Go Configure™ Software Hub

User Guide

Rev. 2.13.0

Contents

1 Introduction

Renesas Go Configure Software Hub is a full-featured Integrated Development Environment (IDE) that enables a completely graphical design process, allowing custom circuits to be configured, programmed, and simulated in minutes.

This guide provides the information for the usage of Go Configure Software Hub, describing the features for Renesas products, such as GreenPAK™, AnalogPAK™, HVPAK™, Power GreenPAK™, Automotive GreenPAK™, and ForgeFPGA™.

1.1 Quick Start

1.1.1 System Requirements

Minimum system requirements for Go Configure Software Hub:

- CPU: Dual core 2.5 GHz.
- RAM: 8 GB.
- GPU: 128 MB.
- Free disk space: 2 GB.
- OS: Windows 10 or 11; macOS 11 or higher; Ubuntu 22.04 or 24.04; Debian 13 or Testing.

1.1.2 Getting Started

1. Download and install Go Configure Software Hub.
2. Open the particular chip part number.
3. Select the components for the project.
4. Specify the pinout.
5. Configure and interconnect the components.
6. Test the design with the **Debug Tool**, using Simulation feature or any of the supported hardware development platforms.

1.1.3 Support

Support is available through multiple channels. Click here to reach the knowledgebase, technical support team, distributor network locations, and additional technical resources. For product-specific assistance, use:

- greenpak@renesas.com for GreenPAK-related inquiries
- fpga@renesas.com for ForgeFPGA-related inquiries

Find more details and latest news on Go Configure Software Hub on Renesas social media. To visit the relevant social media, go to **Help** > **Social** in the main menu of the software instance.

The latest version of the software application is available on the Renesas website, on Software page.

Go Configure Software Hub notifies about pending software updates. More information is available in section 4.1 Software Update.

2 Software Overview

The Go Configure™ Software Hub was designed to provide direct access to supported device features and complete control over the routing and configuration options.

This chapter provides detailed explanations of various tools designed to simplify the workflow. These tools allow a chip to be programmed with a custom design, data to be imported from a programmed part, and simulations to be performed with external components. This chapter also serves as a guide to the software interface and to essential tools such as debug tools, software simulation, and the FPGA editor.

2.1 Design Tools

The Go Configure Software Hub provides an interface for project management, device selection, and access to technical documentation. The **Hub** window is organized into the following primary tabs:

- **Home** – The starting page, designed for project initialization and providing access to recent work.
  - **Open my file** and **Start new project** – Provide options to open an existing project or initiate a new project on the supported part numbers.
  - **Example projects** – Offers a list of design examples. Clicking **Show more** redirects to the **Library tab**.
  - **Recent files** – Displays a list of previously opened projects.
  - **Recovered files** – Lists the restored project files after a crash or freeze. To read more refer to section 2.1.15 Settings.

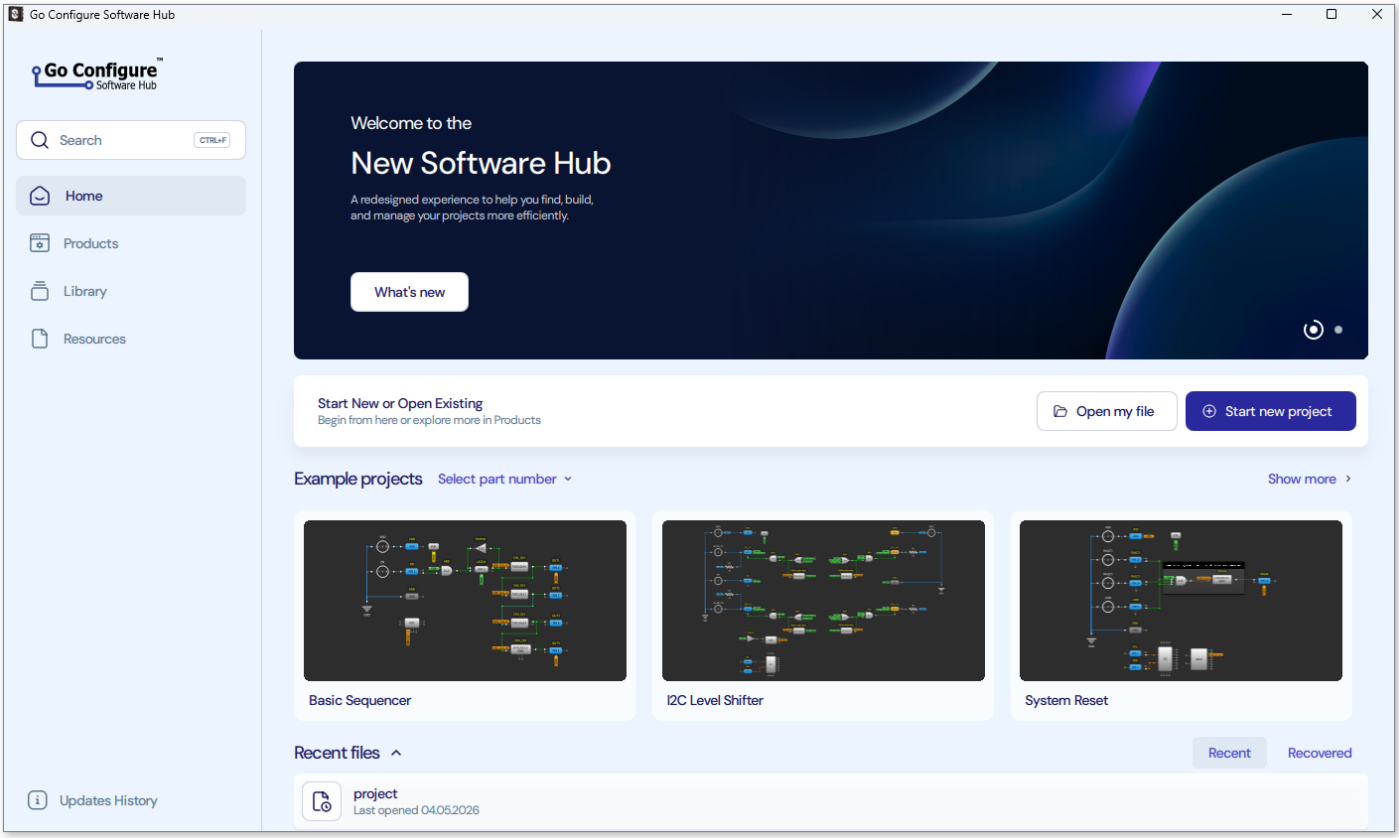


Figure 1. Home Tab of the Hub Window

- **Products** – A searchable database of all supported part numbers.
  - **Products table** – Introduces available part numbers, their key characteristics, and links to datasheets.

- **Chip information panel** – Selecting a part number provides detailed characteristics, supported platforms, and relevant technical documentation.
- **Filtering and sorting** – Use filtering by product family within the table. Click the table headers to sort it by VDD, temperature, special features.

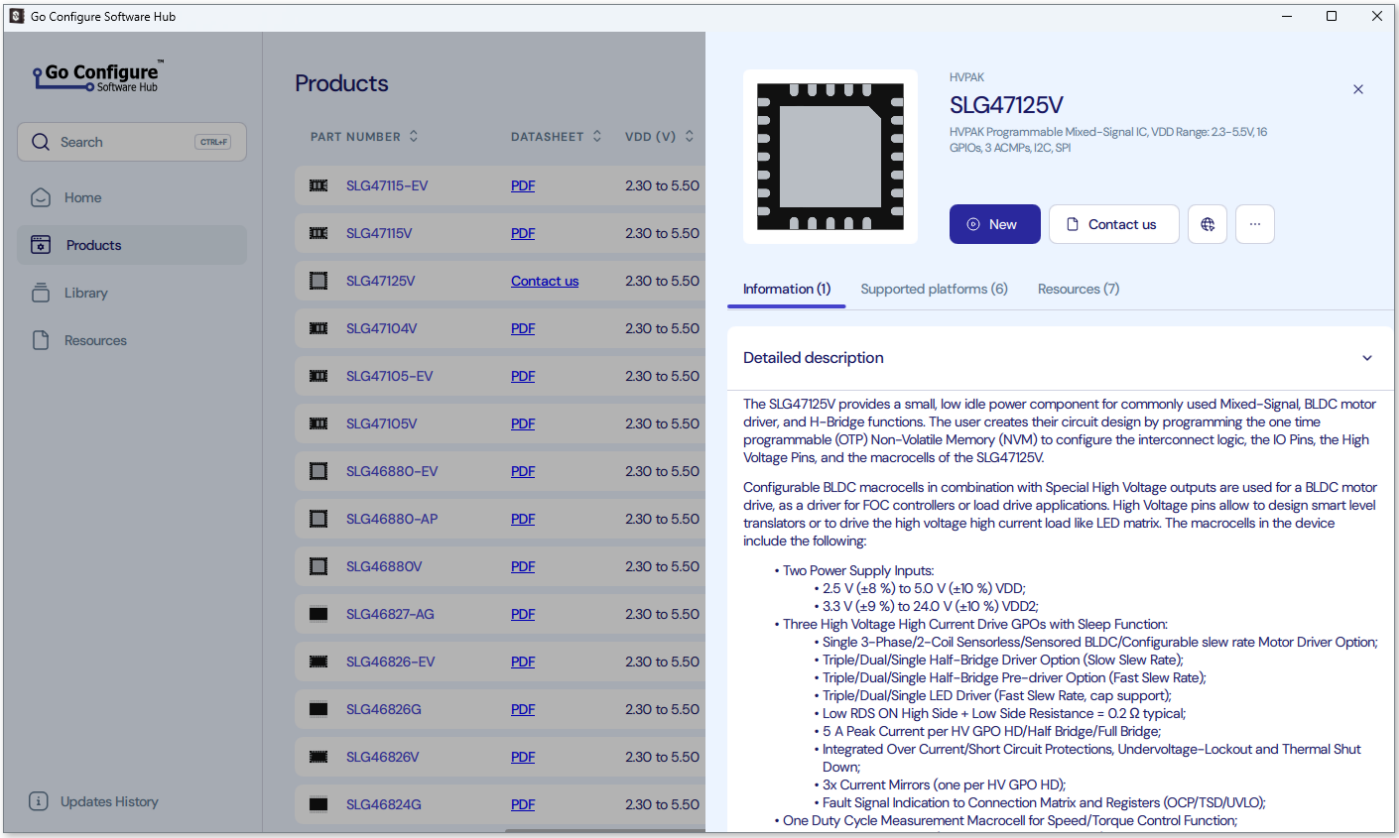


Figure 2. Products Tab of the Hub Window

- **Library** – A repository for design assets, reference designs, and debugging resources. This is essentially a collection of all top-level documents used to build the designs.
  - **Example projects** – Contains modular design blocks or partial project segments for integration into larger system designs.
  - **Application notes** – Provides complete, ready-to-use design examples, including design descriptions and preconfigured circuit projects.
  - **Demo projects** – A collection of projects configured for use with specific demonstration boards for hardware debugging and validation (read more in section 2.2.18 Demo Board and Demo Mode).

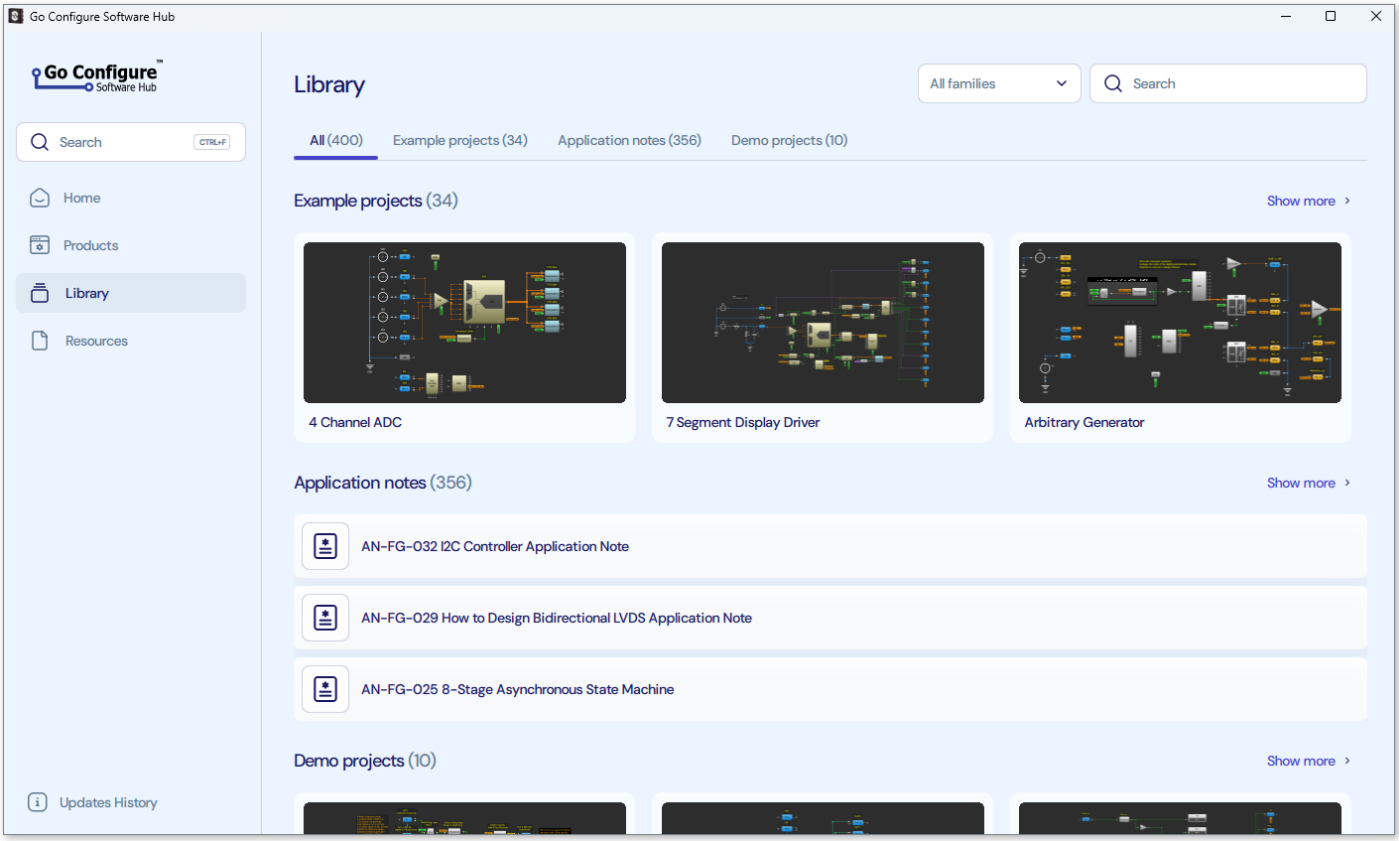


Figure 3. Library Tab of the Hub Window

- **Resources** – Space for assets, useful links, documentation, and support content, helping to find both information and contacts for assistance.
  - **Explore our products** – Directs users to the Renesas website for information on specific product lines.
  - **Key sources** – Provides access to technical blogs, white papers, and product family overviews.
  - **Support channels** – Allows to reach the knowledgebase, technical support team, distributor network locations, and additional technical resources.

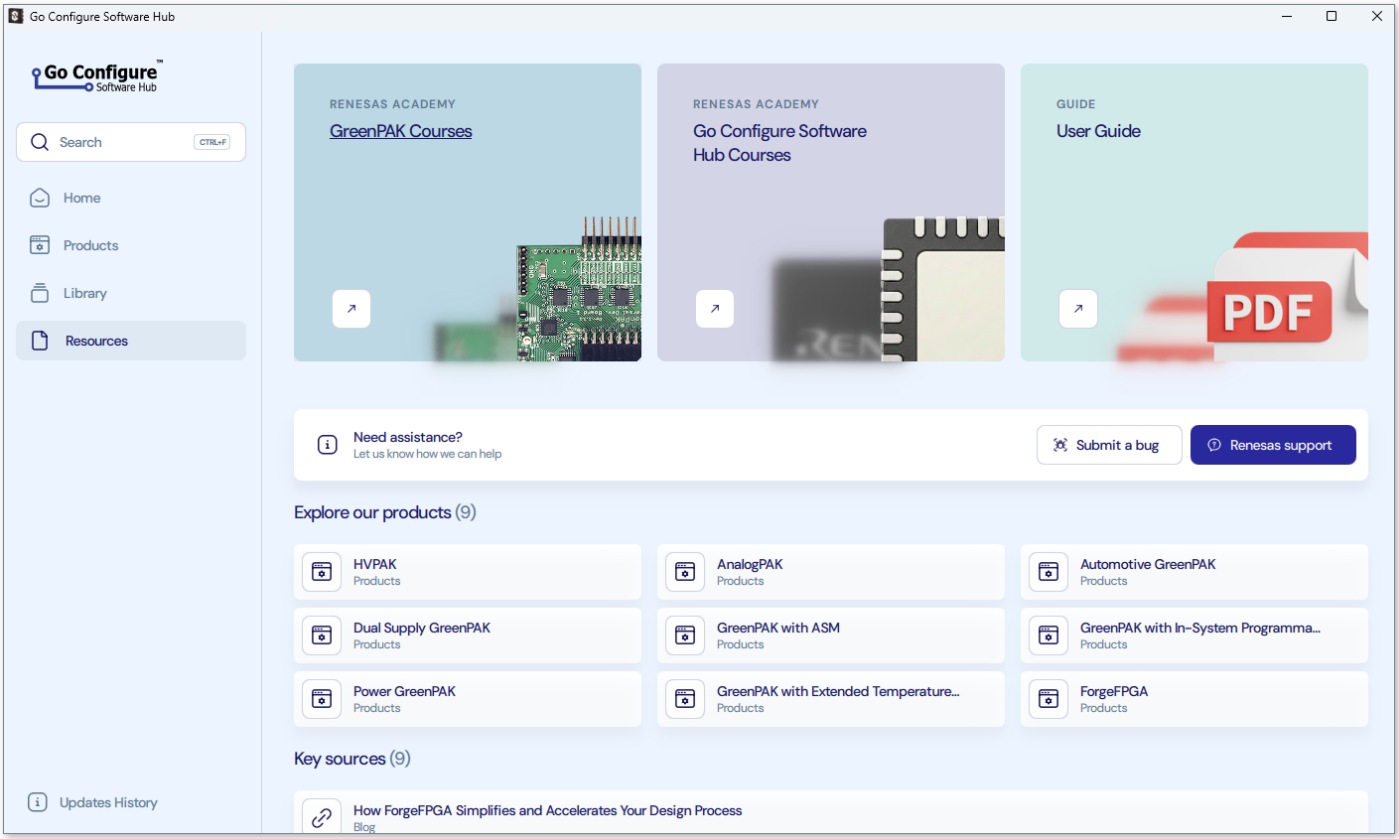


Figure 4. Resources Tab of the Hub Window

- Additional global controls.
  - **Global search** – Located in the upper-left corner, this field allows for application-wide searches across part numbers, datasheets, product families, example projects, application notes, and demo projects.

2.1.1 Interface Overview

The basic interface elements are **main menu**, **toolbar**, **work area**, **work area controls**, **properties panel**, and **component list**.

2.1.1.1 Main Menu

After the part number is selected and the project is started, the menu bar displays the following items: **File**, **Edit**, **View**, **Tools**, **Options** and **Help**. See the full description of all the menu commands in section 6.1 Main Menu Commands.

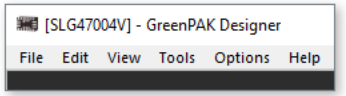


Figure 5. Main Menu

2.1.1.2 Toolbar

The toolbar items are located under the menu bar by default. Move, show, and hide toolbar items to customize the environment. Use **Settings** to reset the **Appearance**.

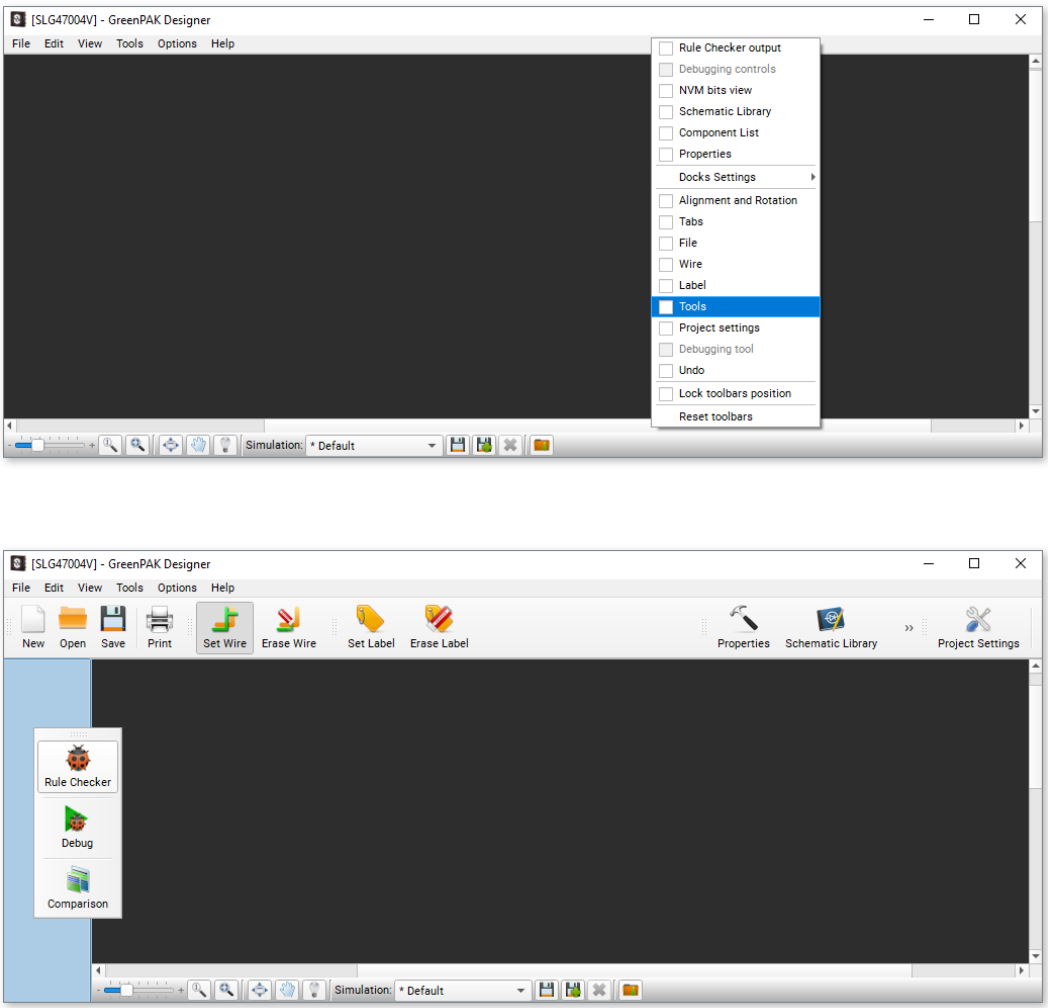


Figure 6. Toolbar Customization

2.1.1.3 Work Area

The work area displays the selected components (macrocells) from the component list and the connections between them. The part numbers contain different sets of macrocells, therefore the IDE's interface varies.

The IDE may contain one or two work areas (an example of part number with two work areas is

SLG46620V).

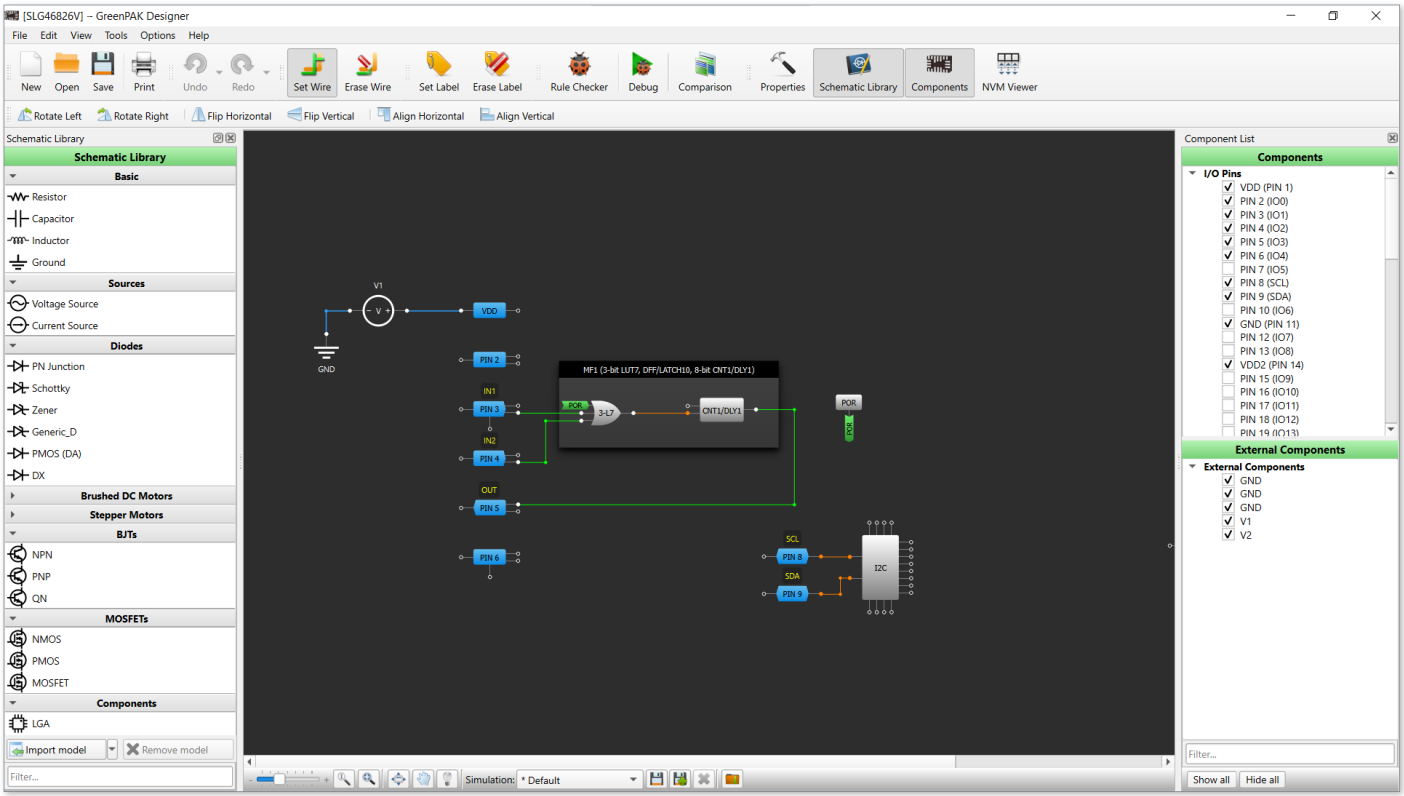


Figure 7. Part Number with One Work Area

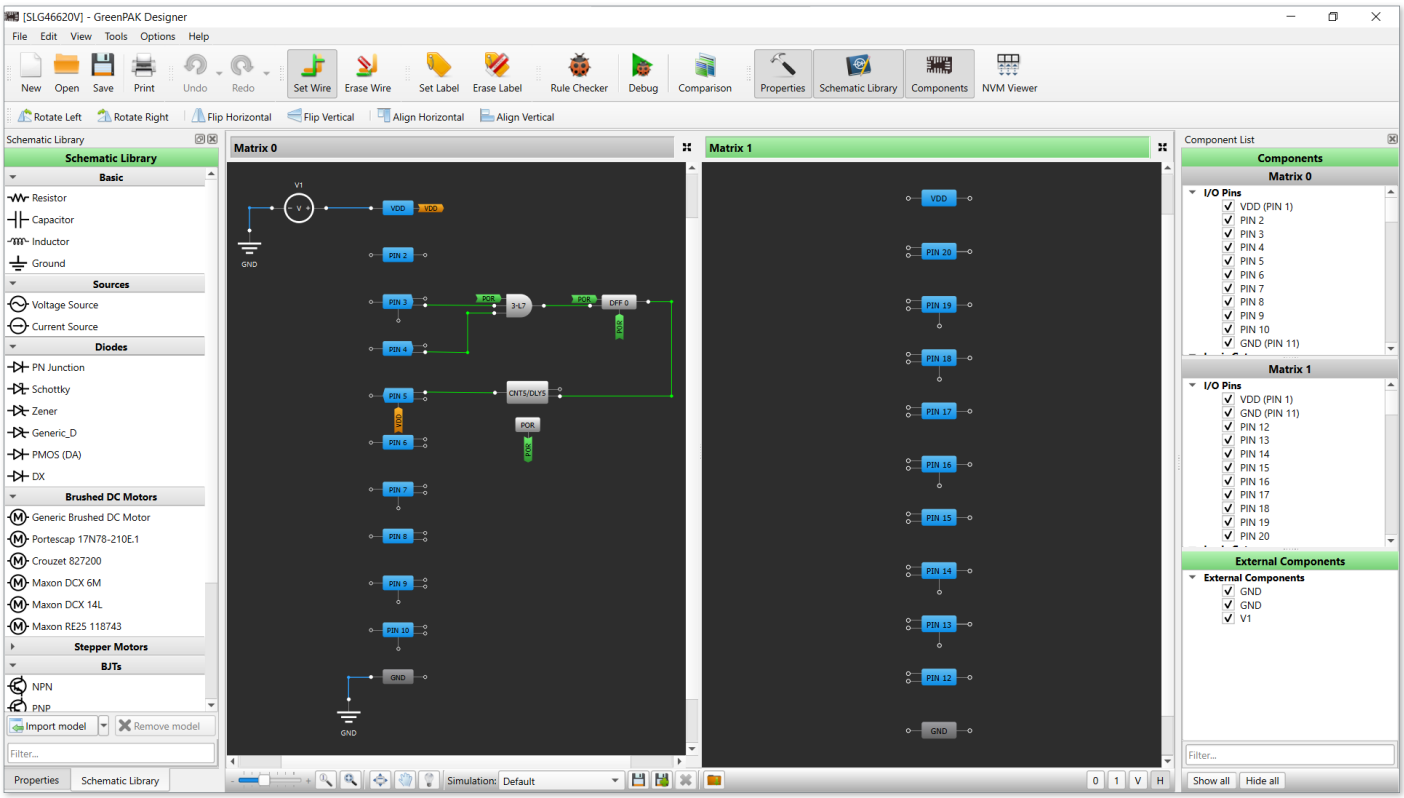


Figure 8. Part Number with Two Work Areas

The part numbers with two work areas have several distinctive buttons, which define the matrix window placement:

Figure 9. Matrices' Window Placement

- **0** – Only **Matrix 0** is displayed.
- **1** – Only **Matrix 1** is displayed.
- **V** – Both matrices are displayed vertically (on top of each other).
- **H** – Both matrices are displayed horizontally (beside each other).
- – Makes one of the matrices a separate window.

The **Matrix** title bar turns green when the work area is in focus.

2.1.1.4 Work Area Controls

The additional work area controls are located on the bottom toolbar.

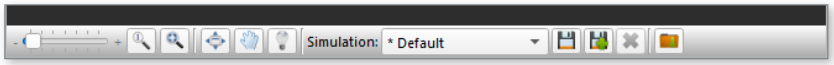


Figure 10. Work Area Controls

Adjust the work area (zoom, fit the work area or make it full-screen).

Enable the **Pan mode** to move the work area (click and hold the middle mouse button as an alternative).

Activate the possibility to see the hint box with the block's property info when hovering the mouse over a component.

2.1.1.5 Component List

This panel contains a list of available macrocells in a chip. Components in the list can be shown, hidden, or searched.

Show or hide a component by using the checkbox next to its name.

Drag and drop a block from the list to the work area.

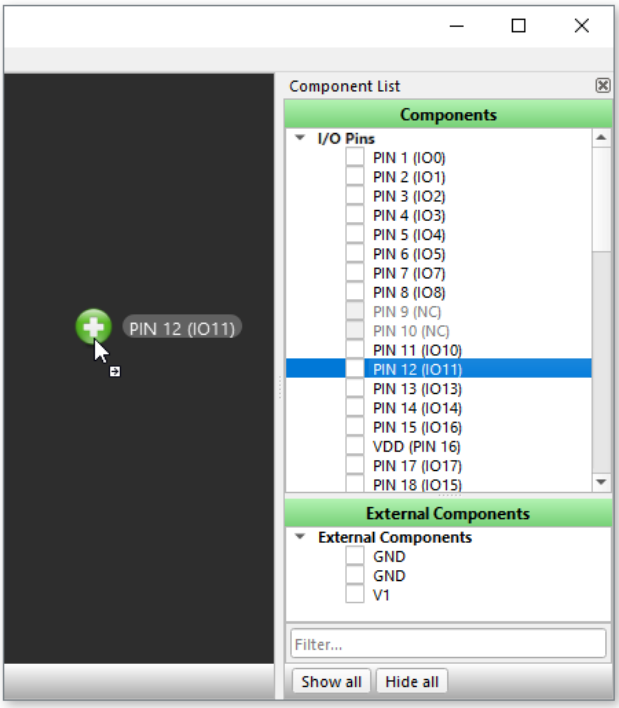


Figure 11. Checkbox. Drag and Drop

The filter can be used to search for the required component more easily. Show or hide all the components by using **Show all** and **Hide all** buttons.

Note 1: A hidden component retains the configurations set on its properties panel (see more info about the **Properties** panel).
Note 2: It is not possible to hide connected blocks (see sections 2.1.5 Components and 2.1.6 Connections to find out more).

2.1.1.6 Properties Panel

The panel contains all settings available for a selected component. The set of properties may vary depending on the macrocell.

The **Properties** panel may contain:

- Settings and parameters that can be specified for a selected block.
- Settings that control the predefined connections to a selected block.

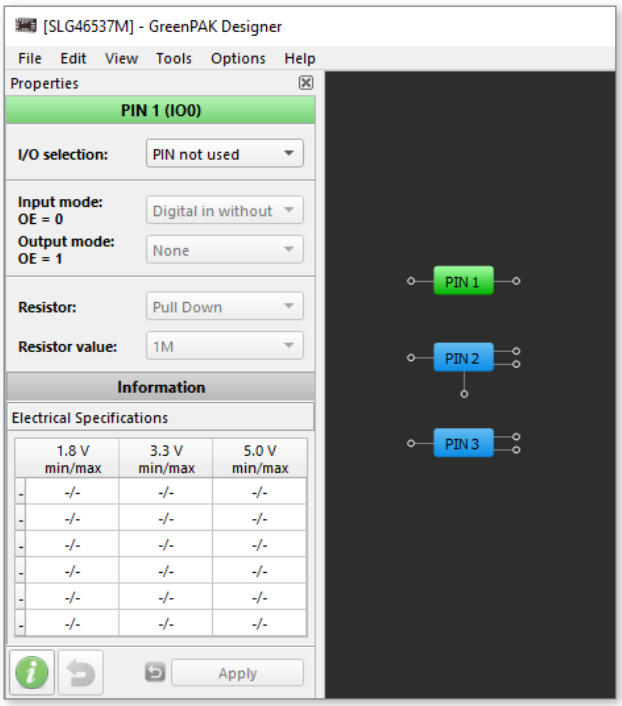


Figure 12. Properties Panel

After the property values are changed, the modifications can be **Applied** or **Reverted**.

After changes are applied, they can also be **Reset**. The following options may be available: **Reset connections to default**, **Reset settings to default** or **Reset configurations to default** (the latter is available only for the components with more than one mode, such as LUT).

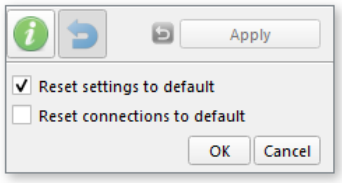


Figure 13. Property Reset Feature

The block can also be **Reset** from the component's context menu.

2.1.2 Working with Project Files

A project file is the latest saved state of the designed project. The file contains component configurations, connections between blocks, component location on the work area, and other project information

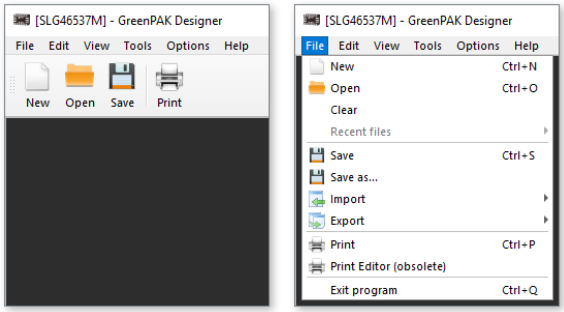


Figure 14. File Operations

2.1.2.1 Create, Save, and Open a Project

A new project can be started from the Hub window or from an opened instance (click the **New** icon on the toolbar or **File** > **New**).

Note: For an FPGA part number (for example, SLG47921), selecting it from the part number list in the Hub window opens the **New Project** window, where a project can be created. After clicking **Create**, the FPGA project directory appears in the specified location.

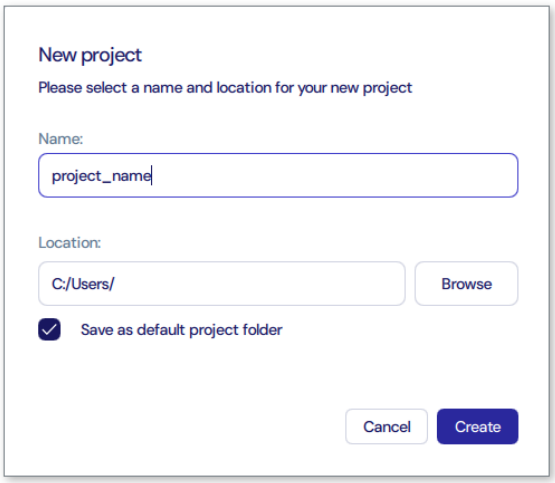


Figure 15. Window for FPGA Project Creation

After a new project is started, the **Project settings** window appears. This step can be skipped, but if the data is needed later, it will have to be entered again. Read more about **Project settings** window here.

The default project (state of the work area right after a new project is opened) is usually configured for minimal power consumption. That is why some components are disabled.

To save the project, click the **Save** icon on the toolbar, or click **File** > **Save/Save as**.

Change the folder to which the project is saved in **Settings**.

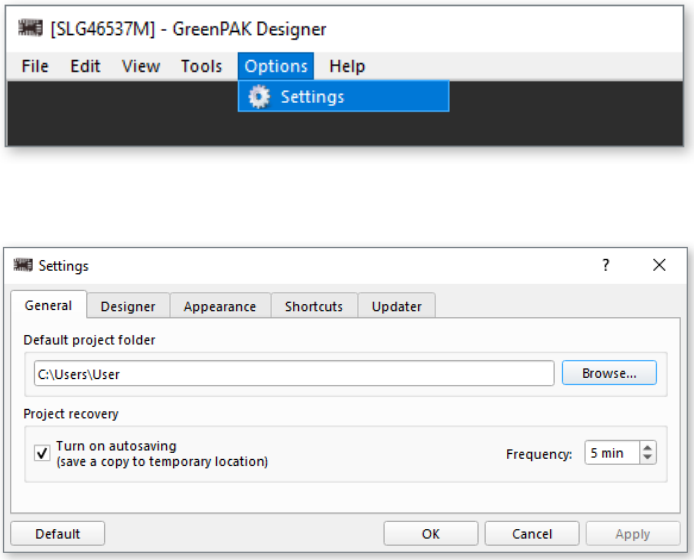


Figure 16. Project File Folder

The supported project file extensions are: .aap, .can, .gp3, .gp4, .gp5, .gp6, .hvp, .ppak, and .ffpga (.gp1, .gp2, .gpp, and .ldo are the obsolete file extensions).

There are several ways to open a project file:

- From the Hub window.
- **Open** icon on the toolbar (or **File** > **Open**).
- Drag and drop the project file to the Hub > **Products**.
- Drag and drop the project file to the work area.
- Double-click the project file.

Note: Manual changes to the file are strongly discouraged, as they may corrupt it.

2.1.3 Project Settings Window

The **Project settings** window appears right after a new project is started. It contains the following tabs: **Specs**, **Information**, **General**, and **Security**.

2.1.3.1 Specs

In **Specs**, the fields for entering chip operating conditions, such as VDD and Temperature (Min., Typ. and Max), are available. The part numbers may have one or multiple VDDs. Click the **information** button to see the recommended range within which the chip can operate safely.

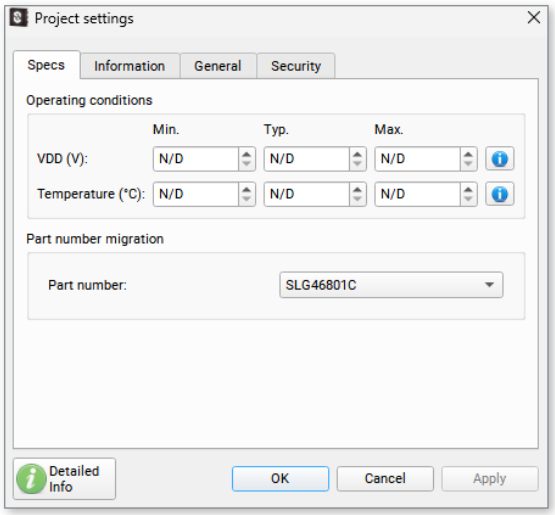


Figure 17. Project Settings Window Examples

Note: The **Project settings** window can be closed without entering the operating conditions. Working with **Simulation** or hardware development platforms in the **Debug tool** requires entering the specs. If **Debug** is clicked on the toolbar while specs are not specified, a warning notification prompting for specs appears. Click the **OK** button on the warning window or the **Project Settings** icon on the toolbar to enter specs.

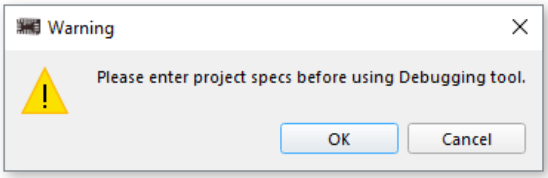


Figure 18. Warning to Add Specs to Use Debug Tool

In case the improper operating conditions are filled in, a warning with the corresponding text message will appear. Some warning notifications do not stop the specs from being applied, while

others require the issues to be fixed first. Here are the warning examples:

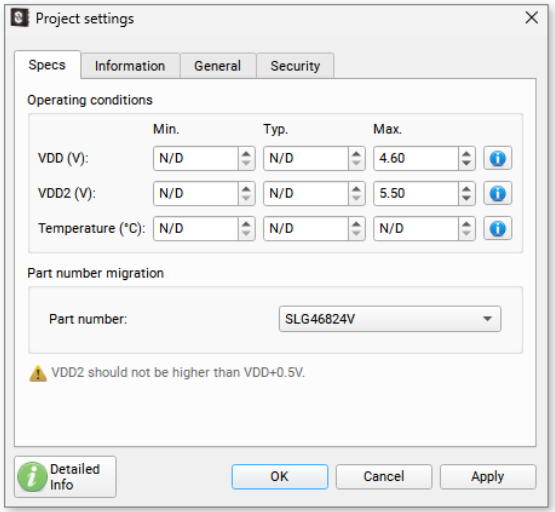


Figure 19. Specs-Related Warning Examples

If, for example, Min. value is higher than Typ., and in other similar cases, the affected fields become highlighted in red. The specs cannot be applied until the issue is fixed.

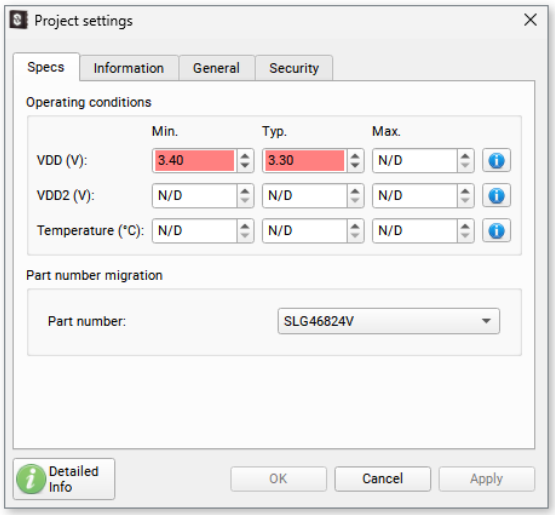


Figure 20. Improper Specs Values

Part Number Migration

In the **Specs** tab, the project can also be migrated to a different chip part number. The part number migration feature allows the design to be transferred between compatible part numbers without starting a new project. The software automatically migrates the configuration data, resources, and settings after part number switch.

Note: If the target device does not provide sufficient resources or lacks features required by the

existing design, the migration proceeds, and an appropriate warning is displayed.

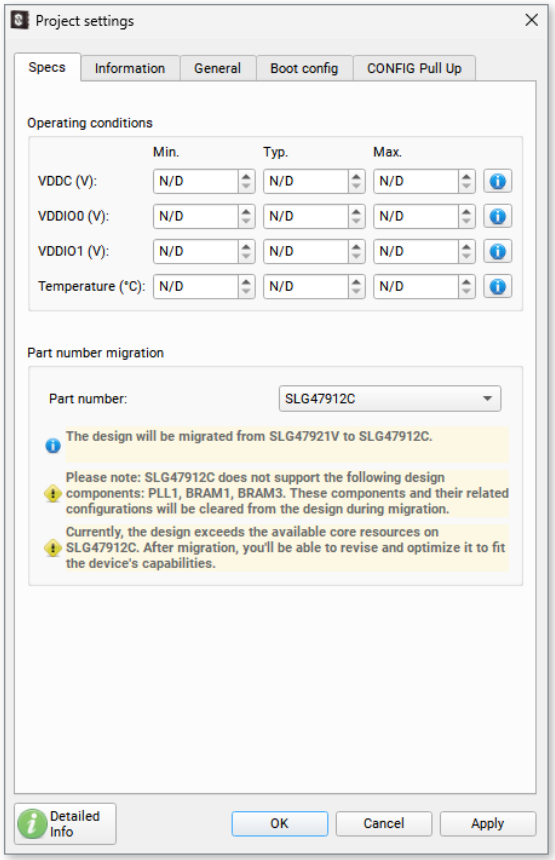


Figure 21. Part Number Migration

2.1.3.2 Information

The **Information** tab contains a short piece of information about the part number. To view the information, click the **Info** button.

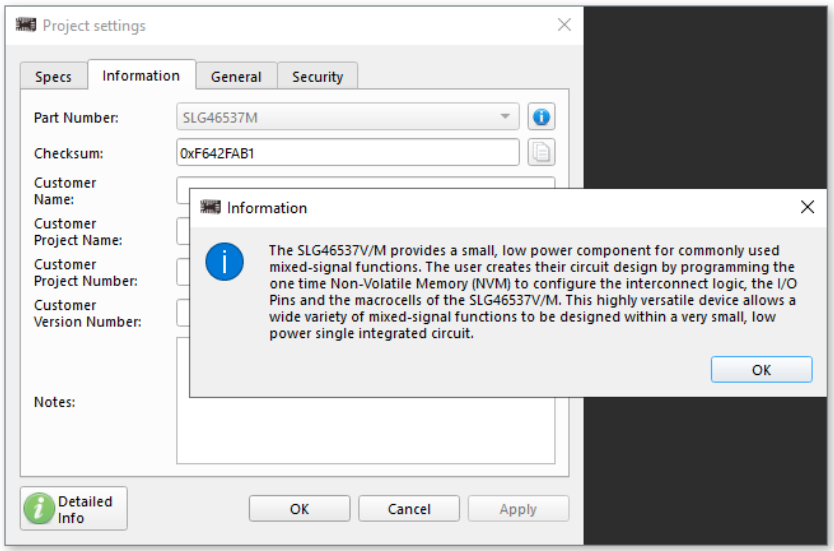


Figure 22. Information Tab

The checksum is a unique project identifier that can be used to distinguish between the projects. The **Checksum** field value is generated based on the project bits state in NVM. The project bits are chip configurations, such as macrocell settings, connectivity matrix and the project settings.

For FPGA part numbers (for instance, SLG47910), the OTP registers [0:191] are ignored during calculation. The FPGA core configuration bitstream is also taken into account when computing the checksum.

Note: The same project may show different checksums when opened in different software versions. The discrepancy occurs because the set of project bits used to compute the checksum, may vary between software versions. In this case, the warning will appear when the project loads. The checksum will be updated in the project file during the next save.

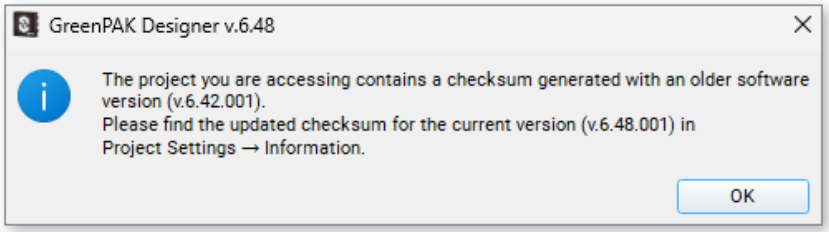


Figure 23. Project Checksum Update Warning

Input fields for entering project information are also available. The entry in the **Customer project**

name field is also displayed in **Print Preview**.

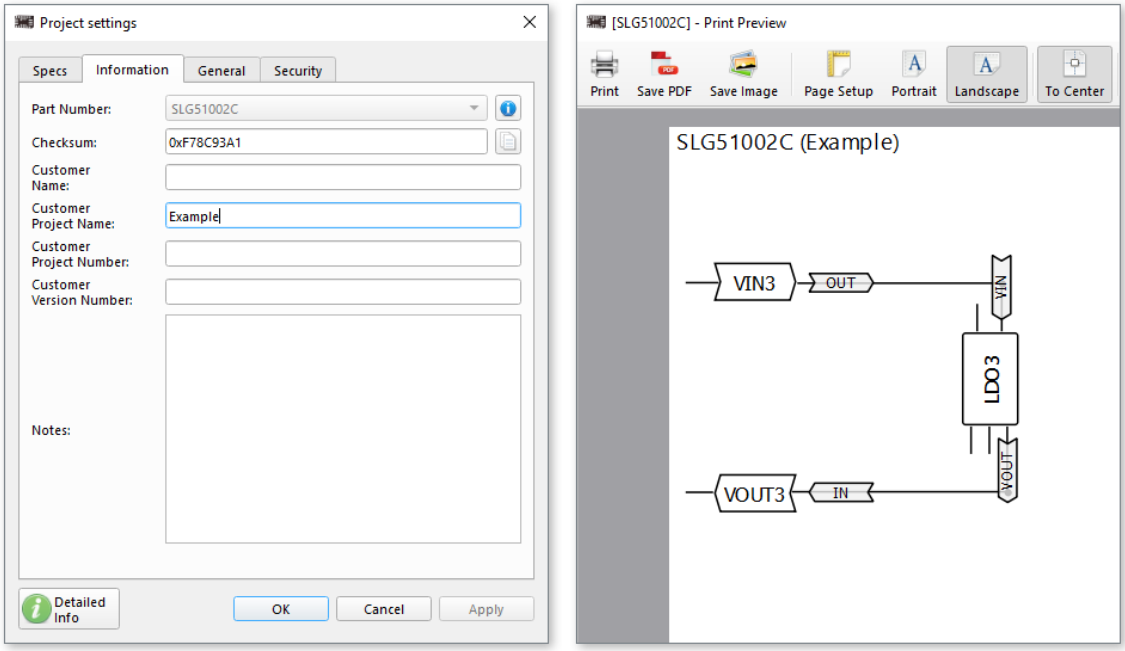


Figure 24. Customer Project Name in Print Preview

2.1.3.3 General

The **General** tab contains the global chip configurations. The set of settings varies depending on the part number.

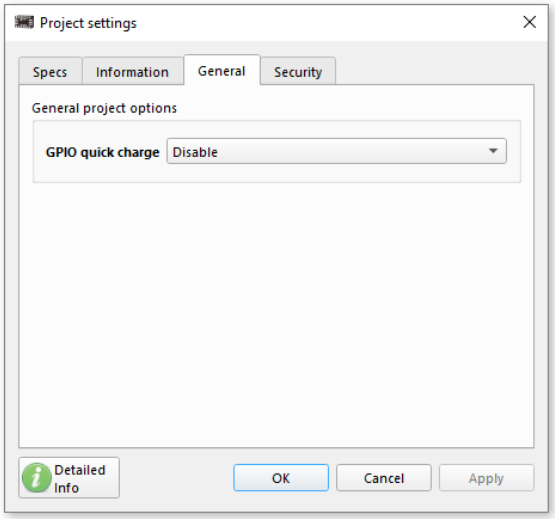


Figure 25. General Tab

2.1.3.4 Security

Same as **General**, the **Security** tab includes global settings which typically grant access to the chip's data protection features. The settings may vary between different PNs based on their specific characteristics, such as available memory type (OTP or MTP). The common and important **Security** tab configurations are project identification and memory lock and protection options. These may include:

- Project Identification (**Pattern ID**) – Can be used for project versioning (the **Pattern ID** data can be read even for chips with read protection enabled).
- Lock and Protection Options – Allow to protect read or write chip operations (manage configurations for specific memory type if applicable).

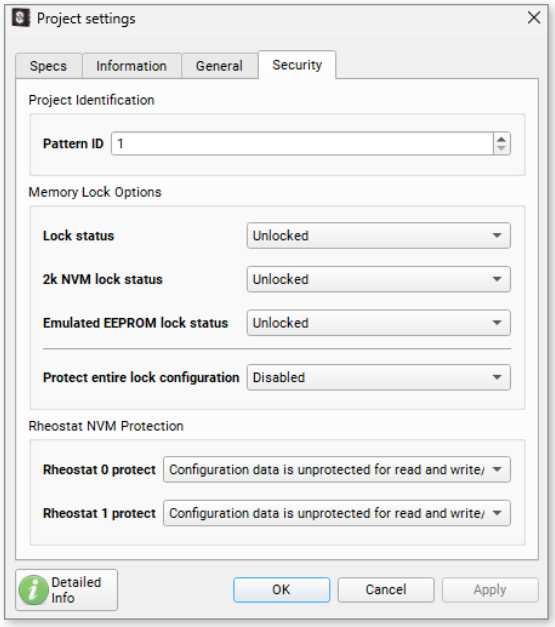


Figure 26. Example of Security Tab (SLG47004)

Note: Enable memory locking options (for read, write, or both) to enhance security when handling sensitive or critical data. See step-by-step instruction on how to lock the project's memory.

To find out more about the particular chip configurations, click the **Detailed info** button on the **Project settings** window or access the datasheets for the desired part number through the Hub window > **Products**.

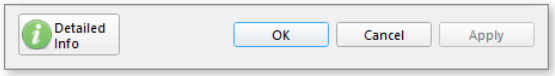


Figure 27. Detailed Info Button

2.1.4 NVM Viewer

The non-volatile memory (NVM) tool represents the permanent memory of a chip. The NVM is grouped into bytes. Each byte is an 8-bit sequence. Above each byte, the byte's address is shown along with its value in hexadecimal and decimal formats. NVM changes occur after modifying a macrocell's property or setting a connection between the blocks.

