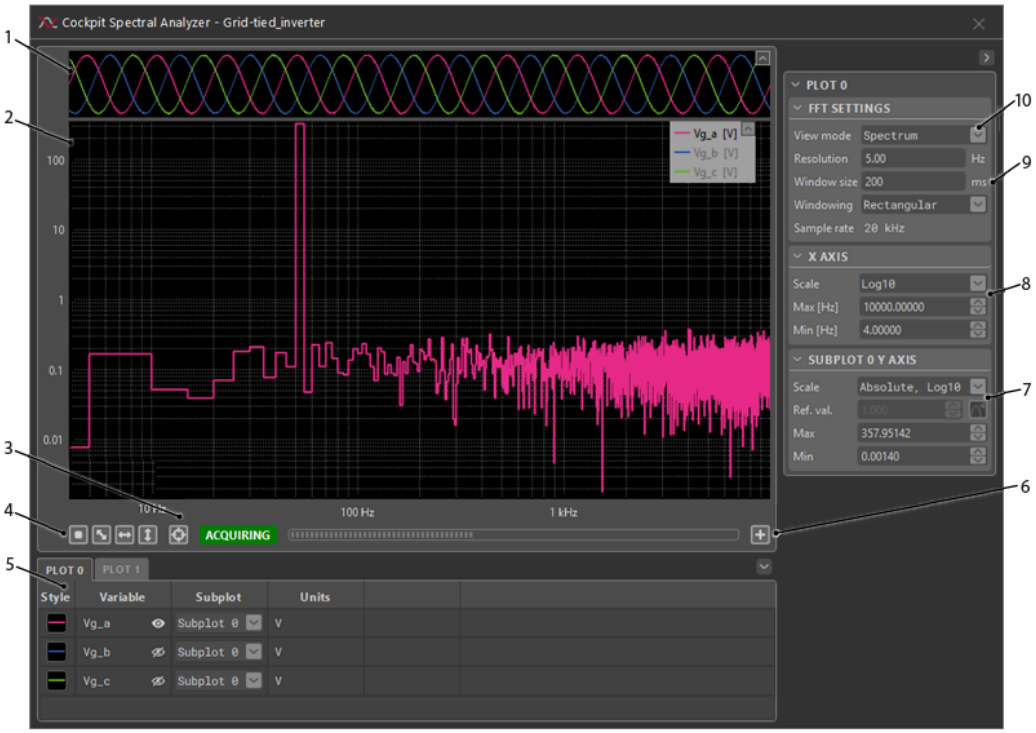
Clicking on *Cancel* will close the Formula Builder window without saving anything.

Pressing the *Esc* key while the focus is anywhere in Cockpit is a shortcut to clicking *Cancel*.

Cockpit Spectral Analyzer

The Spectral Analyzer allows the user to examine Scope variables in the frequency domain. This includes all of the user and math variables that are currently present in the module. To add a variable to the Spectral Analyzer, add it to the Scope module and it will show up automatically. Conversely, removing a variable from the Scope will make it unavailable in the Spectral Analyzer as well.

The signal spectra are calculated using a variation of the Fast Fourier Transform (FFT) algorithm. Similarly to math variables, the calculations are performed on the PC side, after the acquisition of a scope window. The displayed frequency domain signal is the **magnitude of the one-sided spectrum** of the scoped variable, with the frequency range $f \in [0, \frac{f_s}{2}]$. The Scope sampling rate, f_s, corresponds to the control task frequency defined in the user code. The frequency resolution of the resulting spectra depends on the window in time over which the Fourier transform is performed and can be adjusted in the window's accompanying menus.



1 – Time domain signal preview

This area displays a miniature version of the signals displayed in the Scope module. The main difference compared to the Scope plot preview is that only variables from the currently selected plot are shown here, as opposed to all of the signals added to the Scope. This preview can be used to visually adjust the size and position of the window in time over which the Fourier transform is performed.

Zooming with the mouse scroll wheel sets the size of the Fourier Transform window, while dragging the window with the mouse pointer can set its position within the acquisition window.

2 – Plot

The plot shows the scoped signals in the frequency domain. When accessing the Spectral Analyzer from the Scope, typically only one spectrum will be shown. The hide/show options in the bottom bar control the visibility of all of the spectra displayed for a given Scope plot area.

3 – Annotation mode toggle

When switched on, an annotation box for the spectral value closest to the mouse cursor is shown. The annotation contains the x and y axis position corresponding to the closest sample and the name of the variable to whom the sample belongs to.

4 – Start/stop button

Starts or stops the acquisition of the Scope. The button in the Spectral Analyzer and the start/stop button in the Scope module always have the same state. The same

holds for the Acquisition state and the Acquisition loading bar.

5 – Bottom bar

The scoped variables for a given Scope plot area are displayed here. To switch between Scope plots areas, click on the tabs in the header of the bottom bar. All of the settings in the Spectral Analyzer are done separately for each of the Scope plot areas.

6 – Add subplot button

Creates an empty plot area at the bottom of the scope module. The area is labeled as Subplot so as not to be confused with the Scope plot areas that are used for sorting the spectral signals into tabs.

7 – Y-axis scaling options

Besides the usual fields that allow the precise definition of the displayed y-axis range, the Spectral Analyzer offers the option to switch between different scaling settings for the value axis:

- *'Absolute, linear'*, which displays the magnitudes of the calculated Fourier transform, without any visual or mathematical rescaling
- *'Absolute, log10'*, which arranges the magnitude values in a log-scale
- *'Relative, dB'*, which transforms the magnitude values according to the Amplitude dB formula: $20\log_{10}(\frac{|X|}{X_{ref}})$, where $|X|$ is the magnitude calculated by the Fourier transform for any given $f \in [0, \frac{f_s}{2}]$
- *'Relative, log10'*, which divides all of the magnitude values with X_{ref} and arranges the resulting values in a log-scale

For the scaling options that require it, X_{ref} can be set manually in the corresponding field below. If the continuous normalization option is toggled on, X_{ref} will be set automatically to the biggest magnitude value seen since the option was turned on.

8 – X-axis scaling options

Besides the usual fields that allow the precise definition of the displayed x-axis range, the Spectral Analyzer offers the option to switch between linear and log-scaling for the frequency axis.

9 – FFT window settings

The results of the Fourier Transform depend on the size and and position of the window in time over which its performed. While the position is adjusted through the plot preview, the window size can be set more precisely through this menu, either directly or by setting the desired frequency resolution. The typical windowing

functions (Rectangular, Hann, Hamming, Blackman, Blackman-Harris and Flat top) are provided and can be applied to the samples within the window before transform.

10 – Spectrum view mode

Two aspects of viewing the same spectral content are provided:

- *Spectrum mode*, where the default FFT window is the same as the acquisition window. This is to give the best overview of the spectral content of the scoped signals, since the widest window results in the highest frequency resolution.
- *Harmonics mode*, where the default FFT window is such that the frequency resolution is 50 Hz. This allows us to visualize the spectral components in a manner typical for the Power Electronics domain. The Total Harmonic Distortion (THD) and Weighted THD (WTHD) metrics are calculated and updated in the bottom bar menu for the variables that are currently displayed in the plot.

Here are some shortcuts for the scope module:

- To **add multiple variables to a plot**, open the user variable section in the project pane.
  Keep the ctrl key pressed and click on the desired variables to select them. Alternatively, click on the first variable to select it, keep the Shift key pressed, and click on the last variable to select. These selected variables can then be dragged and dropped into a plot all at once.
- To **zoom in and out along the horizontal axis**, place the mouse cursor where to zoom. Then, use the mouse wheel to zoom in or out around the location of the mouse cursor.
- To zoom in and out along the vertical axis, place the mouse cursor where to zoom. Then press the ctrl key and use the mouse wheel to zoom in or out around the location of the mouse cursor.
- To **zoom on a specific area**, click and drag to draw a blue rectangle over the zoom area.
- To achieve a **horizontal autoscale**, right-click and drag horizontally. A light grey horizontal strip will appear. Release the mouse button to perform the horizontal autoscale.
- To achieve a **vertical autoscale**, right-click and drag vertically. A light grey vertical strip will appear. Release the mouse button to perform the vertical autoscale.
- Many of the Scope functionalities can also be accessed through **context menus** by right-clicking on a plotted variable or on the empty space in the plots.