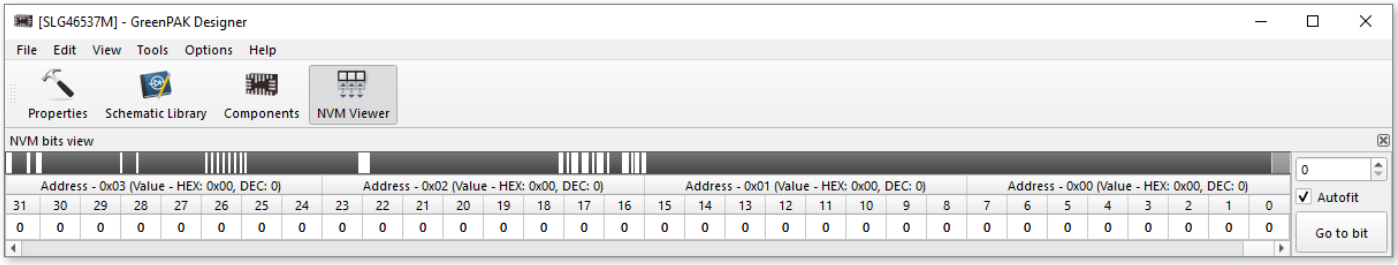


Figure 28. NVM Viewer

NVM viewer cells can be in the following states:

0	Bit range of a selected block.
0	Bits with 0 value.
1	Bits with 1 value.
0	Latest bit changed to 0 value.
1	Latest bit changed to 1 value.

The **NVM viewer** controls can also be used for quick navigation through the bit table. Use the **Autofit** feature to jump directly to the bit range of the selected component.

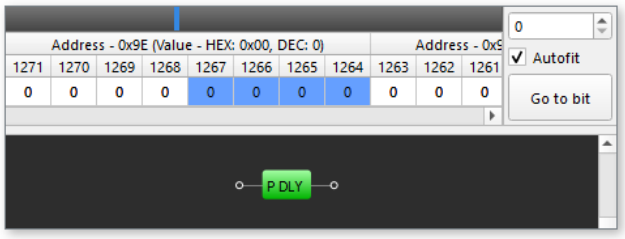


Figure 29. Autofit

Enter a bit number (in decimal format), then click **Go to bit** to find the required bit. The bit order number becomes highlighted after clicking a bit or using the **Go to bit** button.

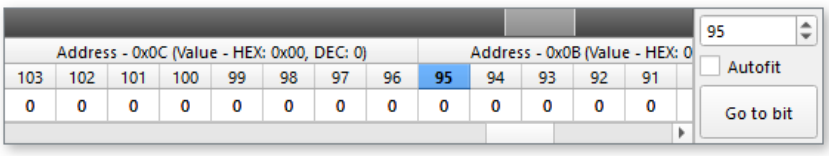


Figure 30. Bit Order Number Highlight

The **NVM viewer** navigation bar can also indicate the location of selected or changed bits in the table (the color on the bar corresponds the cells color described above). In addition, black color represents 0 and white shows 1 values.

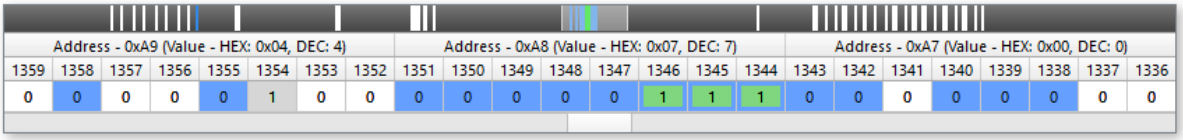


Figure 31. NVM Viewer Top Bar

Hover the mouse cursor over the bit to see which macrocell this bit belongs to.

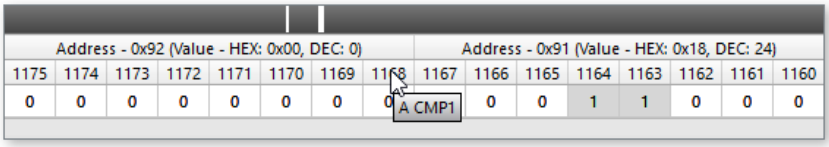


Figure 32. NVM Hint

To save the NVM sequence to a file, select **File** > **Export** from the main menu. To load an NVM sequence, select **File** > **Import** (see section 6.1 Main Menu Commands for a description of all main menu items). If the imported file contains a bit sequence whose length differs from that of the selected part number, a confirmation dialog appears, allowing the data to be loaded.

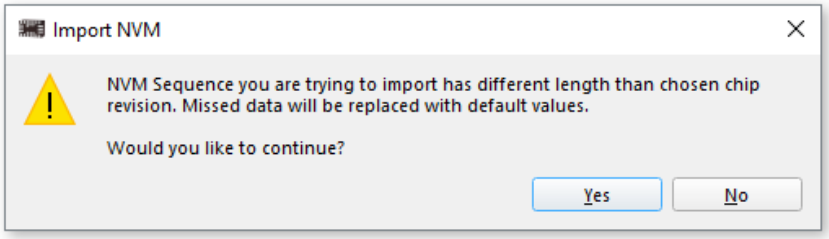


Figure 33. Import Different NVM Sequence Length

2.1.5 Components

After a macrocell is added from the component list to the work area, different interaction options become available. Move the component(s) using a mouse or a keyboard (see section 6.2 Keyboard and Mouse Controls). The toolbar widgets can also be used to rotate, flip, and align the selected block(s). To select more than one block, click, hold, and drag over the components to be selected.

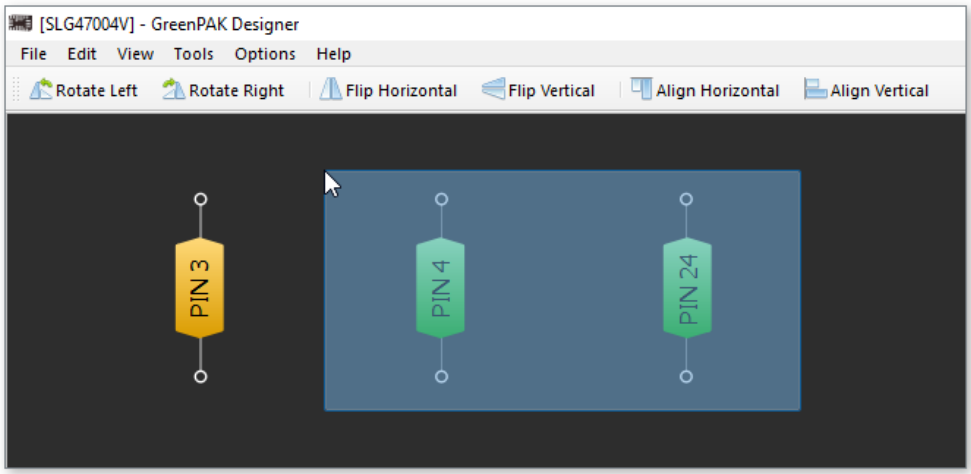
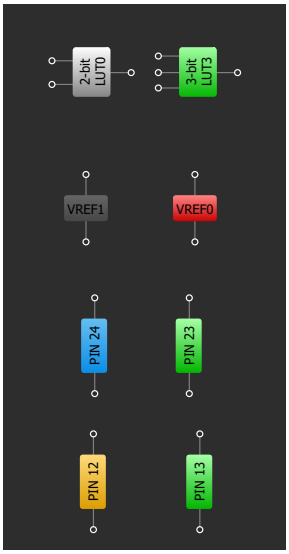


Figure 34. Selected Block Adjustment

2.1.5.1 GreenPAK Chips Components Highlight

The macrocell color provides information about its mode (enabled or disabled) and state (deselected or selected). The Input/Output (I/O) Pin block color indicates its relation to a particular VDD.



Enabled block UI.

Disabled block UI.

Pin block UI.

Pin for connection with VDD2 (analog).

Components are shown in deselected (left) and selected (right) states.

Macrocells can be connected using the pins. Hover the cursor over the block to see the pin hints.

For more info about managing pin hints behavior, refer to section 2.1.15 Settings.

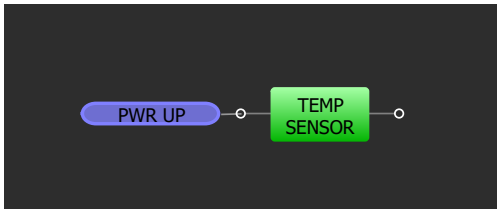
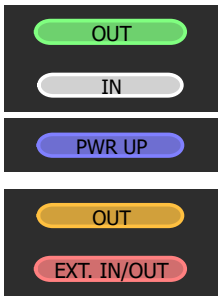


Figure 35. Pin Hint

Macrocell pin hint color indicates the connection possibility between the blocks: whether a wire can or cannot be added, or which connection type can be set.



Open for connection.

Connection limit is reached.

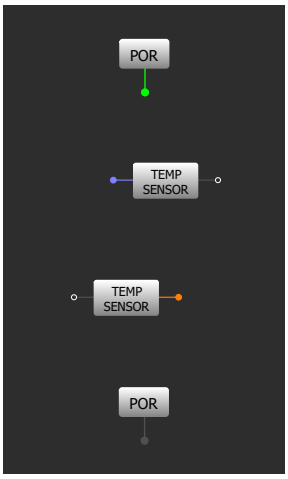
Temporarily closed for connection (macrocell properties can be changed to make it available for connection).

Used for hard-wired connections.

External IN/OUT.

Read about connection types in section 2.1.6 Connections.

When a connection is being set, the pin highlight becomes visible. Just like the pin hint color, the pin color indicates the connection possibility.



Open for connection.

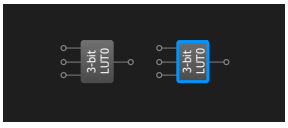
Temporarily closed for connection (macrocell settings on its **Properties** panel can be changed to make it available for connection).

Used for hard-wired connection.

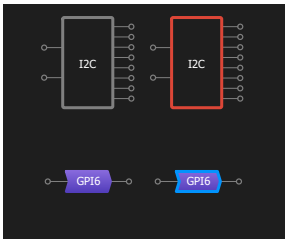
Connection is not allowed.

2.1.5.2 SLG51000/1 Chip Components Highlight

Component highlight for SLG51000/1 chips in different modes or states is as follows:



Enabled block UI.



Disabled block UI.

I/O Pin block UI.

Components are shown in deselected (left) and selected (right) states. Pin and pin tip colors are the same as for chips described above.

2.1.5.3 Component Pin Indicators

Components in the work area display visual markers for easier identification of the specific pin functions:

- **o** – The higher-order input pin.
- **>** – A clock input pin.

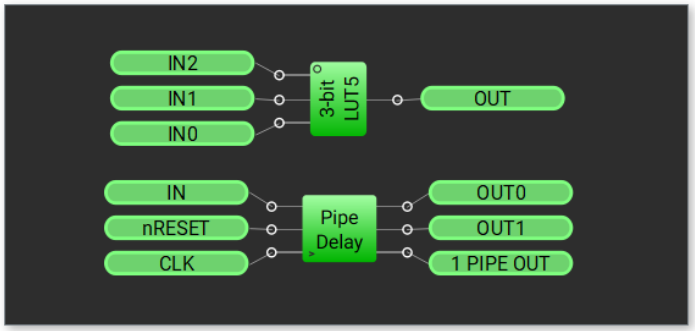


Figure 36. Pin Indicators

2.1.6 Connections

The **Set Wire** widget helps establish a connection between blocks. To add a wire, click the two pins between which the connection is being set. More than one connection set from one OUT pin is a network. Connection creation can be canceled by clicking the right mouse button.

To remove the connection, enable the **Erase Wire** tool and click the wire (the connection or network

can also be deleted from the context menu, triggered by right-clicking the wire).

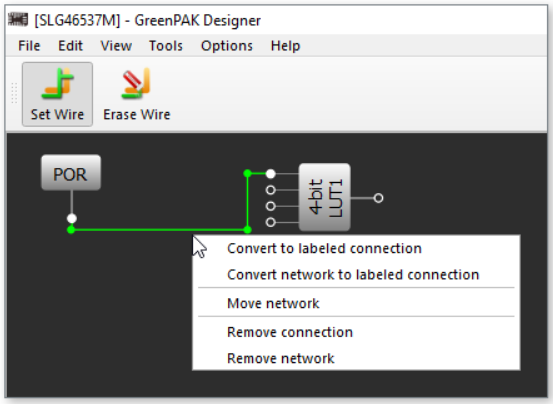
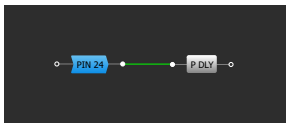
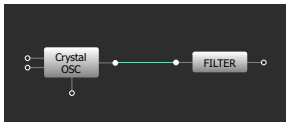
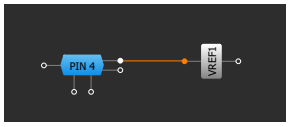
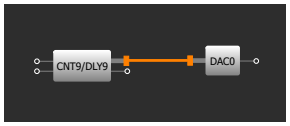
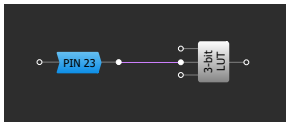
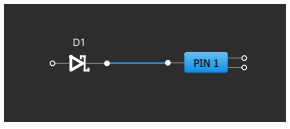


Figure 37. Set or Erase Wire (Network)

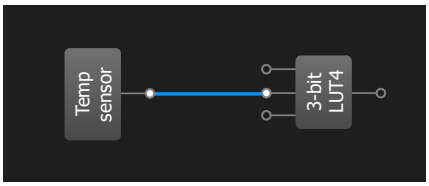
2.1.6.1 GreenPAK Chips Connections Highlight

While working with different part numbers the following component connection types may be encountered:

	**Matrix**	The most common connection which can be set between green pins.
	**Shared**	Matrix connection which is physically shared with more than one component.
	**Hard-wired**	Connections which are predefined and depend on block settings.
	**Bus**	Same as hard-wired.
	**State dependent / Cross-matrix**	Connections set to state dependent components (such as **Dynamic Memory**, **F** or **State blocks** within ASM subsystem in SLG46880V chip) / connections set between Matrix 0 and Matrix 1 components (SLG47011V).
	**External**	Connection set to external components (read more in section 2.4.2 External Components).

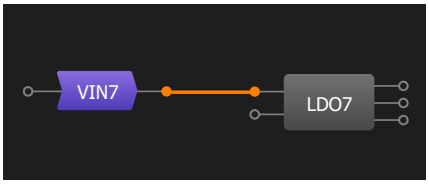
2.1.6.2 SLG51000/1 Chip Connection Highlight

In SLG51000/1 part number the connection types are as follows:



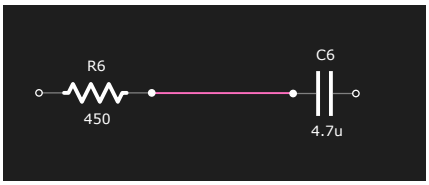
Matrix

The most common connection which can be set between green pins.



Hard-wired

Connection which is predefined and depend on block settings.

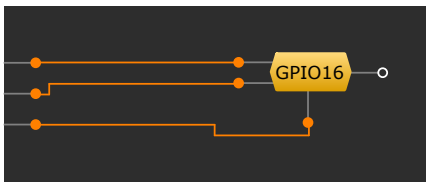


External

Connection set to external components (applicable only for SLG51001), read more in section 2.4.2 External Components.

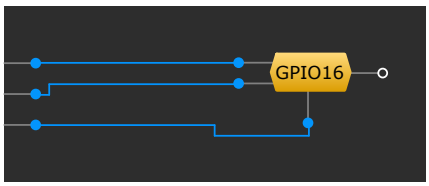
2.1.6.3 FPGA Chip Connection Highlight

Two connection types for FPGA part number (SLG47910V (Rev BB)) are as follows:



Not activated

Connection to FPGA Core not used in configuration.



Activated

Connection to FPGA Core used in configuration. This type displays the current state of the I/O Planner, and therefore Floorplan. The **Activated** connections are reset to **Not activated** after each Synthesis procedure.

2.1.6.4 Common Connection-Related Behavior for All Part Numbers

A wired connection or network can be converted to labeled one. Trigger the context menu by right-clicking the wire to see this option.

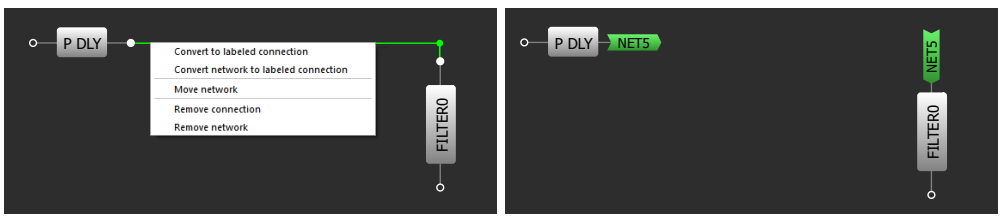


Figure 38. Convert a Wire to Label

After conversion, the label color remains the same as its wired version (Exception: different connection types set from one OUT. In this case, label is yellow. Applicable only to GreenPAK part numbers). Labels and pin colors during connection are also the same.

Connections can be labeled by default, but they can be changed to wired if needed. The name of a labeled connection can also be changed. Double-click the label and enter a new name in a pop-up window.

After hovering the mouse over a label, it becomes highlighted along with all other parts of the same connection.

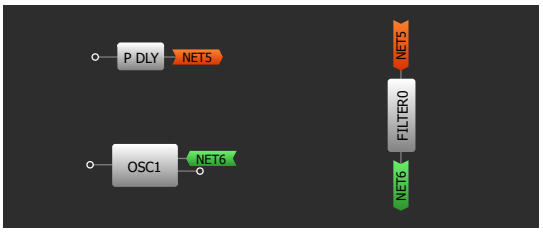


Figure 39. Highlighted Label

The hint with connection or network information is also displayed when the cursor hovers over the wire or label.

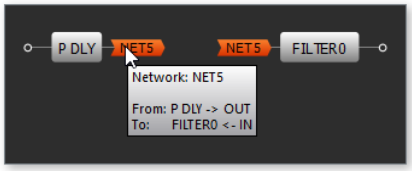


Figure 40. Network Connection Hint

A connection (network) can be moved to another OUT. Select **Move network** from the context menu. Choose the new component in the **Move network** window and click **Move**.

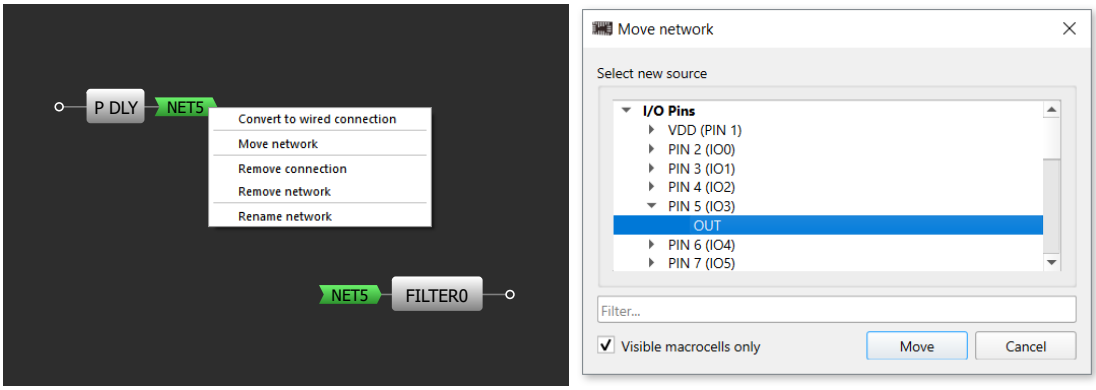


Figure 41. Move Connection (Networks)

Information about the components, pins, connection colors, and their descriptions is also available in Go Configure Software Hub in the **Legend box**. Read more in section 2.1.7 Legend Box.

2.1.7 Legend Box

The **Legend box** window shows the color scheme of the components and connections-related features in Go Configure Software Hub. The **Legend box** view may vary depending on the part number. The window is available from the main menu under **Help** > **Legend box**.

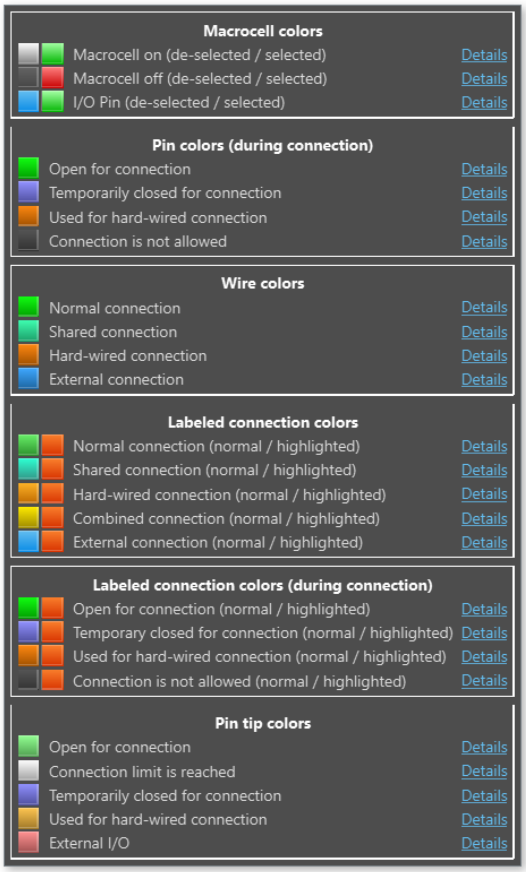


Figure 42. Legend Box

Find out more in sections 2.1.5 Components and 2.1.6 Connections.

2.1.8 Draw Frame Tool

The **Draw Frame** and **Draw Custom Frame** tools enable visual grouping of elements on the work area. Both tools are available on the toolbar, in the **Edit** menu, or using customizable keyboard shortcuts.

- **Draw Frame** – Creates standard rectangular frames. Click and drag on the working area to define the frame. Then select the frame by its border to move it, or drag its edges and corners to resize.
- **Draw Custom Frame** – Creates polygonal frames to outline irregular regions. Click to place the first point, then click to add subsequent segments. Right-click to undo the last segment, or press **ESC** to cancel the draft.

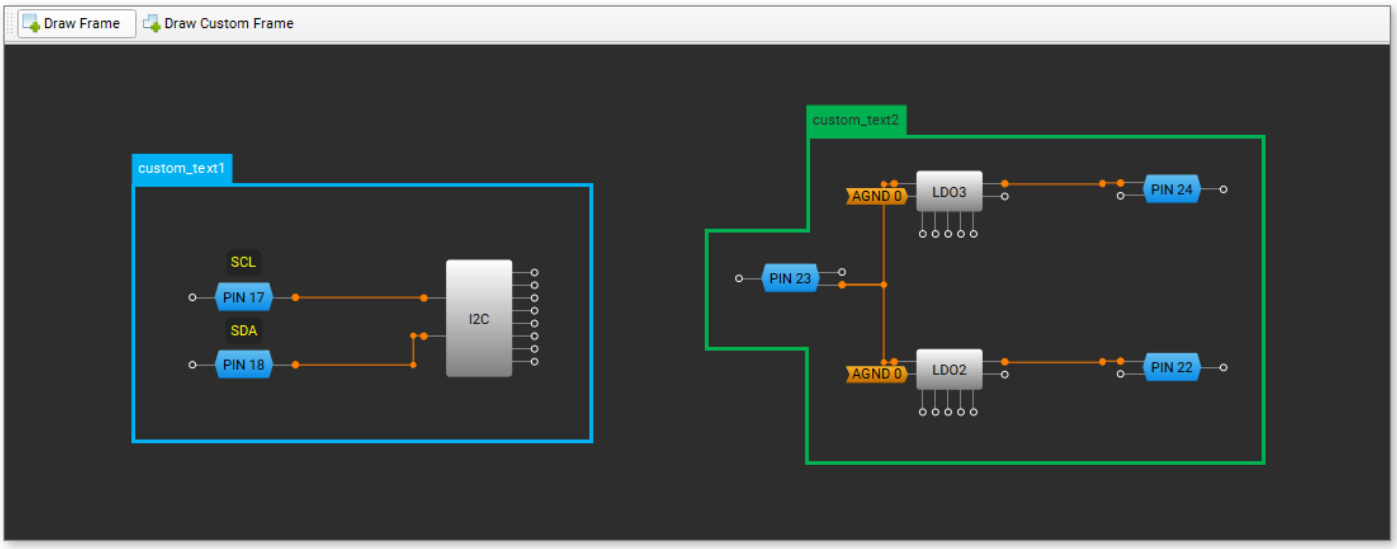


Figure 43. Draw Frame Tool

To modify or remove the frame, refer to its context menu. Customize the frame in the **Edit frame(s)** window. The frames will also appear in the **Print Preview**.

2.1.9 Rule Checker

The **Rule Checker** tool scans a project for errors related to incorrect block connections or settings. To check the design, click **Rule Checker** on the toolbar (the tool is also available from the main menu under **View** > **Tools**).

The **Rule Checker** output window consists of three main columns:

- **Event** – Shows the message type (Fail, Warning, Note).
- **Rule** – Explains the essence of the issue.
- **Note** – Suggests the way to correct the error or provides more details about the issue.

Rule Checker output

Time	Event	Rule	Note
23:37:22	Fail	3-bit LUT2/DFF/LATCH2: The truth table is configured incorrectly.	The truth table is configured so that all combinations of the inputs that are connected to the macrocells, do not cause changes on the output.
23:37:22	Fail	3-bit LUT3/DFF/LATCH3: The truth table is configured incorrectly.	The truth table is configured so that all combinations of the inputs that are connected to the macrocells, do not cause changes on the output.
23:37:22	Fail	VDD and temperature parameters must be filled in.	Please check project specs in Project Settings.
23:37:22	Warning	3-bit LUT2/DFF/LATCH2: No input connected.	3-bit LUT2/DFF/LATCH2's input is not connected.
23:37:22	Note	3-bit LUT3/DFF/LATCH3: macrocell is placed in the schematic but not used.	3-bit LUT3/DFF/LATCH3 is placed in the schematic but has no connection. Please hide the macrocell or make some connection.

Refresh Checking is done with: **3 fails, 1 warning** and **1 note.**

Figure 44. Rule Checker Window

Use the controls to sort the messages by type. Click the **Refresh** button to display the latest

events.

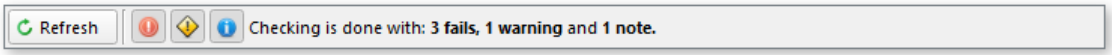
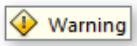


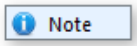
Figure 45. Rule Checker Controls

Rule Checker window shows three message types:

A critical error that may cause the project to fail is present in the design. In some cases, this message type can be ignored (the program may consider the design solution erroneous, although it is correct for the intended application).



The project contains improper block(s) connections or settings. This message does not necessarily imply an error, but it notifies about a potential problem and urges checking the block(s) connections and settings.



This message type suggests minor improvements in the design. The suggestions are optional and can be ignored.

2.1.10 Snipping Tool

Snipping tool is a work area screenshot-maker. The tool is available from the main menu under **Tools** > **Snipping tool**.

Choose between the **Datasheet** and **Regular** screenshot style (the **Datasheet** style view is the same as in **Print Preview**)

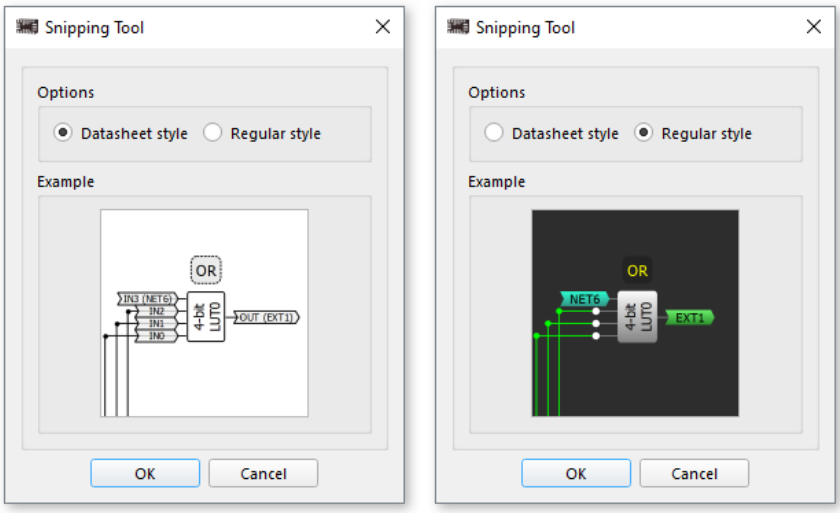


Figure 46. Screenshot Style

After selecting the area, copy it to the clipboard or save it in a supported format.

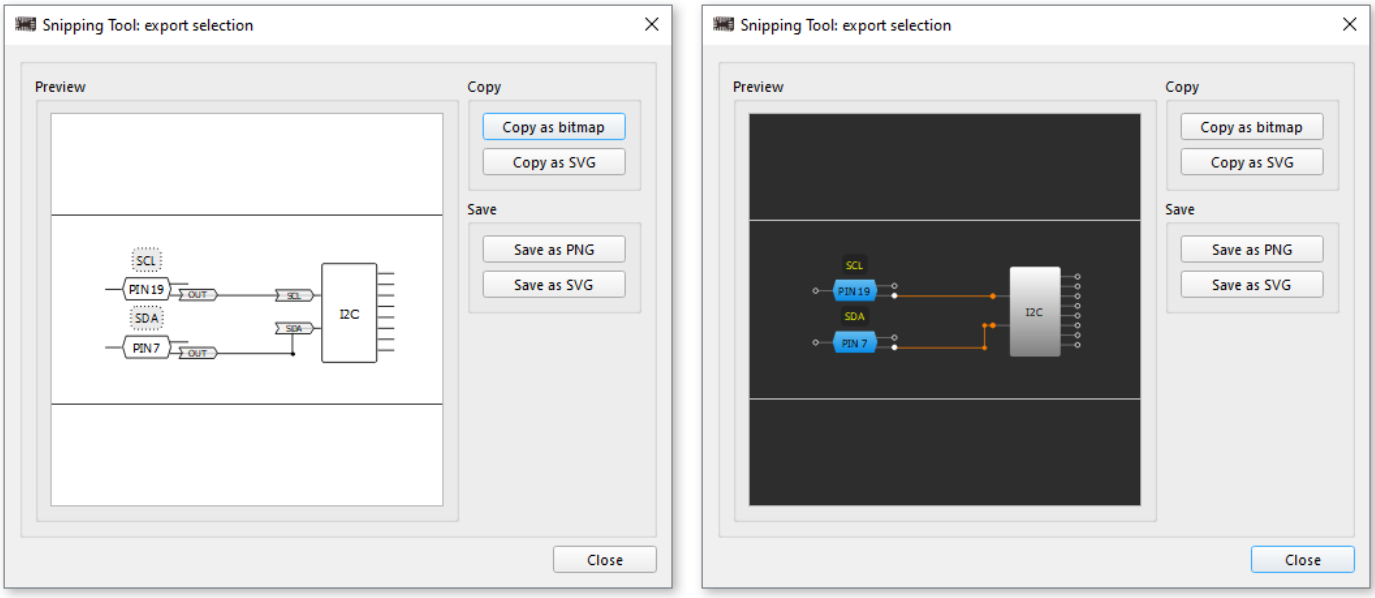


Figure 47. Supported Formats

2.1.11 Comparison Tool

The **Comparison tool** allows comparison of the NVM state of two projects. The tool can be opened by clicking the **Comparison tool** widget on the toolbar or by selecting **Tools** > **Comparison tool**

from the main menu.

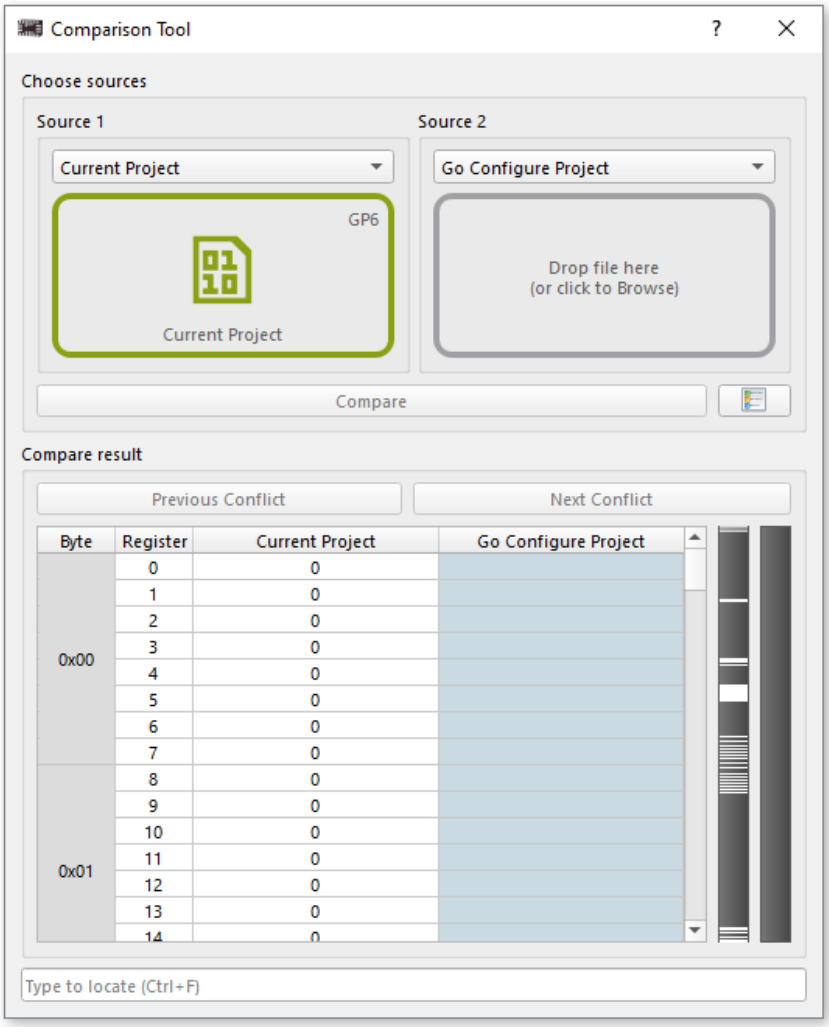


Figure 48. Comparison Tool

The tool consists of two main parts: **Choose sources** and **Compare result**.

2.1.11.1 Choose Sources

Here, the sources for the NVM comparison can be selected. Select the source types from the dropdowns. Then click the designated area to choose the file or drag and drop it. The available sources are as follows:

- **Current Project** – Current state of the design.
- **Go Configure Project** – Go Configure Software Hub project file with the following extensions: `.aap`, `.can`, `.gp3`, `.gp4`, `.gp5`, `.gp6`, `.hvp`, and `.ppak`.
- **Text File** – Exported NVM file in .txt format.

- **Chip Project** – Configurations programmed on a chip.

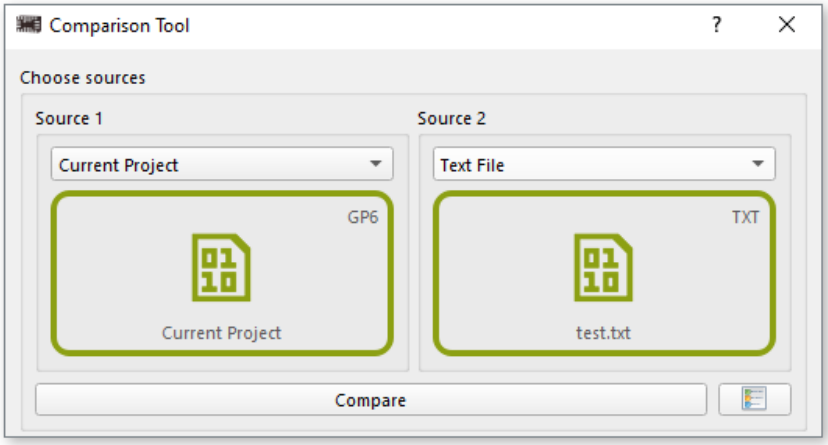


Figure 49. Choose Sources

If **Chip Project** is selected as a source, make sure the supported platform with the inserted IC is connected to the computer. The board info appears in the designated area. Click **Read** to add the programmed data to the tool. Clicking the **Blink** button triggers the blue LED on the connected platform. This feature is useful for checking which board is currently detected by the Go Configure Software Hub instance. If multiple platforms are connected, select the desired one in the dropdown. Note that **Debug** tool should be disabled to use **Chip Project** as a source.

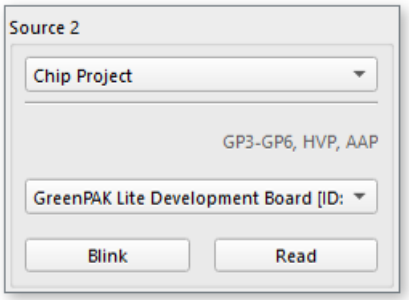


Figure 50. Chip Source

The **Compare** button becomes active when two sources are selected (for **Chip Project**, after the source is selected, click **Read**). After clicking **Compare**, a pop-up appears where the ranges to ignore while counting conflicts can be selected.

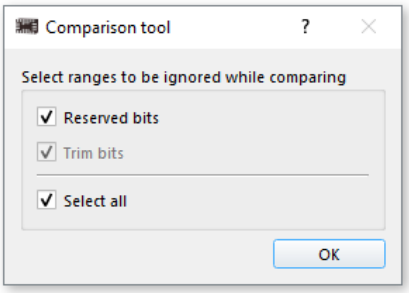


Figure 51. Range Selection

After the pop-up is closed, the **Comparing** window with the results appears.

2.1.11.2 Compare Results

After the sources are compared, the table with the results is displayed. Navigate through the unmatched bits using the **Previous Conflict** and **Next Conflict** buttons.

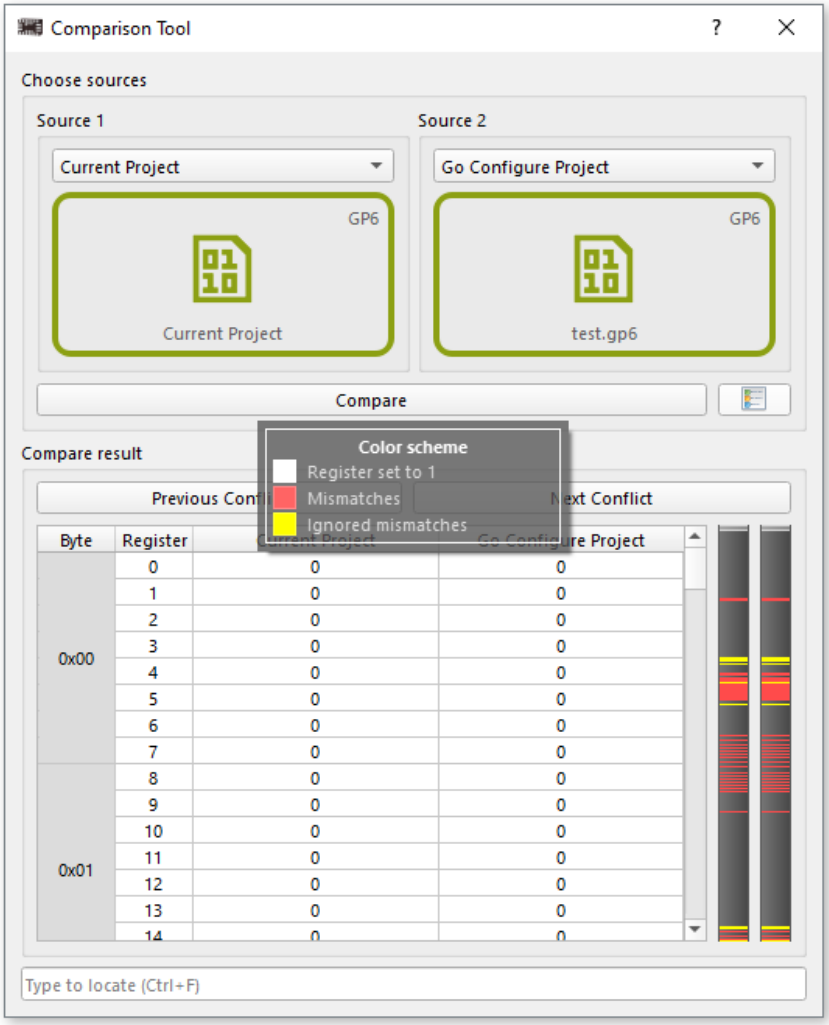


Figure 52. Compare Results and Color Scheme

The table contains the information about a byte, register, and NVM of the selected sources. The vertical navigation bar indicates the location of `0` and `1` register values, along with conflicts and ignored bits. See the color explanations by clicking the **Color scheme** button.

The search field can also be used to find a particular byte or register.

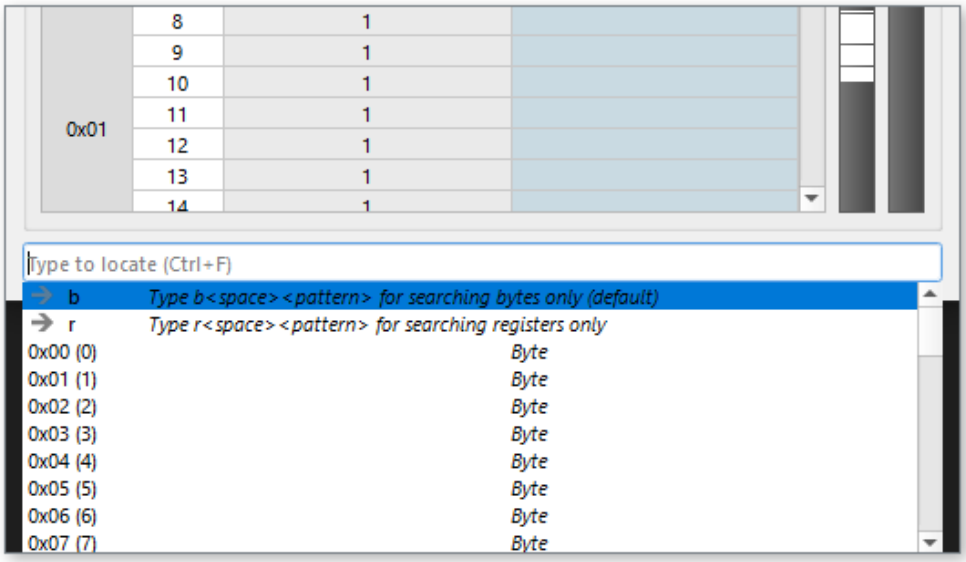


Figure 53. Search Field

2.1.12 Power Dissipation Calculator

The tool helps to determine the amount of heat dissipation of the device, which is an important step in developing an effective thermal management solution. **Power Dissipation Calculator** is available only for the HVPAK part numbers and mainly applies to the HV OUT CTRL macrocell and HV OUT pins.

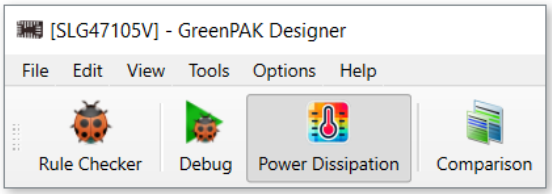


Figure 54. Power Dissipation Calculator on the Toolbar

To launch the tool, click the **Power Dissipation** button on the toolbar or select it from the main

menu under **Tools**.

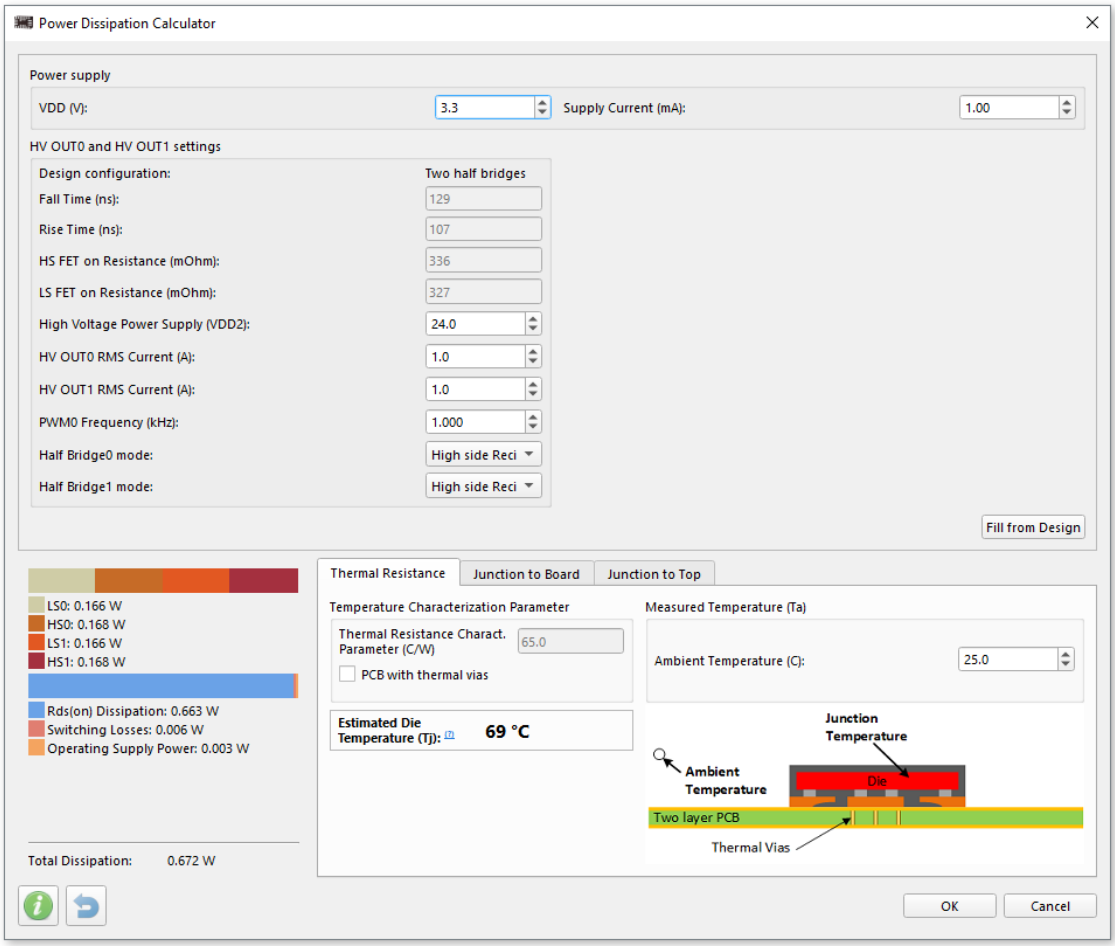


Figure 55. Power Dissipation Calculator Preview Window

The tool consists of four main parts:

- **Power supply** – Voltage applied to VDD pin and estimated current consumption parameters.

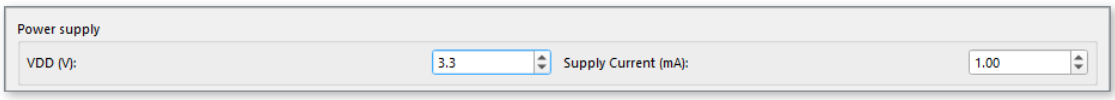


Figure 56. Power Supply

- **HV OUT 0 and HV OUT 1 settings** – The main parameters of the power pins used in the design.
  - **Design configuration** – Reflects the **HV OUT mode** parameter selection, set on the HV OUT CTRL macrocell's **Properties** panel.
  - **Fall Time**, **Rise Time**, **HS FET on resistance (mOhm)**, **LS FET on resistance (mOhm)** – Configurations, which are automatically adjusted based on the conditions and configuration of the HV OUT CTRL component.

- **High Voltage Power Supply** – Set the supply voltage (VDD2) for HV OUT pins.
- **HV OUT RMS Current** – Responsible for the output current and cannot exceed 1.5 A per HV OUT.
- **PWM 0 Frequency (kHz)** – Set the frequency at which HV OUT pins operate. The frequency value can be set using the PWM 0 macrocell. Additionally, the frequency value can be transferred automatically using the **Fill from Design** button in the lower right corner of the **HV OUT 0 and HV OUT 1 settings** section.

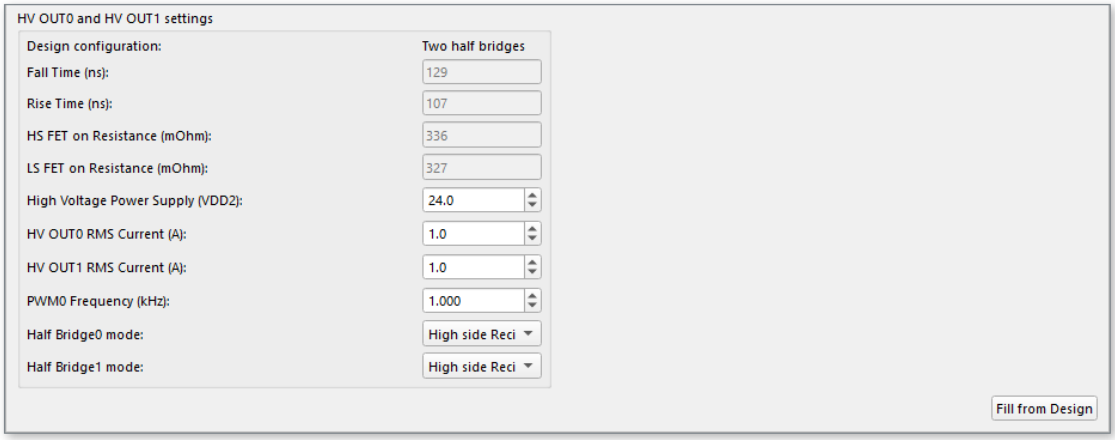


Figure 57. HV OUT 0 and HV OUT 1 Settings

Note: When the HV OUT pins are in the **Hi-Z** state, **HV OUT 0 and HV OUT 1 settings** are displayed as locked. Change the HV Pin's **Output mode** on its **Properties** panel to unlock the settings.

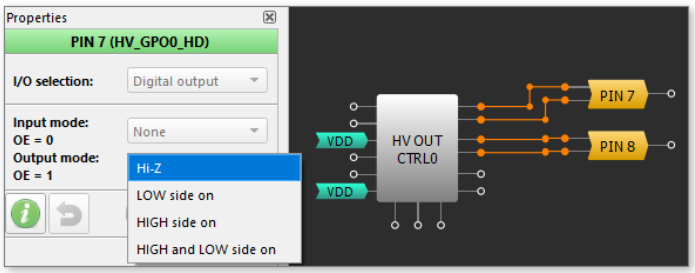


Figure 58. HV OUT Properties Panel

- **Power dissipation bar** – Indication of the power dissipation for each output FET: **LSx (low side)** and **HSx (high side)**. Below, the total power dissipation is shown, including open FET resistance Rds(on), FET switching losses, and the power dissipated by the chip even when

the HV OUT pins are in **Hi-Z**.

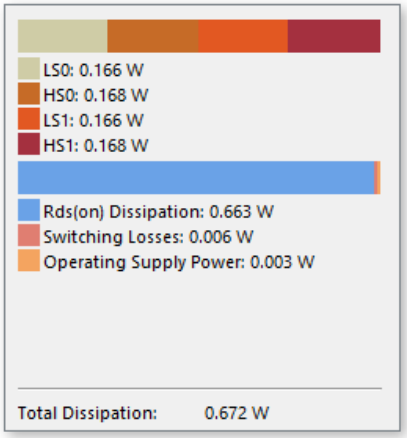


Figure 59. Power Dissipation Bar

- **Temperature Characterization** – Provides three options for calculating the device **Die** temperature:
  - **Thermal Resistance** – Only the ambient temperature of the device is required for calculations. This method is the least accurate. If the PCB in use contains thermal vias, the respective checkbox can be selected to take this into account

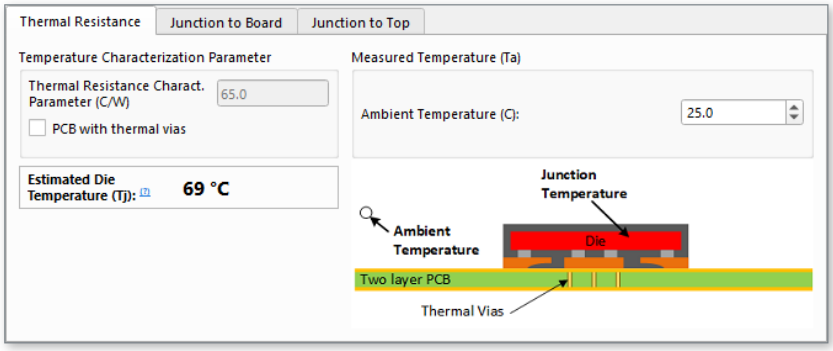


Figure 60. Thermal Resistance Window

  - **Junction to Board** – The PCB temperature is used to calculate the **Die** temperature. It should be measured 1 mm from the device. This method has a higher accuracy

compared to **Thermal Resistance**

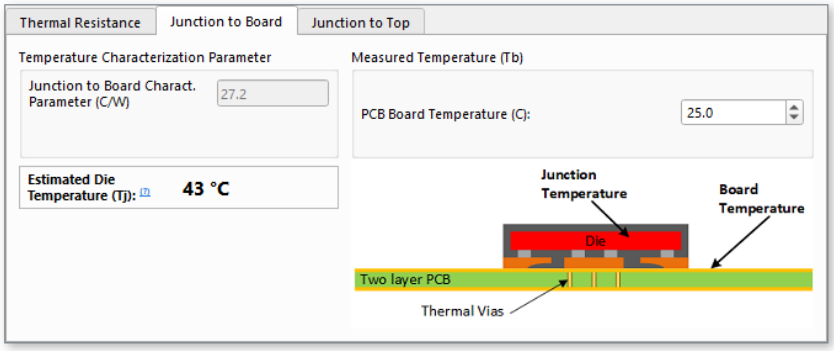


Figure 61. Junction to Board Window

- **Junction to Top** – The temperature of the **Die** is determined based on the upper surface temperature of the chip. This method has the highest accuracy.

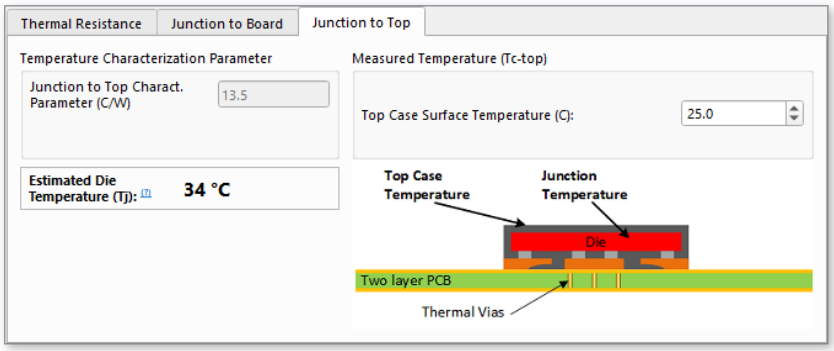


Figure 62. Using Junction to Top Window

Keep the **Die** temperature below 150°C to ensure the chip operates correctly. An indication of overheating appears if the temperature exceeds the limit.