Scope Variable context menu and plot area context menu

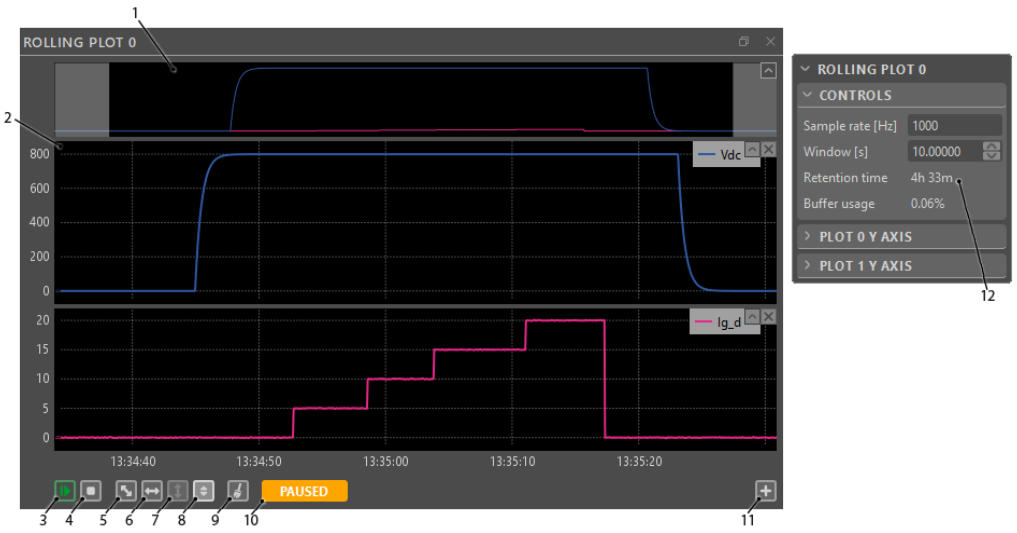
Rolling plot module

The rolling plot module allows for more long-term monitoring of the selected variables. A typical use case is monitoring the long-term evolution of the converter state in order to keep an eye on critical variables.

The sampling frequency of the Rolling plot can range from 10Hz up to the CPU control task frequency. The maximal amount of the recorded data depends on this value and the number of acquired variables. Once the allocated memory buffer fills up, the oldest acquired points will be deleted.

The buffer allocated to each instance of the Rolling plot module is set at 500Mb and corresponds directly to the memory occupied by the acquired data in the PC running Cockpit. Should this be insufficient, simply instantiate another Rolling plot, since their buffers are independent.

Rolling plot module interface



Rolling plot module

1 – **Plot preview**

The plot preview shows an overview of all of the monitored signals. The vertical axis of the preview is automatically scaled so all of the signals fit the plot. The plot preview can be collapsed to optimize space when using Cockpit on a small monitor.

2 – **Plot**

This is the area where the monitored signals are displayed. User variables can be dragged and dropped from the project pane into a plot to monitor them. If multiple plots are created, their x-axis will always remain synchronized, including when zooming. The user can remove a variable from a plot from the *Rolling Plot* tab of the bottom bar or through the variable plot context menu accessed by right-clicking on the plotted signal.

3 – **Pause/Resume** button

Pauses/resumes the rolling of the plot window. Data acquisition will continue in the background regardless of whether the plot window is rolling or not.

4 – Stop acquisition button

Stops the data acquisition of the rolling plot. Clicking on the resume button from the stopped state will restart the acquisition.

5 – **Vertical and horizontal autoscale button**

Autoscales both the horizontal and vertical axis, ensuring the acquired signals fit in the plots.

6 – **Horizontal autoscale button**

Autoscales the horizontal axis of every plot while keeping the vertical axis unchanged. Additionally, if the monitoring is paused, it will be automatically resumed.

7 – **Vertical autoscale button**

Autoscales the vertical axis of every plot while keeping the horizontal axis unchanged.

8 – **Continuous vertical autoscale** switch

When switched on, this feature continuously auto-scales every plot, ensuring that the monitored variables never go out of scope.

9 – Clear history button

Clears all of the data acquired by the rolling plot. If the Rolling plot is acquiring, the acquisition will continue.

10 – Rolling plot status

Displays the current rolling plot state. The possible states are:

- *Offline*: the target is disconnected from the host computer or the user code is not running on the target.
- *Stopped*: the acquisition is stopped. The data that what was acquired before the acquisition was stopped remains on display .
- *Paused:* the rolling plot window is not rolling with the acquisition.
- *Live:* the rolling plot window is rolling with newly acquired data.

11 – **Add plot** button

Creates an empty plot area at the bottom of the rolling plot module.

12 – Rolling plot settings

The sampling frequency of the Rolling plot can be set through this menu. Since the lower sampling rates are achieved by downsampling, the sampling rate may be rounded to the closest value which the control task frequency is divisible by.

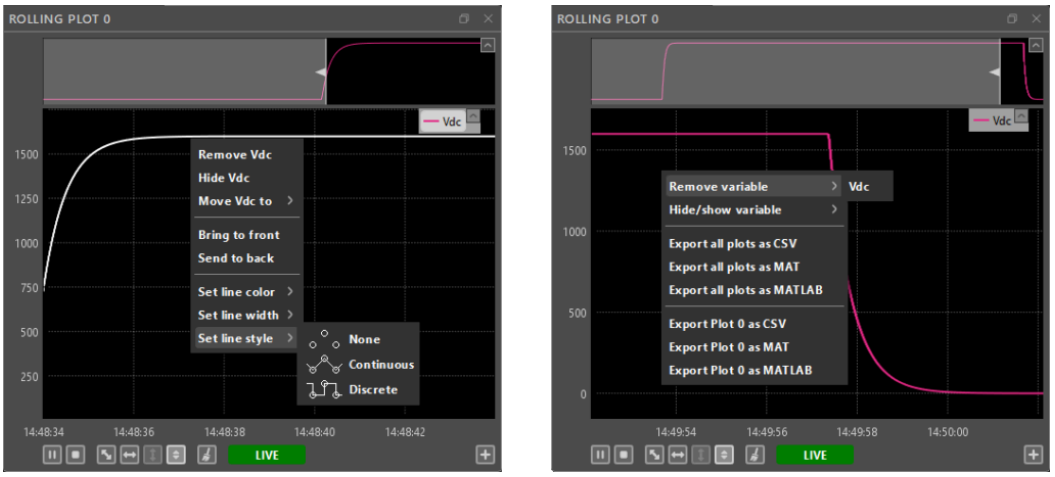
The maximal length of the recorded data in time is displayed in the ‘Retention time’ field. The state of the memory buffer can be tracked through the ‘Buffer usage’ field.

Starting from Cockpit version 2024.3, the Rolling Plot module acquisition procedure was upgraded from an approach of sampling at 20Hz in a “best-effort” manner with no strict timing guarantees to a synchronized sampling of all variables up to the control task frequency.

Rolling plot module shortcuts

- To **add multiple variables to a plot**, open the user variable section of the project pane. Keep the ctrl key pressed and click on the desired variables to select them. Alternatively, click on the first variable to select, keep the Shift key pressed, and click on the last variable to select. These selected variables can then be dragged and dropped into a plot all at once.
- To **zoom in and out along the horizontal axis**, place the mouse cursor where to zoom. Then, use the mouse wheel to zoom in or out around the mouse cursor.
- To **zoom in and out along the vertical axis**, place the mouse cursor where to zoom. Then, press the ctrl key and use the mouse wheel to zoom in or out around the mouse cursor.
- To **zoom on a specific area**, click and drag to draw a blue rectangle over the zoom area.

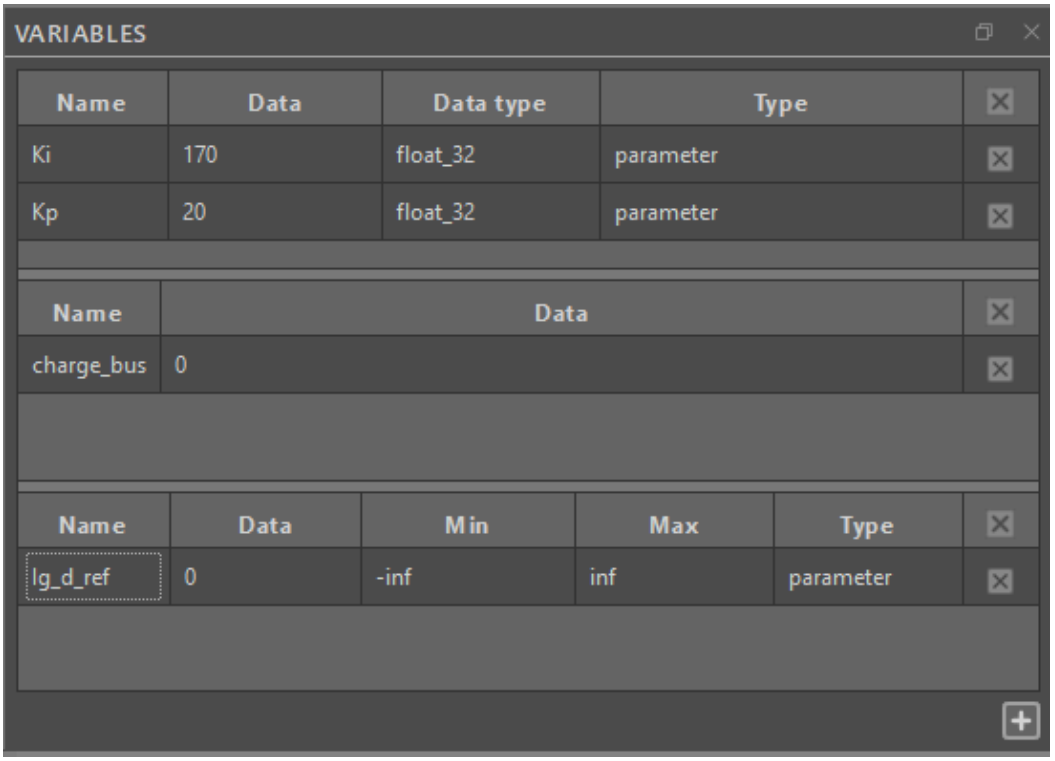
- To achieve a **horizontal autoscale**, right-click and drag horizontally. A light grey horizontal strip will appear. Release the mouse button to perform the horizontal autoscale.
- To achieve a vertical autoscale, right-click and drag vertically. A light grey vertical strip will appear. Release the mouse button to perform the vertical autoscale.
- Many of the Rolling Plot functionalities can also be accessed through **context menus** by right-clicking on a plotted variable or on the empty space in the plots.