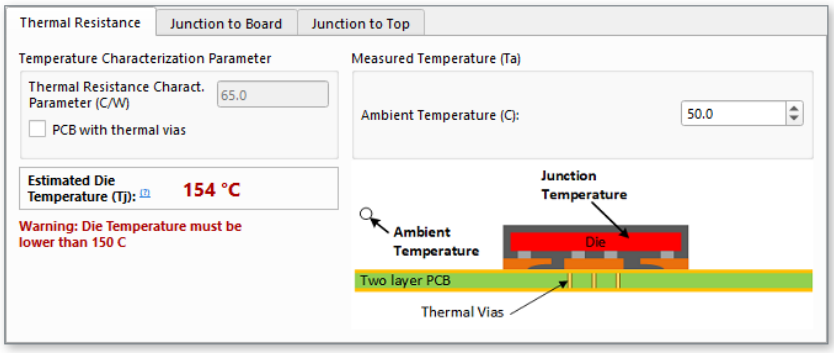


Figure 63. Overheating Warning

2.1.13 Acceleration Profile Configurator

The **Acceleration Profile Configurator** tool is designed for linear motor speedup calculation based on specified time and motor speed range for the Commutation block. Additionally, Timer macrocells can use **Acceleration Profile** as a counter data source. Launch the tool by clicking the **Acceleration Profile** button on the toolbar or by selecting it from the main menu under **Tools**.

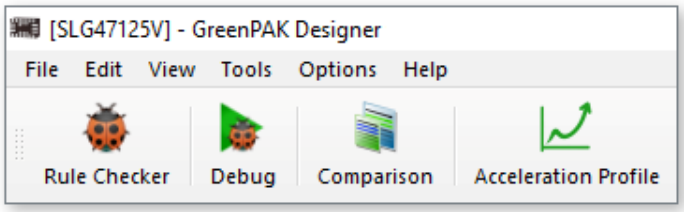


Figure 64. Acceleration Profile on the Toolbar

Select the desired macrocell, then set the parameters for calculations.

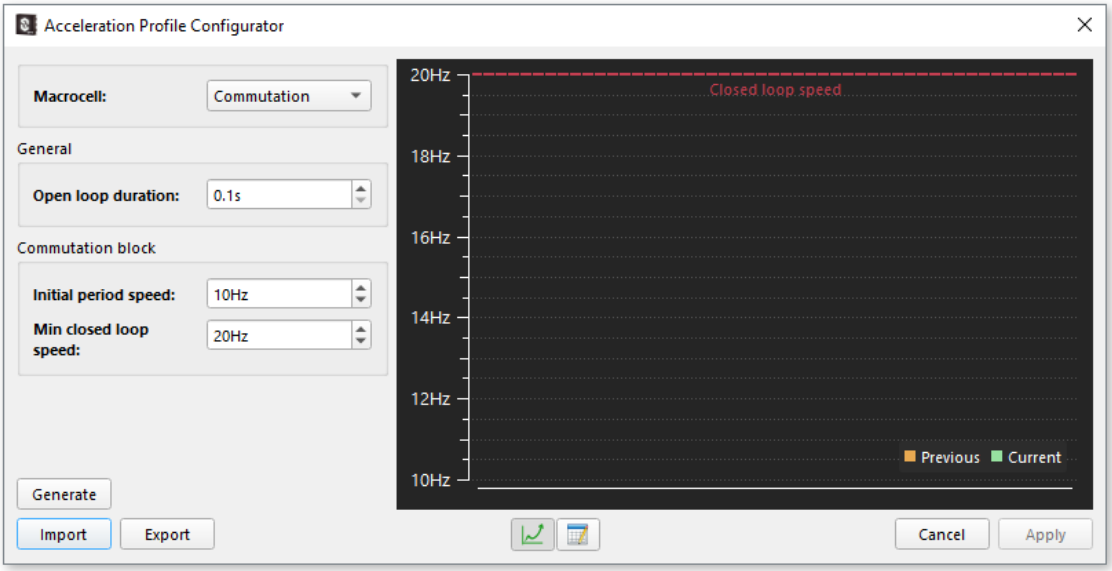


Figure 65. Acceleration Profile Configurator Tool

2.1.13.1 Commutation

The parameters available for the Commutation macrocell are listed below.

- **Open loop duration** – Time interval of motor acceleration time (x-axis on the graph).
- **Initial period speed** – Minimum motor rotation frequency (y-axis on the graph) during the open loop acceleration.
- **Min closed loop speed** – Maximum motor rotation frequency (y-axis on the graph) during the open loop acceleration.

The two latter parameters can be configured in the tool directly or on the Commutation block **Properties** panel and are mutually syncronized.

Click the **Generate** button to perform the calculations on the basis of the provided data.

The results are represented in the following ways:

- **Plot** – Graphical visualization of open loop acceleration. It provides the following information and possibilities:
  - x-axis (time interval set to the **Open loop duration**) and y-axis (frequency values set to the **Initial period speed** and the **Min closed loop speed**).
  - Current graph (based on latest added parameter values) and previous one (based on the previously set values).
  - **Closed loop speed** dotted line showing the upper frequency limit.
  - Tracker with labels.
  - Zoom by x-axis scale with **Ctrl** + **mouse wheel**.

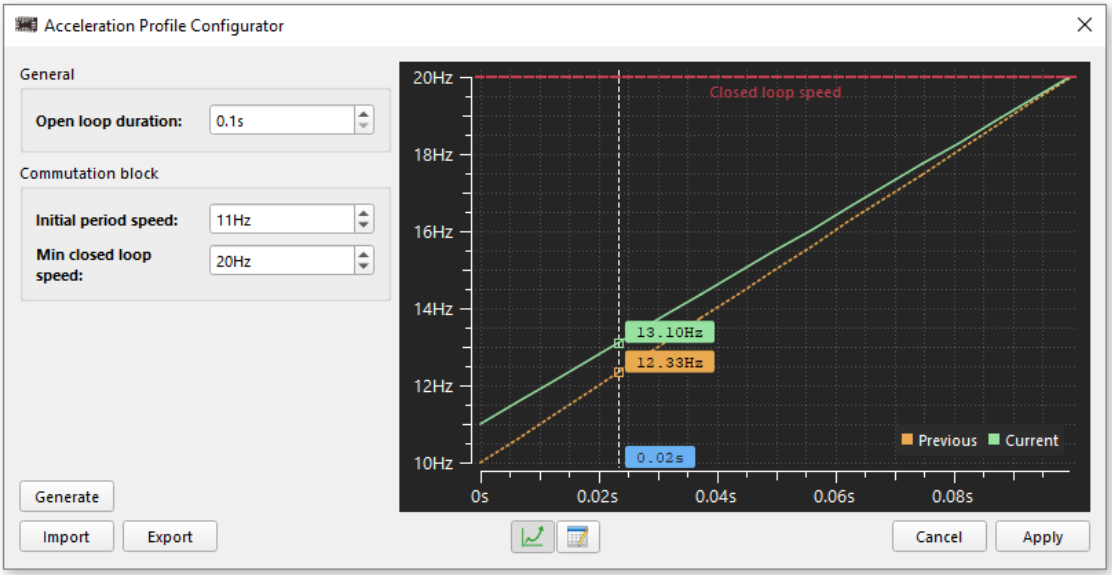
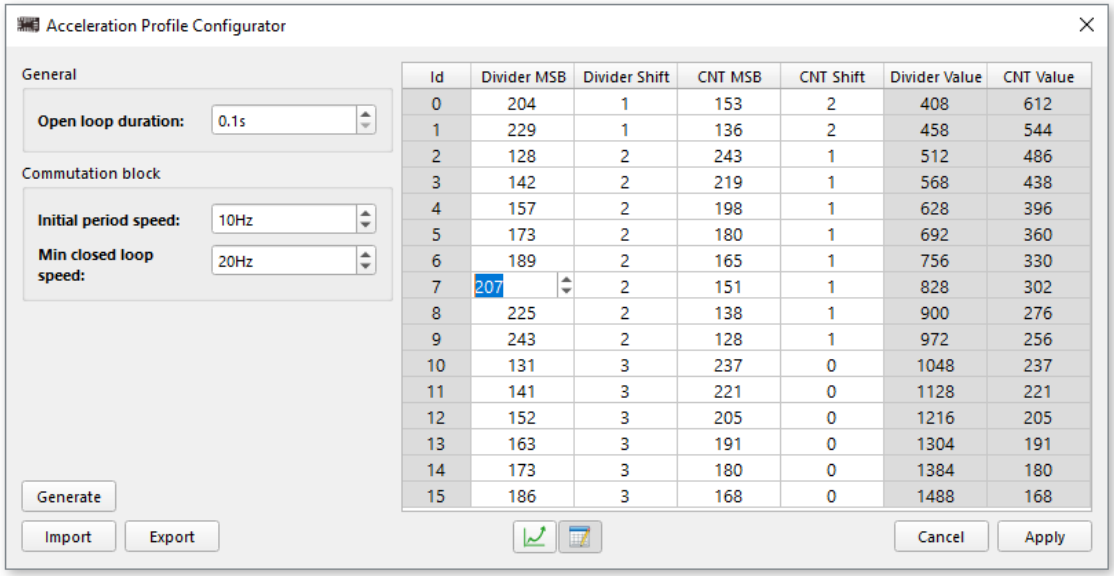


Figure 66. Commutation Waveform

- **Table** – The table contains 16 points. Each point consists of two values: **Divider Value** and **Counter (CNT) Value**. Each value consists of two parts: MSB 8bit and shift 4bit to reduce memory space. It is also possible to modify the **MSB** and **Shift** cells manually by double-clicking them.



Id	Divider MSB	Divider Shift	CNT MSB	CNT Shift	Divider Value	CNT Value
0	204	1	153	2	408	612
1	229	1	136	2	458	544
2	128	2	243	1	512	486
3	142	2	219	1	568	438
4	157	2	198	1	628	396
5	173	2	180	1	692	360
6	189	2	165	1	756	330
7	207	2	151	1	828	302
8	225	2	138	1	900	276
9	243	2	128	1	972	256
10	131	3	237	0	1048	237
11	141	3	221	0	1128	221
12	152	3	205	0	1216	205
13	163	3	191	0	1304	191
14	173	3	180	0	1384	180
15	186	3	168	0	1488	168

Figure 67. Manual Data Editing for Commutation Table

2.1.13.2 Timers

The Timer macrocell parameters in the tool are also synchronized with the ones on the Timers **Properties** panel.

- **Dynamic mode** – Enable the Dynamic mode.
- **Counter Data (CD) Source** – Select the Timer source counter, Acceleration profile, or Scaler (where applicable) to read the counter data value from.

Same as for Commutation, the Timers' data is also represented in two ways:

- **Plot** – Visualizes the counter data values.

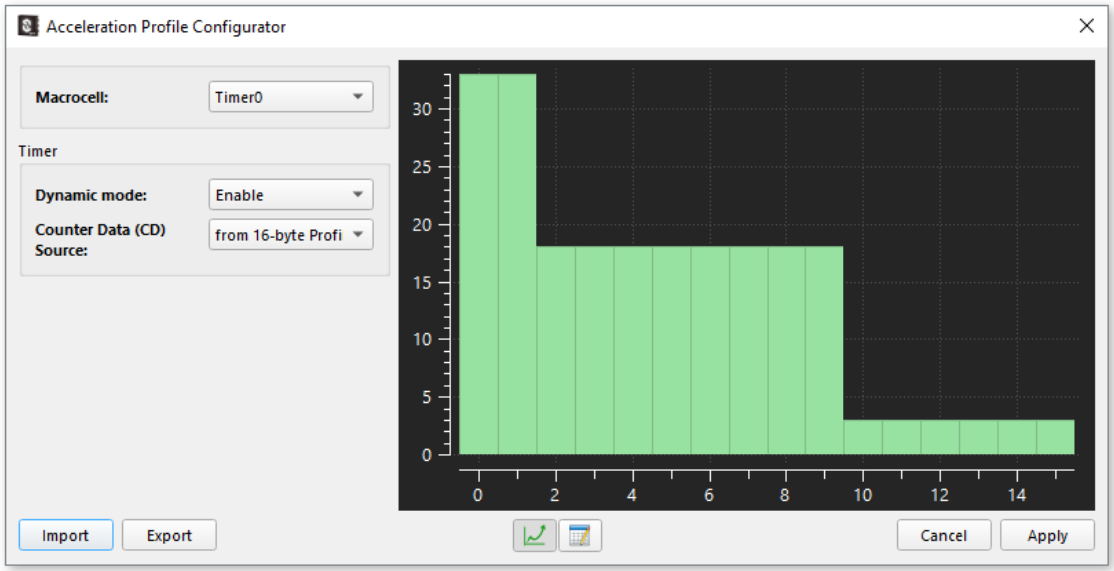
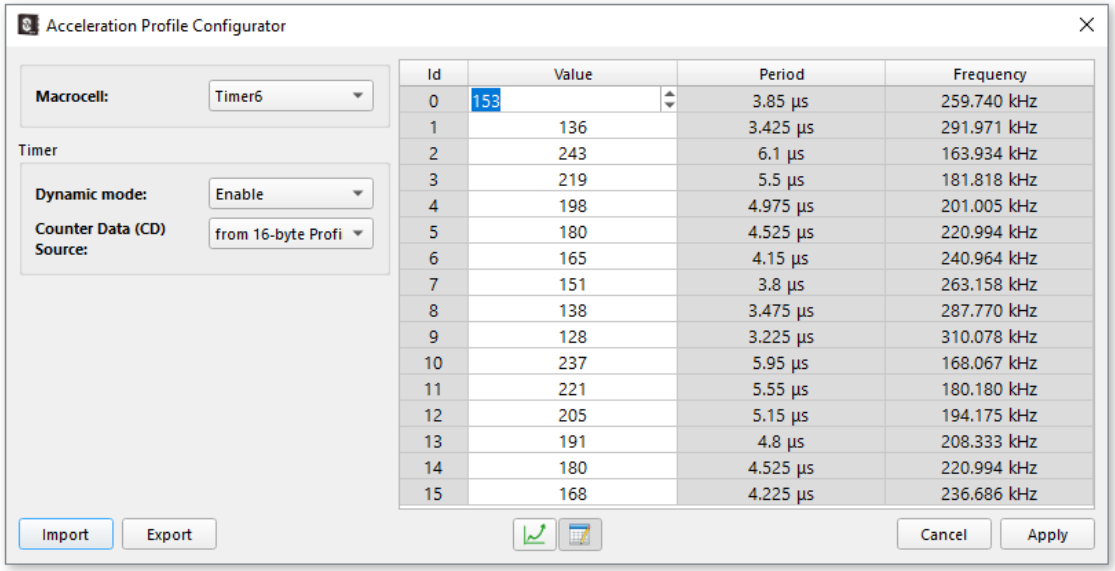


Figure 68. Plot

- **Table** – The table also consists of 16 points, which can be entered manually while the **Counter Data (CD) Source** is set as **from 16-byte Profile**. **Period** and **Frequency** are calculated based on the cell value.



Id	Value	Period	Frequency
0	153	3.85 µs	259.740 kHz
1	136	3.425 µs	291.971 kHz
2	243	6.1 µs	163.934 kHz
3	219	5.5 µs	181.818 kHz
4	198	4.975 µs	201.005 kHz
5	180	4.525 µs	220.994 kHz
6	165	4.15 µs	240.964 kHz
7	151	3.8 µs	263.158 kHz
8	138	3.475 µs	287.770 kHz
9	128	3.225 µs	310.078 kHz
10	237	5.95 µs	168.067 kHz
11	221	5.55 µs	180.180 kHz
12	205	5.15 µs	194.175 kHz
13	191	4.8 µs	208.333 kHz
14	180	4.525 µs	220.994 kHz
15	168	4.225 µs	236.686 kHz

Figure 69. Timer Macrocell Table

Import or Export the calculation results in .csv or .txt format using the corresponding buttons at the bottom.

Note 1: To write the table data to NVM, click the **Apply** button.

Note 2: The Commutation and Timers table values share the same NVM bits.

2.1.14 Reg File Configurator

The tool interacts with Commutation block and PWM macrocells. For Commutation block, the tool provides the motor control waveform calculation according to the given amplitude and the Reg File size. For PWM, it performs the automatic fill-up of the Reg File for the specified waveform.

Start **Reg File Configurator** from the toolbar or find it in the main menu, **Tools**.

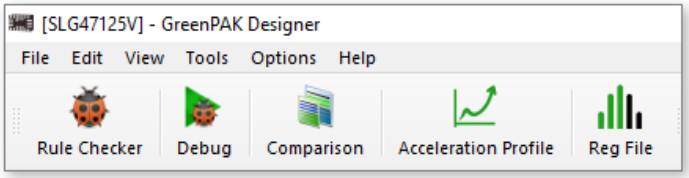


Figure 70. Reg File Configurator on the Toolbar

The parameter set can be used to:

- Specify the used amount of bytes that can be read from Reg File by the selected macrocell.

- Select the waveform type in which the generated data is represented (this option is available only while Commutation block macrocell is selected).
- Set the PWM resolution and the waveform amplitude.

Click the **Generate** button to perform the calculations on the basis of the provided data (this option is available only while Commutation block macrocell is selected).

The tool provides two ways of data representation:

- **Plot view** – X-axis represents the amount of bytes that can be read by the selected block; y-axis is the waveform amplitude. The green part represents data stored in Ref File, while orange shows data for motor control generated automatically, based on Reg File data.

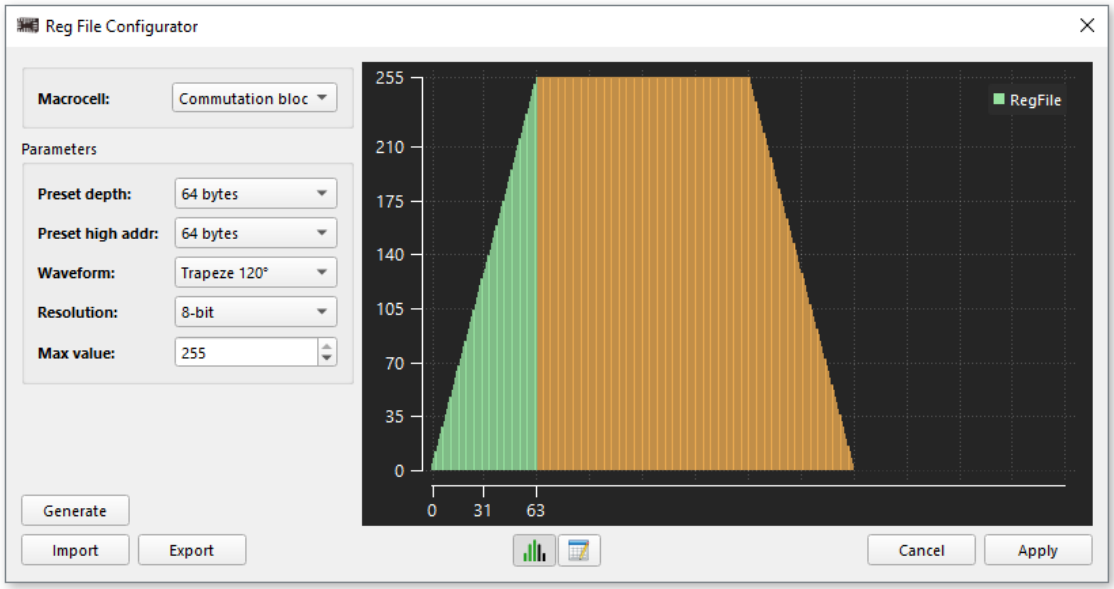


Figure 71. Reg File Configurator Window

- **Table view** – In addition to generated values, the **Value** column cells can also be modified

manually by double-clicking.

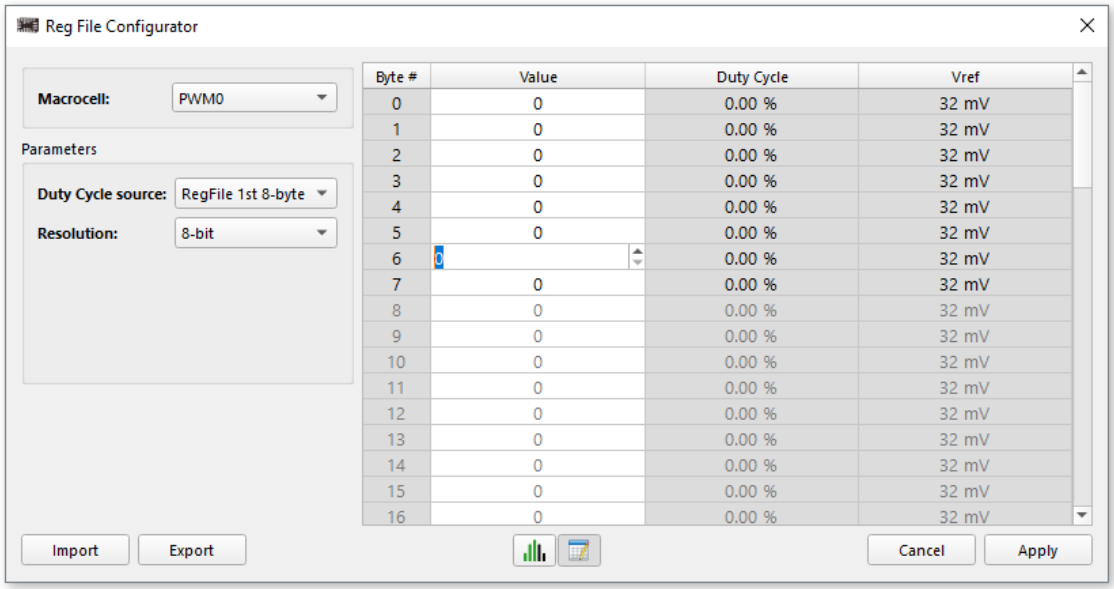


Figure 72. Manual Data Editing

Import or export the calculation results in .csv or .txt format using the corresponding buttons at the bottom.

Note: To write the table data to NVM, click the **Apply** button.

2.1.15 Settings

To reach the settings window open **Options** > **Settings** (on macOS open **App menu** > **Preferences**). The window contains the following tabs: **General**, **Designer**, **Appearance**, **Shortcuts**, and **Updater**.

2.1.15.1 General

- **Default projects folder** – Define the path to project files.
- **Projects recovery** – Activate project file autosaving.

This feature reduces the risk or impact of data loss in case of a crash or freeze. The project file copy is saved to a temporary location at a predefined interval. If a critical issue occurs, the file appears in the **Recovery Files** tab in the Hub window (see the location of the **Recovery files** tab in section 2.1 Design Tools).

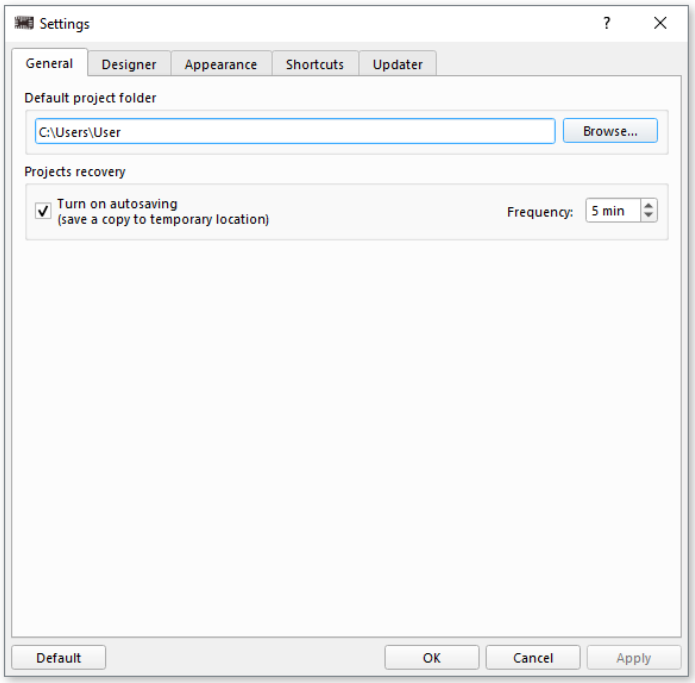


Figure 73. General Tab

2.1.15.2 Designer

- **Pin hints** – Show pin hints while a block is selected or the properties panel of a component is visible (find out more about pin hints in section 2.1.5 Components).

- **Look-Up Table (LUT)** – Select the preferred LUT shape (regular, ANSI or IEC) using different standard gates.

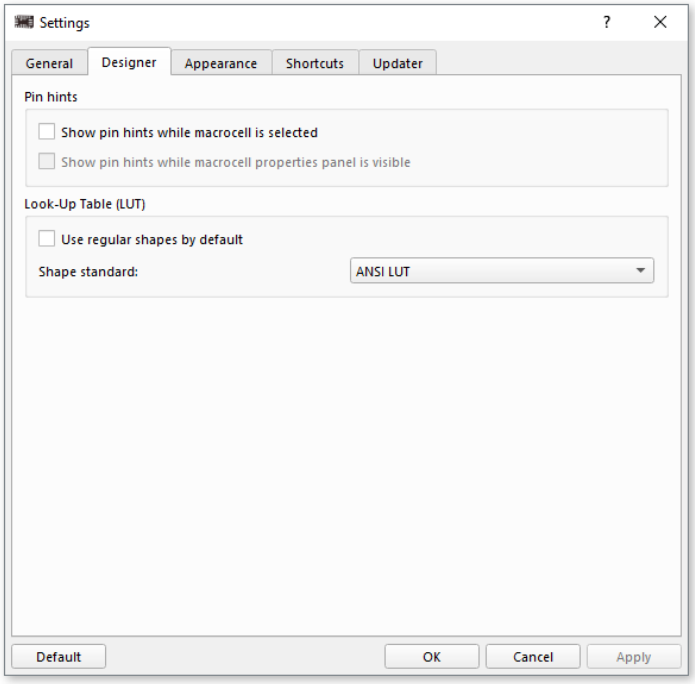


Figure 74. Designer Tab

2.1.15.3 Appearance

- **Window appearance** – Save position of the toolbars, dock widgets, and window geometry of the workspace.

  Note: If multiple GCSH instances are open, the **Window appearance** settings of the last closed instance are saved.

- **UI Scale** – Select from a predefined list of scale options to adjust the interface size. Changes to the scale require an application restart to take effect.

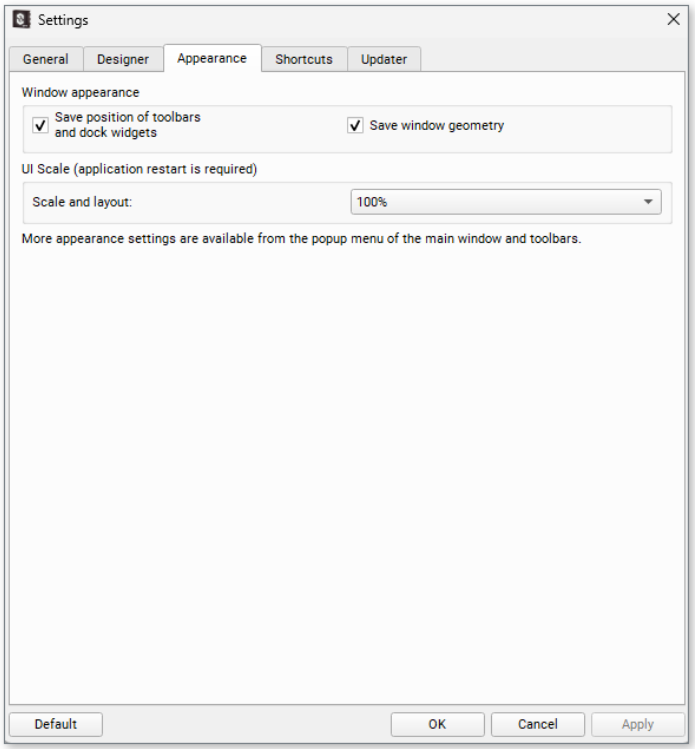


Figure 75. Appearance Tab

2.1.15.4 Shortcuts

- **Shortcuts configuration** – Assign or change the shortcut for the available actions.

Read more about shortcuts in section 2.1.17 Keyboard Commands.

2.1.15.5 Updater

- **Scheduler** – Set frequency of the updates check.
- **Path** – Define a location to download updates to.
- **Proxy** – Configure a proxy for updates.
- **Check configuration button** – Test the connection to the server.

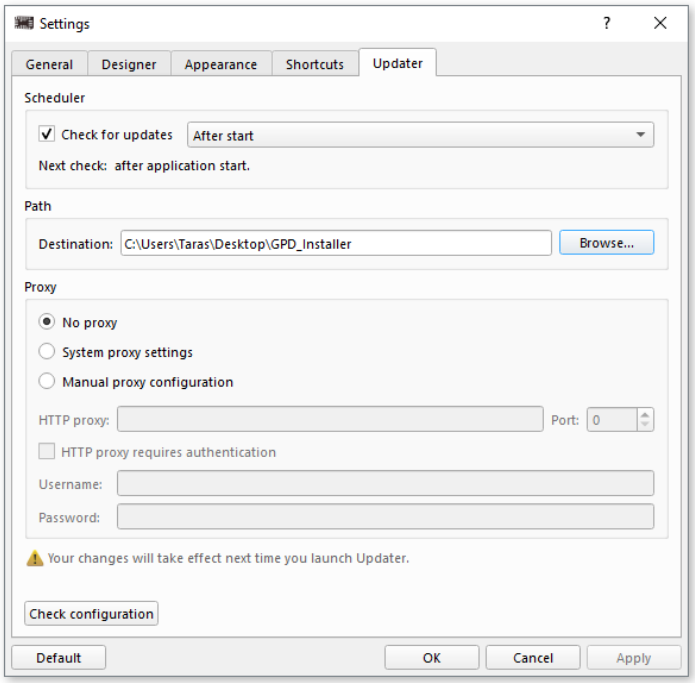


Figure 76. Updater Tab

To reset the settings, click the **Default** button at the bottom left corner of the **Settings** window. Settings can be reset for a particular category or for all categories at once.

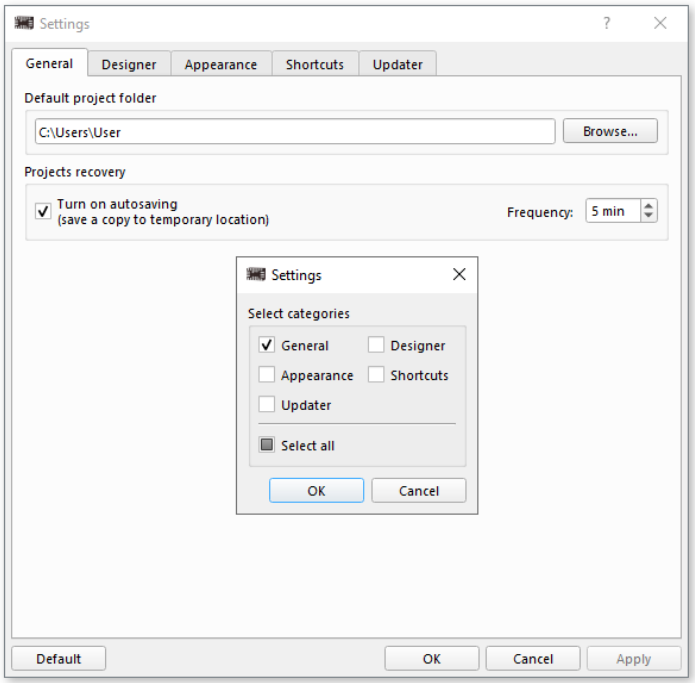


Figure 77. Default Button

2.1.16 Help window

Help materials provide information about the IC's parameters, components, and tools. There are several ways to reach the **Help** window.

To open the unified **Help** materials for a particular part number, go to the main menu > **Help** > **Help** (**F1**). Walk through the categories to find the required information.

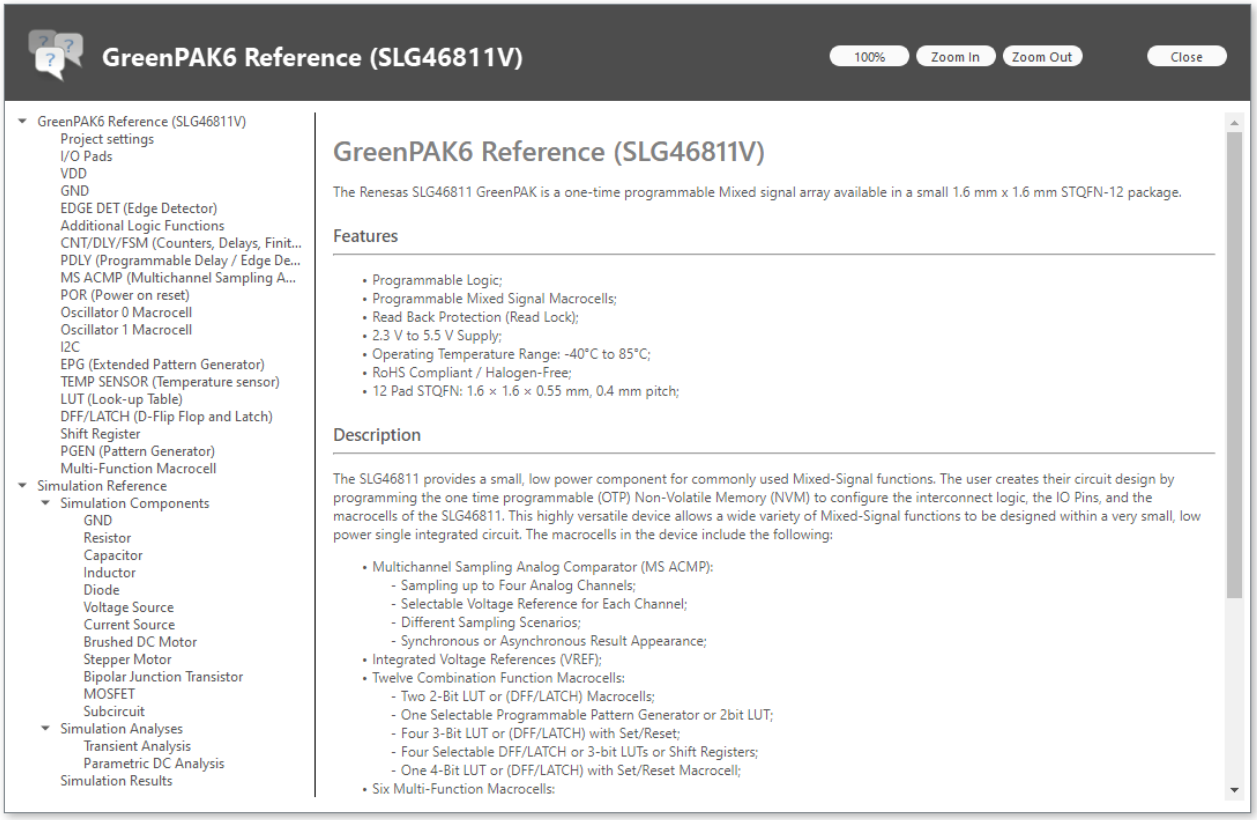


Figure 78. Help Window

To open the **Help** window for a selected block, right-click the component and select the **Detailed info** option (**F1**) in the context menu.

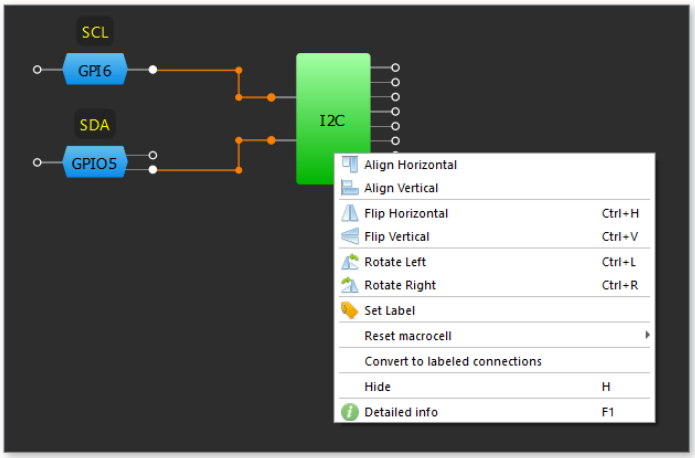


Figure 79. Detailed Info

Information buttons (ⓘ) may also appear in different workspace locations, such as on a component's **Properties** panel or **Project settings** window. Clicking the button also opens the **Help** window.

2.1.17 Keyboard Commands

There are configurable and non-configurable keyboard commands. The configurable commands can be managed in **Settings** (for more info, see section 2.1.15 Settings). To find a command easily, use the search field.

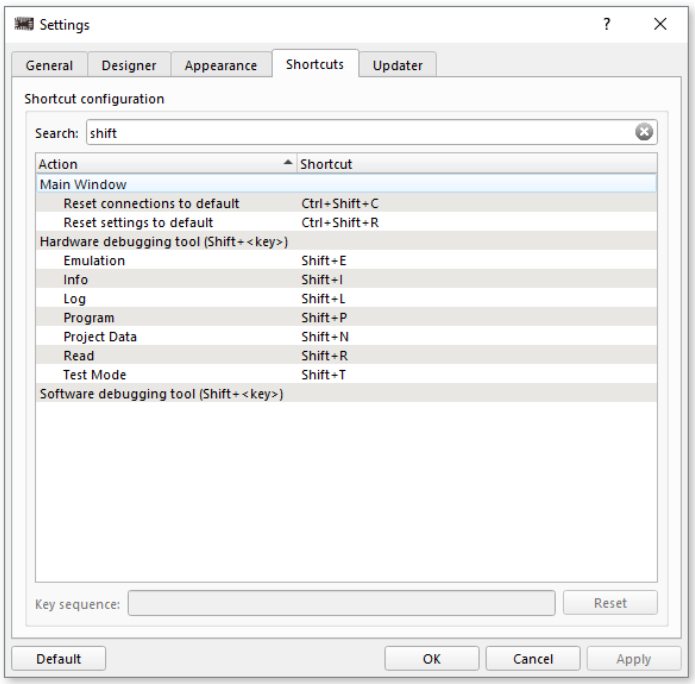


Figure 80. Shortcuts Table

To change or add a shortcut, double-click the action and press the key sequence on the keyboard

to add it to the corresponding field. To discard changes, use the **Reset** button.

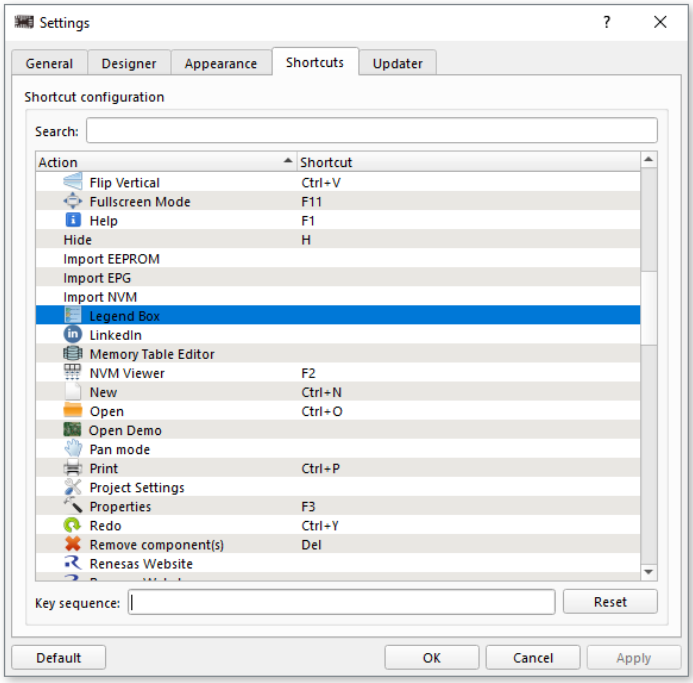


Figure 81. Assign a Shortcut

In **Debug tools**, a hotkey can be assigned to some hardware sources in the context menu. Use a hotkey to change the source state (read more in section 2.2 Debug Tools). Assign a hotkey from the context menu or create a custom one.

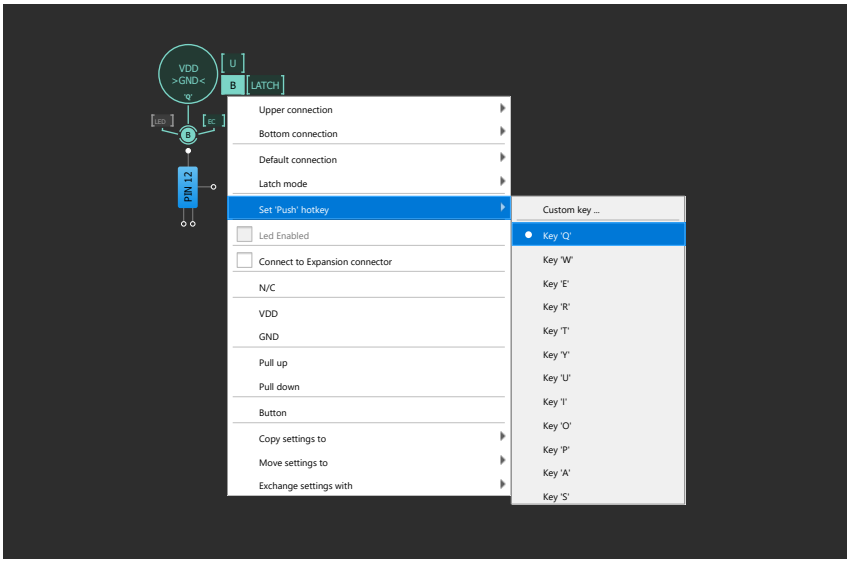


Figure 82. Hardware Sources Hotkeys

See the list of configurable and non-configurable hotkeys in section 6.2 Keyboard and Mouse Controls.

2.2 Debug Tools

Debug tools are a set of instruments that allow testing and debugging of a design. To access **Debug tools**, click the **Debug** button on the toolbar. Since different hardware platforms are available for a specific part number, select the platform most suitable for the project.

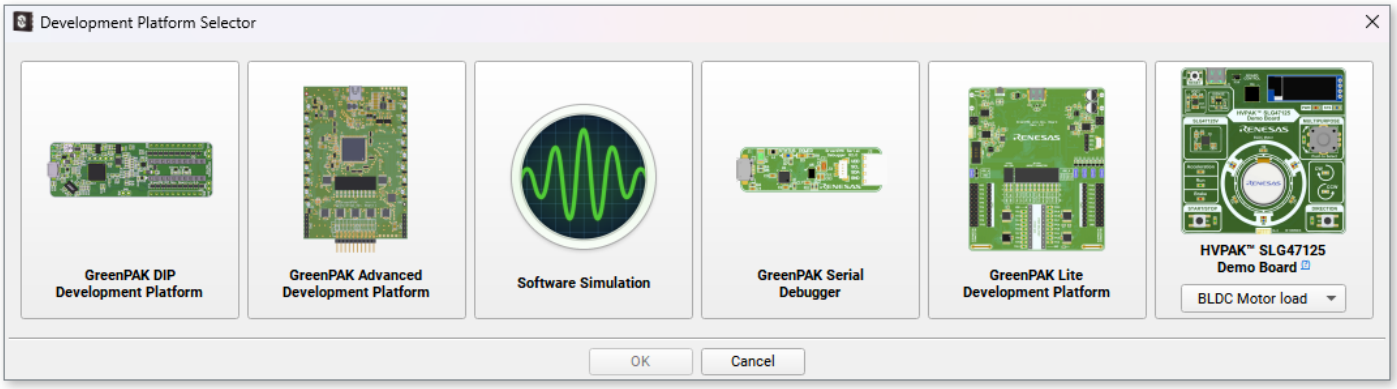


Figure 83. Platform Selector Window for SLG47125V

Note: The hardware can only be used in one application instance at a time. If it is necessary to transfer control from one instance to another, disable **Debug tools** on the controlling instance.

After a platform is selected, the software activates a toolbar and a panel with controls for main procedures, including emulation and chip programming.

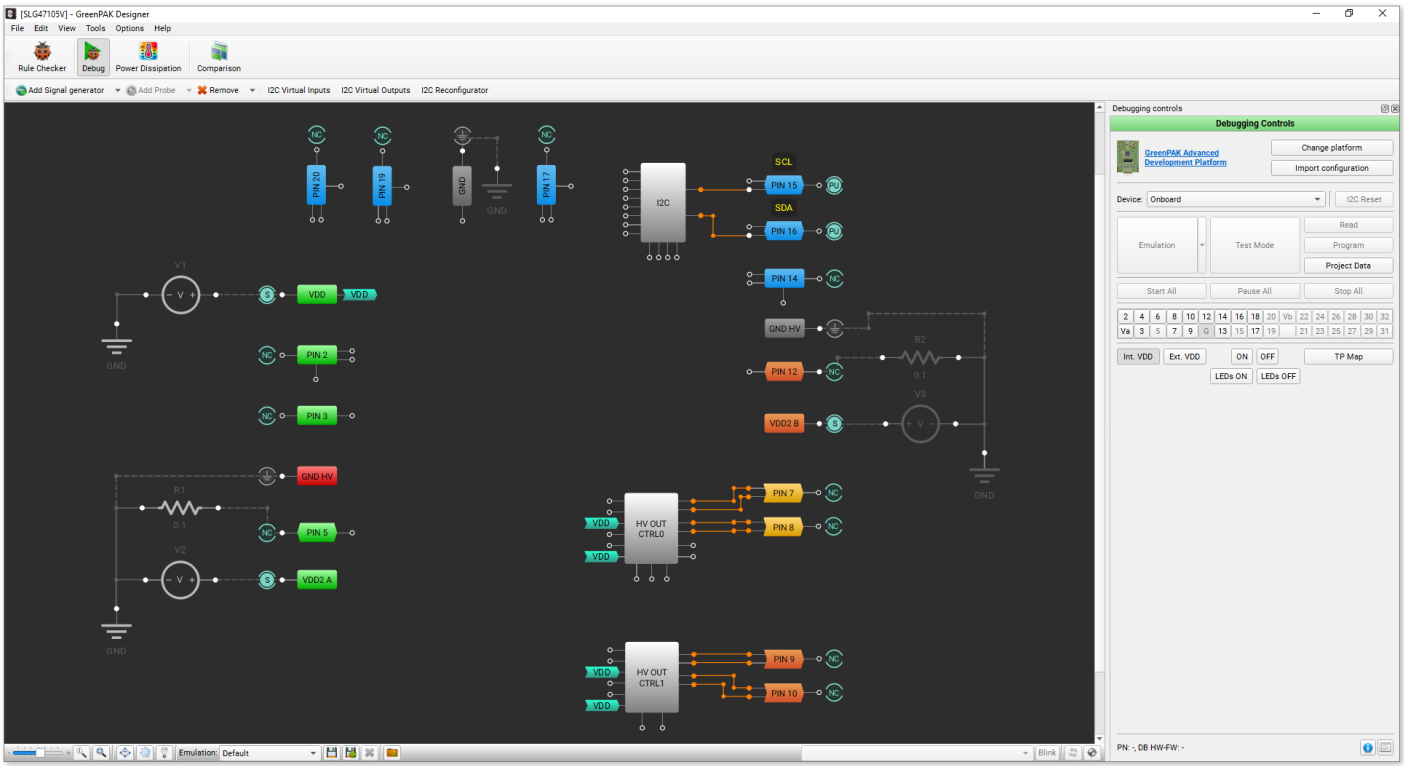


Figure 84. Software UI After Debug is Enabled

2.2.1 Hardware Sources

Go Configure Software Hub provides software tools to configure varied hardware sources that manage or generate input and test signals for a chip. Each signal source is connected to an external pin on a chip.

Below is the complete list of all hardware sources.

No source (Not Connected)

VDD

GND

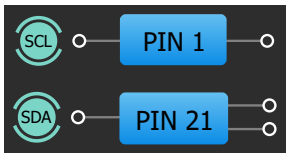
Button

Pull up

Pull down

Signal Generator

Logic Generator



I2C Generator

Constant Voltage

Parametric Generator

Synchronous Logic Generator

Note: The choice of hardware sources depends on the development platform features and chip restrictions.

The controls can be switched on or off from the context menu or from the toolbar by clicking the **Add Signal Generator** or **Add VDD** button (the button view depends on the available sources).

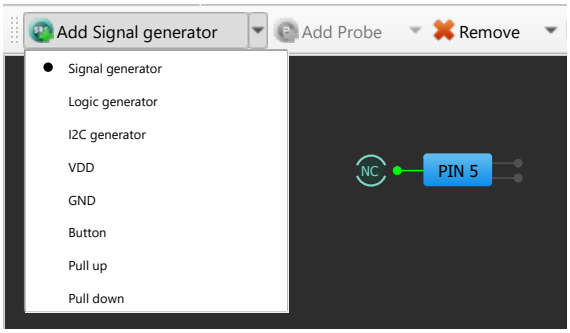


Figure 85. Debug Toolbar

To remove the source, click the **Remove** button or select **N/C** (Not Connected) in the context menu.

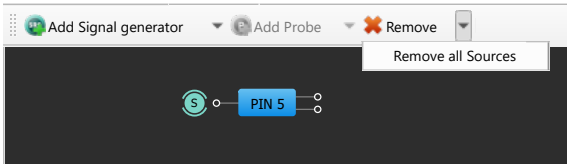


Figure 86. Remove Button on Debug Toolbar

Some chip pins may have additional controls, such as **LED indicators** or **Expansion Connectors (ECs)**, which are available even if a development platform is not yet connected. These controls can be enabled or disabled by hovering over the source and clicking the required control.

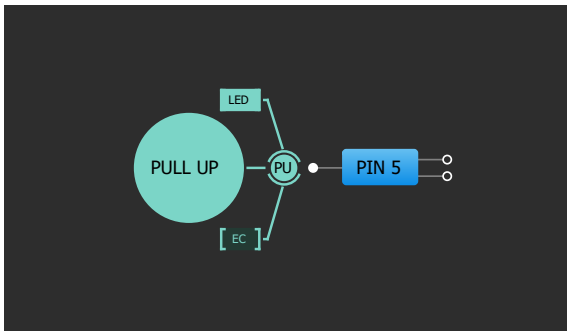


Figure 87. Buffered Led

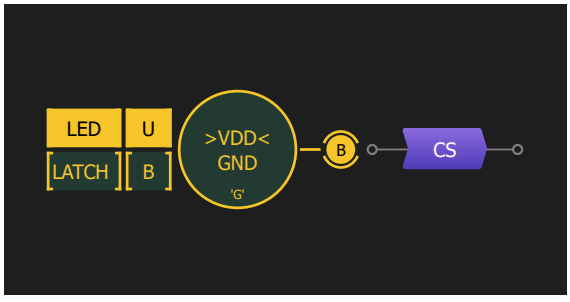


Figure 88. Power GreenPAK Buffered Led

If it is necessary to identify the chip pin connected to a specific test point on a board, use the **Test Point (TP) Map** tool. Note that the pin-to-test-point mapping varies depending on the chip type.

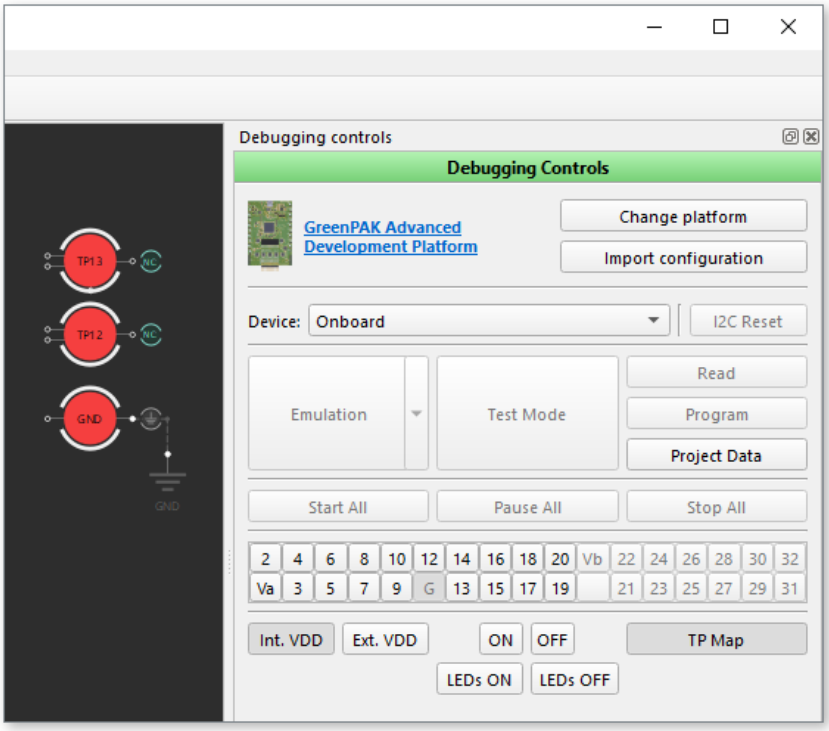


Figure 89. TP Map

Hardware sources can be divided into two categories: **basic Hardware Sources** and **Generators**. **Generators** provide a comprehensive solution for creating analog or digital signals with configurable settings and are discussed in more detail in section 2.2.2 Generators. Later in this chapter, the **basic Hardware Sources** are described.

2.2.1.1 Basic Hardware Sources

The available **basic Hardware Sources** are: **VDD**, **GND**, **Pull up**, **Pull down**, and **Button**. Unlike the others, **Button** offers additional configuration options (for the remaining basic sources, all necessary information is already described above).

The **Button** hardware source allows quick switching between two predefined states: **Upper connection (U)** and **Bottom connection (B)**. These predefined states can be set to **VDD/GND**, **High-Z**, **Pull Up**, or **Pull Down**. Hover the mouse cursor over the **Button** control to view its

configuration.

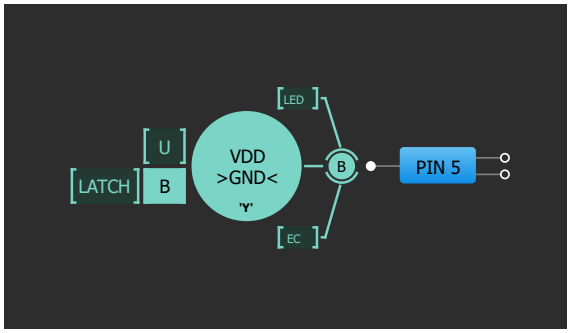


Figure 90. Configurable Button

The default connection can be set to either the **Upper connection (U)** or the **Bottom connection (B)**.

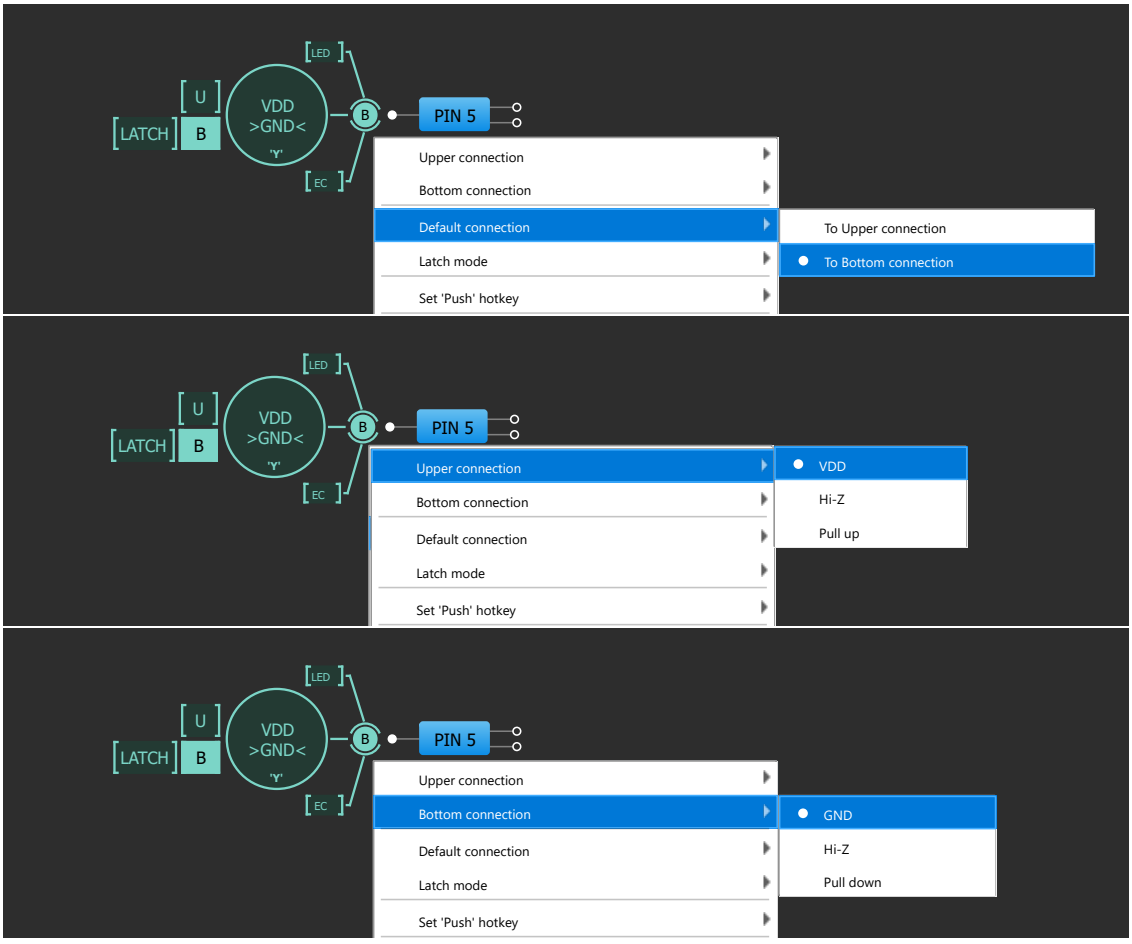


Figure 91. Default Key Connection

Latch control has two modes: **Latched** and **Not latched**. These modes can be configured from the context menu or by clicking the **LATCH** button to change the value.

In **Latched** mode, the **Button** toggles its state on click and remains in the new state until the next click.

In **Not Latched** mode, the **Button** changes its state when left-clicked and returns to its previous state after the mouse button is released.

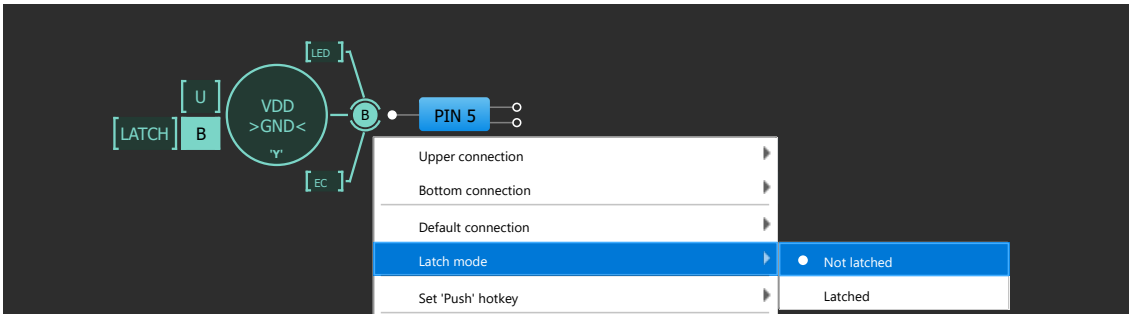


Figure 92. Key Mode

A hotkey can be assigned for the **Push** action. Pressing the hotkey is equivalent to a mouse click.

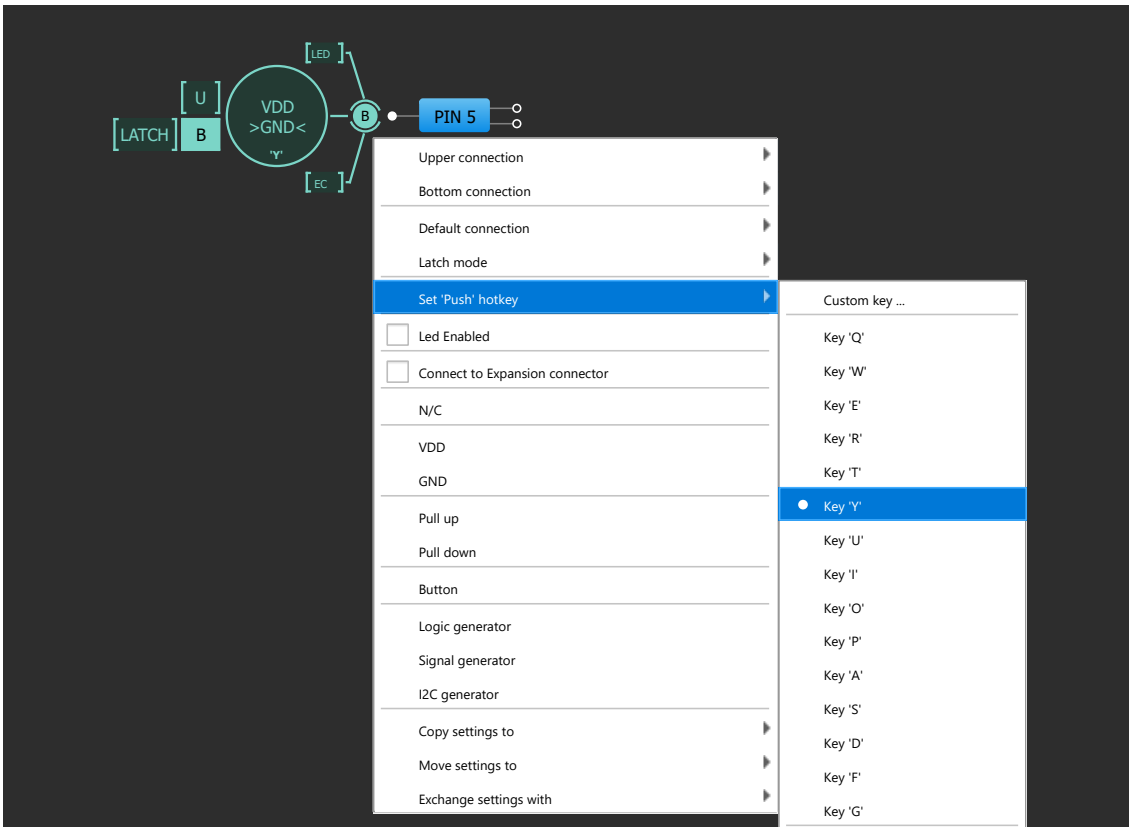


Figure 93. Choosing Hotkey

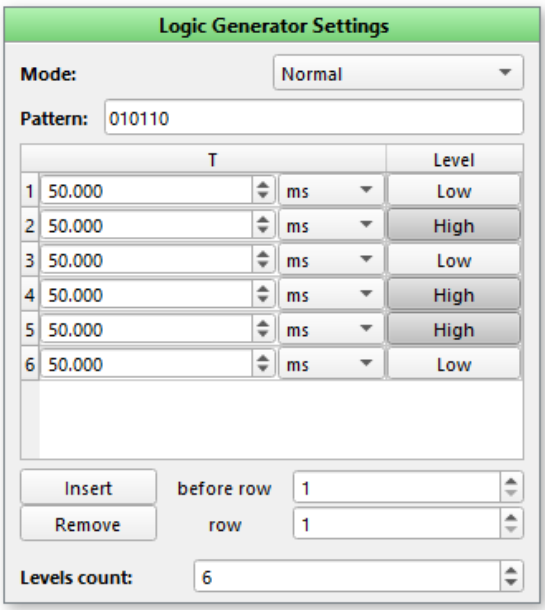
The same hotkey can be assigned to multiple **Buttons**, which allows the states of all such **Buttons** to be changed simultaneously.

2.2.2 Generators

A **Generator** is a hardware source type that produces analog or digital electronic signals on test points on a board. Go Configure Software Hub allows generator setup in an easy and convenient way with visual control of settings and states. The **Generator** can be controlled using the sticker that appears when hovering over the corresponding icon. For more advanced configuration, use the **Signal Wizard** tool. For details, refer to section 2.2.3 Signal Wizard.

2.2.2.1 Logic Generator

The **Logic Generator** generates logic pulses. A logic pulse is one of two voltages that correspond to two logic states (**low state** and **high state**, **0** and **1**).



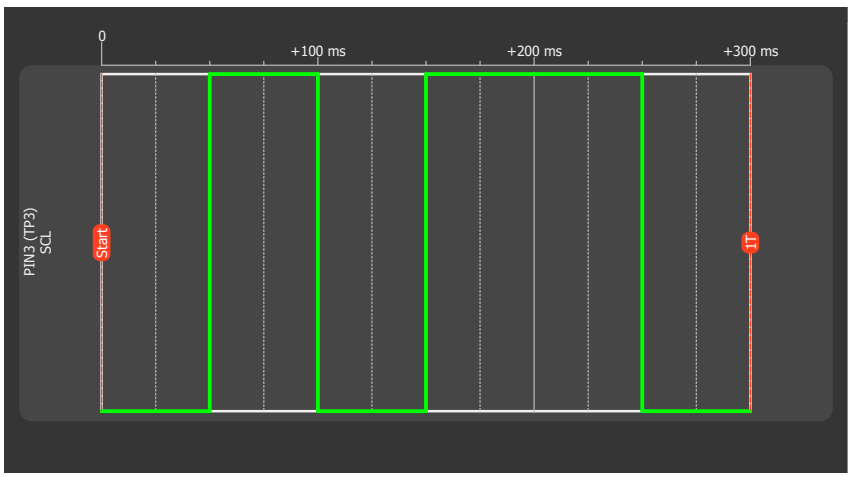


Figure 94. Logic Generator in Signal Wizard

All **Logic Generator** settings can be exported or imported. To do this, copy the **Pattern** field content and paste it into a text editor. The content is automatically converted to **XML** format text. This feature can be used to save custom generators or load them from an external file.

2.2.2.2 I2C Generator

The **I2C Generator** is intended to create **I2C** (Inter-Integrated Circuit) communication patterns based on **Logic Generators**. Two **Logic Generators** are combined as **SDA** (Serial Data) and **SCL** (Serial Clock) lines. Predefined **I2C** primitives can be combined to generate the required waveform appropriately, and the **SCL** frequency can be selected.

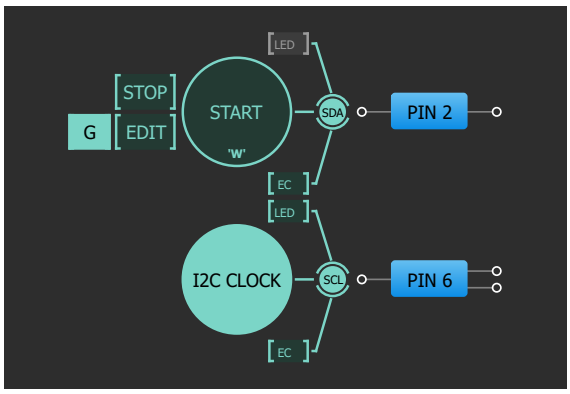


Figure 95. I2C Generator

SDA signal is a special case of **Logic Generator** used for sending data over **I2C**. The **Signal Wizard** editor shows the sequence of commands.

SCL signal is a particular **Logic Generator** that can be used for board configuration only. The **SCL** is configured by choosing a predefined frequency. The set of these frequencies depends on the development platform.

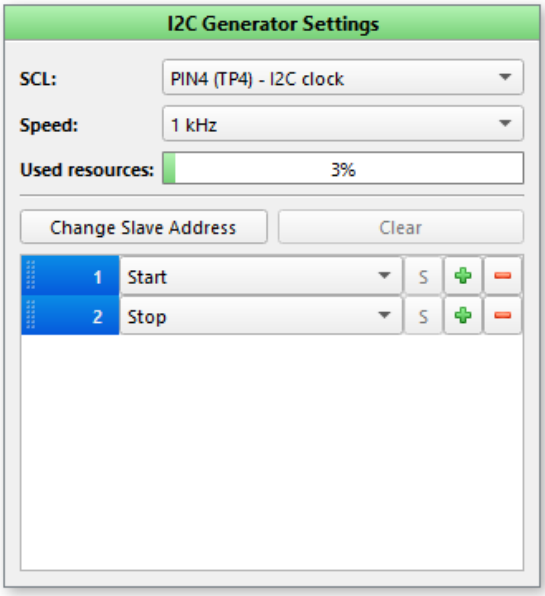


Figure 96. I2C Generator Command Editor

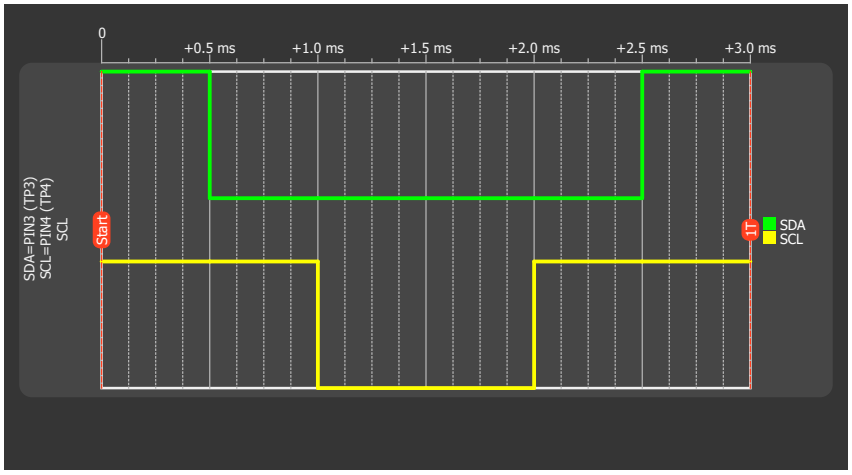


Figure 97. I2C Generator Signal Wizard

If a command type needs to be changed, click the arrow and the drop-down menu shows the available commands.

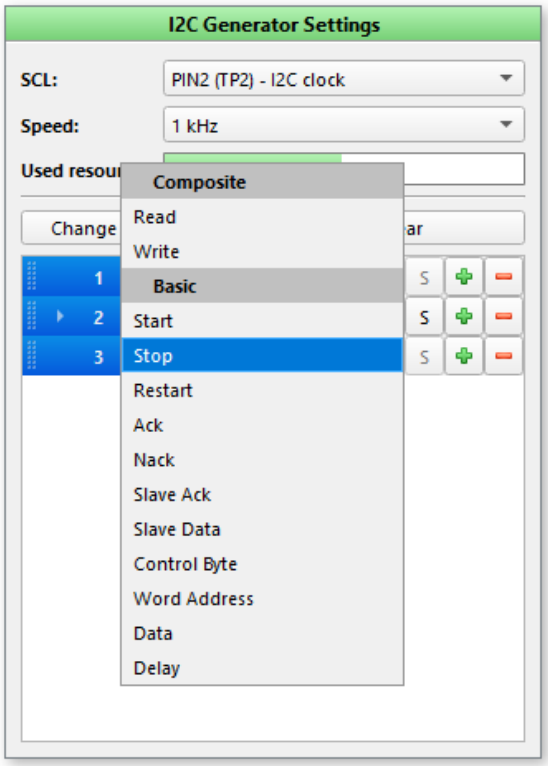


Figure 98. I2C Command List

Composite commands **Read** and **Write** may be split into a sequence of basic commands (all commands are listed in the drop-down menu above).

2.2.2.3 Signal (Analog) Generator

The **Signal Generator** produces an analog signal and can be configured using one of the following pre-defined types: constant voltage, sine, trapeze, logic pattern, or custom signal.

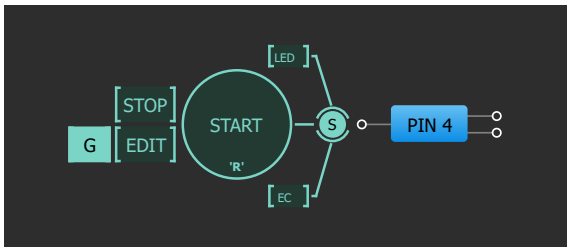
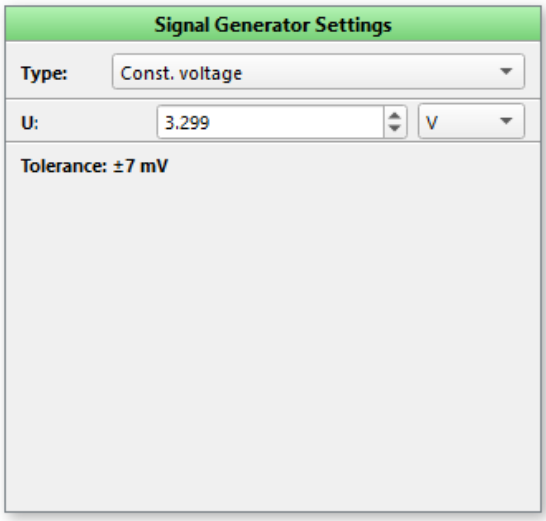


Figure 99. Signal Generator

Below is the **Signal Wizard** view for different signal types.

- **Constant voltage waveform type**



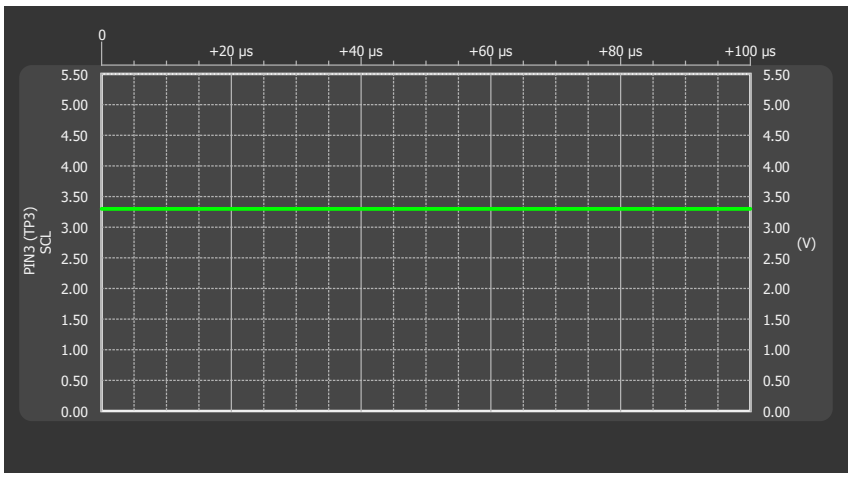


Figure 100. Configuration Options of Signal Generator, Type Constant Voltage

■ Sine waveform type

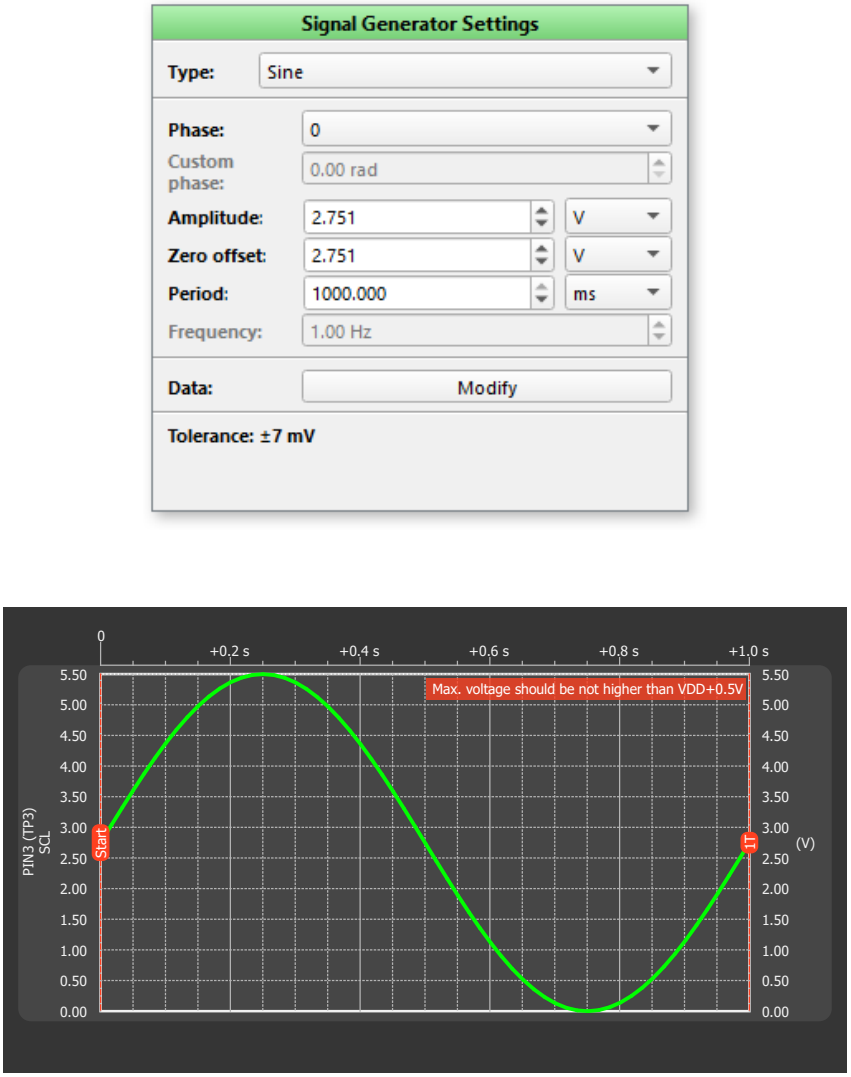
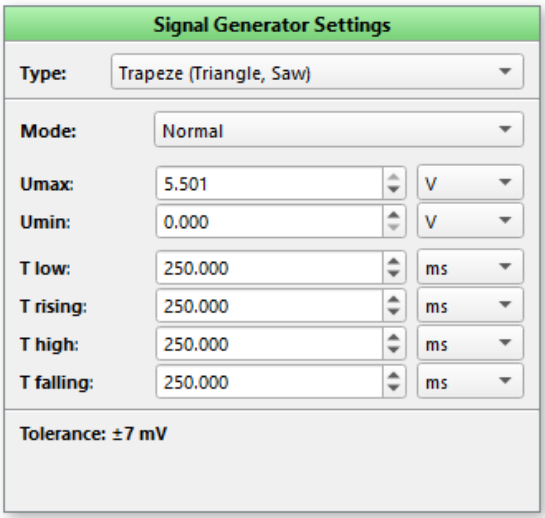


Figure 101. Configuration Options of Signal Generator, Type Sine

■ Trapeze waveform type



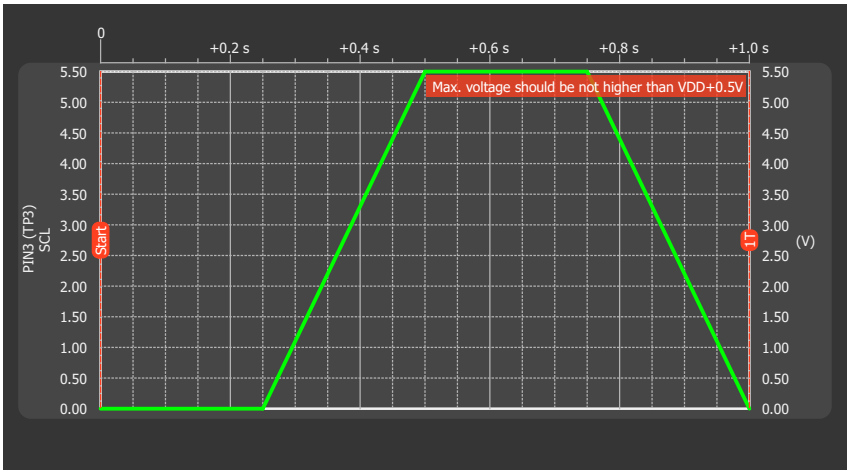
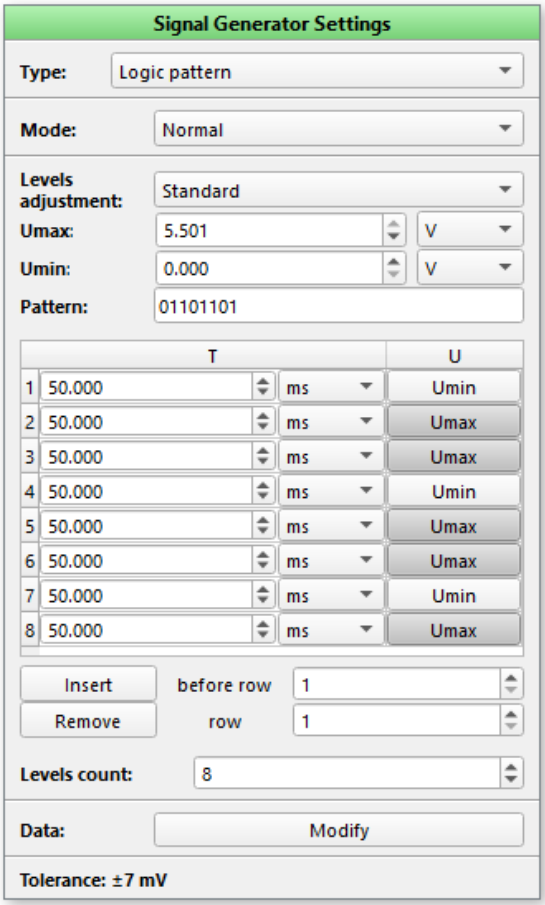


Figure 102. Configuration Options of Signal Generator, Type Trapezoid (Triangle, Sawtooth)

■ Logic pattern waveform type



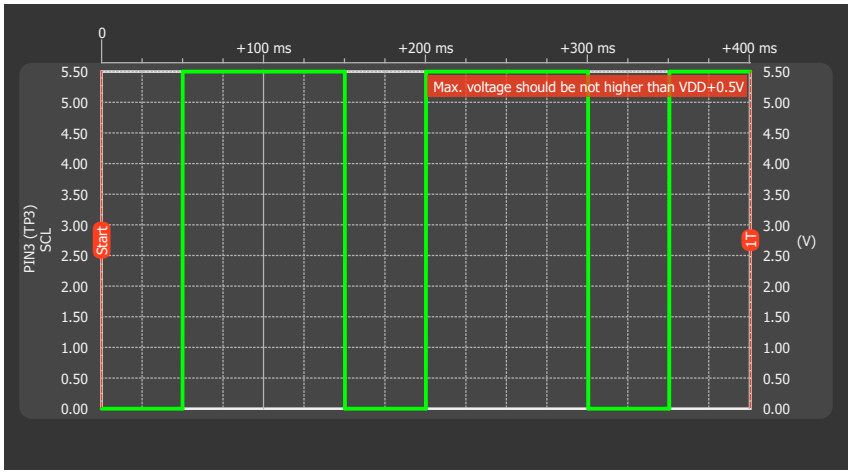
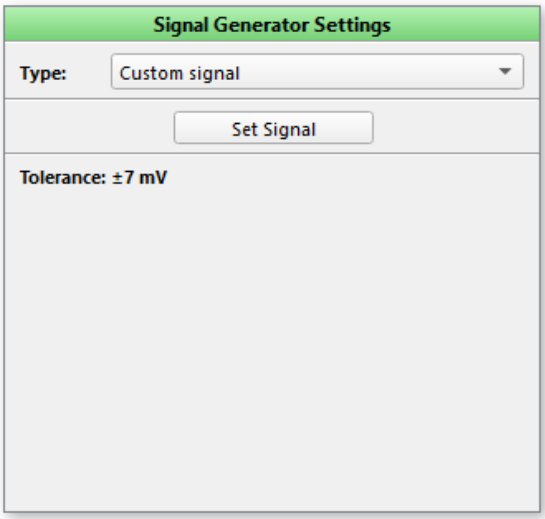


Figure 103. Configuration Options for Signal Generator, Type Logic pattern



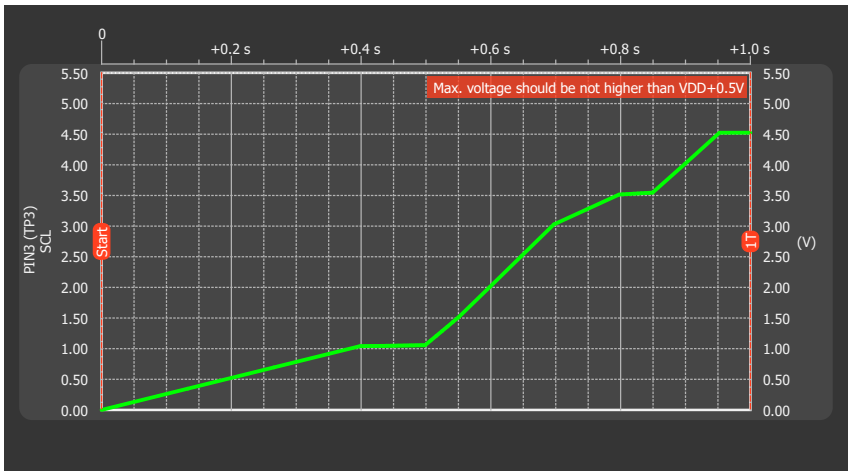


Figure 104. Configuration Options of Custom Signal (Arbitrary Waveform) Generator

The custom signal can be configured in a separate window. Find out more in section 2.2.4 Custom Signal Wizard.

2.2.2.4 VDD/VDD2 Power Signal Generator

Some part numbers have an additional power supply (**VDD2**); if present, it allows two independent voltage domains to be interfaced within the same design. The pins dedicated to each power supply can be configured as inputs, outputs, or both (controlled dynamically by the internal logic) for the **VDD** and **VDD2** voltage domains. Using the available macrocell, mixed-signal functions bridging both domains can be implemented, or level translation can be passed through in **HIGH** to **LOW** and **LOW** to **HIGH** directions.

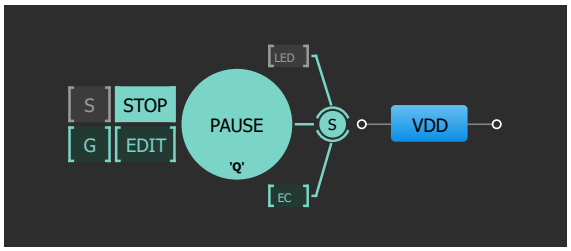


Figure 105. VDD/VDD2 Power Signal Generator

If the **Sync Power Rails [S]** mode is enabled in **Signal Wizard**, **VDD** and **VDD2** share the same power settings. The option is available only for **VDD** / **VDD2** power generators.

2.2.2.5 Synchronous Logic Generator

The **Synchronous Logic Generator** is used for generating logic pulses and waveforms on GPIO pins. It is a 64-channel digital pattern Generator, provided only by ForgeFPGA Deluxe Development Board.

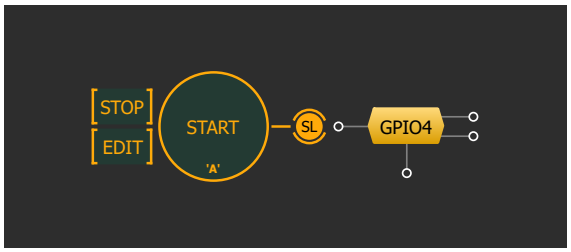


Figure 106. Synchronous Logic Generator

The efficiency of this generator can be improved by using the **Signal Wizard**, which offers two types of resources: **used bandwidth** and **used points**. **Used bandwidth** refers to buffer occupancy and depends on **points density**.

The top complex graph displays two metrics: a green line for **points density** (points per second) and a yellow line for **used bandwidth**. The peak value of the **used bandwidth** line correlates with the used bandwidth value in **Parametric Generator Settings**. This graph precisely summarizes resource usage in all waveforms displayed below over one period.

The screen view can be adjusted and graph visibility can be toggled using the button in the bottom right corner of the window.

While the graph at the top of the window represents all waveforms collectively, the indicators to

the right of the generators display information for each channel individually. Each channel has two different indicators:

- **Points Usage Indicator** – Shows the number of points a channel is using.
- **Resource Percentage Indicator** – Displays the used bandwidth for each channel.

Both indicators are color-coded and change color depending on their values or percentages. Legends are available for each indicator to check the value range.

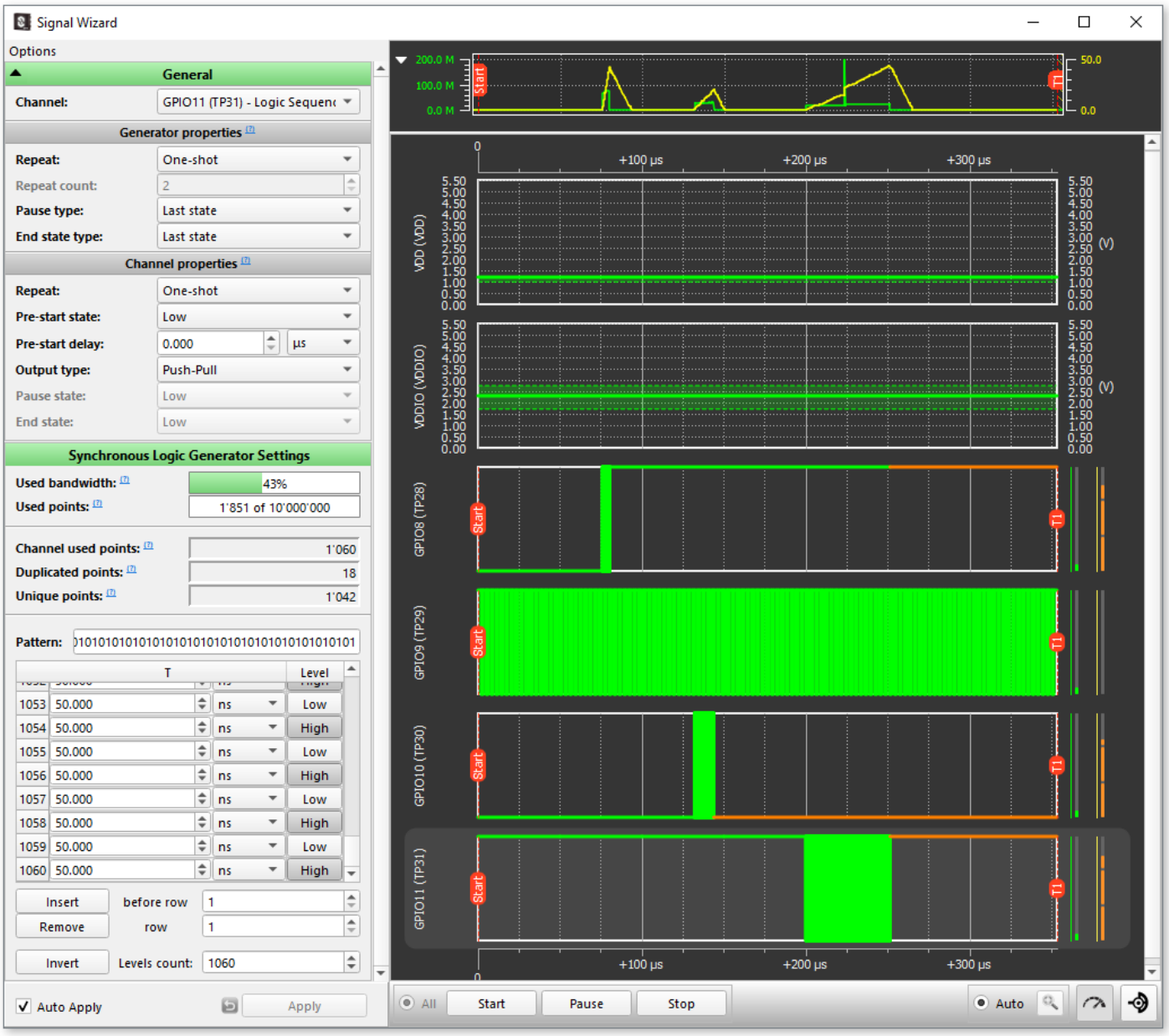


Figure 107. Synchronous Logic Generator Window

The **Used points** metric displays the cumulative number of points used by patterns across all channels. A point represents any change in level across any of the 64 channels, including pre-start delay and endstate. However, if two or more channels change levels simultaneously, it is counted as one generator point.

The Wizard also helps in understanding each channel individually.

- **Channel used points** combine the duplicated points and unique points in a selected channel
- **Duplicated points** show the total duplicated points currently used by the generator
- **Unique points** represent the total unique points currently used by the generator

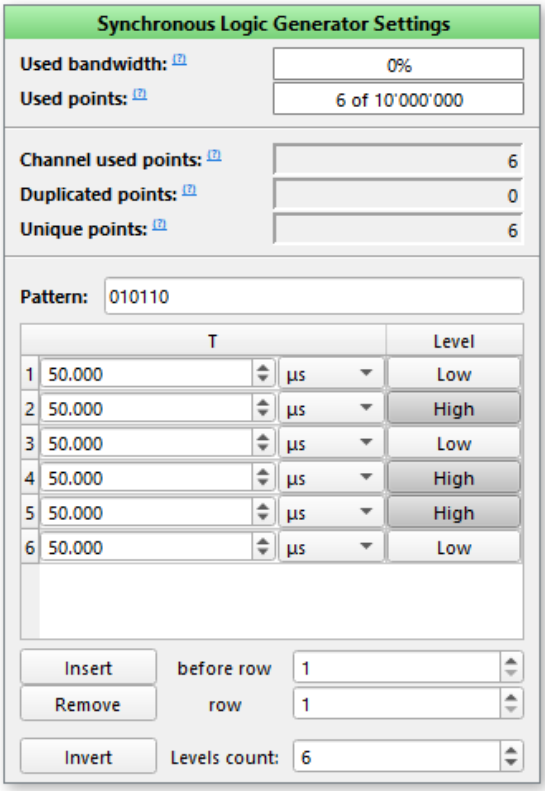


Figure 108. Synchronous Logic Generator Properties

2.2.2.6 Parametric Generator

The **Parametric Generator** generates logic pulses that follow different protocol sequences. This type is also available for ForgeFPGA Deluxe Development Board.

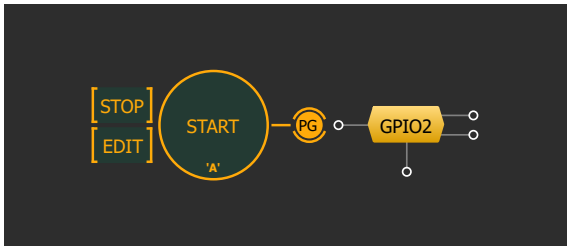


Figure 109. Parametric Generator

In **Signal Wizard**, a special editor shows the sequence of commands.

Actions available in the command editor:

- Add or remove commands by clicking ✚ and ▬.
- Change command parameters.
- Change the order of commands by dragging the command to another position.

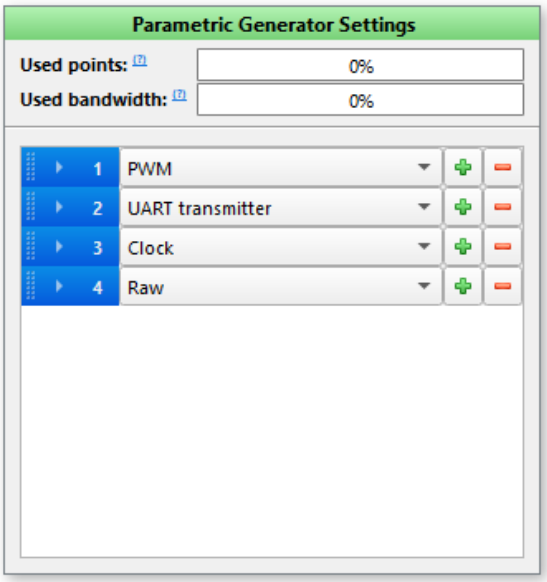


Figure 110. Parametric Generator Command Editor

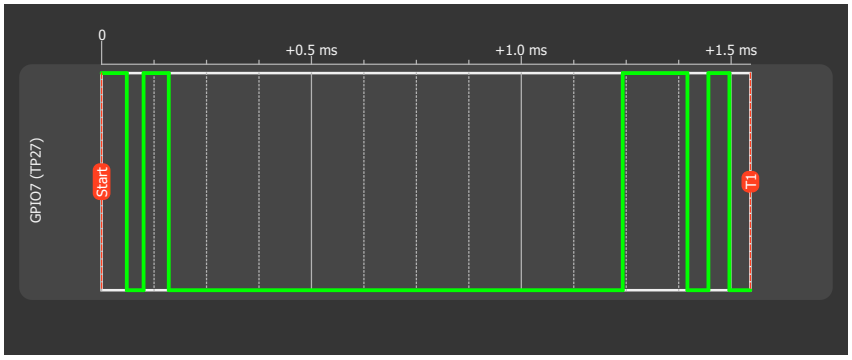


Figure 111. Signal Wizard for the Parametric Generator

The list of available commands:

- **PWM** (Pulse Width Modulation) – The PWM **command editor** has three input fields:
  - **Period** – A united duration of high and low states per repeat.
  - **Duty cycle** – The percentage of the total duration in the high state.
  - **Repeats** – Pattern repeat count.

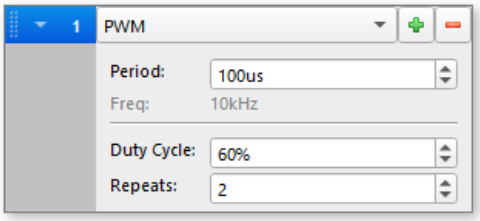


Figure 112. PWM Command Editor

- **Clock** – The command generates a signal oscillating between a high and a low state. The editor has two input fields:
  - **Period** – The united duration of high and low states per repeat.
  - **Repeats** – Pattern repeat count.

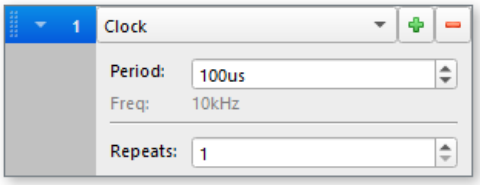


Figure 113. Clock Command Editor

- **UART Transmitter** – Generates a signal according to the **UART** standard:
  - **Baud rate** – Signal frequency.
  - **Data frame** – Number of bits allocated for user input.
  - **Data** – User input in hex format. If the data frame is lower than the number of bits required to represent the data, the more significant bits are ignored.
  - **Parity bit** – Create a parity bit for error detection.
  - **Stop bit size** – Duration of the stop bit.
  - **Bit order** – Serial data transfer format.

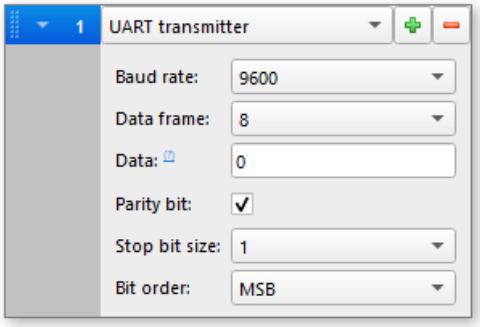


Figure 114. UART Transmitter Command Editor

- **Raw** – This pattern works as a typical **Logic Generator**.

2.2.3 Signal Wizard

To start configuring a **Generator**, double-click its icon next to the pin or find **Edit** in its context menu.

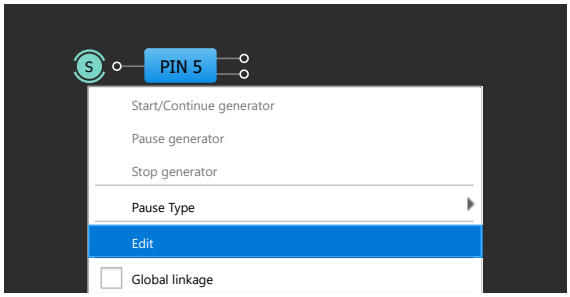


Figure 115. Opening Signal Wizard

The **Signal Wizard** window contains a **Generator** settings panel and the plot (waveform preview). The window is common to all **Generators**, although different sets of settings may be activated, depending on the selected hardware source.

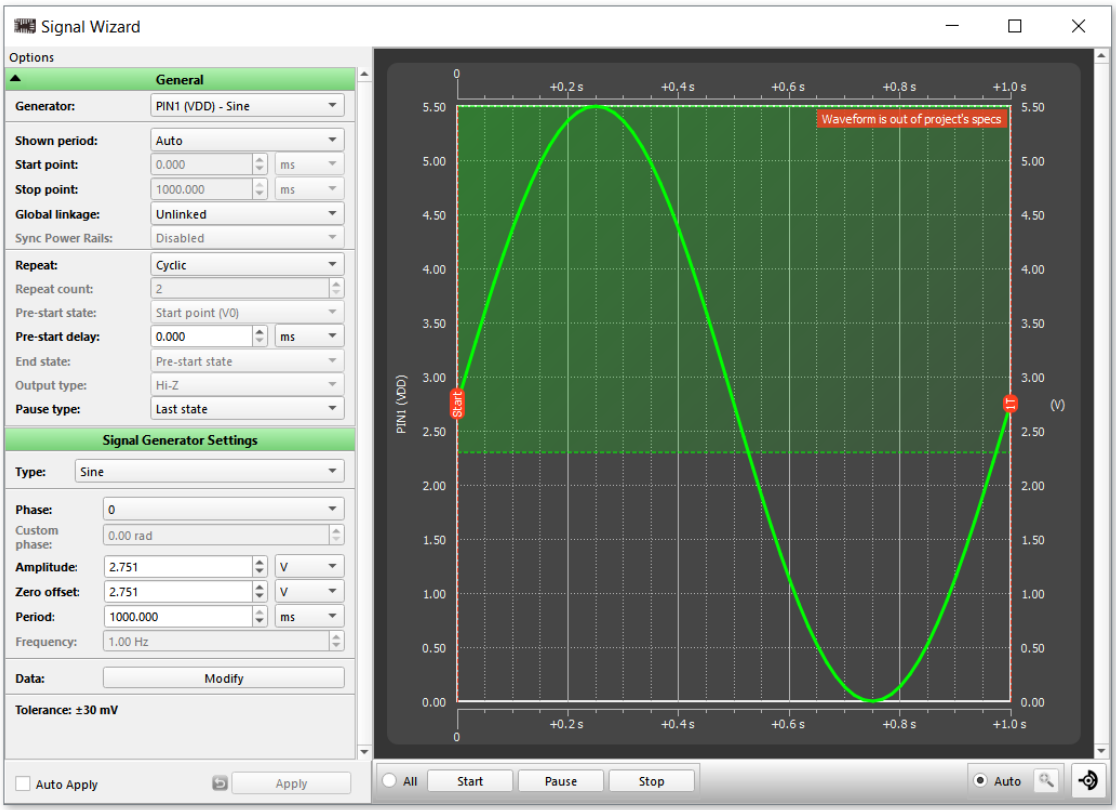


Figure 116. Signal Wizard Window

General settings category mostly provides signal configurations related to time, period, and state. Below are the settings that may require a more detailed description.

- **Global Linkage** – When enabled, allows the **Generator** to be controlled using the **Start**, **Stop**, and **Pause** buttons on the **Debugging Controls** panel.
- **Sync Power Rails** – Enable for VDD and VDD2 to share the same power settings. The option is available only for **VDD / VDD2 Power Generators**.

The lower settings section is generator-specific. Read more in section 2.2.2 Generators.

2.2.3.1 Scaling Controls

Scaling controls are located at the bottom right corner of the **Signal Wizard**. The controls allow adjustment of the number of periods shown in the waveform preview.

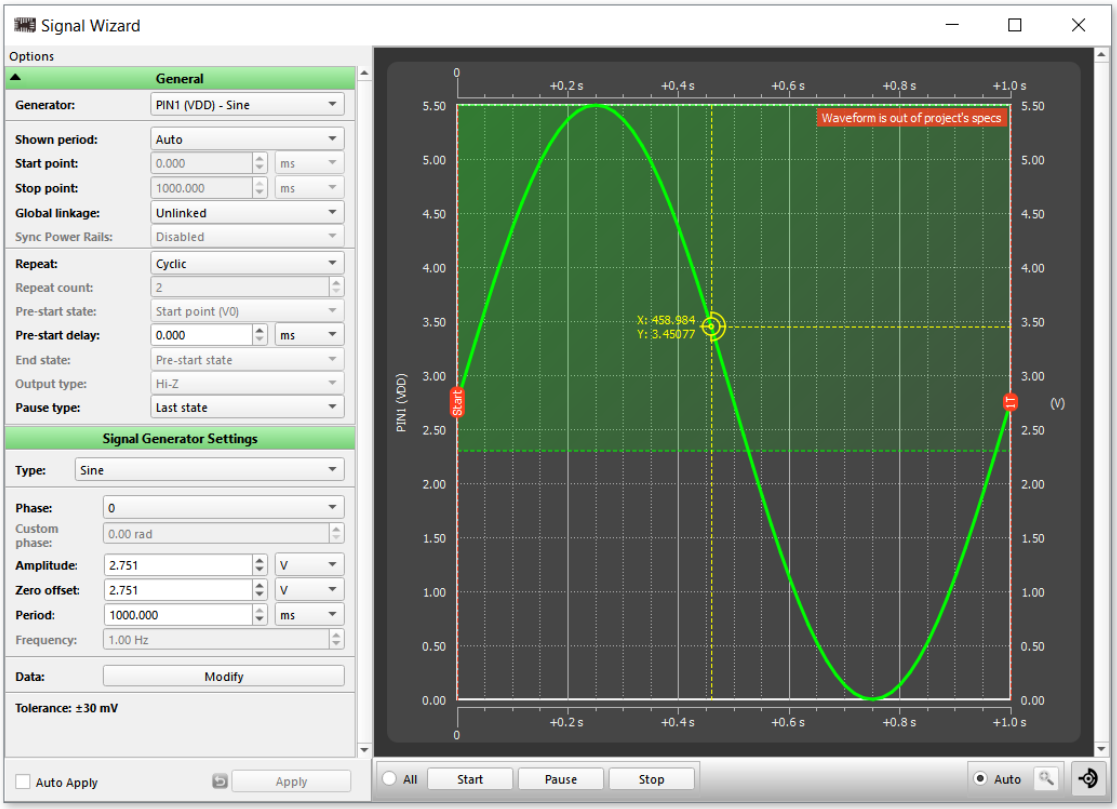


Figure 117. Scaling in Signal Wizard

The **Cursor** button can be used to show or hide the mouse coordinates in the timing diagrams.

- **Auto** – Scale all generators to fit the largest period among them. Only generators with **Shown period** set to **Auto** are affected.
- **Custom** – Waveforms can be rescaled manually. To do that, ensure that **Auto** mode is OFF and then scale by **Ctrl** + **mouse wheel**.

2.2.4 Custom Signal Wizard

Custom Signal Wizard allows creation, import, and editing of signal waveforms (applicable only for **Signal (Analog) Generators**). The signal can be created by manually adding points or by importing a custom list of points from an external source.

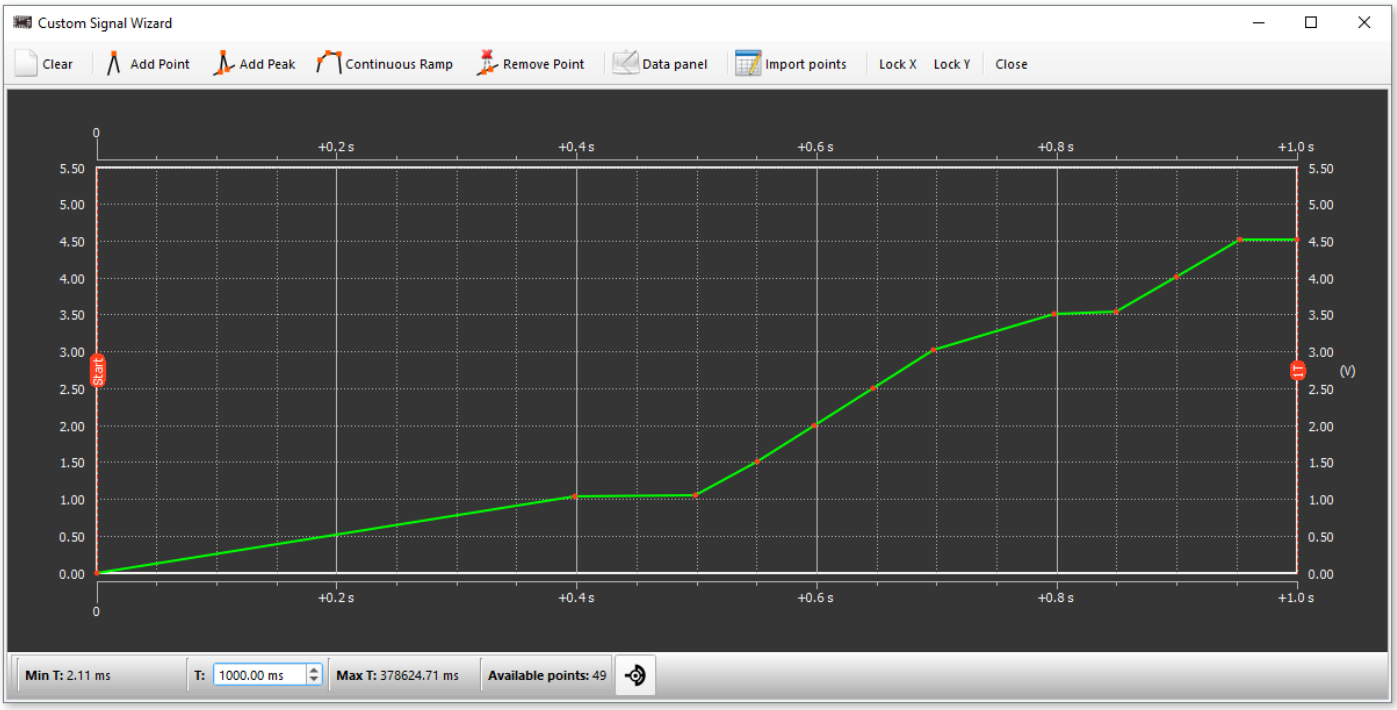


Figure 118. Drawing Signal (Arbitrary Waveform)

Use the controls on the toolbar to manipulate the waveform.

- **Add Point**, **Peak**, and **Continuous Ramp** – Use the buttons to start creating the signal waveform.
- **Remove Point** – Click the button to remove a selected point (or use a right-click).
- **Data Panel** – Show or hide the data table. Values for a selected point can be removed or modified, and a point can be added in between.
- **Import Points** – Click the widget to open the **Import** window and insert the point

coordinates.