Rolling Plot Variable context menu and plot area context menu

Variables module

The Variables module gives quick access to selected variables. It also allows modifying the user variables without rebuilding the control code. Additionally, it displays extra information on user variables, such as their minimum and maximum values.



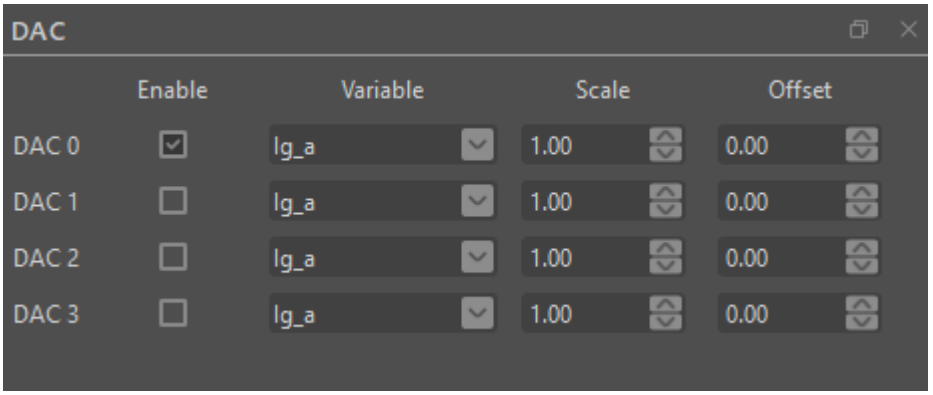
Variables module

This module can contain one or more sections to sort the user variables conveniently.

- To add a new section, click the + button on the bottom left of the module.
- The columns displayed in a section can be managed by right-clicking on the section's header and checking the desired columns.
- To add a variable to a section, drag and drop a variable from the project pane directly into the desired section.
- To remove a variable from a section, right-click on the variable and select "remove variable".
- Signals connected to a Probe block are listed under the *probe* type, and signals connected to a Tunable parameter block are listed as *parameter*. Variables of type *parameter* can be modified in run-time by double-clicking on their value (*Data* column).

DAC module

The DAC module allows the user to apply a given variable to one of the four analog outputs of the B-Box RCP. The values applied to the analog output are specified in volts. The output range is -5V to 5V, and the output is saturated beyond these limits. Further hardware specifications are available in the B-Box RCP datasheet.



DAC module

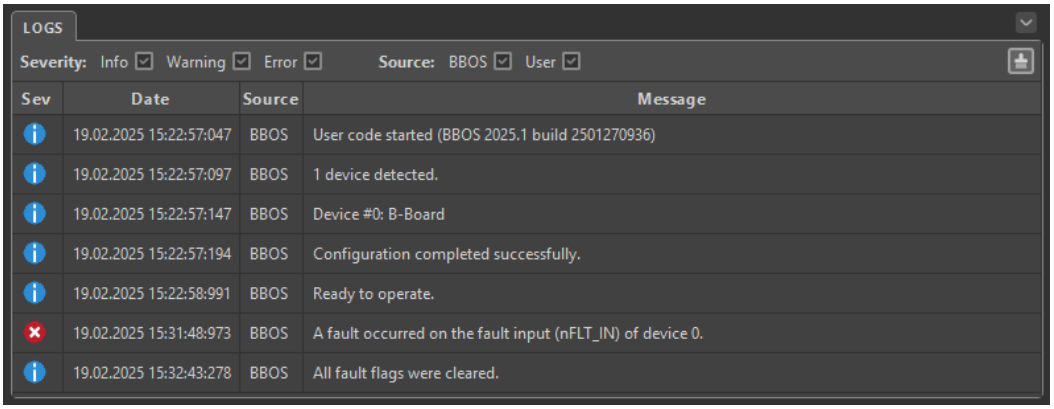
If a DAC block is used in the user code, and the DAC module is also used in Cockpit, the DAC module in Cockpit will have the upper hand.

GUI Builder module

The GUI Builder module enables users to create custom dashboards for interacting with their user code running on an imperix controller.
Read the GUI Builder module page for a more in-depth guide.

Logs tab

The logs tab displays every message reported by the controllers, including useful information such as misconfiguration details, software and hardware faults, or custom user messages.



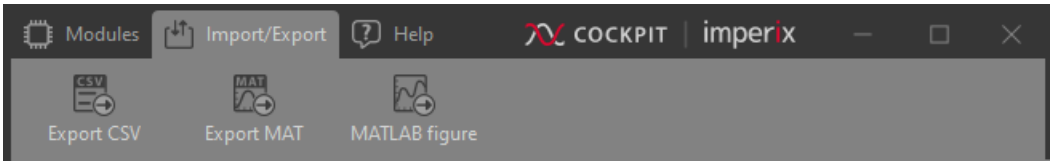
Logs tab

Export data as a file or MATLAB figure

Cockpit offers the possibility to export the plotted signals acquired with the Scope and Rolling plot modules as a CSV or MAT file, or directly as a MATLAB figure.

For the scope module, Cockpit will export each and every sample of the scoped signals. As a reminder, the signal acquisition is performed at the control task rate. For the rolling plot module, Cockpit will also export each and every sample of the acquired signals. In this case, the signals are acquired at 20 Hz. However, this is a best effort, and the data may be acquired at lower rates. Therefore, it is common to observe timestamps in the exported CSV files that are not at 20 Hz.

To export data from a rolling plot or a scope module, click on the Export CSV, Export MAT, or MATLAB figure button located in the Import/Export tab of Cockpit's top bar, as displayed below.



Import/Export menu of Cockpit's top bar

From the menu that appears, select the desired plots to export and choose the save location MATLAB figure or the CSV/MAT file.

A MAT file is also saved when exporting data as a MATLAB figure.

Exporting as MAT file

Exporting as MAT file or MATLAB figure is done through the MAT 7.3 file format. This format is based on the HDF5 standard for hierarchical storing of data, allowing for partial saving and loading and better compression. This makes the export process more efficient in terms of time and memory in most cases, compared to CSV files.

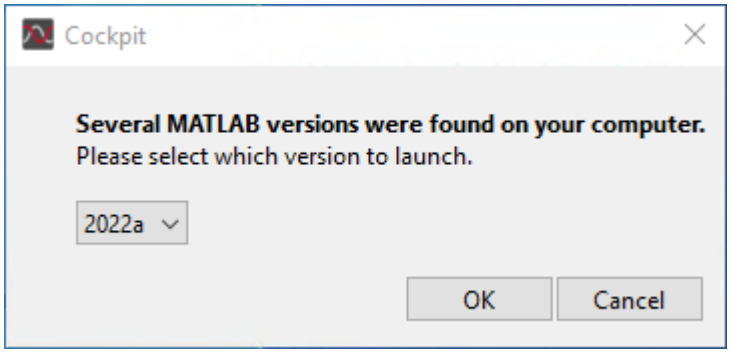
A MAT file can be easily loaded to a MATLAB workspace by finding the file through the "Current folder" menu in MATLAB and double-clicking on it. To learn more about the structure of the MAT files we export and different ways to load and save them, read Working with MAT files exported from Cockpit.

Exporting as MATLAB figure

Exporting as MATLAB figure will first create and save a MAT file, as described in the previous section. Once the data has been exported as a MAT file, MATLAB is automatically launched with a new workspace set to the chosen folder. Finally, the MATLAB figure is automatically displayed.