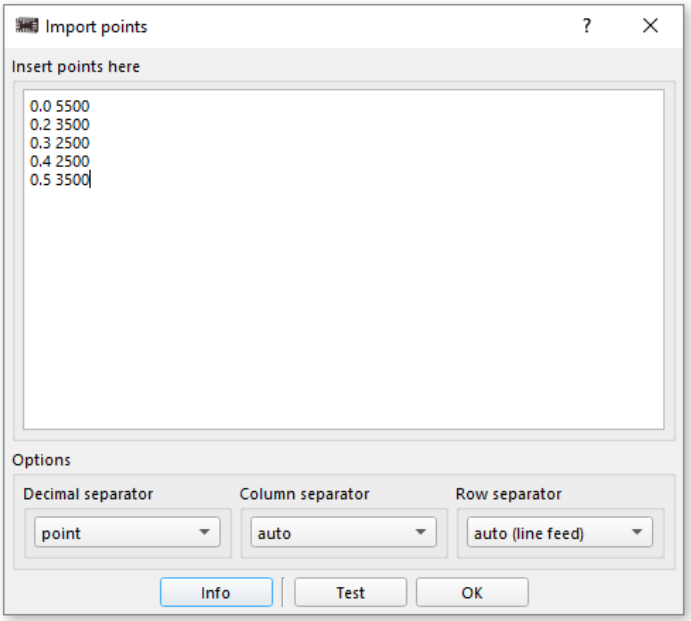


Figure 119. Import Points

See the coordinate formatting options:

- **Decimal separator**: point or comma.
- **Column separator**: auto, tab, or other.
- **Row separator**: auto (line feed), tab, or other.

2.2.5 Hardware Configurations

Configuration state snapshots can be created to restore a previous configuration when needed. The snapshots include hardware source configurations and are saved and loaded with the project file. This option is available for both **Generators** and **basic Hardware Sources**.

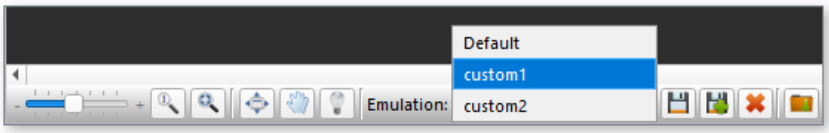


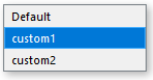
Figure 120. Saved Configurations

 Save the current configuration state.

 Save a new configuration state.

 Delete the selected configuration.

 Import a new configuration.



The list of saved configurations.

2.2.6 Debugging Controls

To access the required development platform through the software, click **Debug** and select the board from the list of supported platforms.

The **Debugging Controls** panel appears. It contains the UI controls for chip communication and programming, brief information about the connected devices, and other features described below.

Different **Debugging Controls** panel types may be encountered in the software. Within the **Standard** debugging controls, built-in validation checks are provided. The list includes, but is not limited to, the following:

- IC detection check – It is verified whether the IC in the socket has the correct part number and revision.
- Socket connection test – The software performs a validation check to confirm the chip's connection status.
- Chip state analysis – The software verifies the chip's state to ensure proper execution of procedures, including checking whether the chip is empty or programmed, and whether the chip protection configuration permits the operation.
- VDDs range validation – The software ensures that the chip operates within the required VDD voltage range.

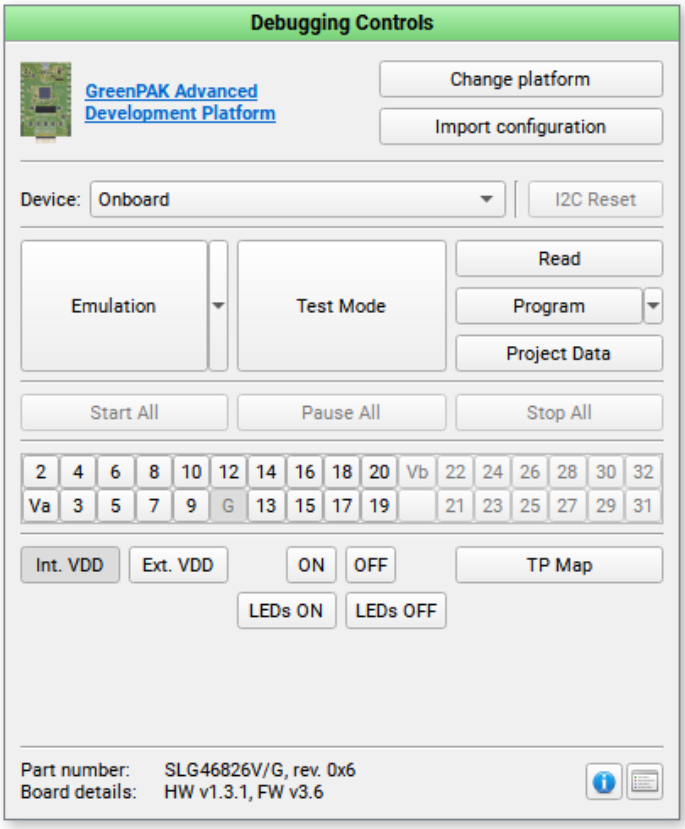


Figure 121. Standard Debugging Controls Panel Example

The **Expert** panel provides the same set of features without the additional validation checks.

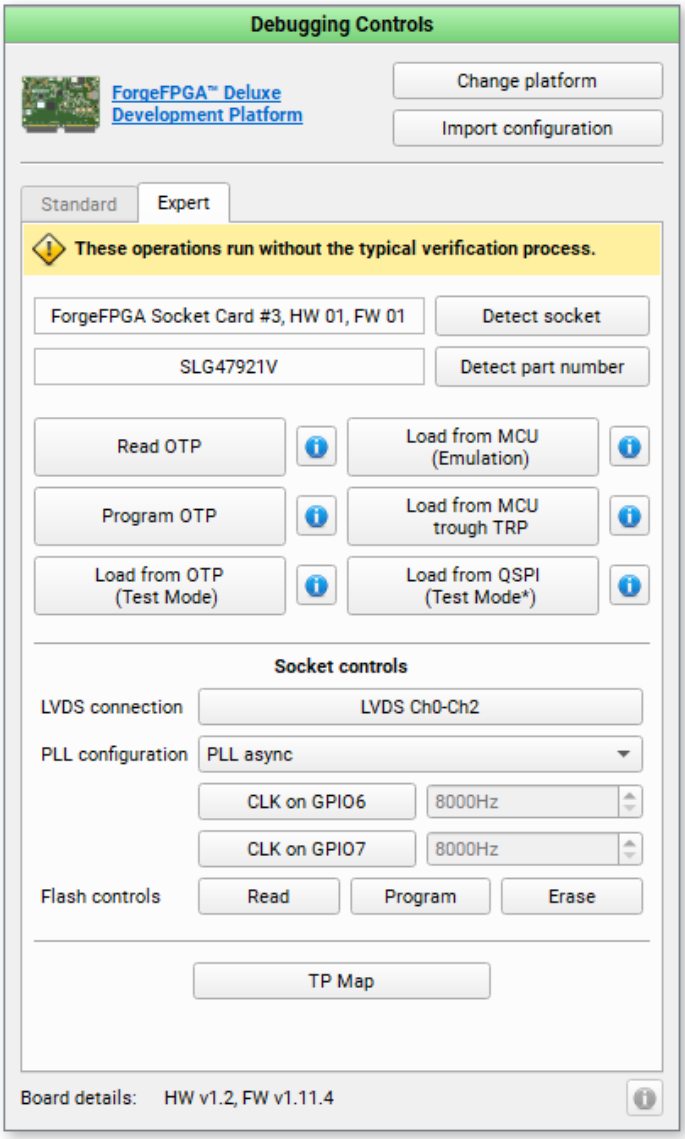


Figure 122. Expert Debugging Controls Panel Example

Later in this section, all **Debugging Controls** panel UI elements that may be encountered while working with different platforms and ICs are described. In chapter 3 Devices, the supported development platforms and the **Debugging Controls** panel for each board are presented.

The table listing **Debugging Controls** element availability for all boards is given in the appendix, in section 6.3 Debugging Controls Feature Availability.

Use the references below to quickly find the **Debugging Controls** panel interface for a specific platform.

- GreenPAK Advanced Development Board
- GreenPAK Lite Development Board
- GreenPAK DIP Development Board
- GreenPAK Serial Debugger Board

- ForgeFPGA Deluxe Development Board
- ForgeFPGA Evaluation Board
- Power GreenPAK Development Mother-board
- Power GreenPAK Demo Board
- GreenPAK Serial Debugger Board (with SLG5100x)
- Go Configure Development Board

2.2.6.1 Debug Configuration

- **Recommended platform configuration** – Find information about the supported adapter, development board, and device ordering information. Click the platform name link to open the **Recommended platform configuration**

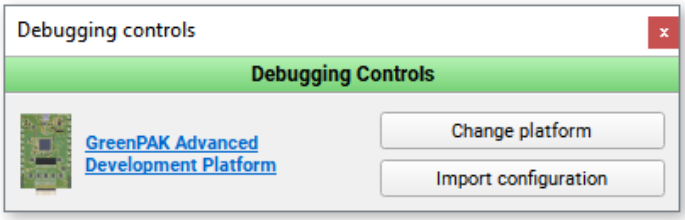


Figure 123. Recommended Platform Configuration

- **Change platform** – Open the list of development platforms supported by a part number.
- **Import configuration** – Upload configurations from a different development platform supported by a part number.

2.2.6.2 Device Selector

- **Onboard**; **Chip in socket** – Search for the device placed on the board.
- **External** – Search for the device on the selected I2C slave address.
- **Auto detect**; **External, connected to 'I2C OUT'** – Search for the device on all available I2C slave addresses, starting from 0 until the first one is found.

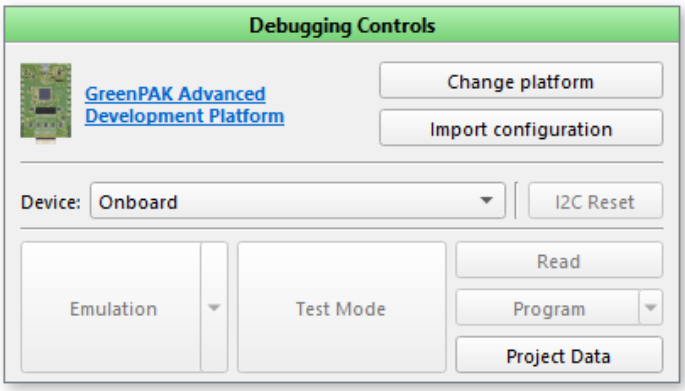


Figure 124. Device Selector

Note: Make sure the socket is not connected to the platform and to an external chip at the same time.

2.2.6.3 External Device Modes

- **GPSD** – Use a separate 4-pin connector (PWR, SCL, SDA, GND) for I2C communication.
- **ECs** – Use the expansion connector for I2C communication:
  - **VDD EC** – Power.
  - **SCL** and **SDA** – EC according to the chip TP map.

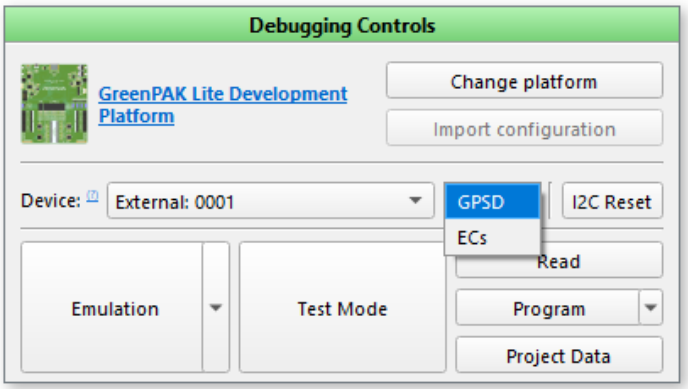


Figure 125. Lite Board Device Selector

2.2.6.4 Chip Procedures

- **Test Mode** – Debug the programmed project. Enable power and load the configured hardware sources.
- **Load from OTP** – Same as **Test Mode** (**Expert** procedure performed without additional validation checks).
- **Emulation** – Debug the current project. Enable power and load the design with configured hardware sources.
- **Load from MCU** – Same as **Emulation** (**Expert** procedure performed without additional validation checks).
- **Load from through TRP** – Same as **Emulation**, but the procedure sequence goes through TRP (**Expert** procedure performed without additional validation checks).
- **Emulation(sync)** – The project changes are automatically loaded to the chip when the control is active.

  Note: **Emulation** bypasses the standard chip startup sequence and is not suitable for normal boot behavior verification. To observe the standard startup process, use **Test Mode** with a pre-programmed chip.

- **Sync** – Load the current project to the chip once.
- **Write** – Load the current project to the chip once.
- **Write (sync)** – Similar to **Write**. Any changes to the project made during the **Write (sync)** are immediately applied to the chip.
- **Read** – Read the programmed chip and open the project in the new software instance or in the **Project Data** window of the current instance.
- **Program** – Program the chip with the current project. For some part numbers, such as SLG47004, **EEPROM** programming is available.
- **I2C Reset** – Change the I2C reset bit (from `1` to `0`). This causes the device to re-enable the Power-On Reset (POR) sequence, including reloading all register data from NVM.

2.2.6.5 Flash Procedures

Flash memory is located on the FPGA sockets. See the available controls for working with it.

- **Test Mode(*)** – Load data from flash to the chip.
- **Load from QSPI** – Same as **Test Mode(*)** (**Expert** procedure performed without additional validation checks).
- **Read** – Read the chip project from flash memory.
- **Program** – Program flash memory with the current project.
- **Erase** – Clear flash memory (erased flash memory address values are read as 0xFF).

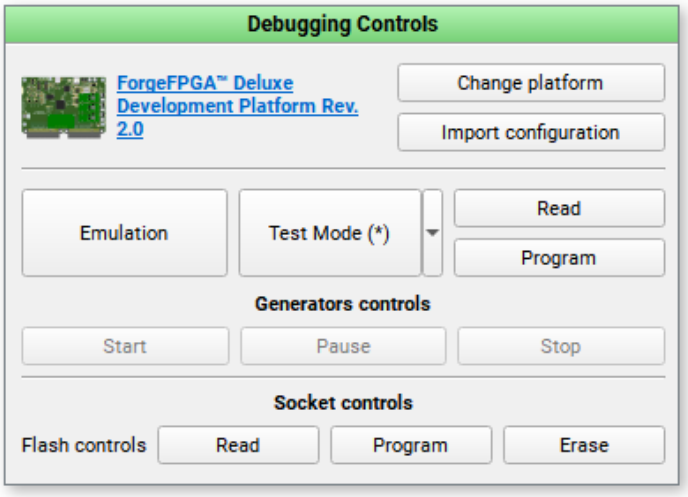


Figure 126. Flash Procedures

2.2.6.6 NVM Programmer Window

- **Project data** – A table with **NVM** or **EEPROM** bit sequences. Bit values can be changed, sequences can be imported or exported, and the data can be used to program the chip (for more information, see section 2.2.7 NVM Programmer Window).

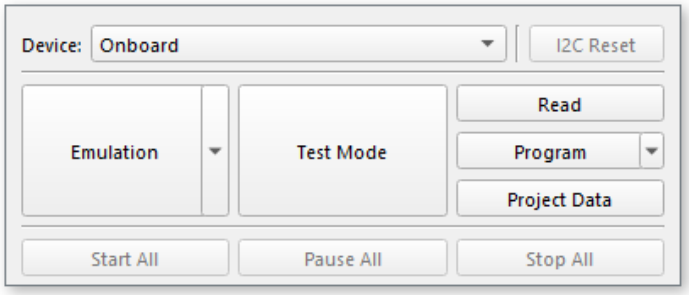


Figure 127. Project Data Control

2.2.6.7 Generator Controls

- **Start**, **Pause**, **Stop All** – Control the generator state while **Global linkage** generator setting is enabled.

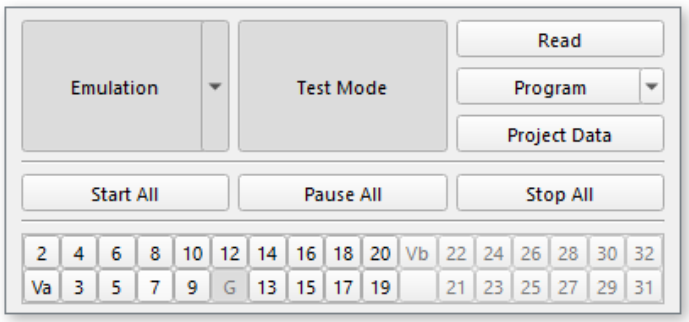
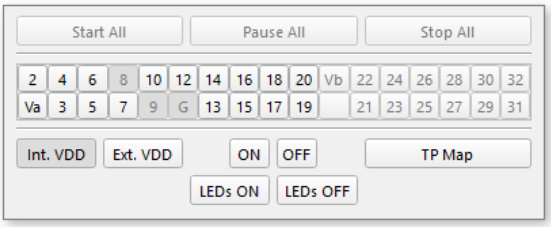


Figure 128. Generator Controls

2.2.6.8 Expansion Connectors (EC)

- **Expansion Connectors** – This port was designed to connect the platform to the external circuits and apply external power, signal sources, and loads. There are several ways to connect or disconnect the expansion connectors (also, control the ECs on the work area using a hardware source):
  - Connect or disconnect all ECs by clicking the **ON** or **OFF** button.
  - Connect or disconnect a specific EC by clicking the ECs **sequence number** button.

- Connect or disconnect **Va EC** by clicking the **Ext. VDD** button.



2.2.6.9 Power Source Selector

- **Internal VDD** – Power is provided by the GreenPAK board.
- **External VDD** – The board measures voltage on the active power connector and provides the same voltage level.

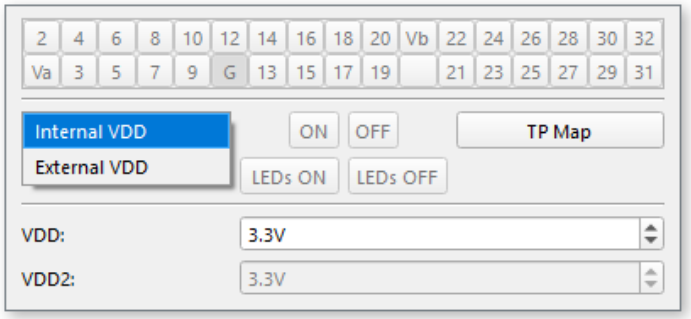


Figure 129. Power Source Selector Example

2.2.6.10 VDD2

Power supply voltage on VDD2 pin for BLDC Motor load or RGB LED load. The selected voltage determines the operating power level.

Note: The VDD2 power supply is automatically disabled if the maximum current limit is reached. To restore operation press **Reset** on the board or reconnect it.

Figure 130. VDD2 Option

2.2.6.11 TP Map

- **TP map** – Show the test point map on the work area to reflect the physical test points on the development platform.

Figure 131. Test Point Control

2.2.6.12 LEDs on and LEDs OFF

- **LEDs ON** and **LEDs OFF** – Enable or disable all LEDs on the current platform (applicable only for development platforms with LED support). A particular LED can be controlled on the work area using a hardware source.

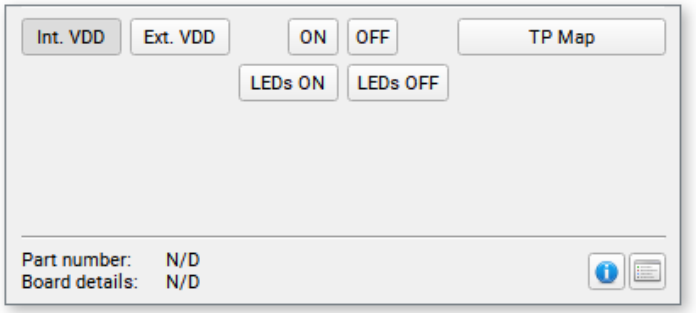


Figure 132. LEDs ON and LEDs OFF Controls

2.2.6.13 Voltage Level Controls

- **VDD** – Set the voltage level on the corresponding test points. If a selected value exceeds any board limitation, the dependent controls are blocked and a warning is shown.

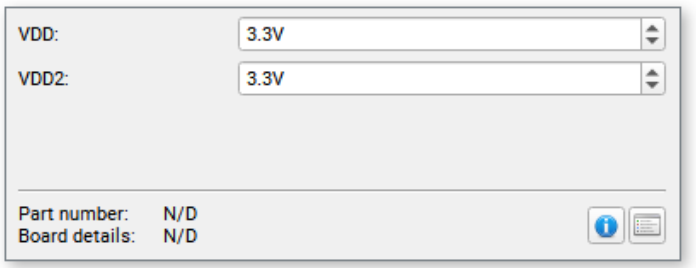


Figure 133. Voltage Level Controls

2.2.6.14 Power GreenPAK Voltage Controls

- **VDD**, **VDDIO**, **VDDLV** – Power supply voltage for the overall chip and GPIOs.
- **VIN** – Power supply voltage on the corresponding LDO component's VINs.
- **VINs ON**, **VINs OFF** – Enable or disable all VINs.

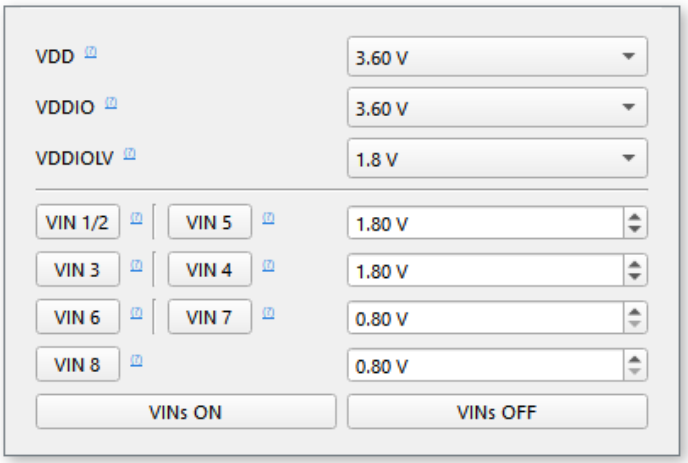


Figure 134. Power GreenPAK Voltage Controls

2.2.6.15 GPIO SW Control

- **GPIO SW Control** – Add buttons for software control of **GPIOs**, **GPIO(SDA)**, **GPI(SCL)**, and **CS**. Read more in section 2.2.1 Hardware Sources

Note: the **GPIO SW Control** feature is permanently enabled in the Power GreenPAK Development Motherboard. For the SLG51002CTR Demo Board, the feature can be activated using the corresponding button in the **Debugging Controls**.

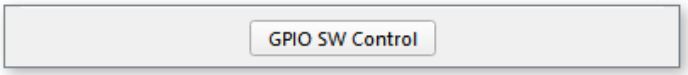


Figure 135. GPIO Software Control

2.2.6.16 Socket Controls

- **LVDS connection** – The LVDS setting enables LVDS switches, configuring all channels simultaneously.

- **PLL configuration** – The PLL generator operates in two modes: sync and async. In sync mode, both generators start simultaneously with a 90-degree phase shift at a set frequency. In async mode, each generator can be enabled or disabled individually and operates independently.

Note: The described socket settings are applied dynamically during the procedure and can be

modified on the fly.

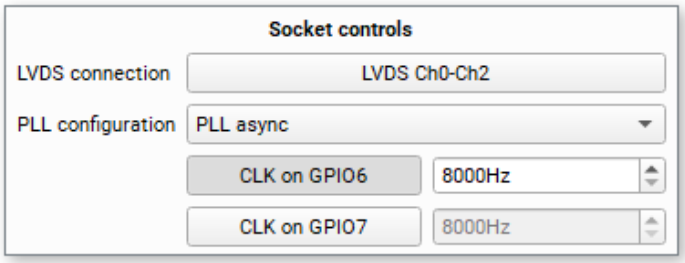


Figure 136. Socket Controls

2.2.6.17 External Bitstream

The option enables the use of a bitstream from another project for Emulation (MCU), Programming (OTP), and configuration from flash (FLASH).

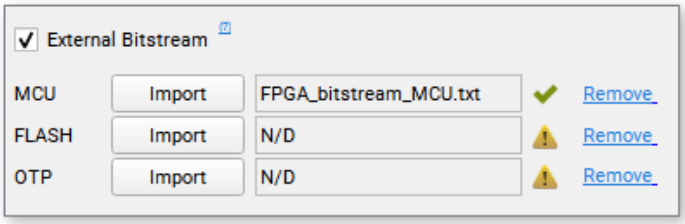


Figure 137. External Bitstream option

2.2.6.18 Motor Measurements Table

This table provides feedback on motor performance and power conditions, helping to monitor and verify motor behavior, where:

- **Motor Speed** – Displays the current speed of the motor in revolutions per minute (RPM).
- **Motor Current** – Indicates the amount of electrical current the motor is drawing, which reflects the motor's load or torque demand.
- **Motor Voltage** – Shows the voltage supplied to the motor for verifying correct power delivery and identifying undervoltage or overvoltage conditions.

Motor measurements	Value
Motor Speed	1665rpm
Motor Current	40mA
Motor Voltage	0.735V

Figure 138. Motor Measurements Table

2.2.6.19 Color Picker

This tool allows configuring RGB LED color output by adjusting the PWM duty cycles for the red, green, and blue channels, with the resulting color shown visually and as an HTML hex code. Click **Write color** to load the specified Duty Cycle values to the corresponding PWM registers.

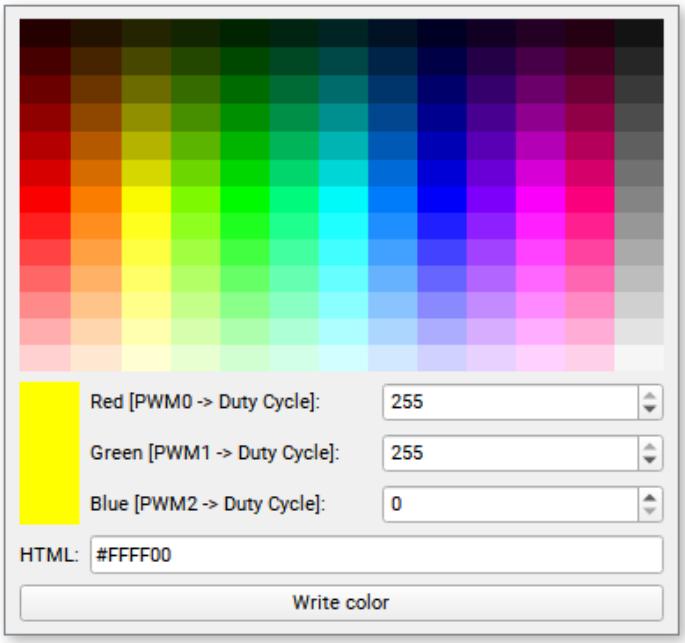


Figure 139. Color Picker Tool

2.2.6.20 Initial Design

The button opens a new software instance containing the preprogrammed design used on the demo board.

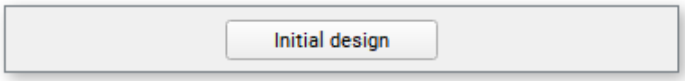


Figure 140. Initial Design Button

2.2.6.21 Info Details

Brief information about the last detected chip (where, for example, V/G is a package name and 0x6 is a metal revision) and platform.

Note: A metal revision of a chip is an updated design that changes only the metal interconnect layers, leaving the underlying silicon unchanged.

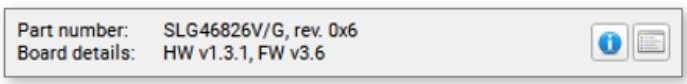


Figure 141. Info Details

Information about the connected part number, development platform, software version, and operating system.

Show operation log.

2.2.6.22 Board Selector

After the board is connected, the platform information appears on the bottom toolbar. To switch to a different platform, use the Board selector.

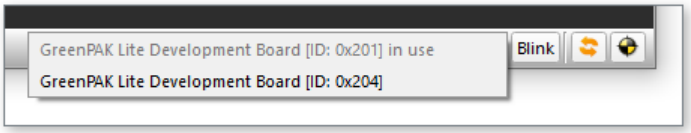


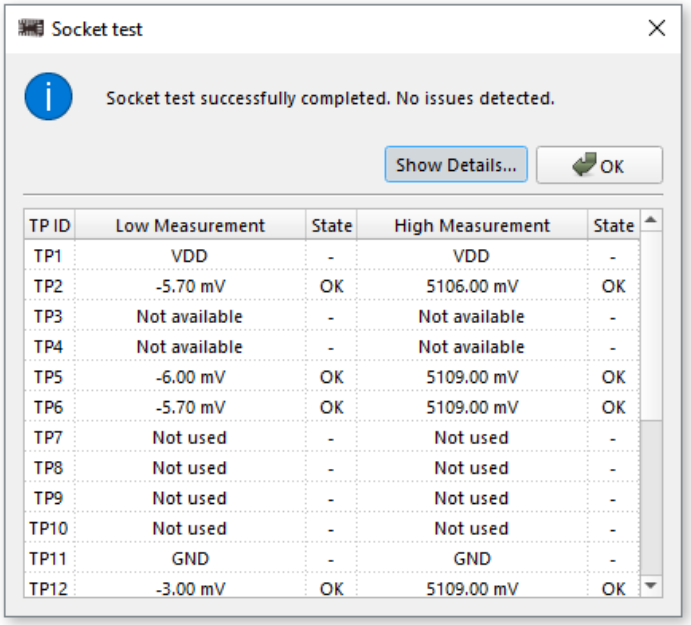
Figure 142. Board Selector

Blink Blink with an LED of the board in use.

Update the chip and board **Info details**.

Run the **Socket Test**.

The **Socket Test** checks whether the chip is successfully connected to the board. Click **Show Details** to view the **Socket Test** report.



TP ID	Low Measurement	State	High Measurement	State
TP1	VDD	-	VDD	-
TP2	-5.70 mV	OK	5106.00 mV	OK
TP3	Not available	-	Not available	-
TP4	Not available	-	Not available	-
TP5	-6.00 mV	OK	5109.00 mV	OK
TP6	-5.70 mV	OK	5109.00 mV	OK
TP7	Not used	-	Not used	-
TP8	Not used	-	Not used	-
TP9	Not used	-	Not used	-
TP10	Not used	-	Not used	-
TP11	GND	-	GND	-
TP12	-3.00 mV	OK	5109.00 mV	OK

Figure 143. Socket Test Report

If the **Socket Test** fails for any reason, refer to the Troubleshooting section for assistance.

2.2.7 NVM Programmer Window

This section contains the description of all **NVM Programmer** window controls.

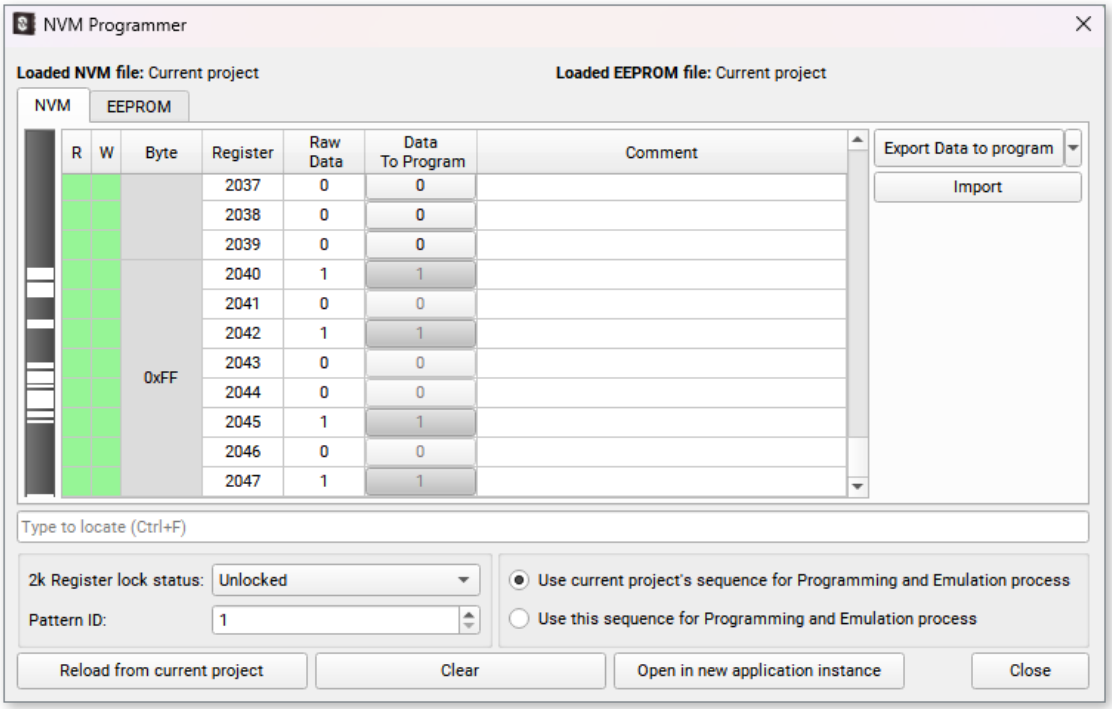


Figure 144. NVM Programmer Window with I2C Read/Write Operations

NVM Programmer Window Table:

- **R** and **W** – Shows whether a register is readable and writable.

  I2C operations allowed.
  I2C operations is not allowed.

- **Raw Data** – Displays the original bit value after data import.

- **Data To Program** – Shows the current NVW that can be changed by the user. It is used for chip procedures, such as emulation or programming.

- **Comment** – Add notes.

  Note 1: The comments are stored neither in chip memory nor in the project file. However, the export option that includes comments can be used.

  Note 2: Service bit values may be modified automatically during NVM import to ensure proper

chip performance.

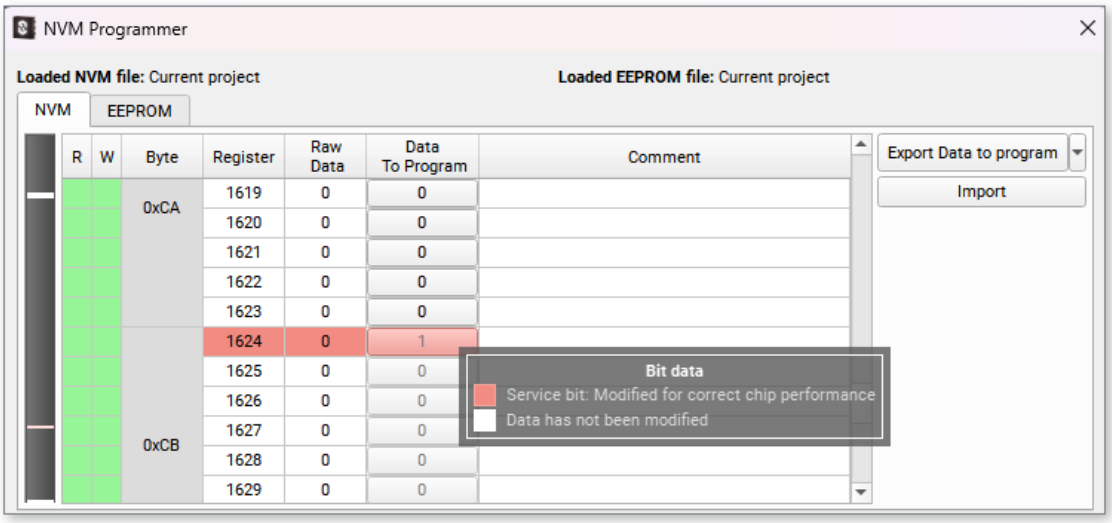


Figure 145. Forced Service Bit Modification Highlight

NVM Programmer Window Controls:

- **Lock status** – Lock **NVM Reading** or **Writing**. Use this control to determine whether **Read** and **Write** operations are possible.
- **Pattern ID** – Assign an **ID** to the current design.
- **Use current project's sequence for Programming and Emulation process** – Choose the bit sequence from **NVM Viewer** for the programming and emulation processes.
- **Use this sequence for Programming and Emulation process** – Choose the bit sequence from the **Project Data** table for the programming and emulation processes.
- **Reload from the current project** – Load the bit sequence from the **NVM Viewer** into the **Project Data** table.
- **Clear** – Set the entire **Project Data** table's bit range to 0.
- **Open in a new application instance** – Open the bit sequence from the **Project Data** table in a new software instance.
- **Export** – Save the **Raw Data** or **Data To Program** bit sequence to a text file.
- **Import** – Load the bit sequence from a text file.

The required byte or register can be found by typing its number in the search field. To filter bytes or registers in the search, use the following markers:

- b – Search for bytes only.
- r – Search for registers only.

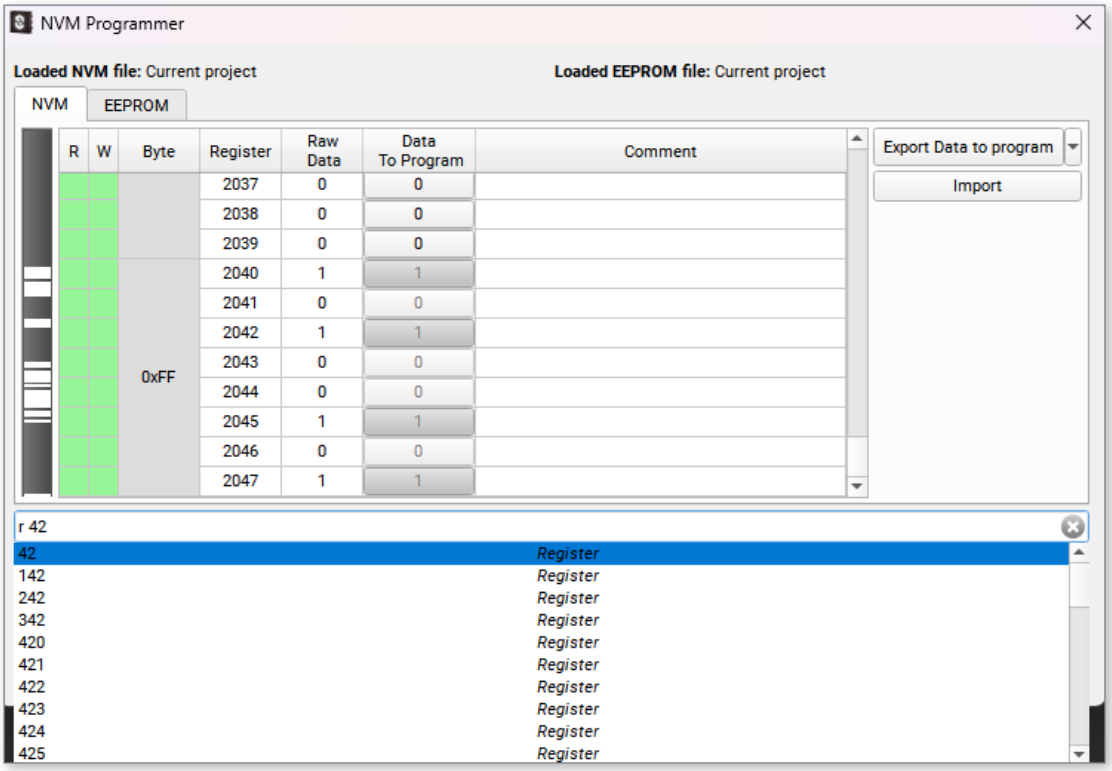


Figure 146. Searching for Registers

2.2.8 I2C Tools

The I2C Tools are instruments that help debug the project configuration by reading or writing register data on the chip.

A chip's I2C communication macrocell allows an I2C bus master to read or write information at any moment over a serial channel directly to the registers using the I2C protocol. The tool allows the macrocell data to be configured on the fly.

After the **Debug** utility is enabled and the platform is selected, the following I2C Tools appear on the toolbar: **I2C Virtual Inputs**, **I2C Virtual Outputs**, and **I2C Reconfigurator**.

Note: To read or write data with the I2C Tools, **Emulation** or **Test Mode** is required.

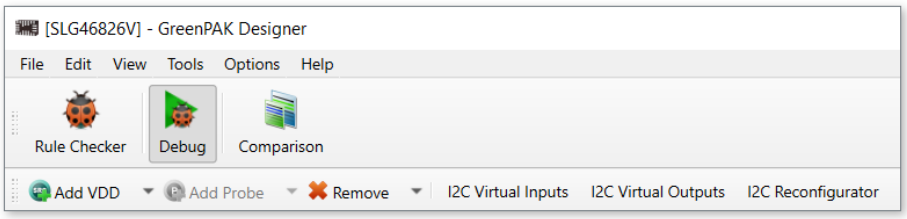


Figure 147. I2C Tools on the Toolbar

2.2.8.1 I2C Virtual Inputs

The I2C Tools contain the following instruments (the set of tools depends on the selected part number):

- **Counters/Delays** – Allows reading or writing the counter data of a specific macrocell configured in CNT/DLY mode.

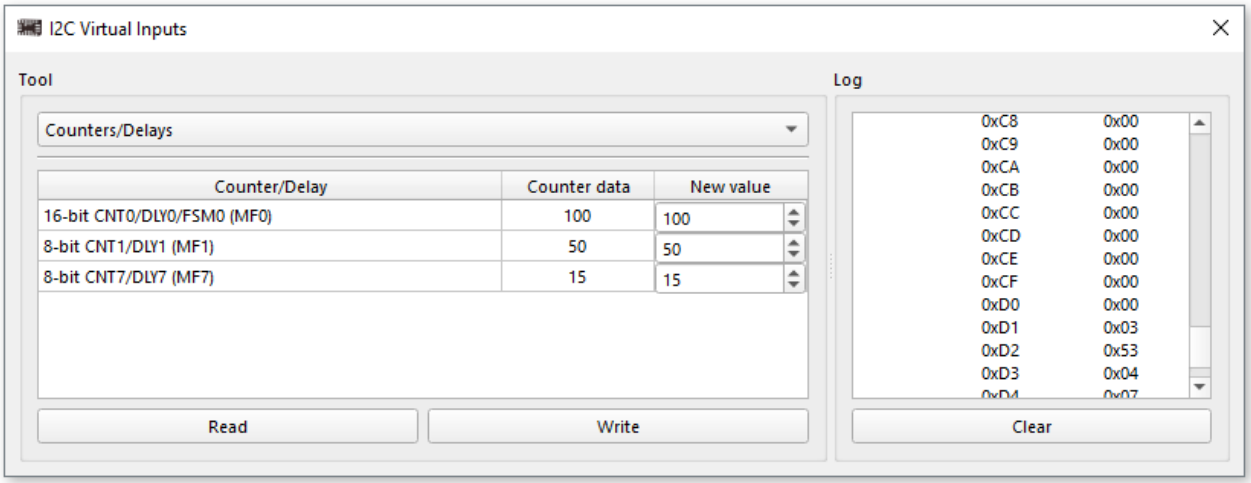


Figure 148. Managing Counters/Delays

- **I2C Virtual Inputs** – Allows reading or writing I2C virtual OUT values of the I2C macrocell.

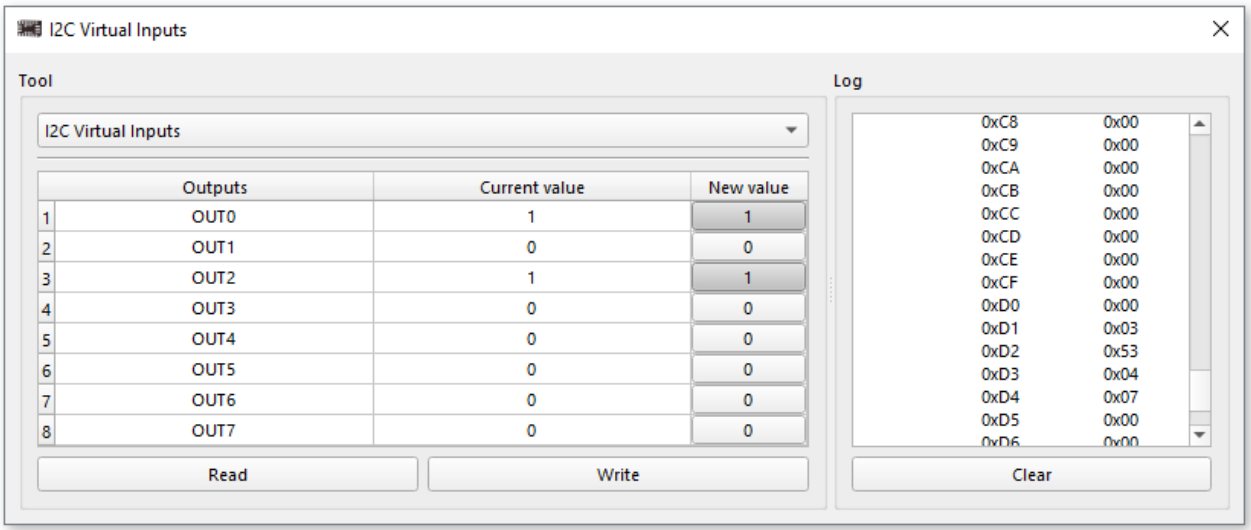


Figure 149. I2C Virtual Inputs

- **Registers** – Allows reading or writing all chip register data. It is possible to read or write:
  - The specific register in each row.
  - All registers using the bottom buttons.

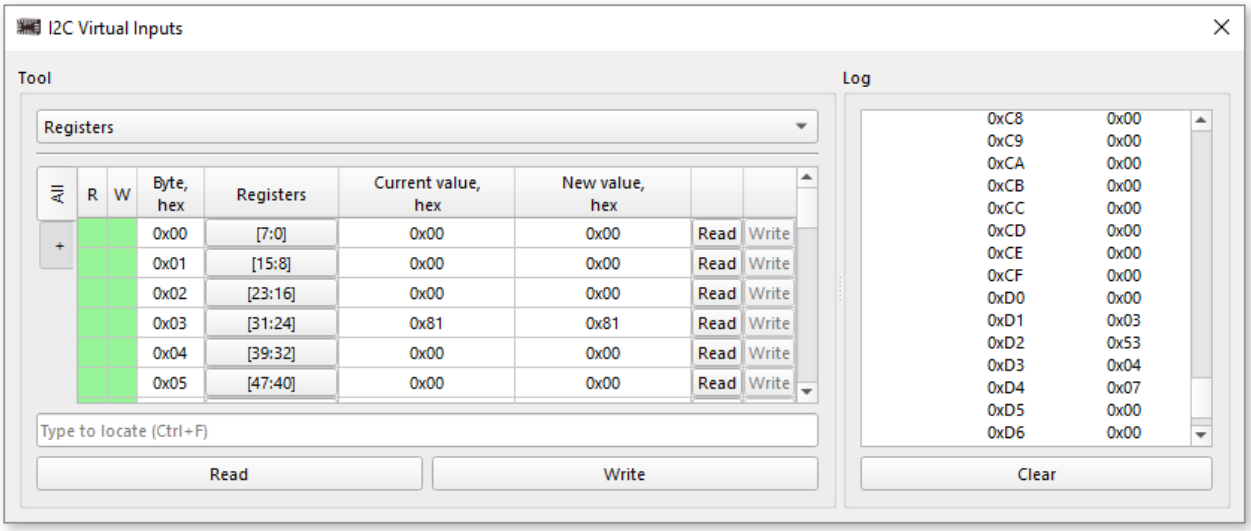


Figure 150. Read/Write the Registers Data

Register values can be edited in the following ways:

- Enter the value in the **New value** column.
- Change the register bits by clicking a cell in the **Registers** column.

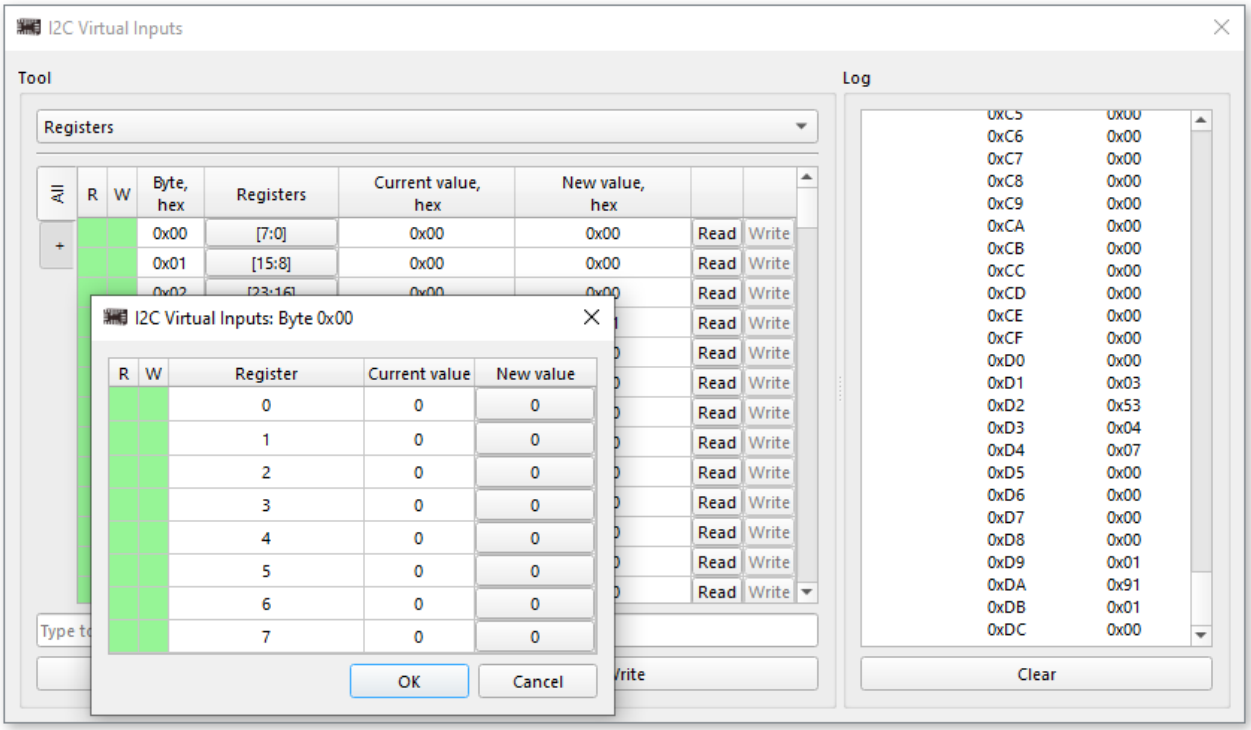


Figure 151. Registers Modification Options

A required byte or register(bit) can be found by using the pattern below:

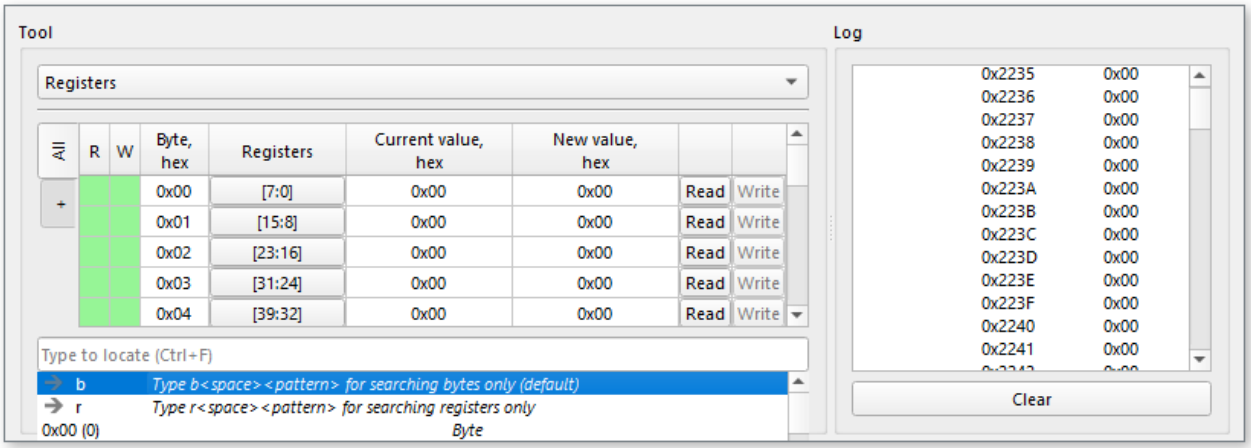


Figure 152. Filter Options

The **Registers** tool allows creation of tabs with specific register ranges. Add a new tab by clicking the button under the **All** tab and the ✚, then add entries with register ranges.

Note: Only the registers under the active tab are read when the bottom **Read** button is clicked.

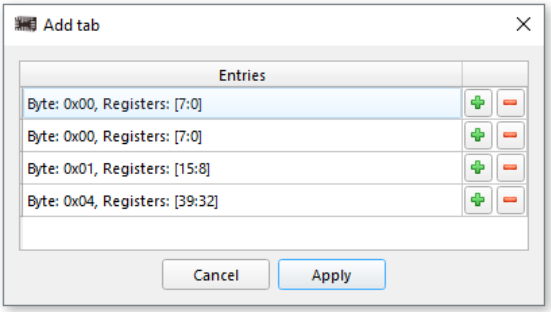


Figure 153. Add Tab

The context menu also provides additional options: edit, rename, or remove tabs; and switch between hexadecimal and decimal value formats.

Remove the tick from the **Update current value after write** checkbox if reading back

overwritten values is not desired.

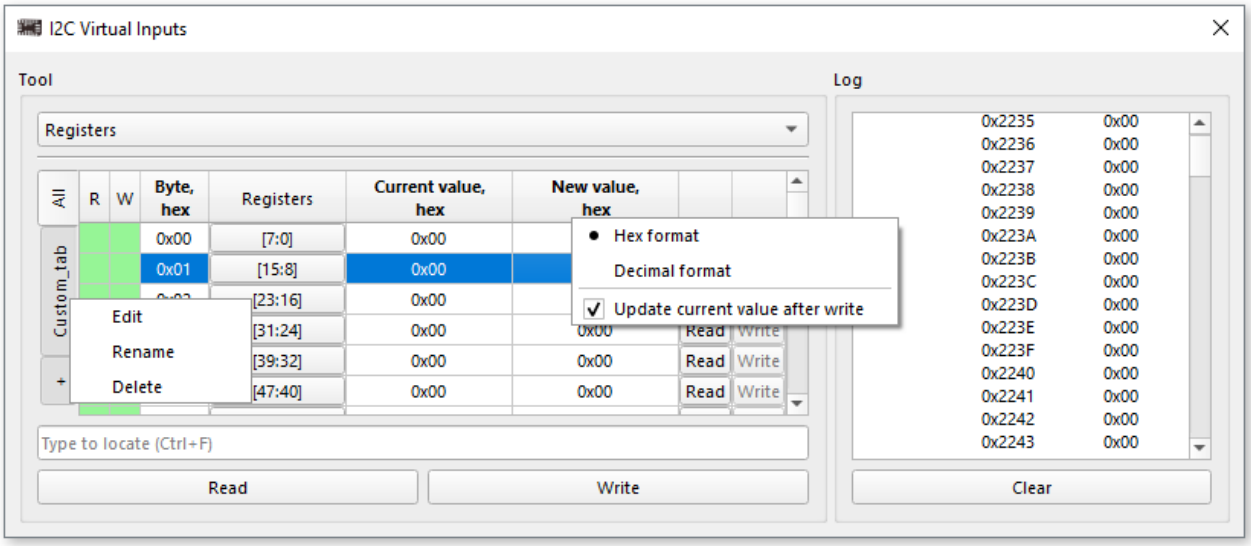


Figure 154. Context Menus

The **R** and **W** columns indicate whether a register is readable or writable:

- I2C operations are supported.
- I2C operations are not supported.
- I2C operations are unsupported for some bits of the register.