If several instances of MATLAB are installed on the computer, the user will be prompted to select the desired MATLAB version with the following pop-up message.

The export as *MATLAB figure* feature only supports MATLAB versions R2019a and newer.



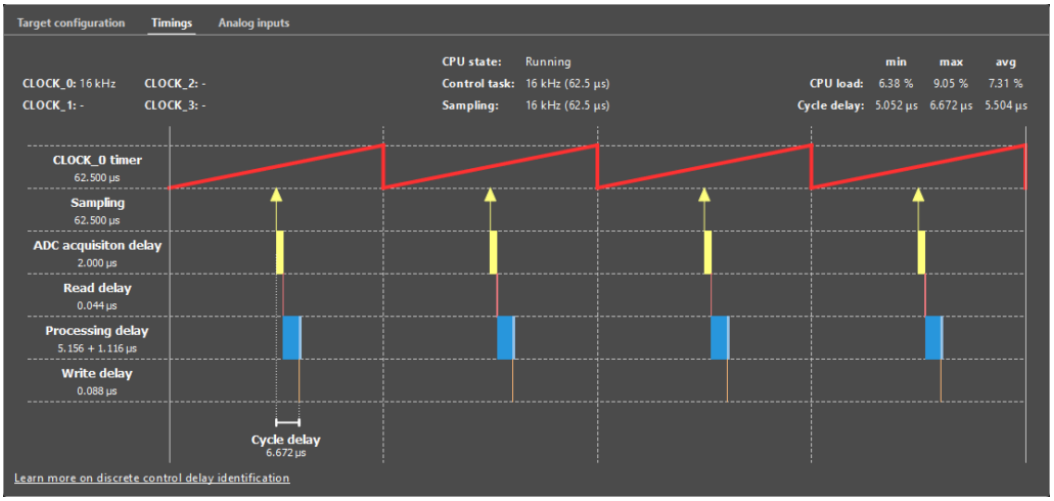
Popup message when multiple MATLAB versions are detected

The export process can take up to several minutes when exporting data from a Rolling plot module, especially if the data spans over a prolonged period of time. Moreover, it is also possible that when exporting this data as a Matlab figure, the sheer amount of data could be too large for the available memory of MATLAB. Thus, it is not excluded that MATLAB could crash. In such cases, partial loading of the saved file might help.

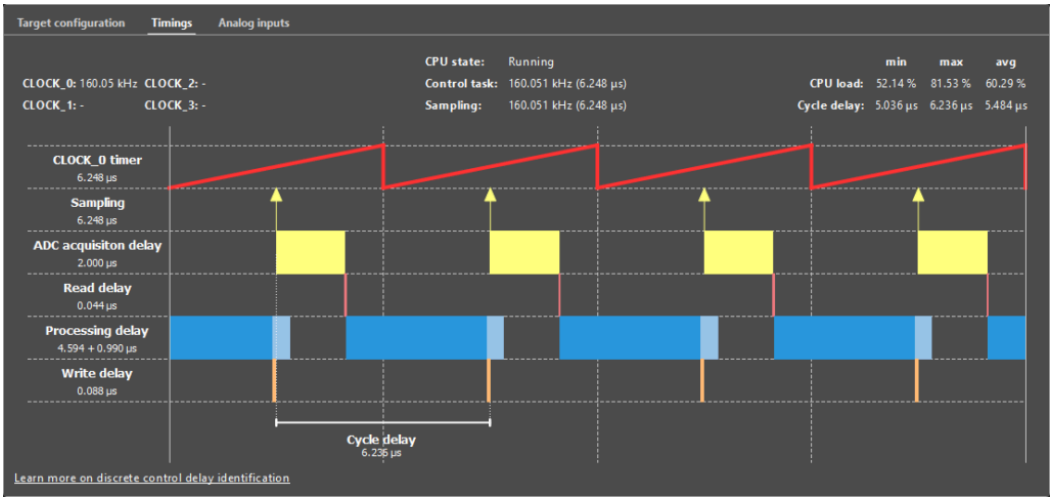
Target timings

The Timings tab, accessible from the three dots menu of a project pane, provides a graphical representation of the various computation and communication delays involved in the controller during run-time. It is particularly useful to observe the delays involved in the control dynamics of the system as explained in Identifying the discrete control delay (PN142). The control parameters, such as the K_p and K_i of a PI controller, can then be adjusted accordingly, as shown in Basic PI control implementation (TN105).

Below are screenshots of the Timings tab when running the model Central PV inverter (AN006) in two scenarios.