- **Data Buffers** – Allows reading data from the Data Buffer macrocells and the **ADC data register**. For automatic data actualization, check the **Auto read every 1s** option and click **Start**.

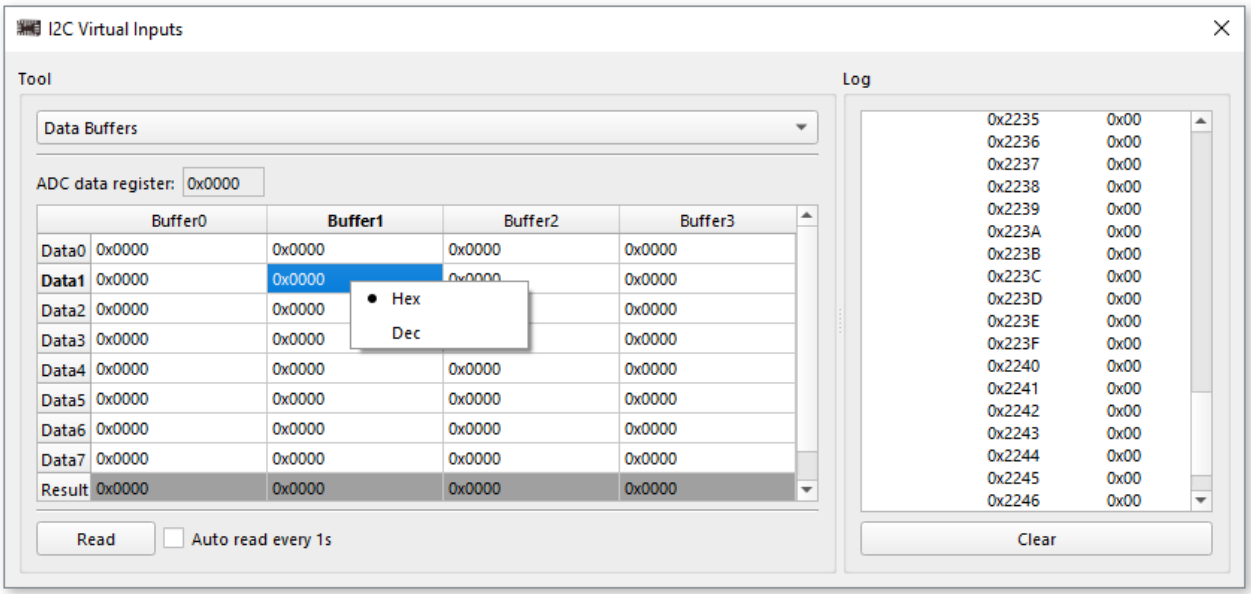


Figure 155. Managing Data Buffers

- **Memory Table** – Read data from the Memory Table macrocell in the **Column X** section. To change the number of displayed columns, use the **Columns** selector. In addition to decimal and hexadecimal formats, which apply to both column types, the **Address** column data view can also be changed using the following options:
  - **Absolute** – Show the actual address of the register in the macrocell.
  - **Relative** – Show the register address starting from zero.

  The read data can be saved in .csv format by clicking the **Export to file** button.

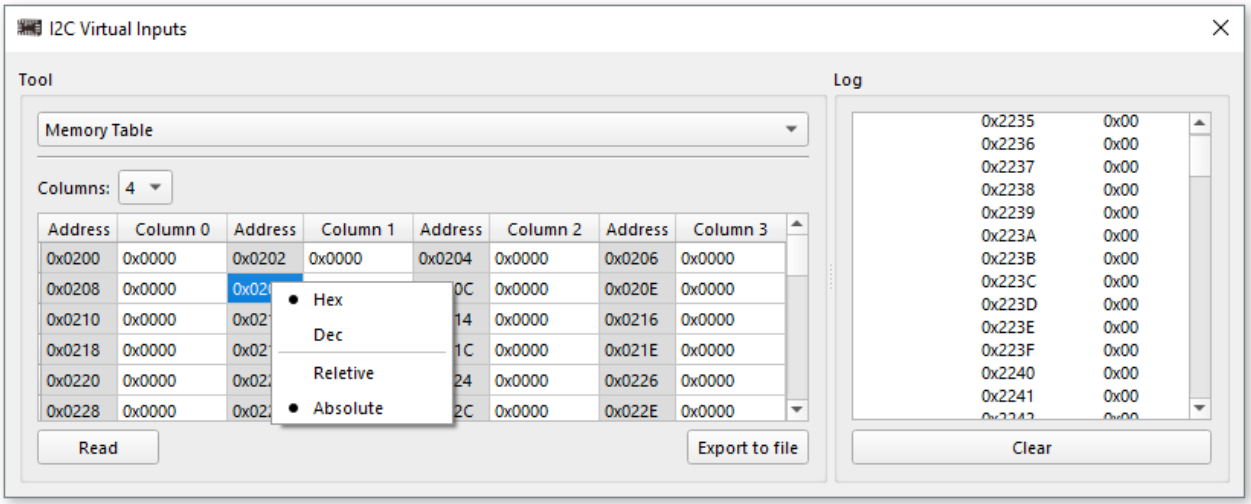


Figure 156. Managing Memory Table

- **Speed control** – Displays and allows reading or writing the constant of the Scalers and Speed/Torque Control macrocells.

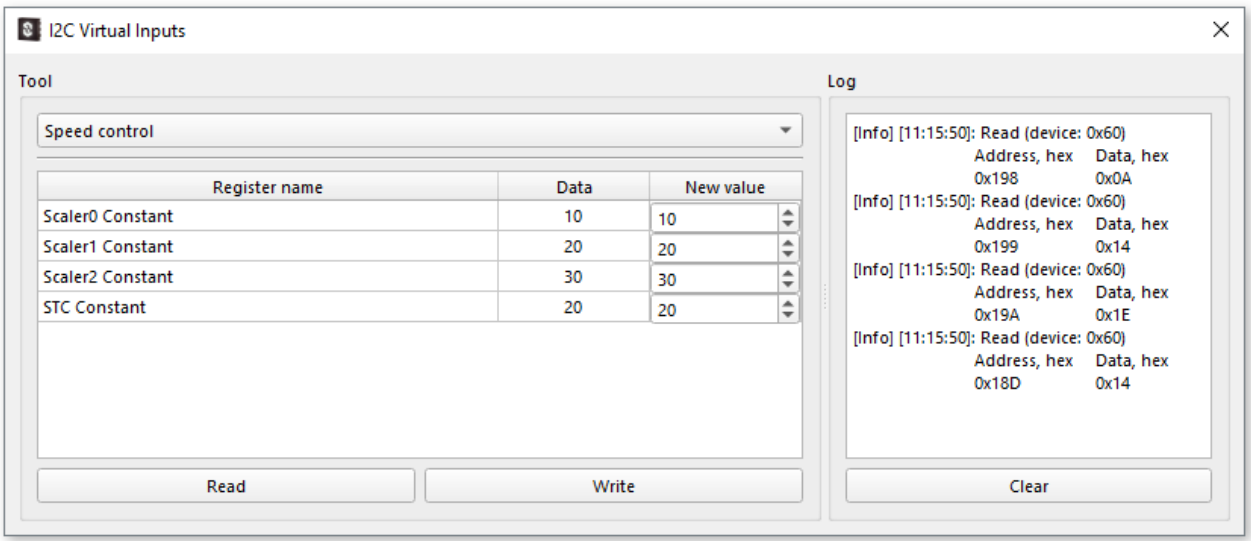


Figure 157. Speed Control Tool

- **Math Core Table** – Is designed to manage the Math Core component data from its

Properties panel. The K and B constants can be set, read, or written, and the results of mathematical operations can also be set. Change the data format in the context menu.

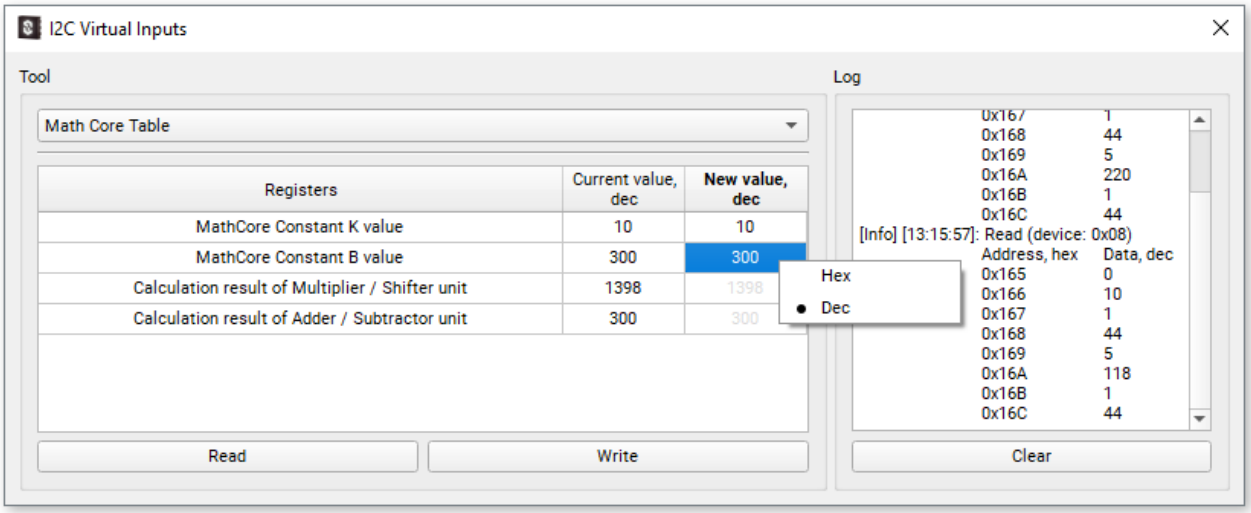


Figure 158. Math Core Table

- **NVM eraser tool**, **EEPROM eraser tool** – Allow clearing selected data programmed onto the chip (only multiple-time programmable (MTP) ICs are supported). After the **Read NVM** or **Read EEPROM** button is clicked, the programmed data is displayed in the respective column. Use **Erase selected** to clear the data. While NVM or EEPROM is being cleared, the **Data** column is not updated until the memory is read again. Certain pages cannot be erased because they are essential for proper chip operation.

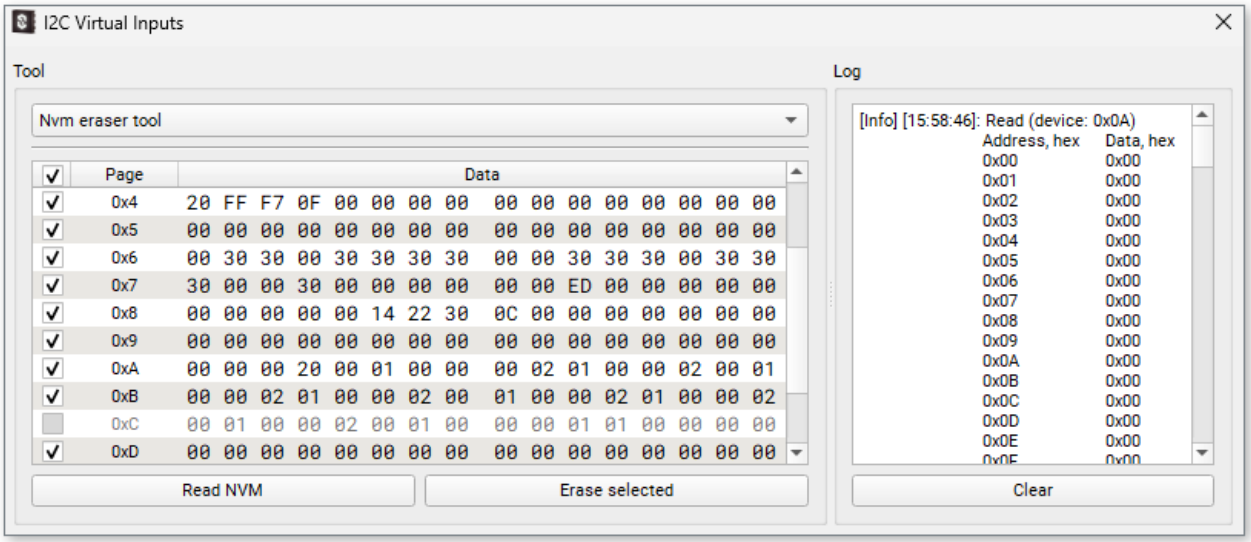


Figure 159. NVM Eraser Tool

- **Log** – Displays the results of recent read and write operations. The **Log level** has three

severity settings in decreasing order of detail: **Debug**, **Error**, and **Info**.

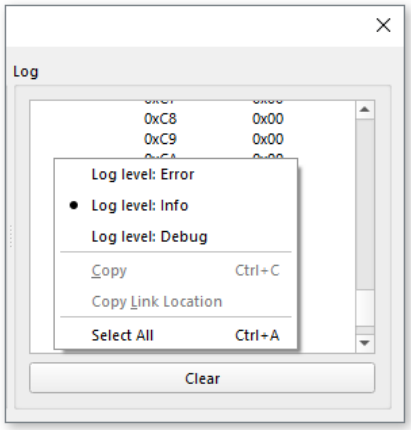
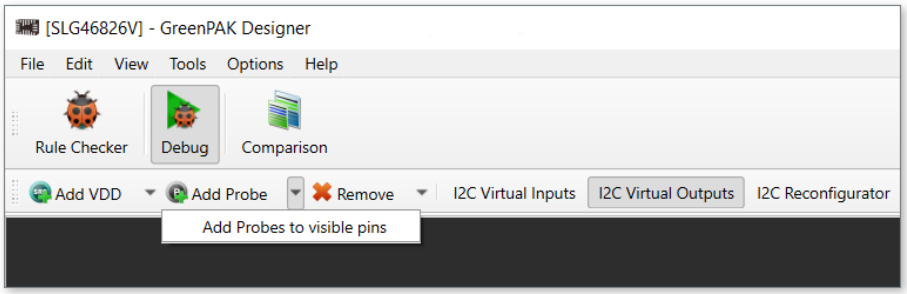


Figure 160. Log Context Menu

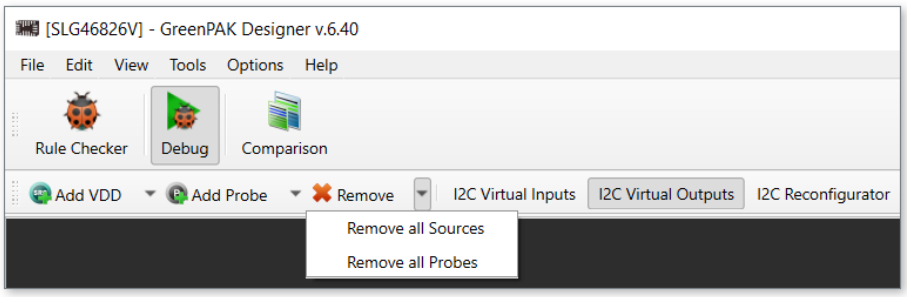
2.2.8.2 I2C Virtual Outputs

I2C Virtual Outputs allows reading the state of certain macrocells. The tool includes hardware probes, ASM state, and the current counted data of specific counters.

- **Probes (Matrix inputs)** – Reflect the current logic level on the macrocell output pins. A probe can be added by clicking **Add Probe** and then the required pin. Internal pins that support probe insertion are highlighted in green. Probes can also be added to all visible pins at once by clicking **Add Probes to visible pins**.



Probes can be removed one by one or all at once.



- **I2C Virtual Outputs** tool – Displays the current value of selected components, for example, CNT/DLY, ASM, Scaler, and DCM, read from the chip registers (see the Datasheet for the

selected part number). To update the data or hide unused blocks, click the respective controls on the tool panel.

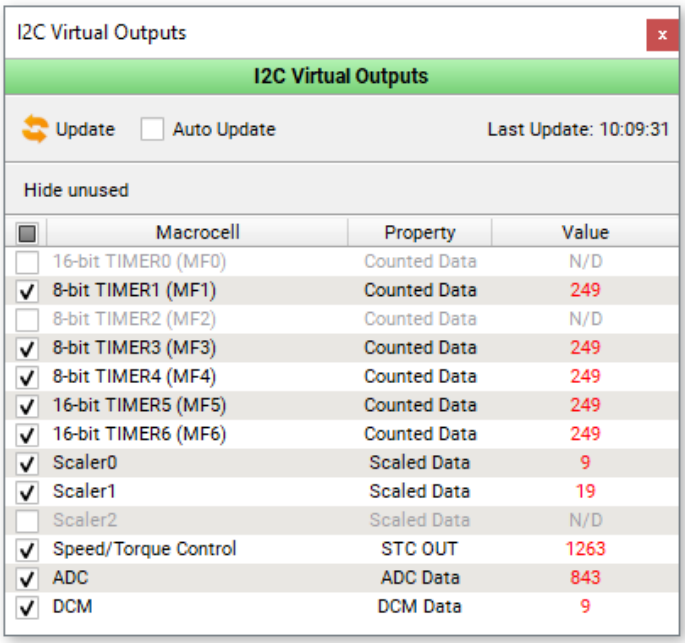


Figure 161. I2C Virtual Outputs Tool

2.2.8.3 I2C Reconfigurator

I2C Reconfigurator allows chip data to be changed dynamically by sending **NVM** snapshots to the chip. Snapshots are diff-based lists of changes. Each includes macrocell configurations and connections.

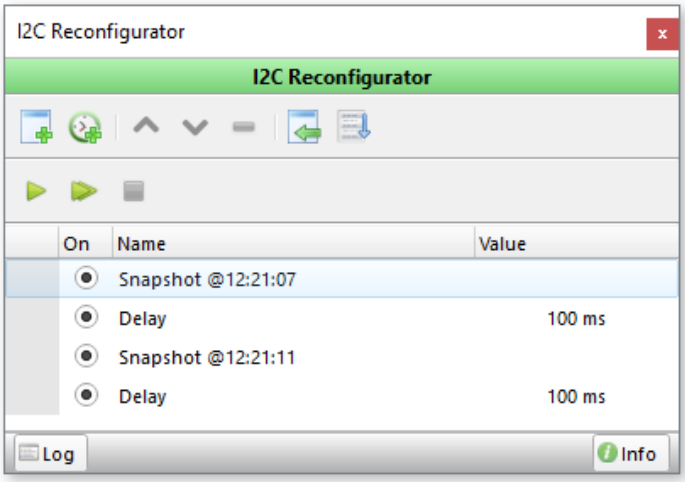


Figure 162. Snapshot Configuration

The **Reconfiguration scenarios** can be managed using the following controls:

- Add a snapshot of the work area.
- Add a delay between snapshots.

Move the selected list item one level up.
Move the selected list item one level down.
Remove the selected list item.

To view the list of applied changes, refer to the **Reconfiguration scenario** controls:

Open a dialog with the list of applied changes; delays are included. The entire list can also be exported to a file.
Import presets from a different project.

Here is the list of snapshot sending controls:

Send the next snapshot to the chip and pause list execution.
Send snapshots along with delays one by one continuously.
Stop sending snapshots and reset the list pointer.

The selected snapshot can also be manipulated using the following buttons:

- **Load** – Send the configurations of the selected snapshot to the chip.
- **Overwrite** – Make the desired changes and replace the existing snapshot data.

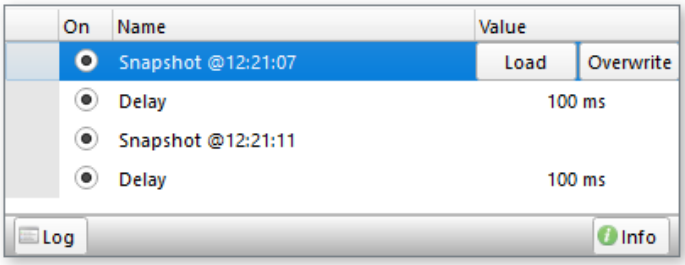


Figure 163. Load/Overwrite Snapshot Configurations

- **Reset** – The button becomes available after a snapshot has been loaded by clicking the **Send one** button and any configuration modifications have been made afterward. Click the button to return the snapshot to its originally saved state.

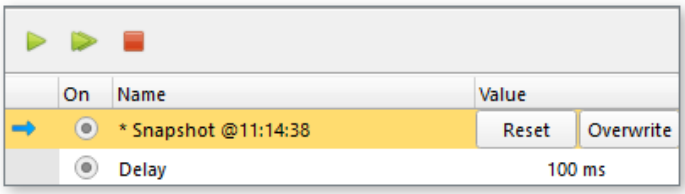


Figure 164. Reset Snapshot Configurations

The **Log** utility displays the changes applied to the chip. As delays do not imply any changes to the chip, they do not appear in the **Log**.

2.2.9 I2C I/O Tool

Just like I2C Tools described above, I2C I/O Tool allows project configuration to be debugged by reading or writing register data on the chip (the tool type depends on the selected part number).

After **Debug** is enabled and the required platform is selected, the **I2C I/O Tool** button appears on the toolbar.

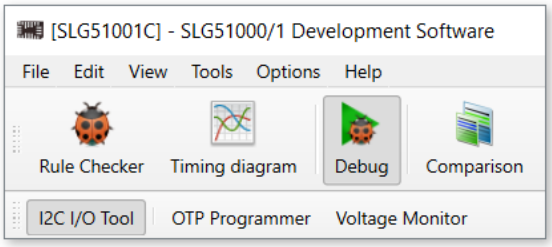


Figure 165. I2C I/O Tool on the Toolbar

The following instruments are included: **Registers**, **Counters/Delays**, **Events / Current states** and **Raw I/O**.

Note: To read or write data with the I2C I/O Tool, **Emulation** or **Test Mode** is required.

- **Registers** – The tool is mostly the same as **Registers** in I2C Tools. The slight differences are described below.

  Click the **Address** column cell to open the **Byte view** window:

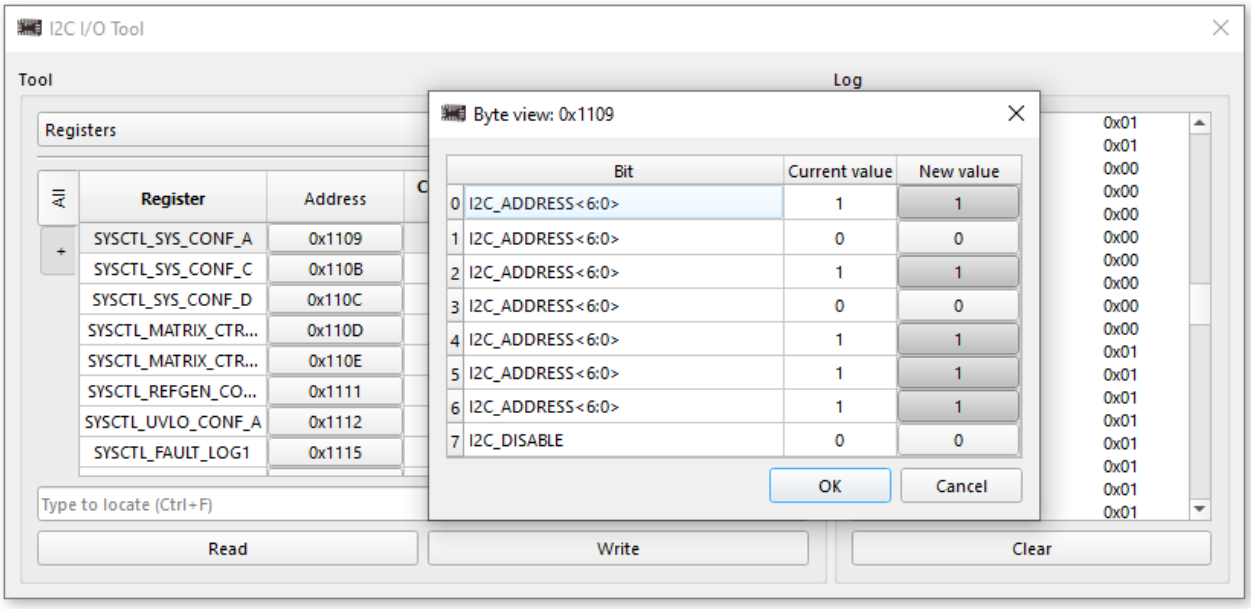


Figure 166. Change Specific Bit

Search by byte or register name is available:

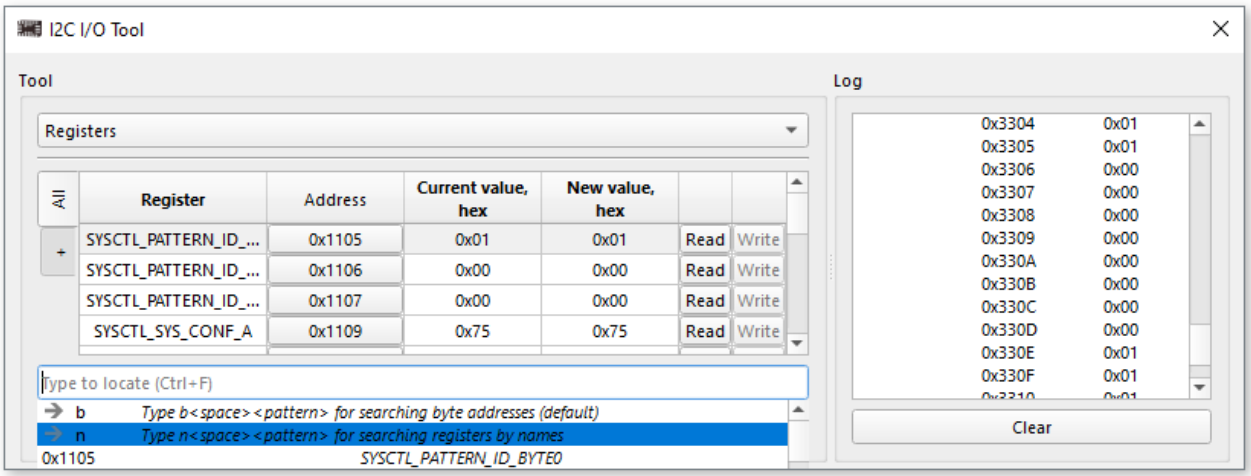


Figure 167. Search by Byte or Register

- **Counters/Delays** – The same description as in the I2C Tools section is applicable.
- **Events/Current states** – Shows the read-only list of events for specific components (for example, LDO, CNT/DLY).

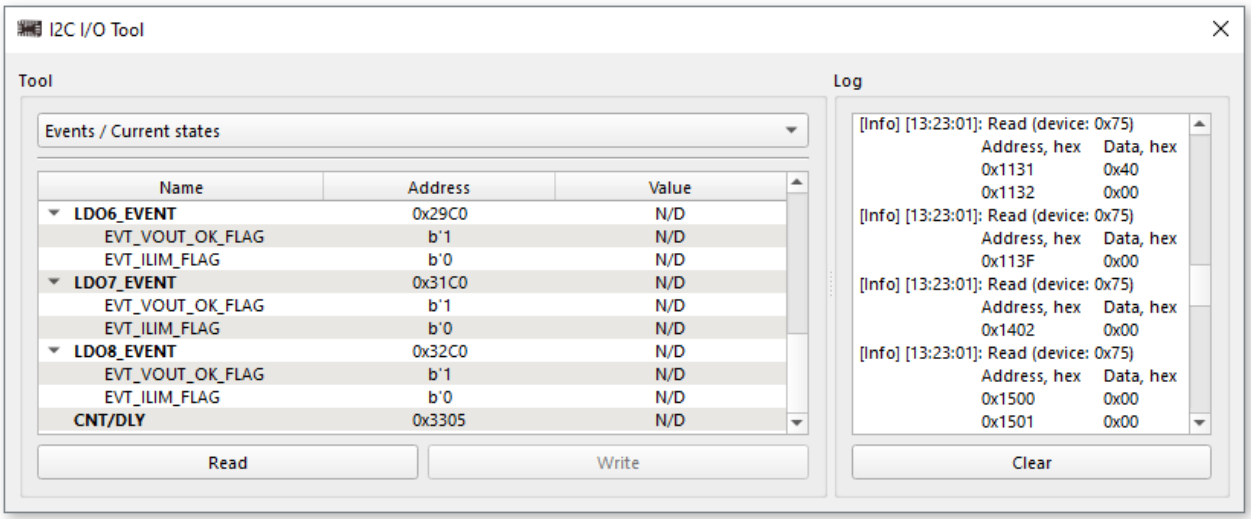


Figure 168. I2C I/O Tool Window with Events

- **Raw I/O** – Provides extended I2C access to chip registers. It allows access to registers that

may be inaccessible from the **Registers** instrument.

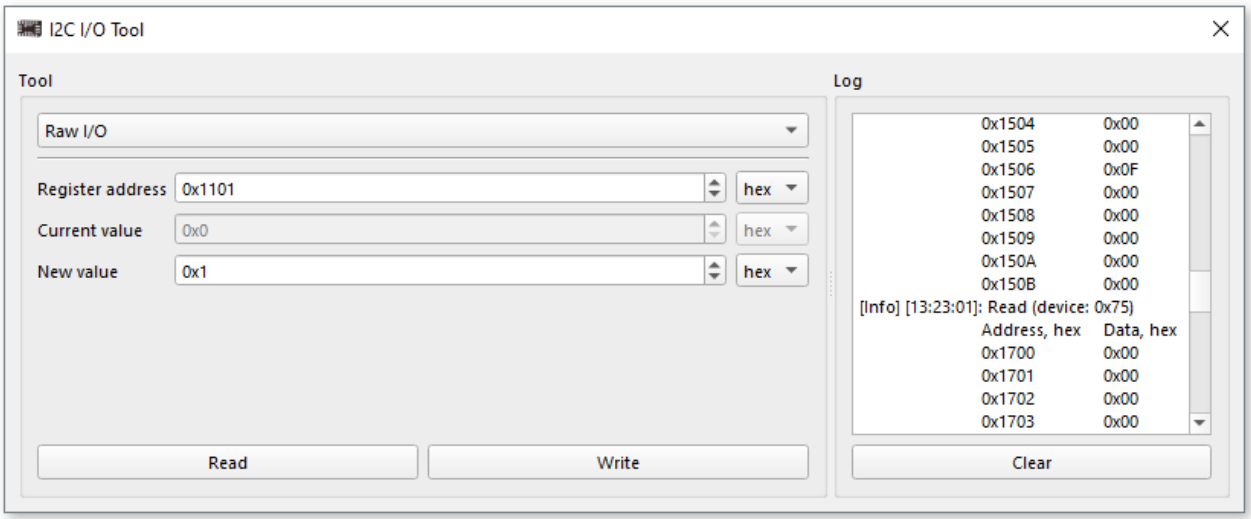


Figure 169. I2C I/O Tool Window with Raw I/O

- **Log** – Refer to the I2C Tools section.

2.2.10 Logic Analyzer

Logic Analyzer tool allows capture of multiple signals from a digital circuit. It includes triggering capabilities and a protocol decoder that helps reveal the timing relationships between multiple signals and decode them.

The characteristics of **Logic Analyzer** are the following:

- Frequency range (500 Hz - 200 MHz).
- Buffer size (16384 samples).
- I2C, Parallel, SPI, and UART protocol support.

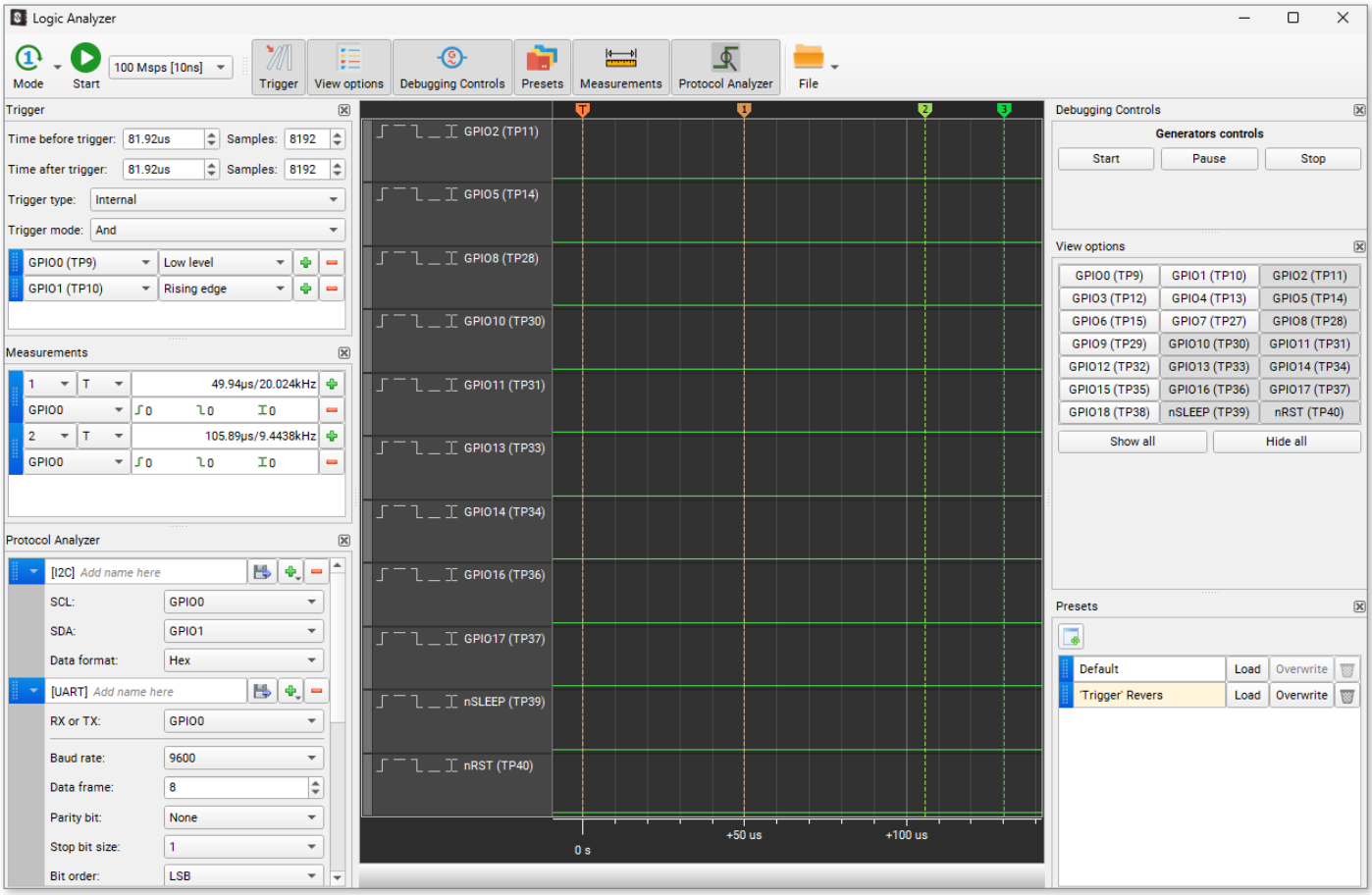


Figure 170. Logic Analyzer Window

2.2.10.1 Main Operational Controls

- **Mode** – Three operating modes are available:
  - **Auto mode** – Trigger events are ignored. The signal on the pins is shown as a continuous waveform.
  - **Single shot** – Refreshes the waveform pattern as soon as a trigger event is detected.
  - **Normal mode** – Refreshes the waveform pattern each time trigger conditions are met.
- **Sampling rate** – Drop-down selector of the signal sampling frequency.
- **Start** – Launch **Logic Analyzer**. The **Start** button becomes active after **Emulation** or **Test Mode** is started.

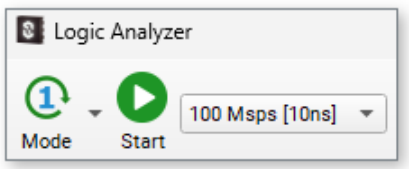


Figure 171. Main Controls

2.2.10.2 Triggers

Set time parameters to choose the trigger time position or a specific sample.

Chose between the following trigger types:

- **Hardware button** – The trigger is activated by pressing a physical button on the board.
- **Internal** – The trigger is activated when the trigger condition, which is set in the trigger list, is met. Additional settings are available for the **Internal** trigger type:
  - **Trigger mode** – Depending on the selected option (**And** or **Or**), any or all of the trigger conditions added to the trigger list will be met.
  - **Trigger list** – Add or remove triggers, assign the trigger to the channel, and select the trigger conditions among the following: **Rising edge**, **Falling edge**, **Both edges**, **High state**, and **Low state**.

Note: Set proper trigger conditions for successful sampling (such as **Rising edge**, **Falling edge** together with **And** trigger mode may result in improper trigger work).

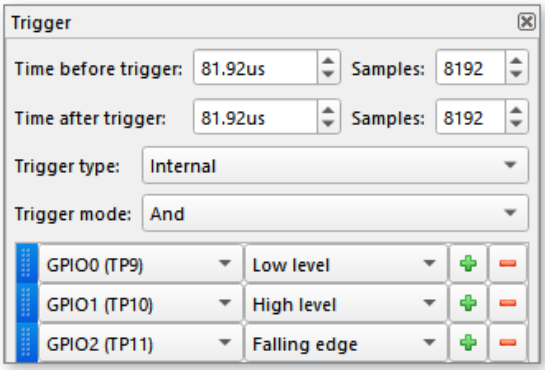


Figure 172. Trigger Parameters

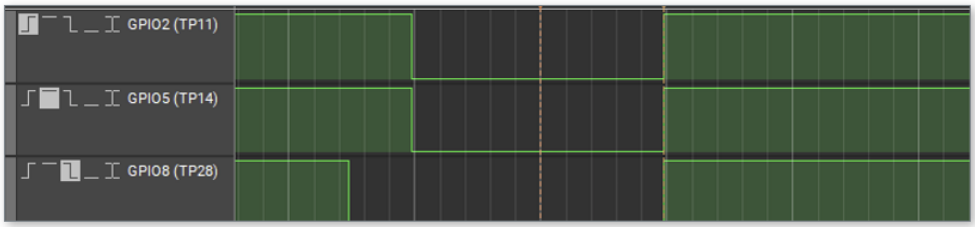


Figure 173. Trigger Configuration

2.2.10.3 Protocol Analyzer

The **Protocol Analyzer** allows data to be decoded according to a protocol.

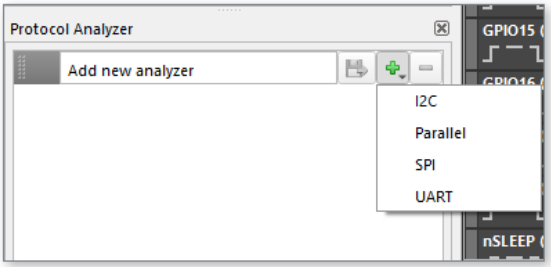


Figure 174. Protocol Analyzer

To analyze captured data, click the ✚ button and select one of the protocols.

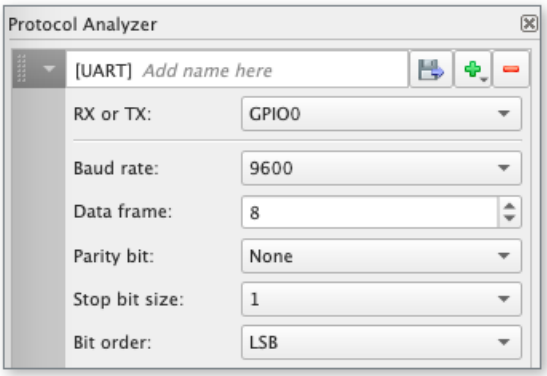


Figure 175. Protocol Settings

Then, choose a channel for analysis and modify protocol settings if necessary. The decoded data is displayed above the corresponding plot.

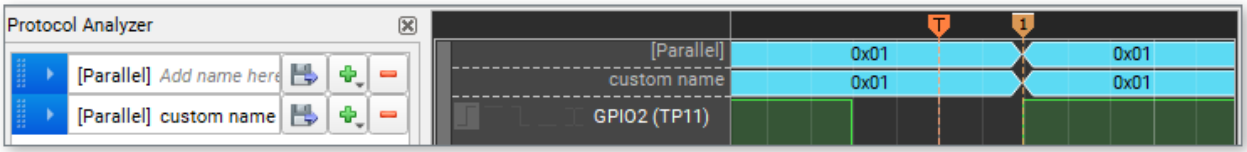


Figure 176. Decoded Data

Data from a **Protocol Analyzer** can be exported easily by clicking the **Save** button next to the **Protocol Analyzer** name field. If the parameters are selected correctly, a CSV file is downloaded automatically.

2.2.10.4 Import and Export Actions

Waveform data can be saved or imported in CSV format. These options are grouped under the **File** button on the top toolbar.

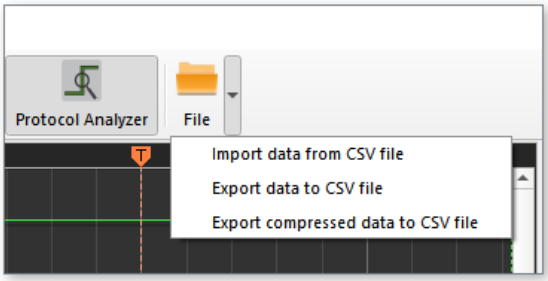


Figure 177. Export and Import Operations

2.2.10.5 Plot Widget

The plot widget displays waveforms in the **Logic Analyzer** window. The way a plot is shown can be changed from the plot context menu. Right-click a plot area to add a marker, show or hide the time scale, change the plot height, and select the cursor width.

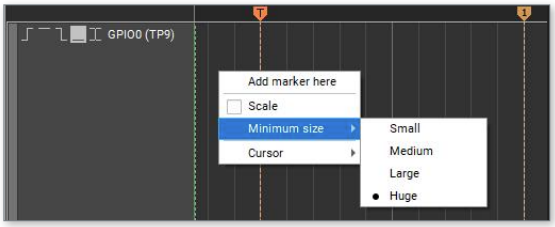
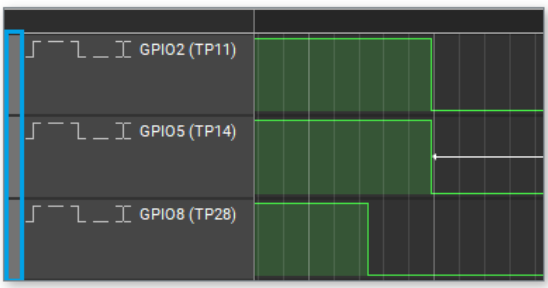


Figure 178. Logic Analyzer Plot Menu

The following actions can be performed in the plot area:

- Move left or right by click and drag.
- Zoom in or out by **Ctrl** + **mouse wheel**.
- Zoom in or out by **MMB** click + drag up or down.
- Reorder waveforms by drag and drop.



2.2.10.6 Markers

- Add a new marker with a **Ctrl** + **LMB** click on the markers panel.
- Set a new marker from the plot's context menu.
- Remove the marker with a **Ctrl** + **RMB** click on the marker head.
- Move the marker by a **LMB** click + drag.
- Move the marker from the context menu by a **LMB** click on the marker's head.

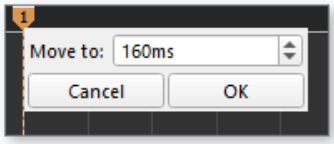


Figure 179. Marker Move To

- Select the marker by clicking the marker head: the selected marker has a white line on top.
- Move the selected marker to the closest visible front by **Ctrl** + **Left**/**Right**.
- Move the selected marker by one sample with **Ctrl** + **Shift** + **Left**/**Right**.

2.2.10.7 Marker Measurements

- Measurements – Period and frequency. Frequency value is rounded to four decimals.
- Additional measurements – The count of **Rising Edges**, **Falling Edges**, and **All Edges** in the period between the markers.

To calculate all measurements between two markers, select the markers and a channel in combo boxes.

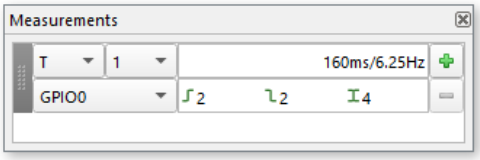


Figure 180. Markers Measurements

2.2.10.8 Cursors

A cursor is a measurement tool for calculating waveform data between the edges of one or more plots. To display the cursor, hover over a measured section of a waveform.

A **Half Period** cursor measures the distance between the two nearest edges at a hovered section

of a waveform.

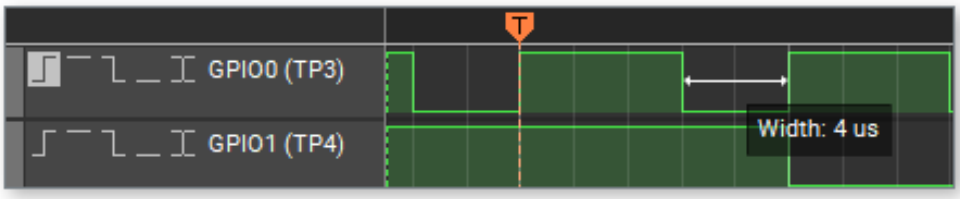


Figure 181. Half Period Cursor and Sample Measurements

A **Period** cursor measures the width of the hovered half period, the duration of a full period, calculates the frequency, and determines which fraction of the full period the hovered part of the period represents, which is the **Duty cycle**.

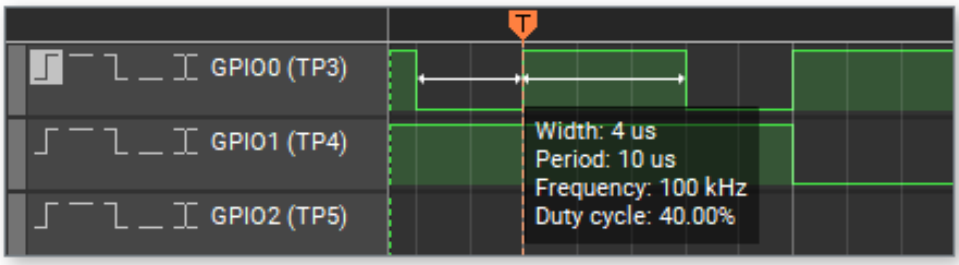
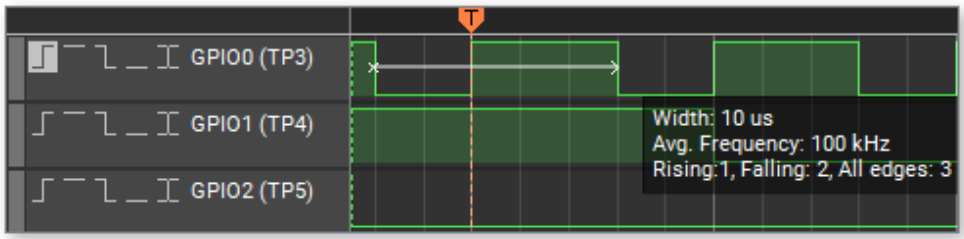


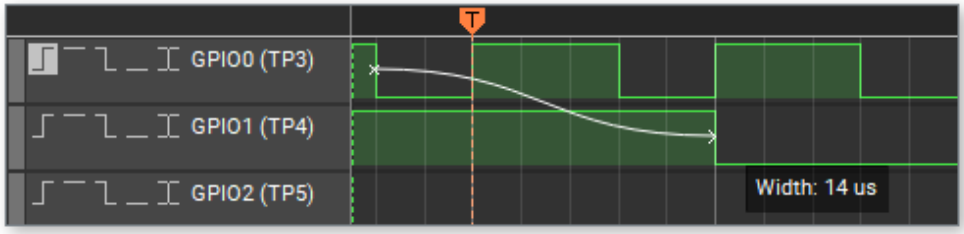
Figure 182. Period Cursor and Sample Measurements

To change the cursor width, open the context menu and select either the **Period** or **Half Period** option.

To measure data across a distance that exceeds one period, click the edge from which the measurement should start and hover over the edge to which the measurement should extend. This provides the total duration of the measured section, the calculated average frequency, and the number of rising and falling edges, as well as their sum.



The width between edges on distinct waveforms can also be measured.



2.2.10.9 View Options

Click the channel in the **View options** panel to show or hide the corresponding plot on the plot widget.

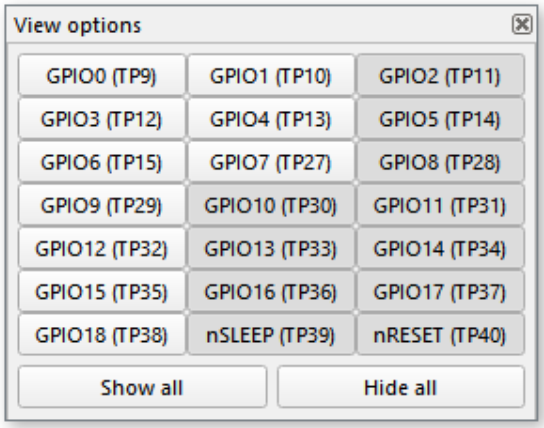


Figure 183. View Options Panel

2.2.10.10 Debugging Controls

The **Debugging Controls** panel is responsible for **Emulation** and **Test mode** chip procedures, along with **Generator controls** functionality.

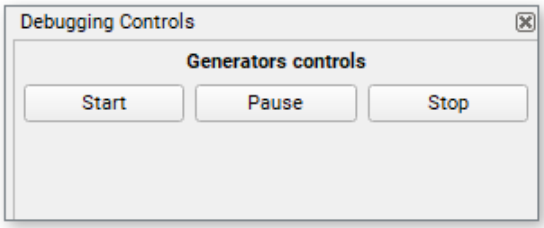


Figure 184. Debugging Controls

2.2.10.11 Presets

Logic Analyzer configurations can be stored in presets and a previous configuration can be restored when needed.

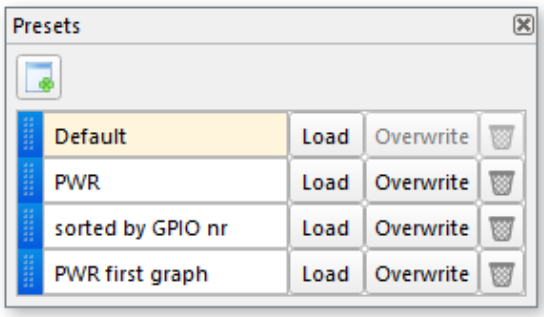


Figure 185. Presets

- – Create a new preset with the current **Logic Analyzer** configuration.

- **Load** – Load a selected preset configuration.
- **Overwrite** – Overwrite the preset with the current **Logic Analyzer** configuration.
- – Remove the selected preset.
- **Default** – Load the Default preset of the **Logic Analyzer** window.
- **Autosaved @ [time]** – The modified preset is saved each time the **Logic Analyzer** window, the **Debug tool**, or ForgeFPGA Workshop is closed.

2.2.11 Flash Programmer

The **Flash programmer** tool allows flexible programming of flash memory across its full addressing range, enabling programming of both the bitstream and custom user data.

To start working with the tool, load data by clicking the **Load Bitstream** or **Load Custom Data** buttons in the upper section:

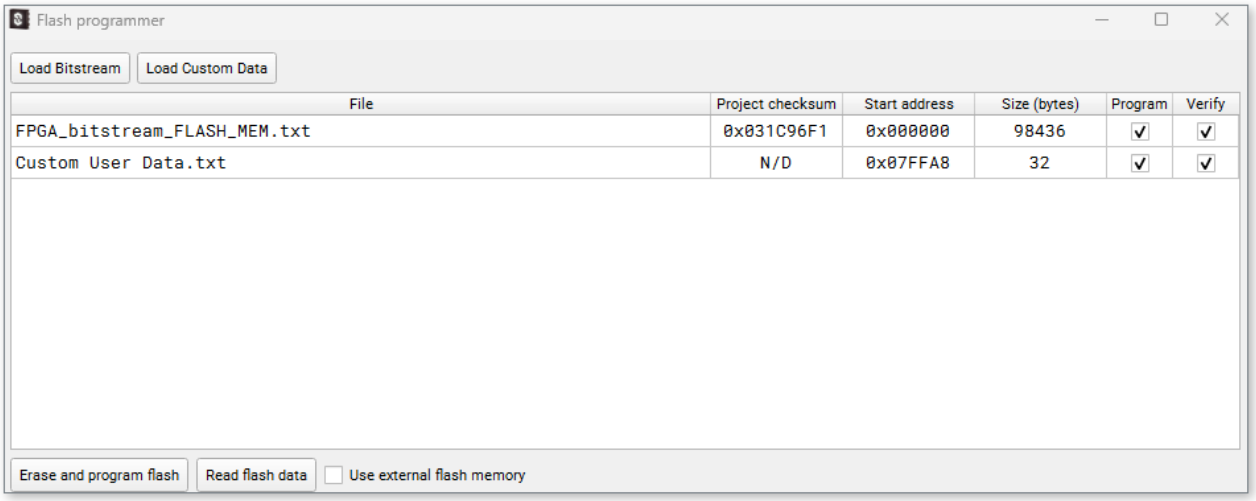


Figure 186. Flash Programmer Tool

- **Load Bitstream** – After a file is added, the tool validates its size against the expected bitstream size and calculates the checksum.
- **Load Custom Data** – Loads user-provided data, which should be in the same format as the data in the bitstream file.

To program the flash, select one or multiple files using checkboxes in the **Program** column.

The **Flash programmer** tool supports programming of external flash memory. To enable this feature, tick the corresponding checkbox in the footer. Click the info icon to view the instructions for flash parameter configuration. Specify the external flash size and set the SPI frequency to match

the device specifications.

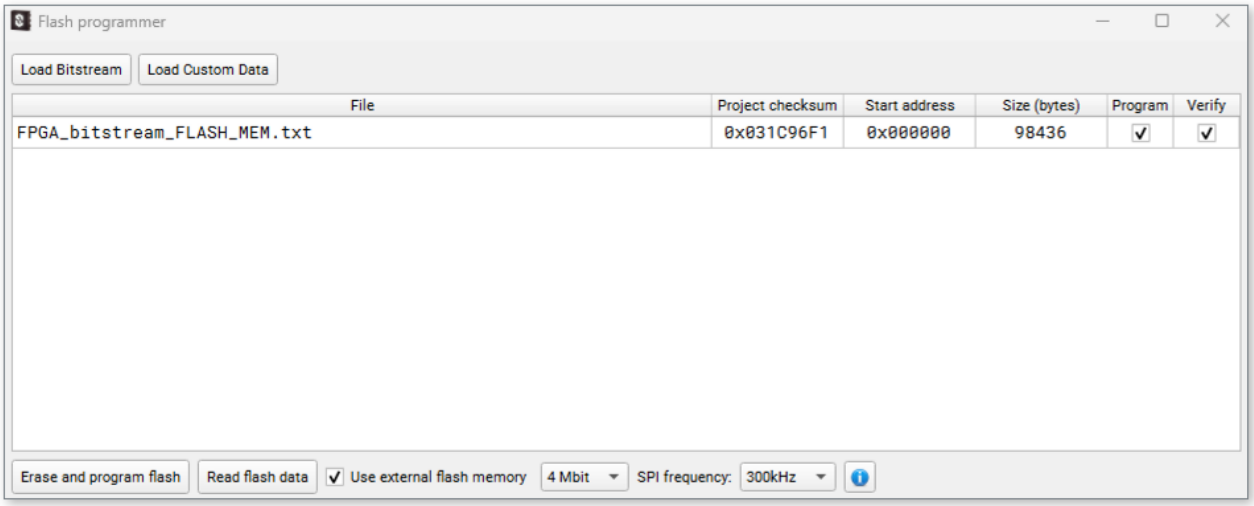


Figure 187. External Flash Programming Option

Observe the hints triggered in case of any inconsistency or other issues during interaction with the tool. Only selected files pass validation.