Central PV inverter running at 16 kHz



Central PV inverter running at 160 kHz

The timing graph accurately represents the delays involved in the execution of the user control code.

- **CLOCK_0 timer** represents the clock generator counter. CLOCK_0 is used as the time base for the sampling events and the control task routine execution. In most cases, the PWM modulators are also based on CLOCK_0.
- **Sampling** shows the sampling events, i.e. the instants where the ADCs sample the input analog signals. In the example above, the sampling phase is set to 0.5, so the sampling occurs in the middle of the period of the PWM signals that are based on CLOCK_0. Multiple sampling events per control period can be configured using oversampling, as explained in Oversampling (PN154).
- **ADC acquisition delay** shows the delay between a sampling event and the availability of the values in the FPGA. It comprises the ADC acquisition delay and the transfer time of the read value to the FPGA. This value is 2000 ns in the B-Box RCP and 500 ns in the B-Box Micro and B-Board PRO. The appropriate value must be set from the CONFIG block.
- The **Read delay** shows the time needed to perform *FPGA-to-CPU transfers*. These tasks are executed right after the ADC results are available in the FPGA. The values transferred are typically the ADC measurements, the GPI values, or the angle decoder output.
- The **Processing delay** shows the time the CPU spends executing the interrupt routine. To execute the *CPU-to-FPGA transfers* as early as possible (and reduce the overall response delay), the interrupt routine is separated into two phases:
    - The *control task execution* is the time necessary to execute the user code and update the modulation parameters and other FPGA values. The *CPU-to-FPGA transfers* are executed right after, in parallel with the *post-processing execution*.

    - The *post-processing execution* is the time necessary to perform all the tasks that are not directly involved in the control algorithm. It includes the datalogging execution, the CAN communication, etc.
- The **Write delay** shows the time needed to perform the *CPU-to-FPGA transfers*. These tasks are performed once the *control task execution* is over. The values transferred are typically the PWM modulation parameters (duty-cycle, phase), the GPO values, or the DAC values.

At the top of the tab, the following information is displayed:

- **CLOCK_0**, **CLOCK_1**, **CLOCK_2,** and **CLOCK_3** show the configured frequency of the four Clock generators.
- The **CPU state** displays the current operating state of the controller.
- The **Control task** routine execution frequency and period are also displayed. Note that the control task is always mapped on CLOCK_0.
- The **Sampling** shows the frequency and period of the ADCs sampling. It allows the user to visualize if the oversampling is configured or not.
- The **CPU load** represents how much time the CPU spends in the interrupt routine relative to the period of CLOCK_0. Safety mechanisms are implemented to detect CPU overload. An overload can result from a control algorithm being too complex or an execution frequency being too high. The min, max, and avg values are computed on a window of one second.
- The **Cycle delay** represents the delay between the sampling event and when the newly computed data are available in FPGA (*ADC acquisition delay + FPGA-to-CPU transfers + control task execution + CPU-to-FPGA transfers*). This value can be used to compute the *control delay* and the *total loop delay*, as explained in Identifying the discrete control delay (PN142). The cycle delay value is precisely measured directly from within the FPGA. The min, max, and avg values are computed on a window of one second.

Other features

How to launch a user code at controller start-up

In some cases, it can be useful to automatically start a user code when turning on the imperix controller. To do so:

- Launch the user code and connect to the target as usual.
- Once the code is running on the target, open the *Target configuration* window from the three dots menu of the project pane.

- In the section *User code saved on SD card*, click on the *Save the currently running code* button to save the current code on the SD card inside of the controller.
- Click on the checkbox *Load at startup* to automatically launch the code when the controller is powered on.

This will save the control algorithm on the SD card inside the controller and automatically load and start it from the SD card at start-up.

How to load a custom FPGA bitstream

In some cases, it is interesting to offload all or parts of the computations from the CPU to the FPGA. Doing so often results in much faster closed-loop control systems.

To do so:

- Launch the user code and connect to the target as usual.
- Once the code is running on the target, open the *Target configuration* window from the three dots menu of the project pane.
- In the section *FPGA bitstream saved on SD card*, click on the *Browse* button and select the desired bitstream. The bitstream will be uploaded into the SD card of the controller. A check in the *Load at startup* checkbox indicates that the target will load the imported customized bitstream at the next power cycle of the target instead of the standard one.

Further information on how to develop power converter control algorithms in FPGA can be found on the Getting started with FPGA control development page.