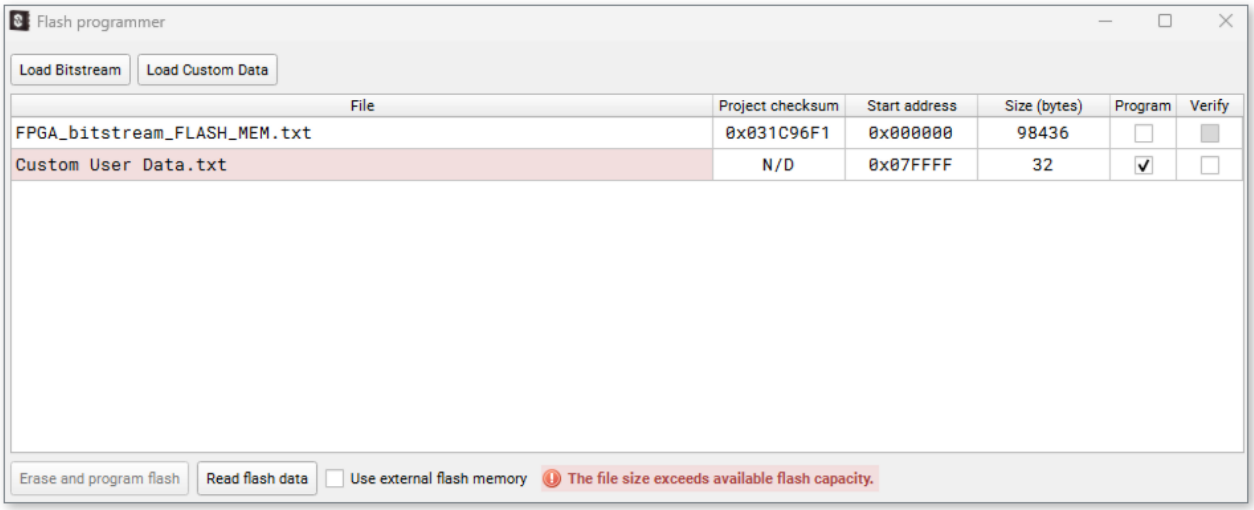


Figure 188. Flash Programmer Tool Info Hint Example

Optionally, enable the **Verify** column. When activated, **Flash programmer** reads the programmed range defined by the **Start address** and **Size** columns and compares it with the loaded file data.

Right-click the file name to access options for removing or replacing the file using the context menu. Hover over the file to see its location on the computer.

Note: The entire flash memory is erased before programming.

2.2.12 UART Terminal

UART Terminal is a console tool used for serial communication between a board and an external device. While developing a project, the tool helps perform **read** and **write** operations over the UART protocol.

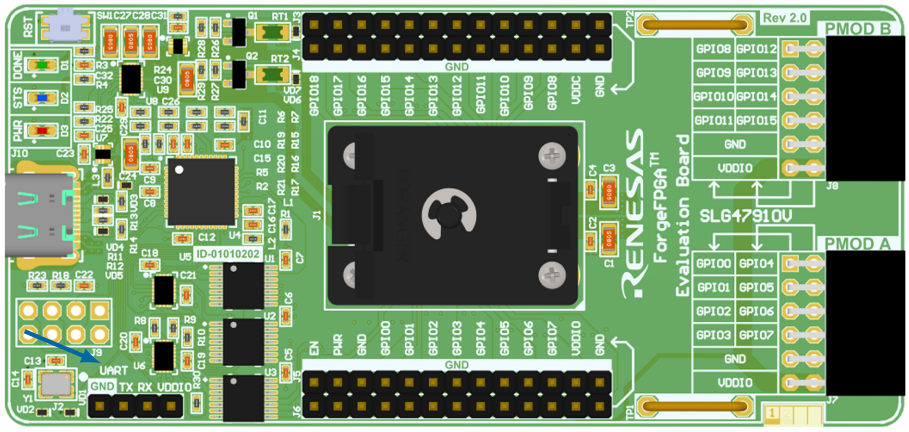


Figure 189. The UART Port on the ForgeFPGA Evaluation Board

To start working with the terminal, make sure the selected platform supports this feature. Open the **UART Terminal** tool from the top toolbar.

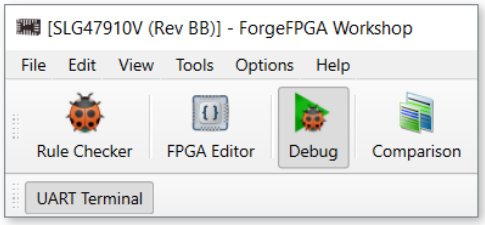


Figure 190. UART Terminal on the Toolbar

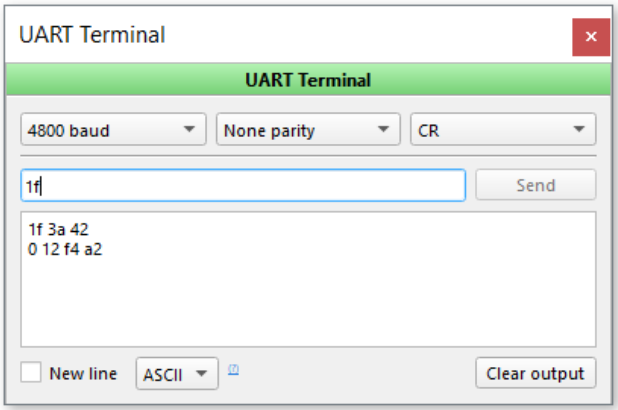


Figure 191. UART Terminal Window

The **UART Terminal** key features are as follows:

- **Baud rate** – Select the baud rate from the list for **read** and **write** operations.
- **Parity control** – Choose whether the parity control bit is used and how.
- **Line ending** – Set which character is used as a new line character: **No line ending**, **New Line (NL)**, **Carriage Return (CR)** or both **New Line** and **Carriage Return**. This option is available in ASCII mode only.
- **New line** – Add a new line after each byte sent if set; otherwise, send the whole message at once.
- **Data Format** – Select the data format: ASCII or hex. For hex format, the input case is independent, and numbers are separated by a space.

The input field supports copy and paste operations. The output field allows copying of the received data.

2.2.13 DAC Tool

The **DAC Tool** allows configuration of the voltage level on special analog output socket pins.

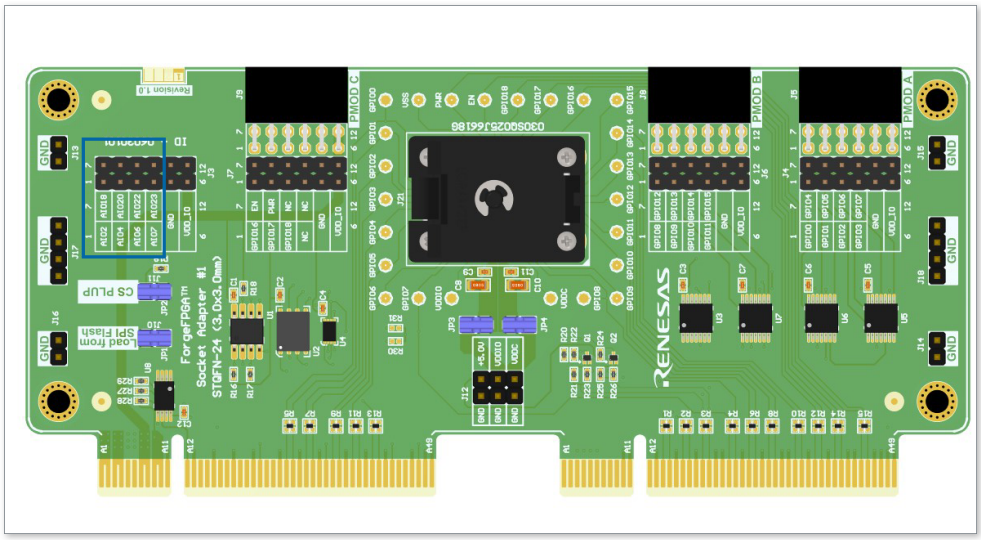


Figure 192. Analog I/O Pins on the ForgeFPGA Socket Adapter

Start by selecting the appropriate development platform and opening the **DAC Tool** from the toolbar.

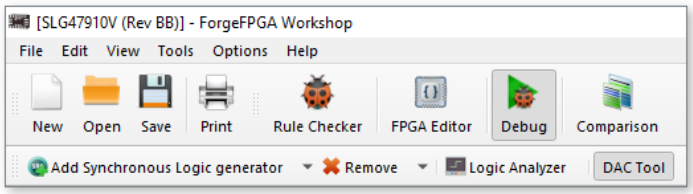


Figure 193. DAC Tool on the Toolbar

All analog I/O pins are disabled by default. Use an **Analog I/O n** button to enable the respective output pin and use the spin box on the right to set the desired voltage level.

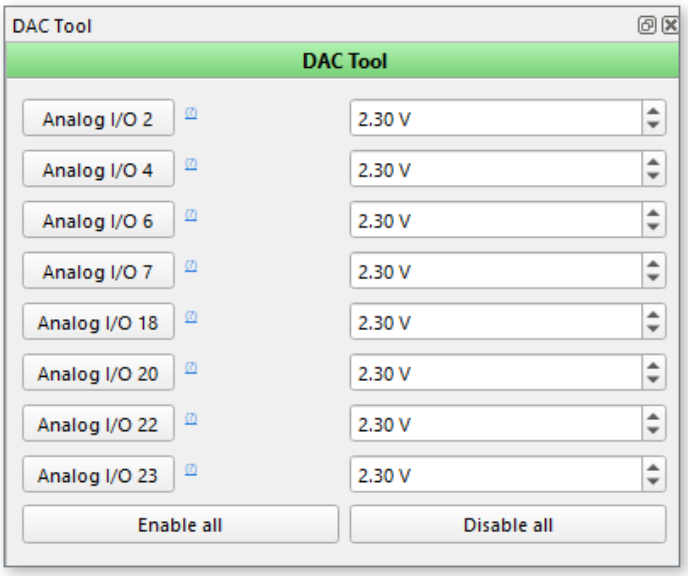


Figure 194. DAC Tool Window

Note: Setting a voltage level on analog I/O pins is enabled only if **Emulation** or **Test Mode** is activated.

2.2.14 OTP Programmer

OTP Programmer allows **OTP (One-Time Programmable) memory** to be read from a chip, modified, and programmed with new data. The tool shows differences and conflicts between the programmed chip project and the current project data.

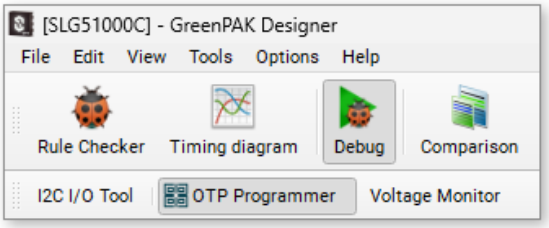


Figure 195. OTP Programmer on the Toolbar

To access the **OTP Programmer** tool, click the **Debug** button and select a suitable development

platform.

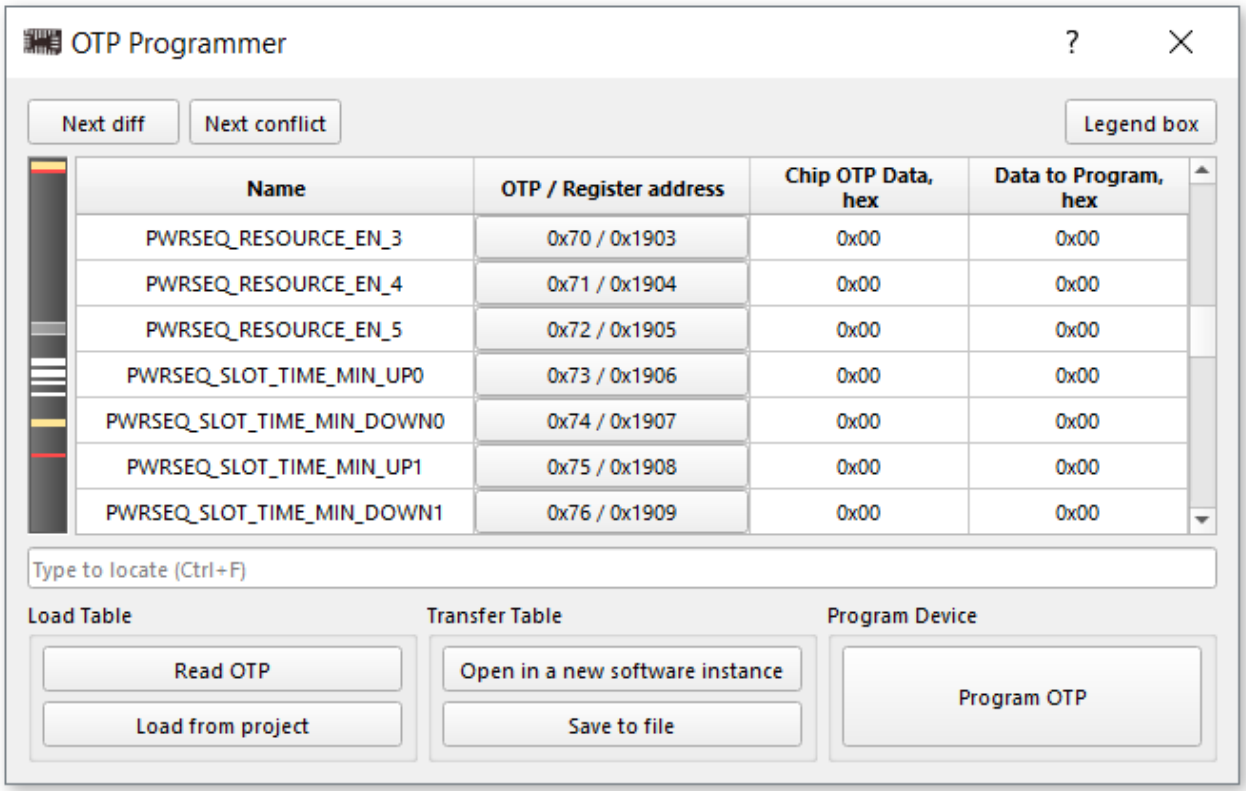


Figure 196. OTP Programmer Tool

Before working with **OTP Programmer**, check whether the chip is detected (platform and chip information should appear in the **Info details** on the **Debugging Controls** panel).

The tool contains the following controls:

- **Read OTP** – Read the data from the chip and load it into the **Chip OTP Data** column.
- **Load from project** – Load the data from the current project into the **Data to Program** column.
- **Open in a new software instance** – Open the **Data to Program** in a new software instance.
- **Save to file** – Save the **Data to Program** to a text file.
- **Program OTP** – Program the chip with the **Data to Program** (*Note:* It is possible to program bits from `0` to `1`, but not vice versa).

Open in a new software instance, **Save to file**, and **Program OTP** buttons become available after the **Read OTP** or **Load from project** buttons are used.

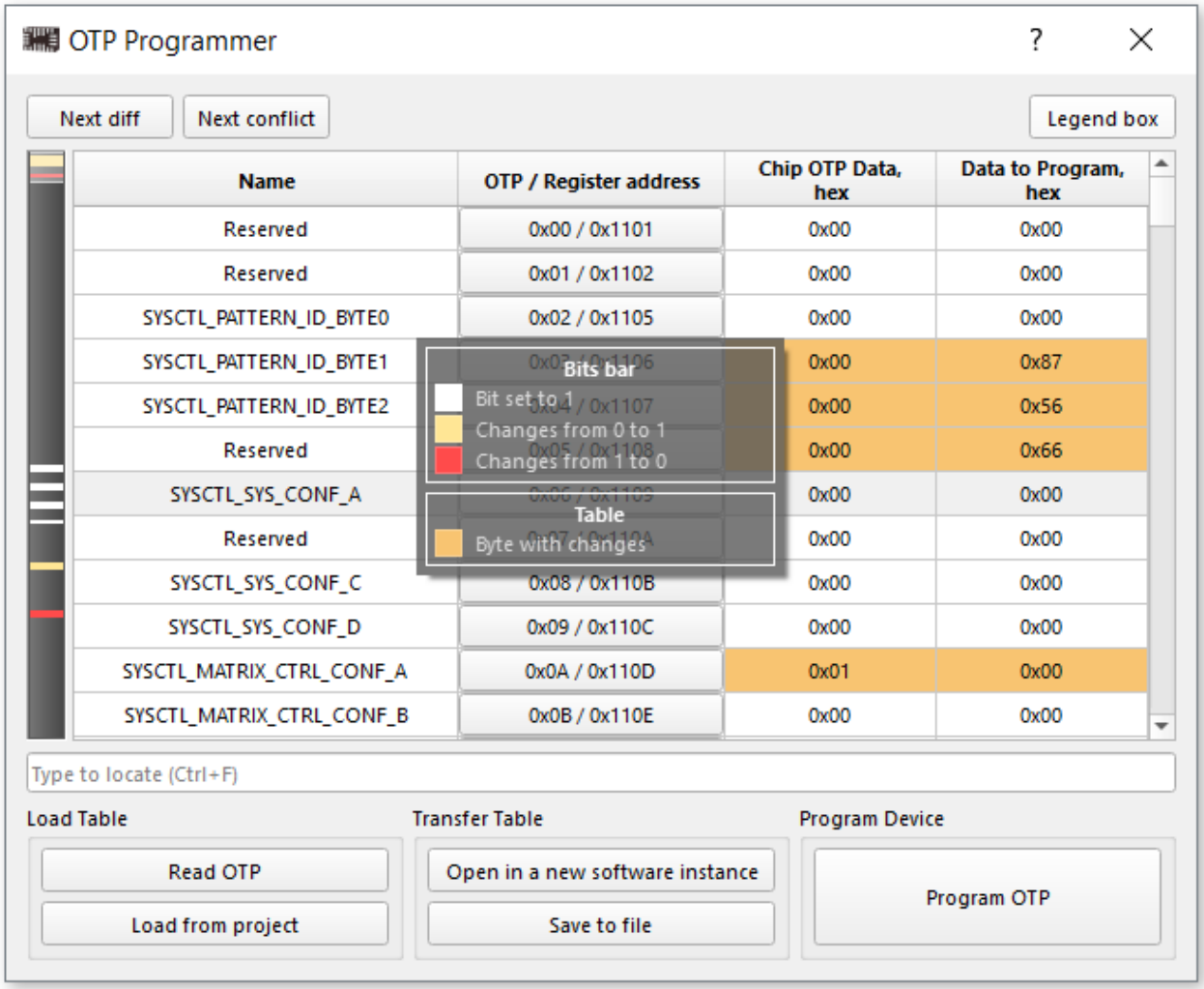


Figure 197. Legend Box

- **Legend Box** – Shows the color scheme of bit changes in the **Bits bar**.
- **Next diff** – Switch focus to the next **Byte with changes**.
- **Next Conflict** – Switch focus to the next **Byte with changes** in which the value was changed from `1` to `0`.
- **Search field** – Find a required byte or register.

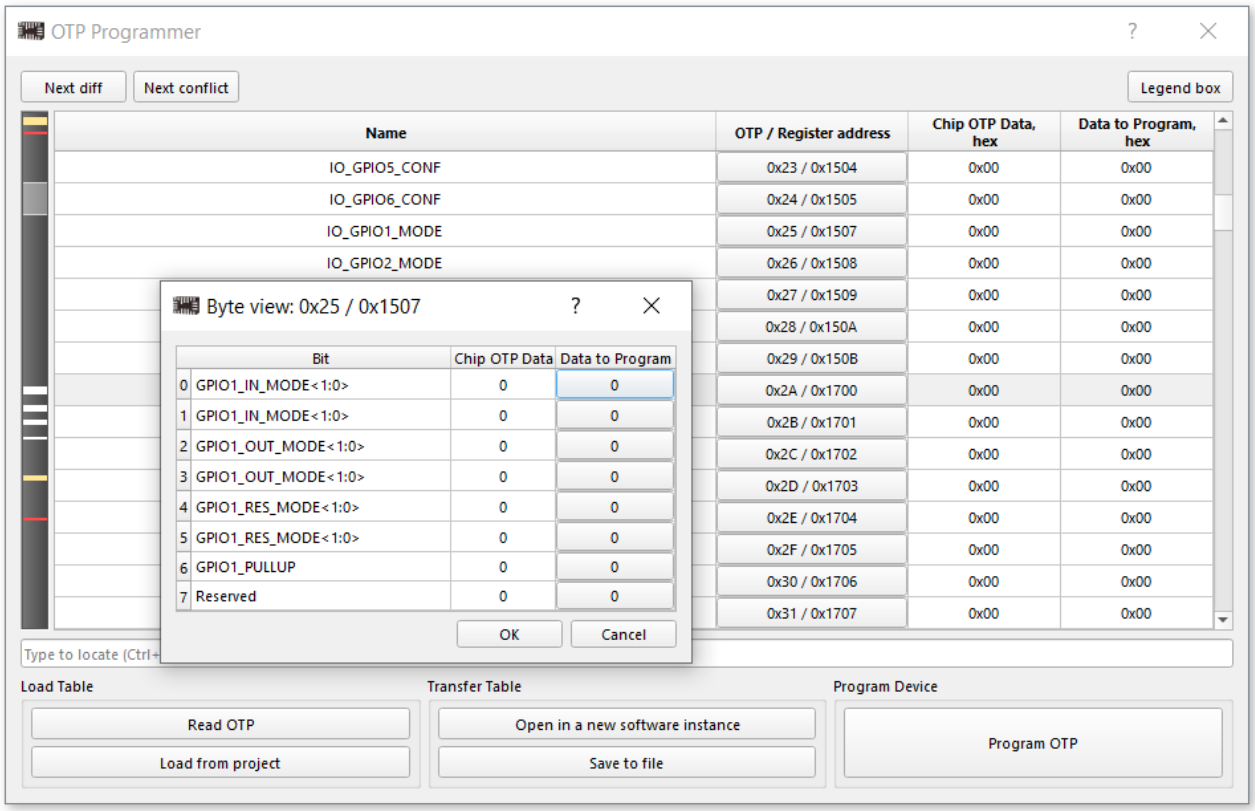


Figure 198. Byte View

Data to Program can be edited by double-clicking a cell in the **Data to Program** column or by opening the **Byte view window** through clicking the required cell in the **OTP/Register address** column.

2.2.15 Voltage Monitor

The **Voltage Monitor** tool helps measure the voltage on the ADC channels. The tool appears on the toolbar after **Debug** is clicked and the appropriate development platform is selected.

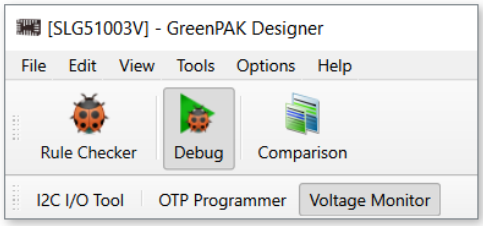


Figure 199. Voltage Monitor on the Toolbar

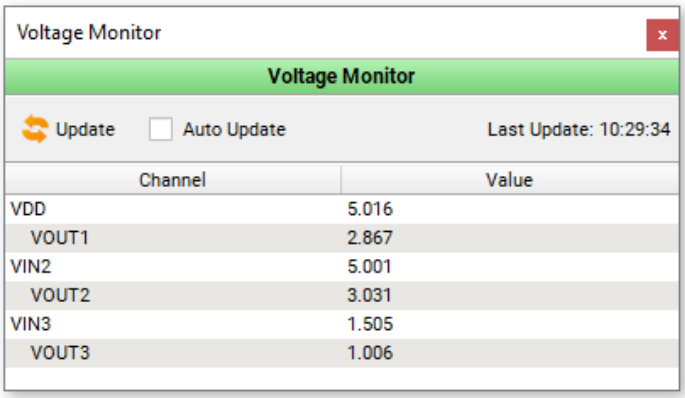


Figure 200. Voltage Monitor Tool

After the platform is connected, the controls on the tool become active. Use the corresponding controls for table update or auto-update.

The tool also provides the following values:

- **Channel** – Channel types with VIN and VOUT relations.
- **Value** – Value measured in volts. The accuracy of measurement is three decimal places.

2.2.16 Power Monitor

The **Power Monitor** tool is designed to measure chip voltage and current for power consumption calculation. The tool is available on the toolbar after clicking **Debug** and selecting the appropriate development platform.

Note: To activate the tool, **Emulation** or **Test Mode** is required.

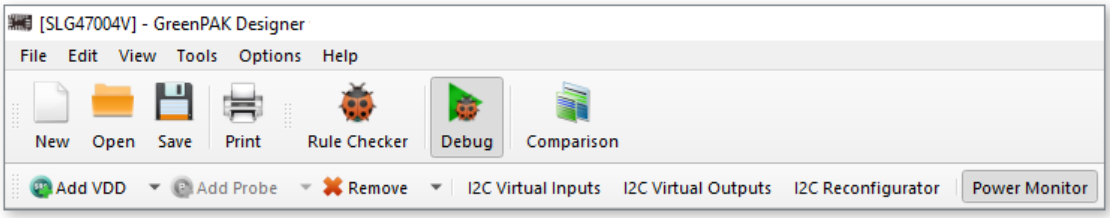


Figure 201. Power Monitor on the Toolbar

Use the corresponding controls to update or auto-update the data. For more details about the tool,

click the information button.

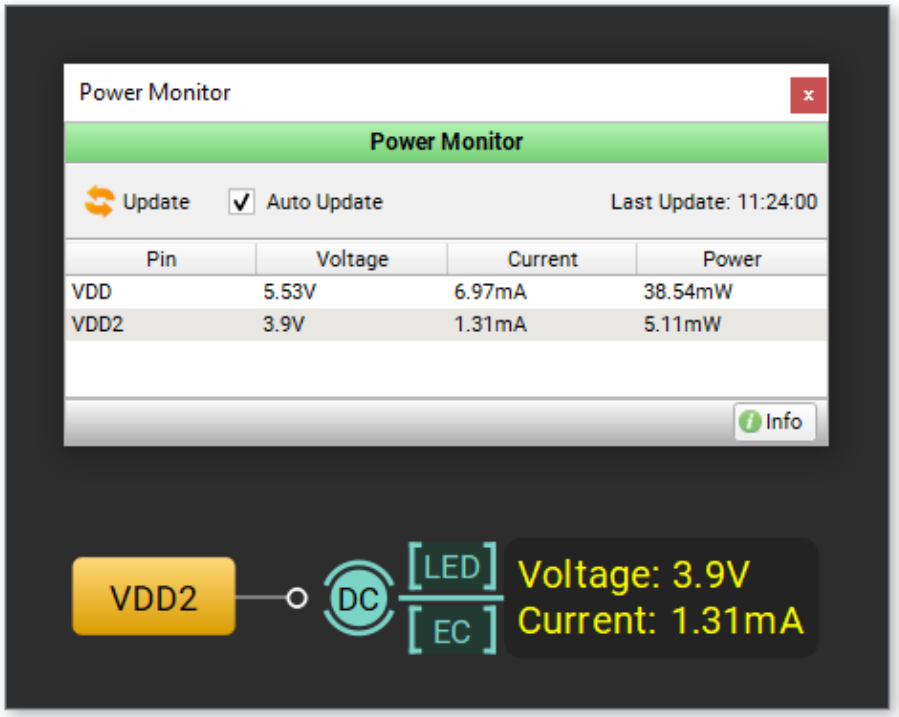


Figure 202. Power Monitor with Work Area Label

Work area labels provide a helpful reference for displaying voltage and current values. After **Emulation** or **Test mode** is activated and the data is refreshed, these labels light up, duplicating values from the **Power Monitor** window for convenient reference during the design process.

Notes:

- For the most precise adjustment of the circuit's current consumption, set all hardware sources connected to the chip outputs to **High-Z** and those connected to inputs to **Pull-Down**.
- For **External device**, if **External VDD** is selected, **Power Monitor** does not provide power consumption calculations, since current cannot be measured by the tool under this condition.

2.2.17 Current Limits

The **Current Limits** utility allows you to restrict the maximum current consumption on the Go Configure Development Board for all VDDs of the selected chip.

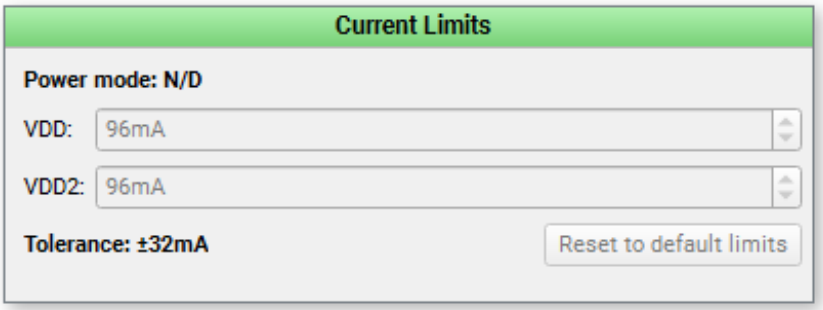


Figure 203. Current Limits Tool for Part Number with Two VDDs

By default, the limit for all VDDs is set to 96mA. This value is read-only if no external power supply is connected to the platform. The limits become editable once an external power supply is connected via DC Input or USB Power.

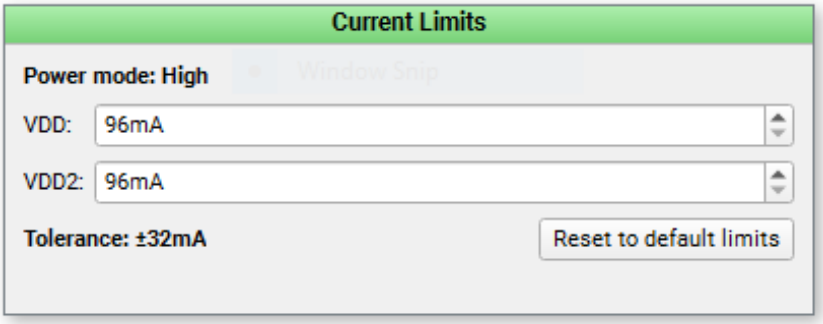


Figure 204. Active Current Limits Tool

The tool also displays the active **Power mode**, which is automatically determined by the connected power source.

If the current consumption exceeds the specified limits, the board power is automatically turned off, and the corresponding notification is displayed.

2.2.18 Demo Board and Demo Mode

The **Demo** mode allows exploration of possible applications for a specific part number. The **Library** tab in the **Hub** window contains a list of preconfigured projects. In **Demo projects** tab hover over

the desired design. Click **Details** to learn more about the design. To open the project, click **Run**.

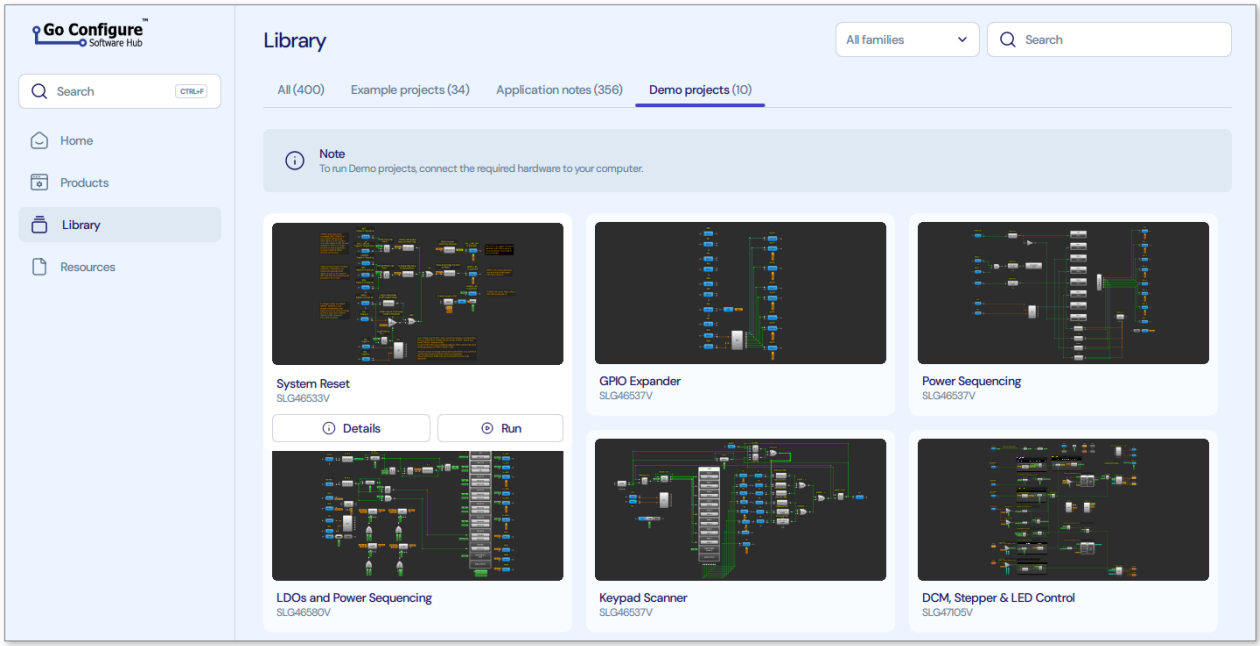


Figure 205. Select a Demo Project

To interact with the design, the corresponding demo board can be connected. Such a board contains a soldered IC with a preprogrammed project.

After the project is opened, the workspace UI depends on whether the demo board is connected.

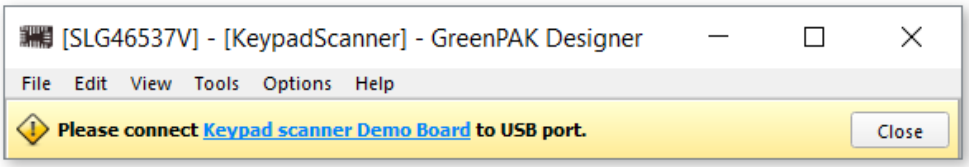


Figure 206. Waiting for the Corresponding Demo Board Connection

After the board is detected successfully, the following controls become available:

- **Write** – Load the current project's NVM sequence to the device.
- **I2C Tools** – The following I2C Tools are available for **Demo** mode:
  - **I2C Virtual Inputs**.
  - **I2C Virtual Outputs**.
  - **I2C Reconfigurator**.

  For more information refer to section 2.2.8 I2C Tools.
- – Information about a part number, development platform, and operating system.

- **Close** – Exit the **Demo** mode.

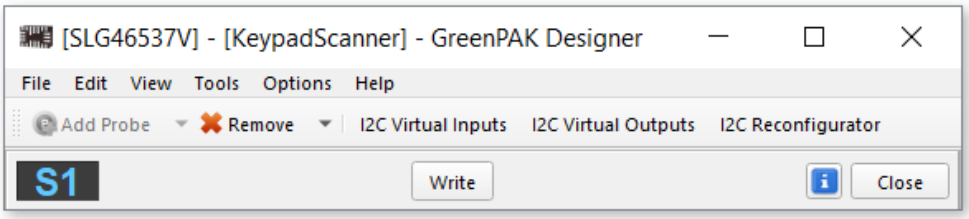


Figure 207. Demo Board Detected

Note: The **Demo** mode applies some limitations to specific features (operations with project files, Debug Tool, **Simulation**). Exiting **Demo** mode removes all limitations while keeping the current project open.

2.2.19 Design Library

The **Design Library** tool provides access to demonstration designs and user-saved configurations for the connected platform. It allows managing and loading configurations into the device and review their corresponding NVM sequences. The interface includes the following tabs: **Demo Designs** for predefined example projects and **User Designs** for stored user configurations.

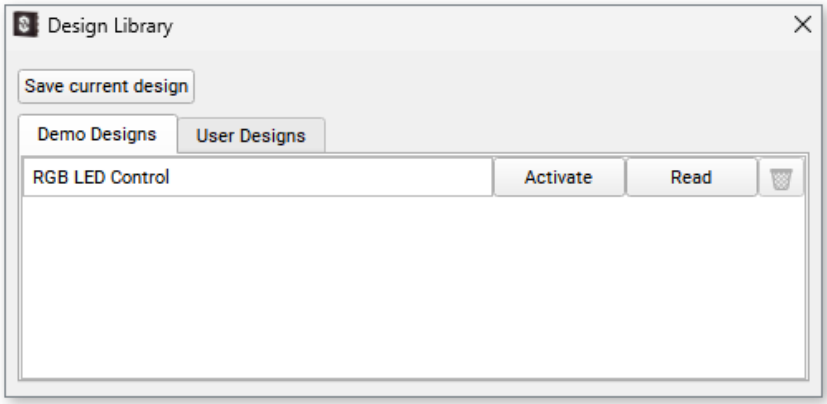


Figure 208. Design Library Demo Designs Tab

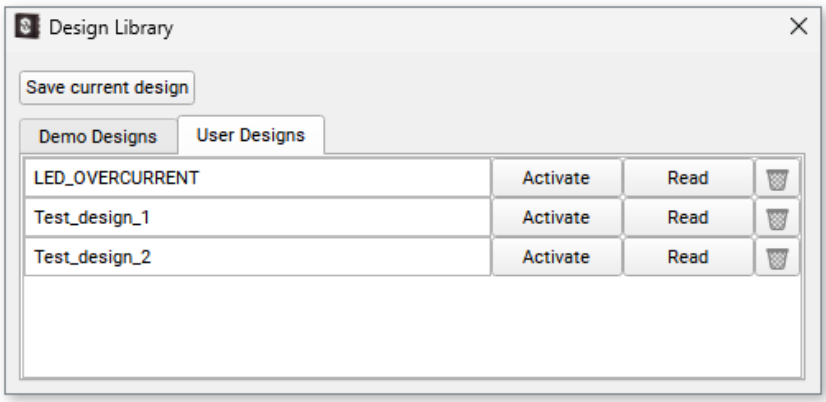


Figure 209. Design Library Users Designs Tab

The main controls are as follows:

- **Save current design** – Saves the active configuration to the **User Designs** tab.
- **Activate** – Loads the selected design into the IC's RAM.
- **Read** – Displays the design's NVM sequence for export or for opening in a new software instance.

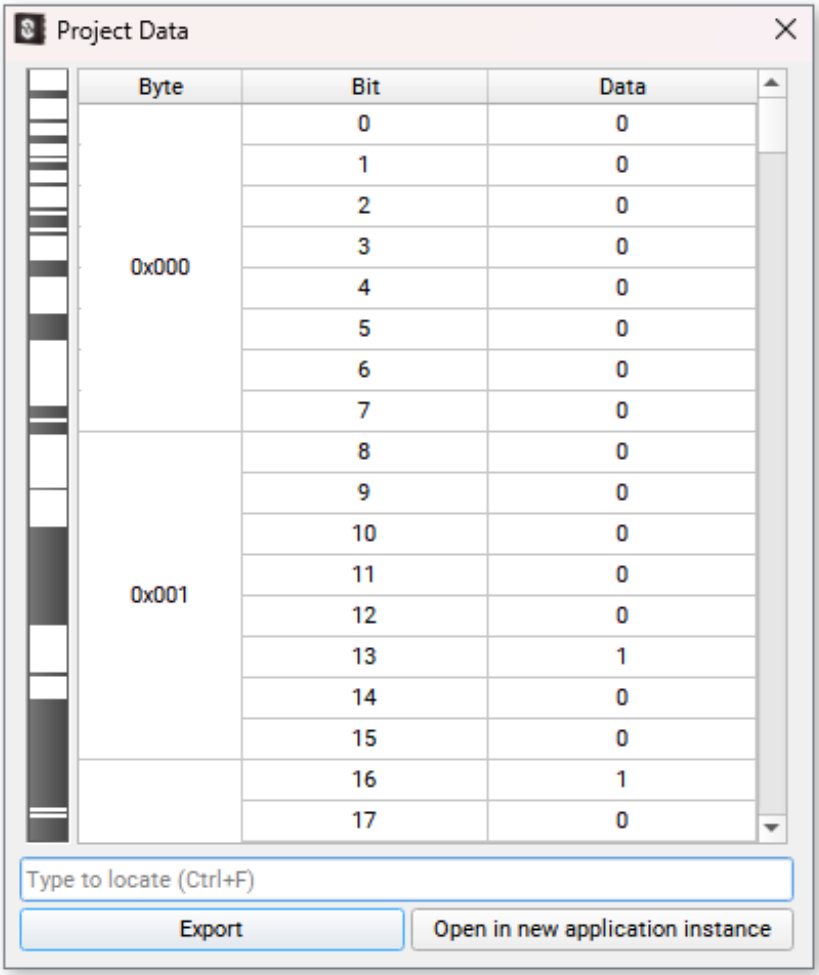


Figure 210. Project Data Window Opened After Read

2.2.20 Go Configure Driver Tool

If the driver required for development platform detection is missing, or if a conflicting driver is present in the system, the board cannot be connected successfully. **Go Configure Driver Tool** is designed to resolve this situation (applicable for Windows OS).

Its main functions are as follows:

- Detect drivers required for successful development platform connection.
- Install missing drivers.

- Remove drivers.
- Resolve driver conflicts (remove conflicting drivers and install missing ones if required).

The tool can be launched manually from the main menu, **Tools** > **Go Configure Driver Tool**. It displays the list of development platforms and the driver statuses (whether the driver is present, missing, or conflicting).

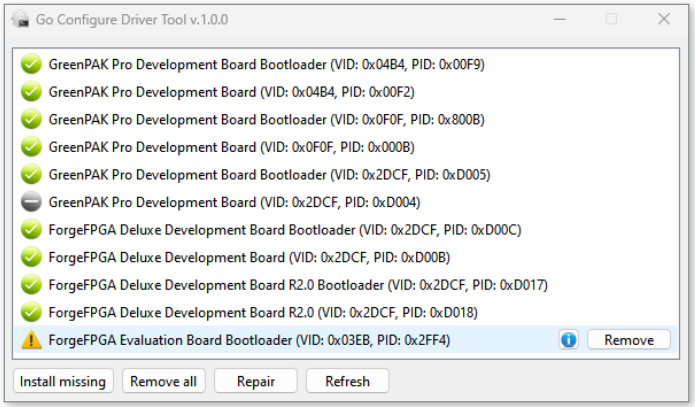


Figure 211. Go Configure Driver Tool

When the platform is selected, the banner on the **Debugging Controls** panel indicates the detected driver issue. Click **Resolve** to launch the **Go Configure Driver Tool** and proceed with the fix. In this case, the tool lists the drivers relevant to the selected platform and their statuses.

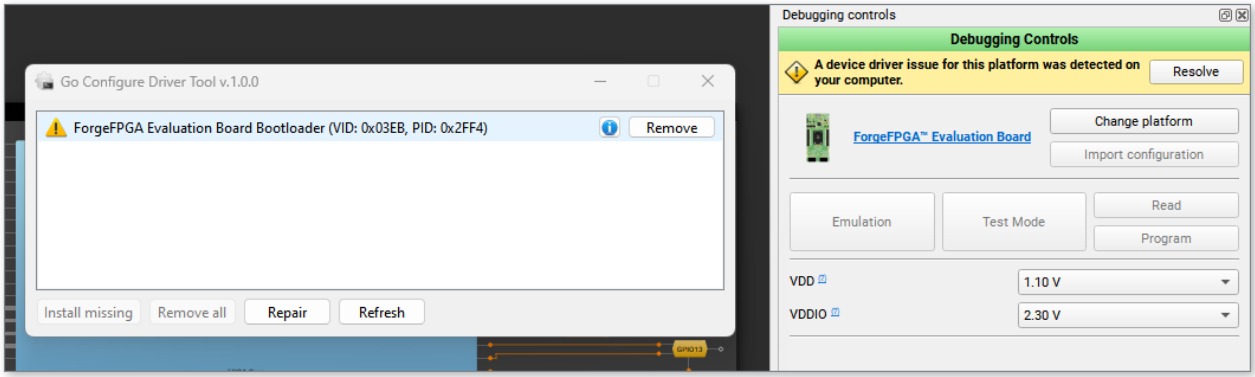


Figure 212. Driver Issue Detection

Click the info icon to find more information about the conflicting driver that prevents successful

board detection.

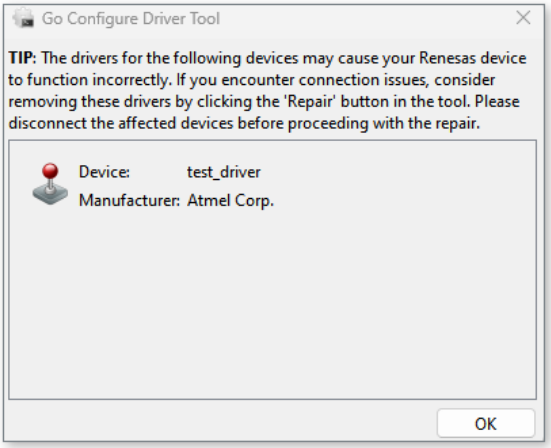


Figure 213. Conflicting Driver Info Pop-Up

After the driver issues are resolved, the pop-up prompts for reconnection of the platform.

2.3 Configuration Tools

2.3.1 Asynchronous State Machine Editor

GreenPAK family devices featuring the **Asynchronous State Machine (ASM)** macrocell allow custom state machine designs to be developed. State definitions can be established, permissible state transitions can be defined, and the signals responsible for initiating each state transition can be specified. Moreover, this macrocell can be linked to various I/O Pins and other internal GreenPAK components to activate inputs for state transitions. Outputs from the macrocell can also be directed to other internal macrocells or I/O Pins as needed.

ASM Editor allows configuring the ASM component using the state diagram and setting the output configuration for the ASM Output macrocell.

To open the ASM Editor, click the **ASM Editor** button on the toolbar or double-click the ASM component.

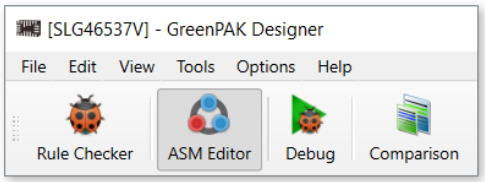


Figure 214. ASM Editor on the Toolbar

2.3.1.1 ASM Editor's Interface

A state's properties can be viewed and configured in the **Properties** panel. The **RAM** properties panel at the right contains the **Connection Matrix Output RAM**. This feature establishes the relationship between controlled outputs and specific states. Additionally, certain part numbers might have their output pin behavior regulated by these states. This behavior is indicated in the **GPOs Output RAM** table.

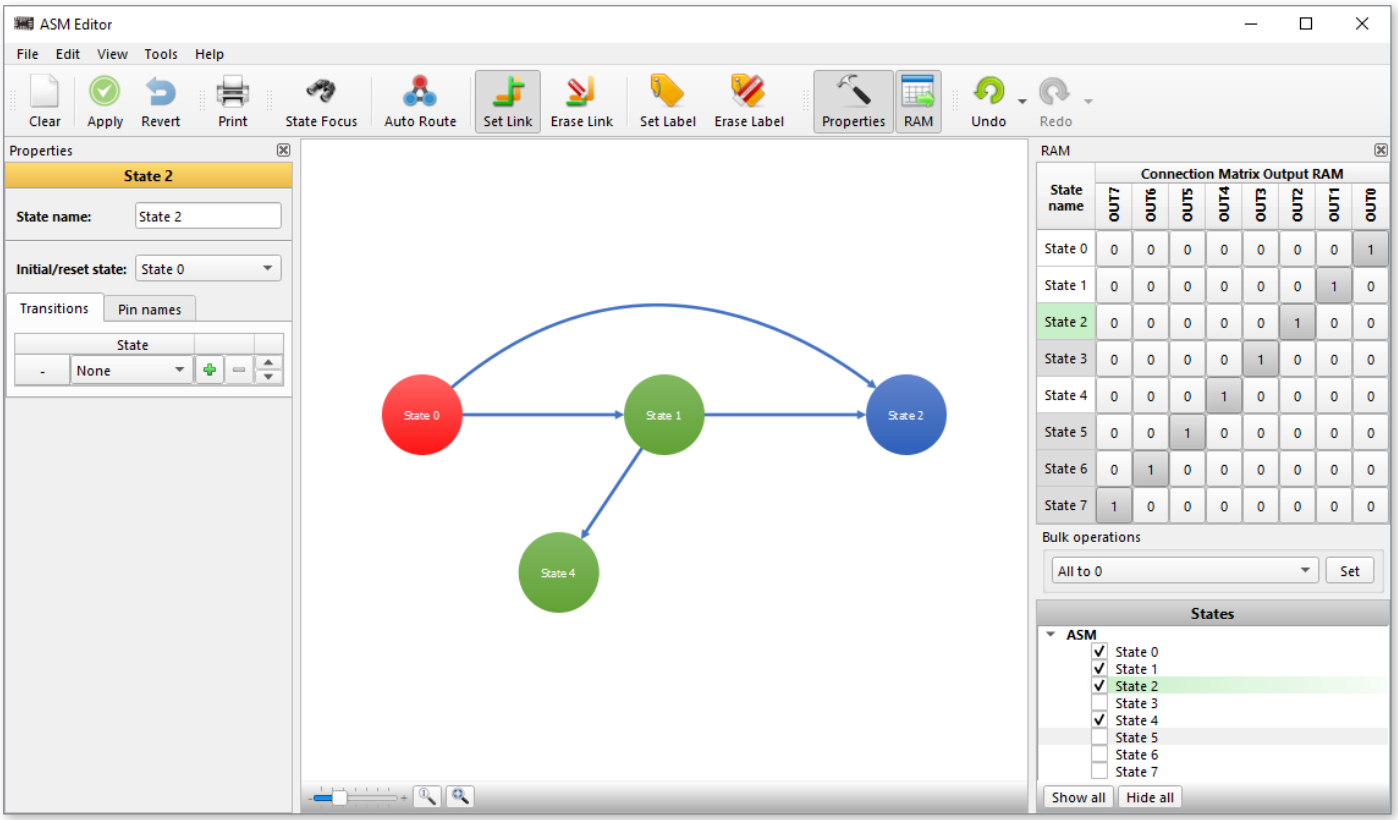


Figure 215. Setting a State Transition

Bulk operations allow applying a selected operation (**All to 0**, **All to 1**, **Invert**, and resetting to **Default**) to the whole table.

States can be shown or hidden by toggling the corresponding checkboxes within the States section located in the lower right area of the **RAM** properties panel.

The **main area** of the ASM editor depicts states. The color of a state changes according to the state's interaction options.

Regular state.

Selected state.

Initial (reset) state.

State transitions limit reached.

2.3.1.2 Configuring States

If states have not yet been established, the **ASM Editor** displays two detached states. These states are illustrated as circles. To relocate a state, use the left mouse button to drag its inner circle. *Note:* dragging the outer circle does not produce any changes. To establish a connection, click the outer ring of one state and then click the ring of another state.

For some chips, a state may be directed to itself by selecting the state's outside ring on both the first and second click. The number of states a particular state can transition to is unrestricted; it can transition to all the utilized states or none at all.

To rename a state, double-click the inside of the state circle or select the state to access the **State name** setting from the **Properties** docker at the left.

Transitions can be adjusted to best fit the design. Drag the link to change its shape. Select the **Edit path** mode from a connection's context menu if further adjustment is required. *Note:* A link's shape is updated automatically and any manual adjustments are discarded every time a state is repositioned.

The software can automatically reorder the states and transitions by clicking **Auto Route** on the toolbar. A label can also be added to a transition. To do that, click **Set label** at the top menu, or right-click an existing transition and select **Set label**.

The display can also be focused on the states by clicking the **State focus** button on the toolbar.

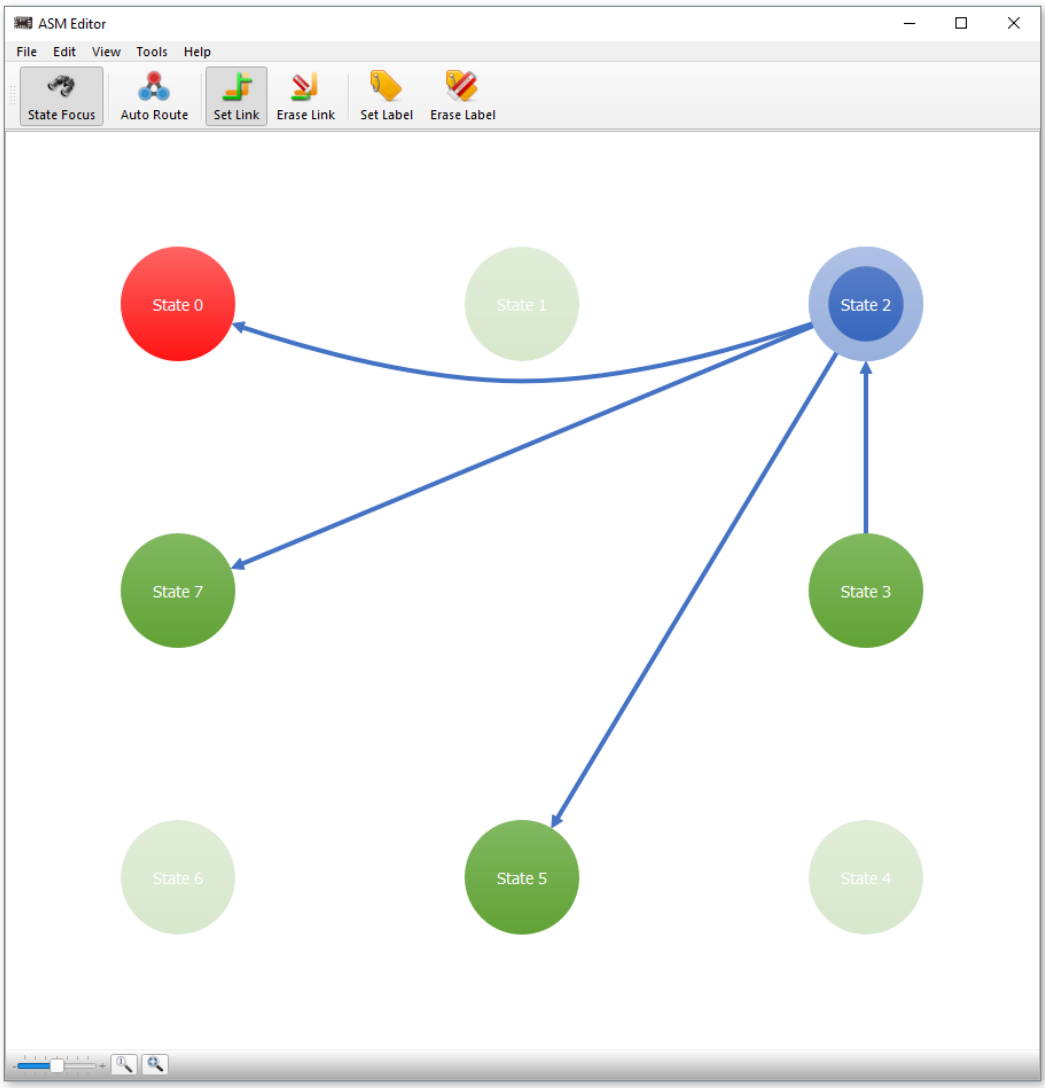


Figure 216. State Focus Mode

In order to set an **ASM** configuration to the **NVM**, apply it by clicking **Apply** on the toolbar. To return to the latest saved state of the ASM, click **Revert**.

The context menu of a state includes:

- **Edit name** – Set state's name.
- **Initial state** – Set the current state as the initial.
- **Hide** – Hides the current state.

A state's properties can be configured in the **Properties** panel at the left. The available tabs are:

- **Transitions** – Lists of the states to which the current state transitions. Allows adding a new transition by selecting another state from the list.
- **Pin names** – This tab's data is common for all states. It lists pin names set the current state

as the initial.

- **RE (Rising Edge)** – Lists the states the current state transitions to and applies **Rising Edge** to the transition; otherwise, ASM is level sensitive.

2.3.2 EPG Waveform Editor

The **Extended Pattern Generator (EPG)** is a component featuring up to 8 outputs. Each can be individually configured using up to a 92-bit large logic pattern. It retrieves data from non-volatile memory **(NVM)** and delivers it to the outputs bit by bit on each rising edge of the **CLK input** signal.

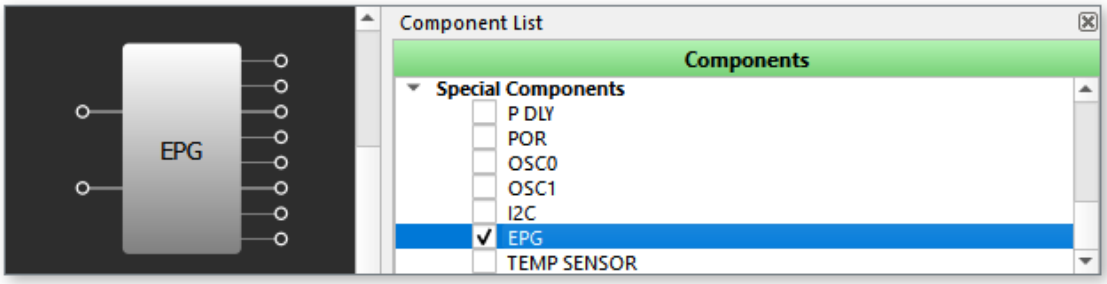


Figure 217. Extended Pattern Generator

The **Extended Pattern Generator** can be configured in the **EPG Waveforms Editor**. This tool allows selection from the available predefined generators, including **SPI**, **I2C**, **PWM**, or **Manual**, and assignment of the output accordingly. Each of these generators offers a range of adjustable

settings, making it more convenient to create the desired patterns.

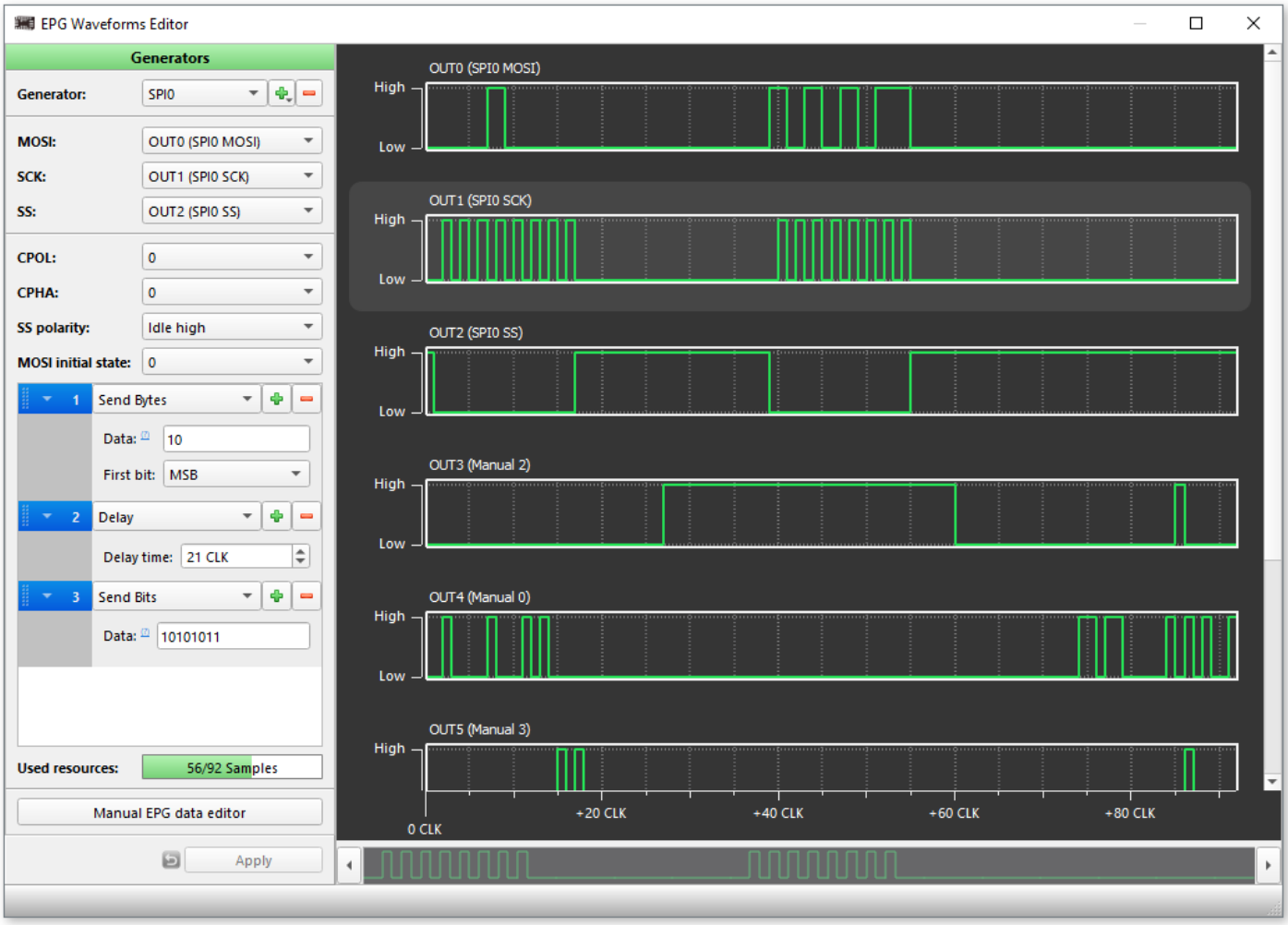


Figure 218. EPG Waveforms Editor

Within the **EPG Waveforms Editor**, a **resource meter** is also available to visually represent the number of bits used in the pattern. This feature provides a transparent view of the memory and resources allocated to the current pattern.

The **SPI** generator allows selection and configuration of the output pin using the **command editor**.

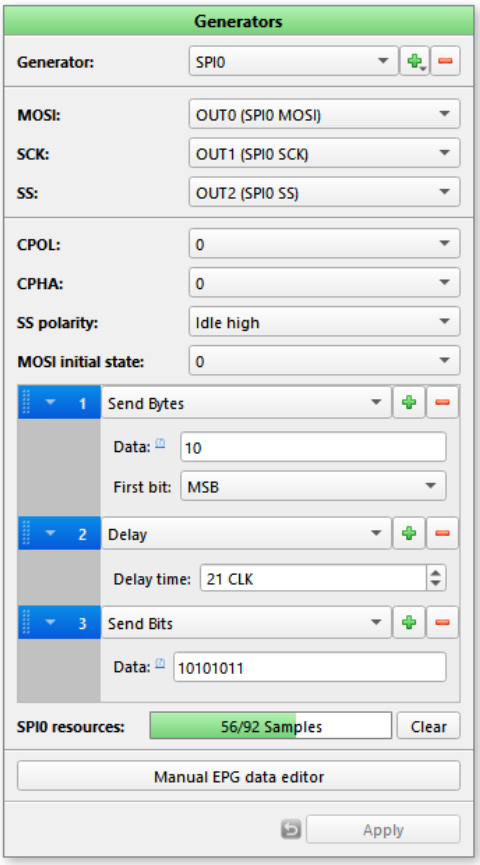


Figure 219. SPI Generator Command Editor and Resource Meter

The **I2C generator** allows selection of the **SDA** and **SCL** output pins and definition of commands using the **command editor**. Within the editor, basic commands can be selected from a menu, such as **Start**, **Stop**, and GreenPAK access operations like **Read** or **Write**. The **I2C generator** has an **S** button that allows complex GreenPAK access commands to be broken down into a series of basic

commands.

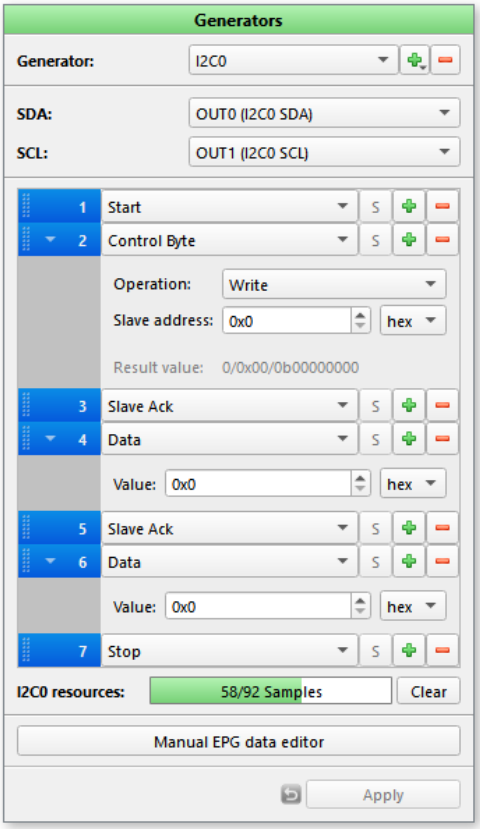


Figure 220. I2C Generator Command Editor and Resources Meter

The **PWM generator** configures the **output pin**, **period duration**, **high** level duration, and **phase shift**.

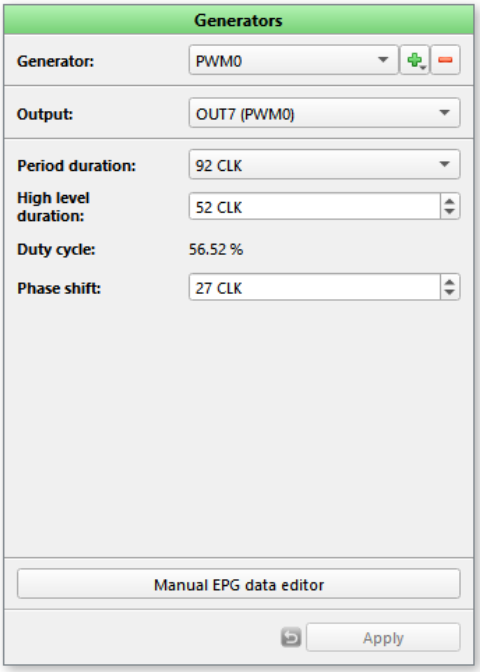


Figure 221. PWM Generator Options

The **Manual** generator allows configuration of the bit values.

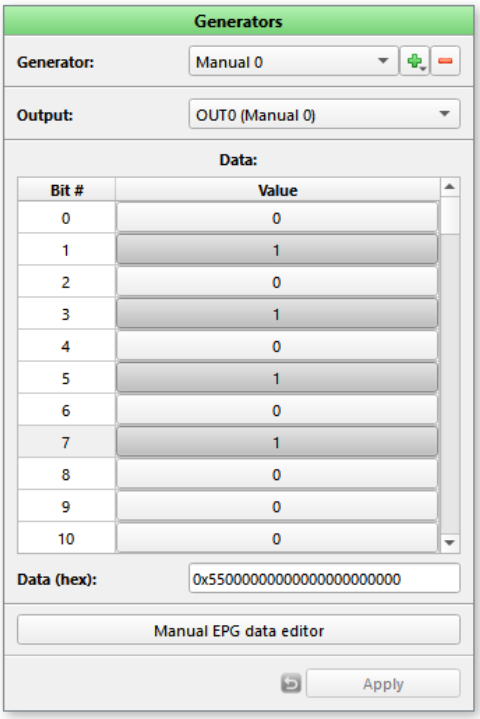


Figure 222. Manual Generator Options

When the system is powered up, the behavior of the EPG varies depending on the signal received at the nReset input. In particular, if the **nReset** input is in an active LOW state, the EPG will display the initial value at the output. Conversely, if the **nReset** input is active HIGH, the EPG will display user-defined patterns at the output. This feature makes it possible to tailor the output of the EPG to specific needs. Moreover, EPG can work continuously when CLK is being applied in **Overflow** mode or keep at the last byte in **Stop** at **Boundary** mode.

2.3.3 Timing Diagram

State control timing diagram is a tool that allows management of the **Power sequencer** block configuration. The **Power sequencer** controls the **power-up** and **power-down** timings for the six resource enable outputs, which feed the matrix interconnect. The timing sequence is divided into six slots, which are periods between the events. The minimum and maximum duration for each slot can be set. The sequence is initiated with the **trigger-up** and **trigger-down** control signals from the matrix interconnect. The **Power sequencer** and, therefore, the **Timing diagram** tool are available,

for example, for SLG51000/1/2.

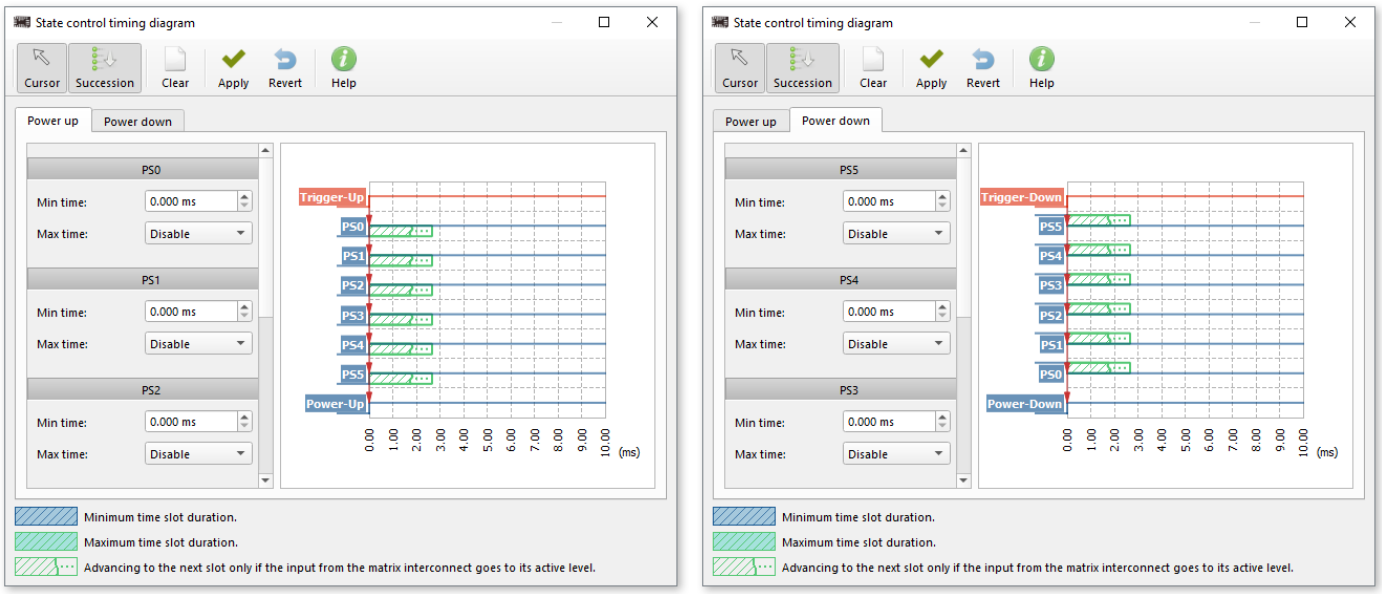


Figure 223. Timing Diagram

The slot view depends on the Min. time and Max. time settings. The legend is shown at the bottom of the **Timing diagram** window.

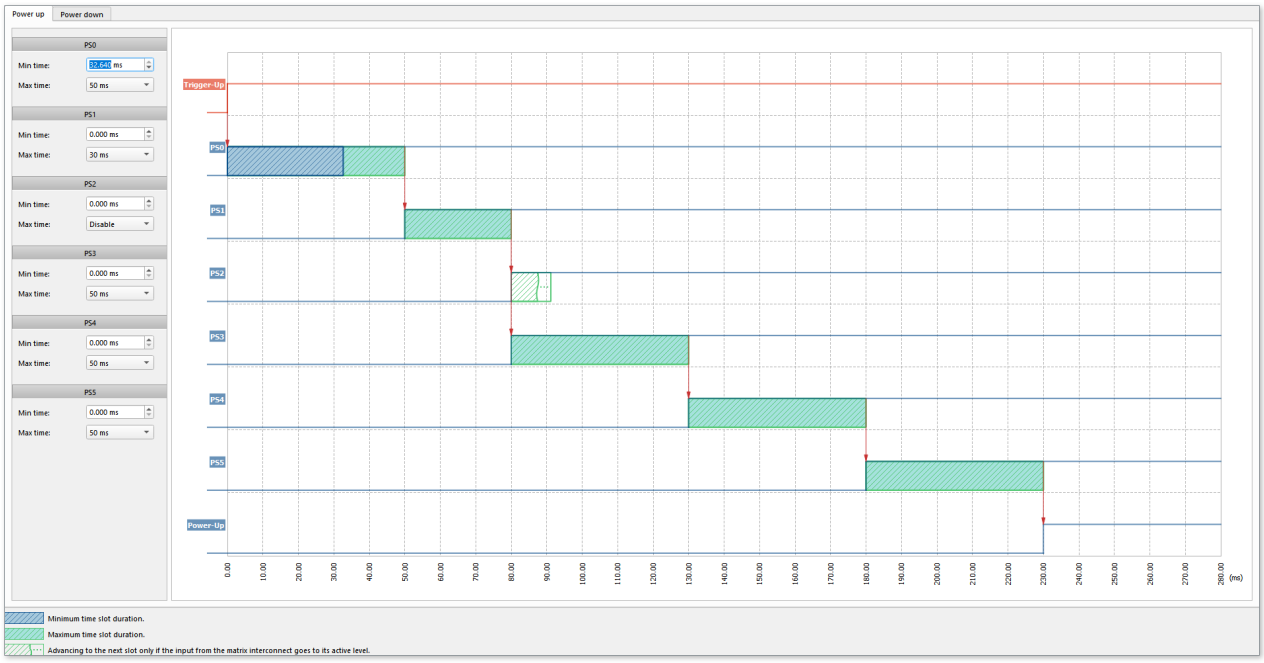


Figure 224. Slot View Examples

Use the toolbar controls to configure the **Power sequencer**.

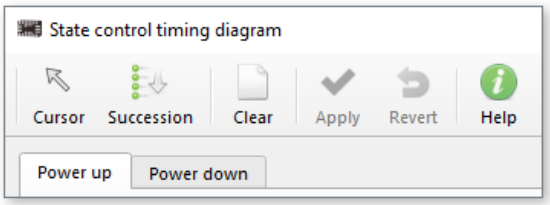


Figure 225. Timing Diagram Options

2.3.4 Memory Table Editor

Memory Table Editor allows configuration of the Memory table macrocell (make sure the ROM mode is selected on the macrocell's **Properties** panel). To open the tool, click the **Memory Table Editor** button on the toolbar or double-click the corresponding component in the work area.

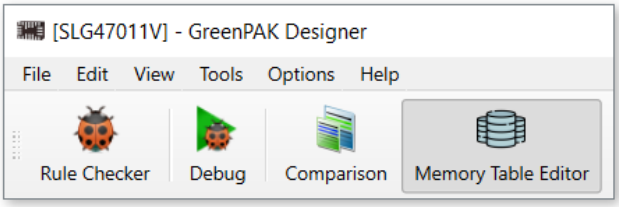


Figure 226. Memory Table Editor on the Toolbar

The **Memory Table Editor** window visualizes the output data of the Width Converter (WC) in graphical format (WC provides data from Memory Table to the connection matrix).

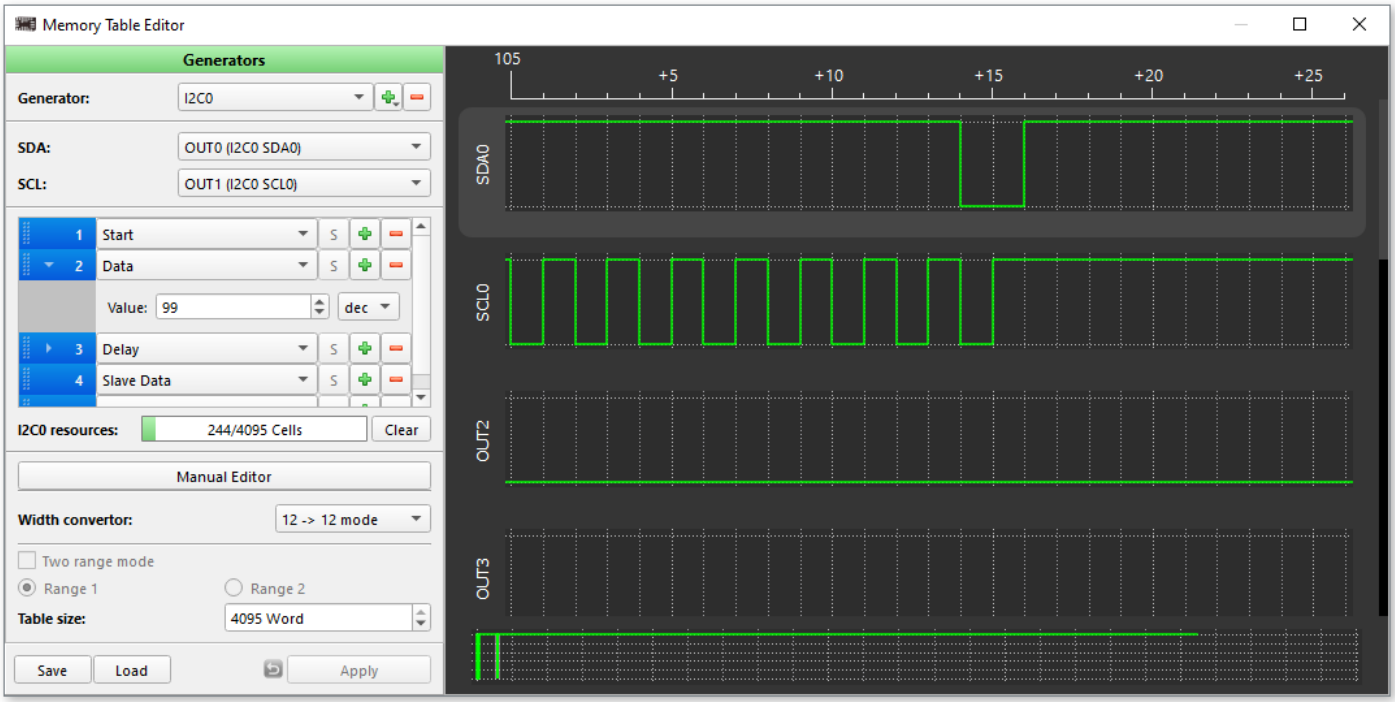


Figure 227. Memory Table Editor Window

The tool provides the following possibilities:

- Add the generator type. For **I2C** and **SPI**, use the available generator-specific settings; for instance, specify which Width Converter output corresponds to which generator signal.

  When **Manual** generator is selected, the word (digit) is represented in binary format. In manual mode, data can be entered in the provided table, which can be bound to a specific output.

  The **Signal** generator represents the word in decimal format.

  The x-axis on the graph is the word sequence number, a memory cell (**Word index**). The y-axis is a word saved in the memory cell (**Word decimal**).

  Use **Manual Editor** to customize the data (read more below).

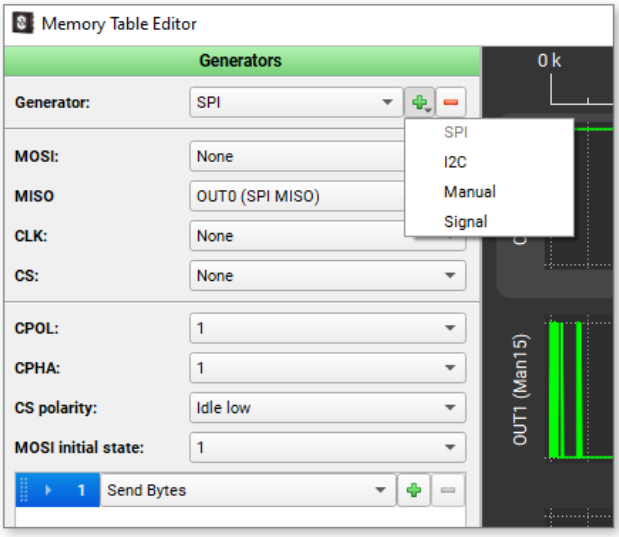


Figure 228. SPI Generator Configuration

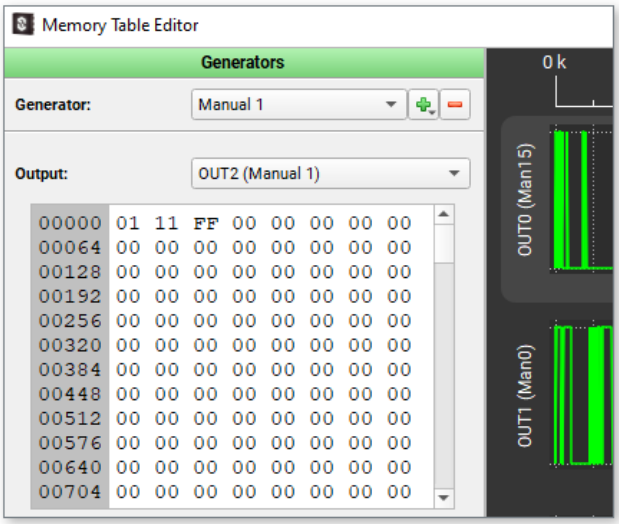


Figure 229. Manual Generator Configuration

Note: If a generator is locked and cannot be added, try changing the **Width convertor** mode.

- Check the **Resources** bar, indicating the number of available cells. The number of cells, as well as the displayed output signals, depends on the selected **Width convertor** mode, which are:
  - 12 > 12 (12-bit parallel output).
  - 12 > 4 (12-bit word to three 4-bit words).
  - 12 > 2 (12-bit word to six 2-bit words).
  - 12 > 1 (12-bit word to serial bit stream).

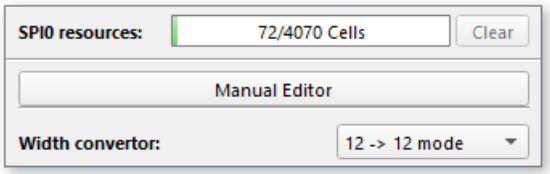


Figure 230. Resources and Width Converter Mode

- **Manual Editor** allows modification of the macrocell's bit values for specific registers, and consists of three main parts:
  - Navigation controls – Enter the bit number and click the **Go to bit** button to jump to the desired bit.
  - Registers value configuration – Configure the registers by setting values in binary, decimal, or hexadecimal format (bits set to `1` are indicated with white highlight on the sidebar).
  - Bottom controls – **Import**/**Export** or **Clear** the data.

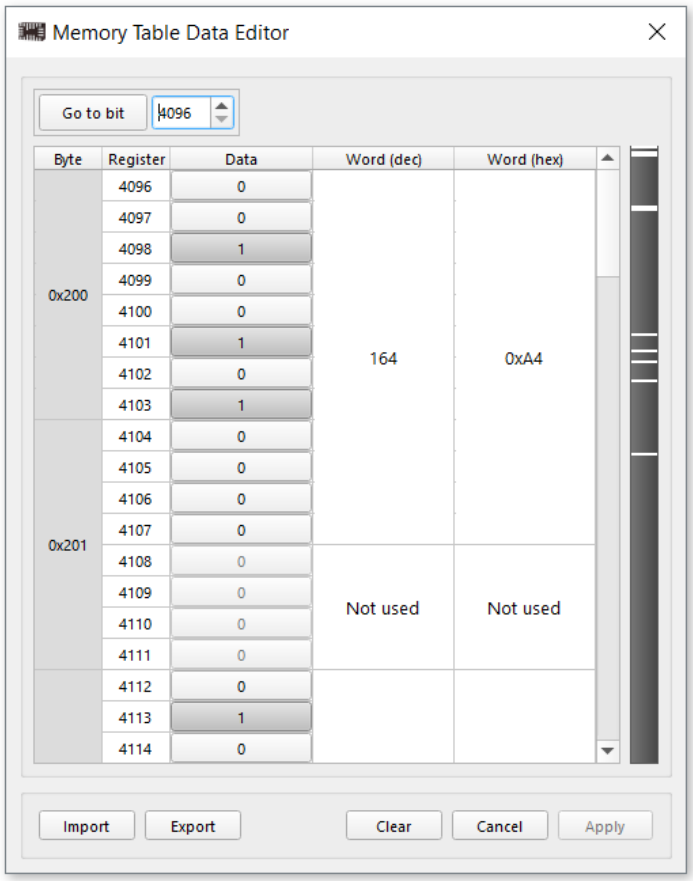


Figure 231. Memory Table Data Editor

The data set in the **Manual Editor** is synchronized with the **Manual** generator properties panel.

- The additional settings provide flexibility in table usage by defining access to the memory cells. To activate the **Two range mode** checkbox, the following conditions should be met:
  - Memory Control Counter (MCC): **Range mode select** = Two ranges; **Initial CNT data value** != 0 (make configurations on MCC's **Properties** panel).
  - Generator is added.

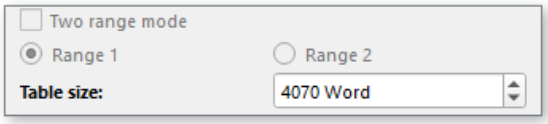


Figure 232. Additional Settings

- Save/Load the file with configurations using the corresponding bottom controls.

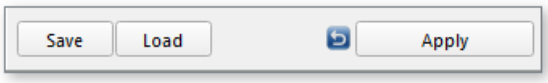


Figure 233. Save/Load Controls

See some additional plot controls:

- Zoom the graphs with **Ctrl** + **mouse wheel**. Reset scale in the context menu.
- Enable legend for each plot in the context menu.

See some additional plot controls:

- Zoom the graphs with **Ctrl** + **mouse wheel**. Reset scale in the context menu.
- Enable legend for each plot in the context menu.

2.3.5 HBRAM OTP Data Editor

HBRAM OTP Data Editor allows modification of the bit values of BRAM and User OTP macrocell registers. Open the tool from the toolbar or from the BRAM/User OTP macrocell **Properties** panel.

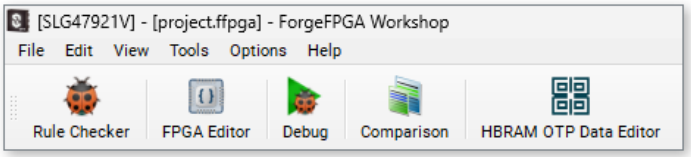


Figure 234. HBRAM OTP Data Editor on the Toolbar

The tool provides separate tables for configuring BRAM and User OTP macrocell bits. The data can

be displayed in either decimal or hexadecimal format.

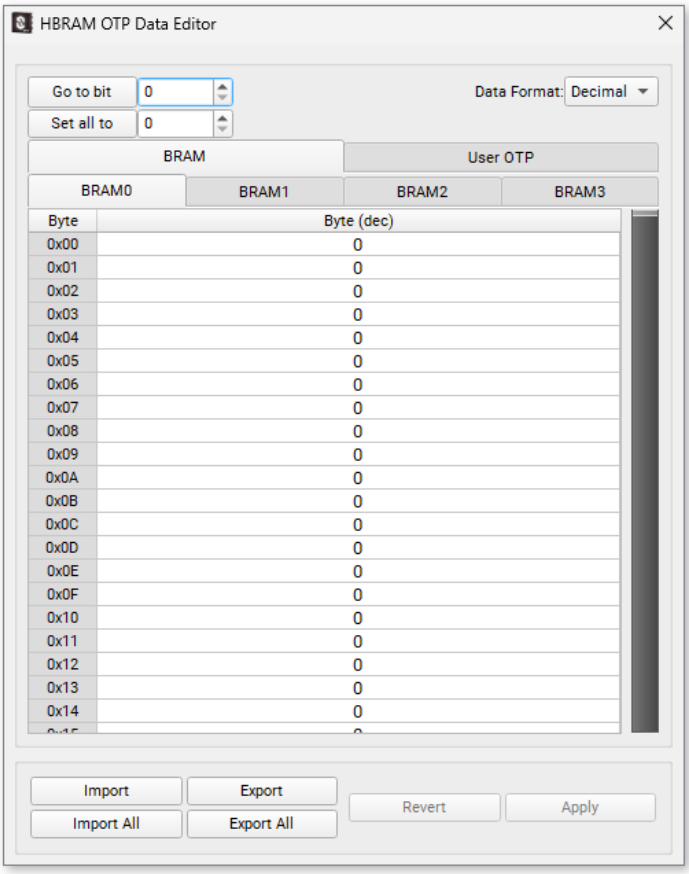


Figure 235. HBRAM OTP Data Editor (BRAM Tab)

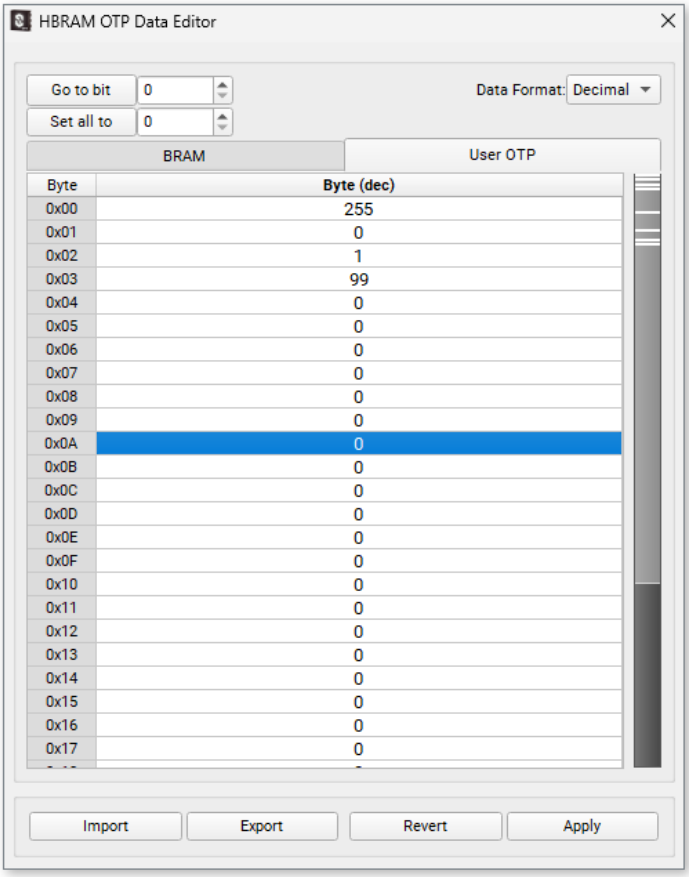


Figure 236. HBRAM OTP Data Editor (User OTP Tab)

The upper controls allow navigation through the table with the **Go to bit** button. A specified value can also be assigned to all bits in the active tab using **Set all to**.

The controls at the bottom allow data to be imported or exported for the active tab. For BRAM, the **Import all** and **Export all** buttons are also available to import or export data for all tabs.

2.4 Software Simulation Tool

The **Software Simulation** mode enables electronic circuit simulation. It uses mathematical models to replicate the behavior of the chip macrocells and the external components. To start **Software Simulation**, click **Debug** on the toolbar and select the tool on the **Development Platform Selector**. Refer to the **Hub** window > **Products** tab > part number click, to check the simulation support availability for a certain part number.

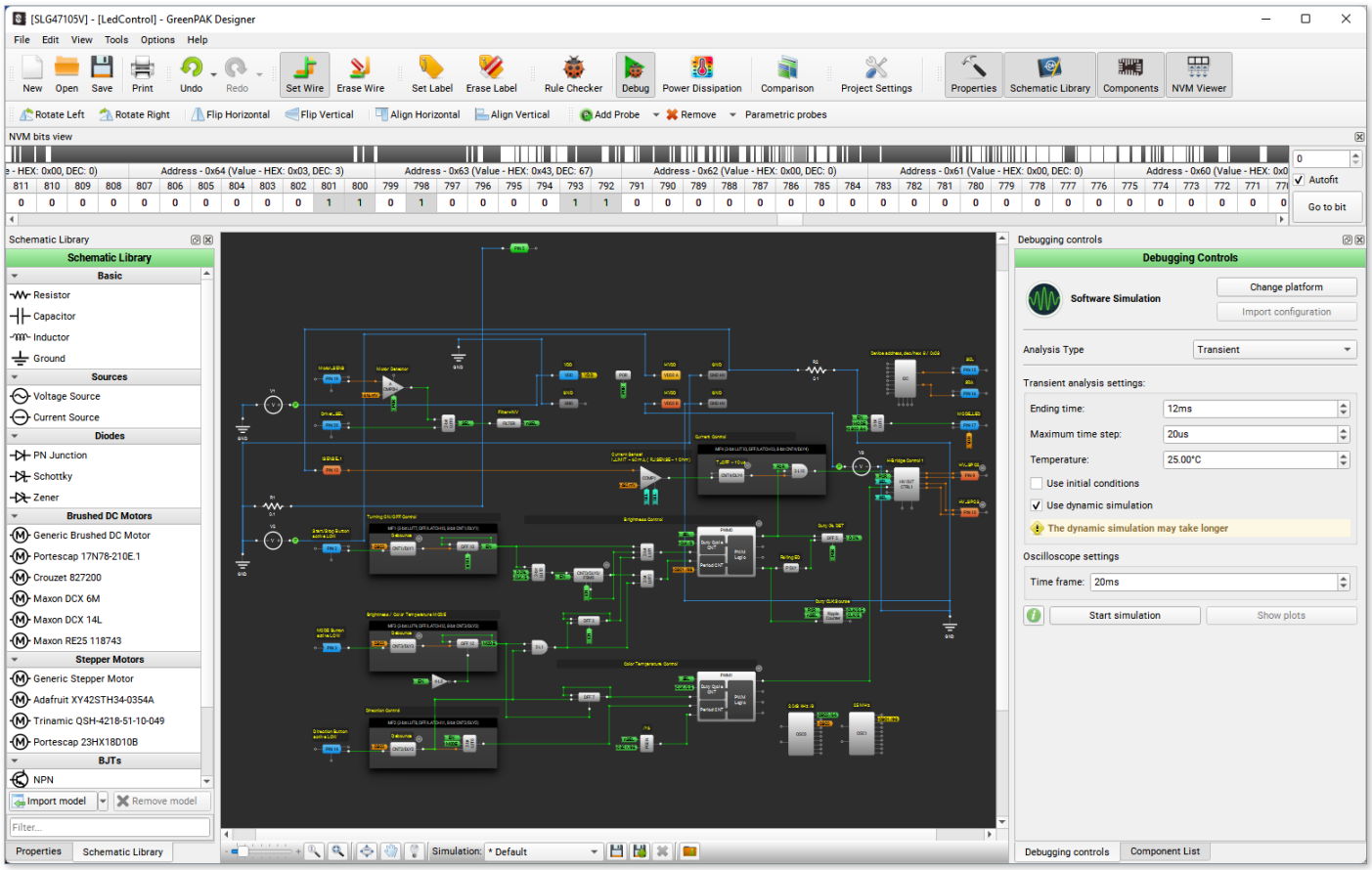


Figure 237. Software Simulation Tools

Before starting an analysis, external components from **Schematic Library** can be used and probes can be added to configure the simulation environment parameters.

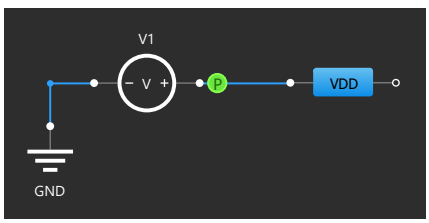


Figure 238. Basic Simulation Tools

2.4.1 Schematic Library

The **Schematic Library** is a repository of external components that can be applied to the design. External components are not part of the chip; however, adding and configuring them allows simulation of the system's behavior.

A custom component can also be added to the **Schematic Library** list by using the **Import model** and **Import subckt** buttons on the **Schematic Library** panel (see section 2.4.3 Working with Models and Subcircuits for details). Imported models and subcircuits appear in the designated location in the **Schematic Library**.

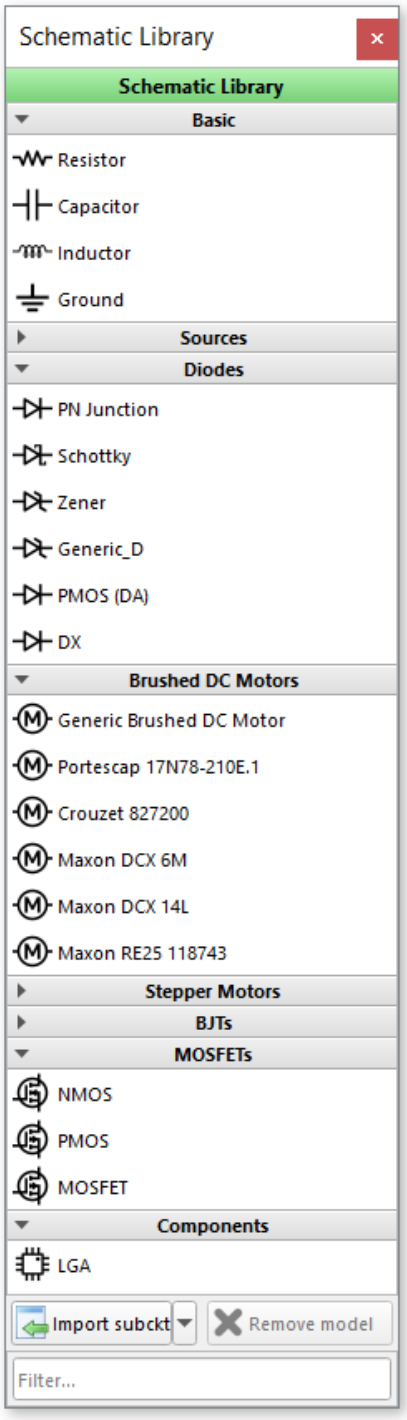


Figure 239. Schematic Library

Unlike the default external components, an imported model or subcircuit can be deleted from the library. Click the component and then **Remove model** below.

2.4.2 External Components

See the list of external components in the Schematic Library panel. Click the component and then the work area to add it (cancel adding with a right-click). After the component is added, the **Copy component(s)** and **Paste component(s)** options become available in the desired component or work area context menu (or use **Ctrl** + **C** / **Ctrl** + **V** shortcuts). The **Copy component(s)**/**Paste component(s)** option works within the same or between different GCSH instances.

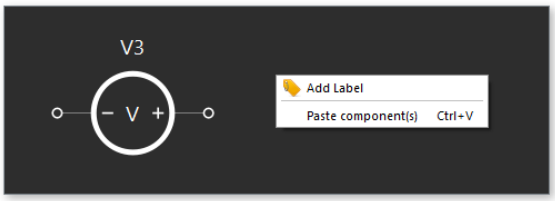


Figure 240. Paste External Components

Double-click a component on the work area to open its **Properties** panel. Click **Configure** on the panel for more advanced settings.

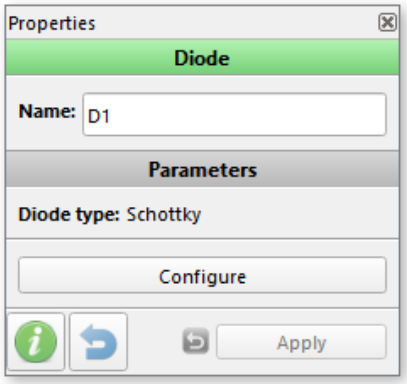


Figure 241. Configuring Basic Parameters

Connect the external components to external I/O macrocell pins or to the pins of other external components using the **Set Wire** tool. When a connection is being set, the available pins become highlighted. By default, voltage source(s) and GND are added to specific pins.

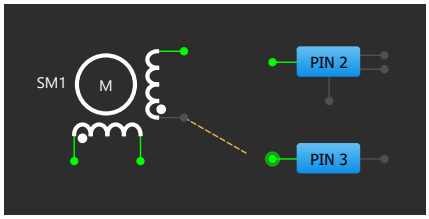


Figure 242. Adding Wire to an External Component

The external connection type can be seen here (all connection types are described in section 2.1.6 Connections).

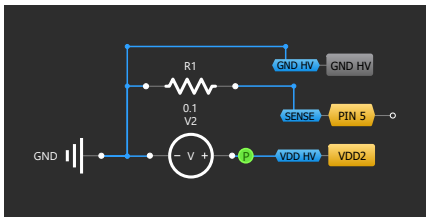


Figure 243. External Connections

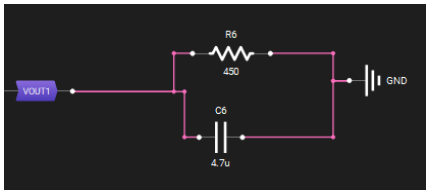


Figure 244. External Connections in SLG5100x

2.4.3 Working with Models and Subcircuits

Third-party SPICE models and subcircuits can be imported into the project. The **Import model** and **Import subcircuit** buttons open the **Import** window, where a SPICE simulation model or subcircuit can be inserted, additional information can be filled in, and the item can be added to the library. The dialog validates the SPICE syntax of a model or subcircuit. Only one model or subcircuit can be imported at a time.

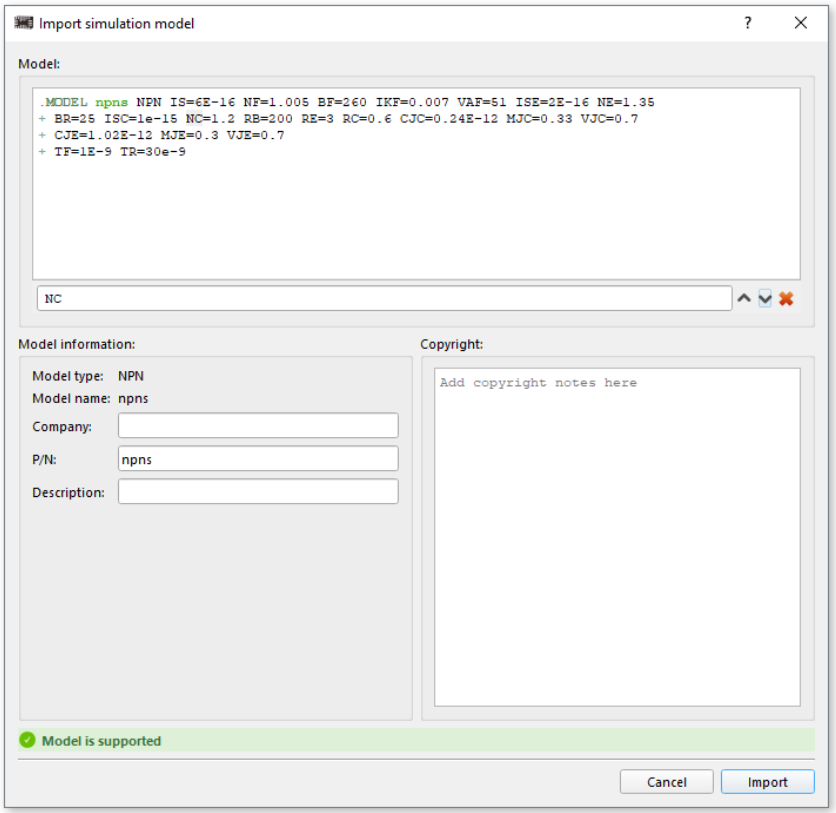


Figure 245. Import Simulation Model Window

Unlike models, multiple subcircuits can be imported and one of them can be selected as the main subcircuit:

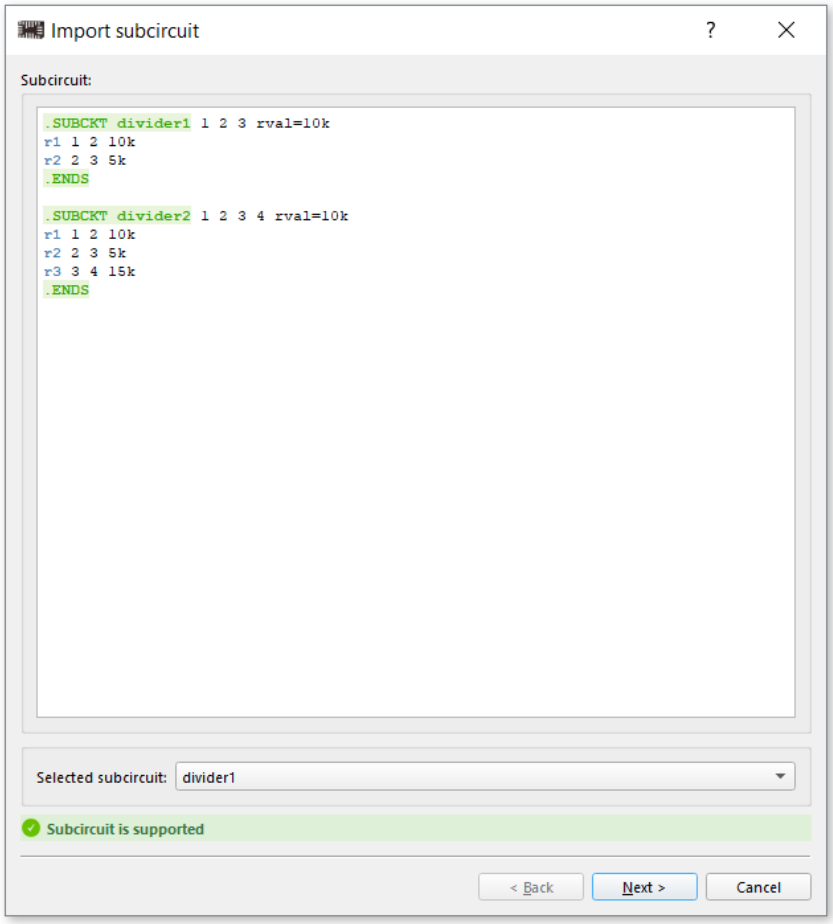


Figure 246. Importing a Model Containing Multiple Subcircuits

Model or subcircuit parameters can be edited in the **Properties** dialog window. Double-click a component in the work area or click **Configure** in the **Properties** panel. The **Properties** window

allows configuration of the external component parameters and editing of the model or subcircuit.

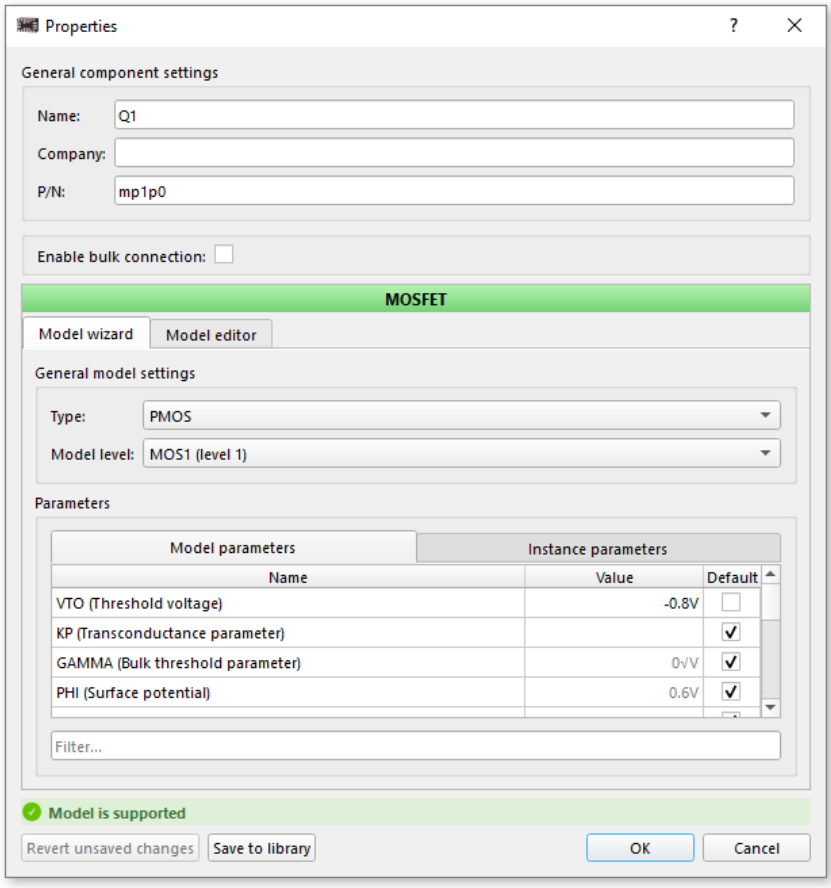


Figure 247. Model Properties Window

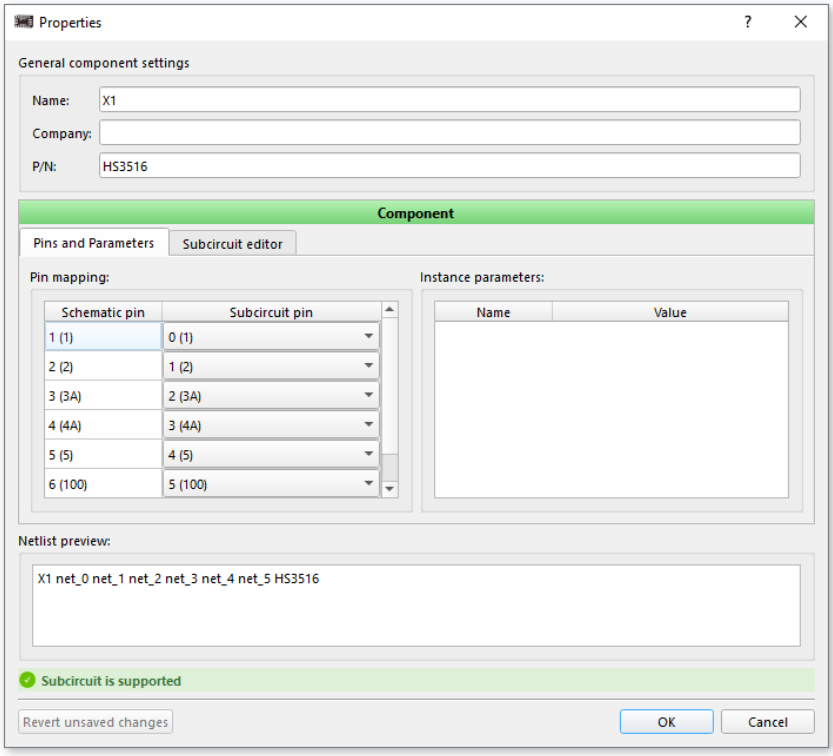


Figure 248. Subcircuit Properties Window

2.4.4 Source Setup

A voltage or current source can be configured in the **Source setup** window. To open the window, double-click a source or click **Configure** on the **Properties** panel.

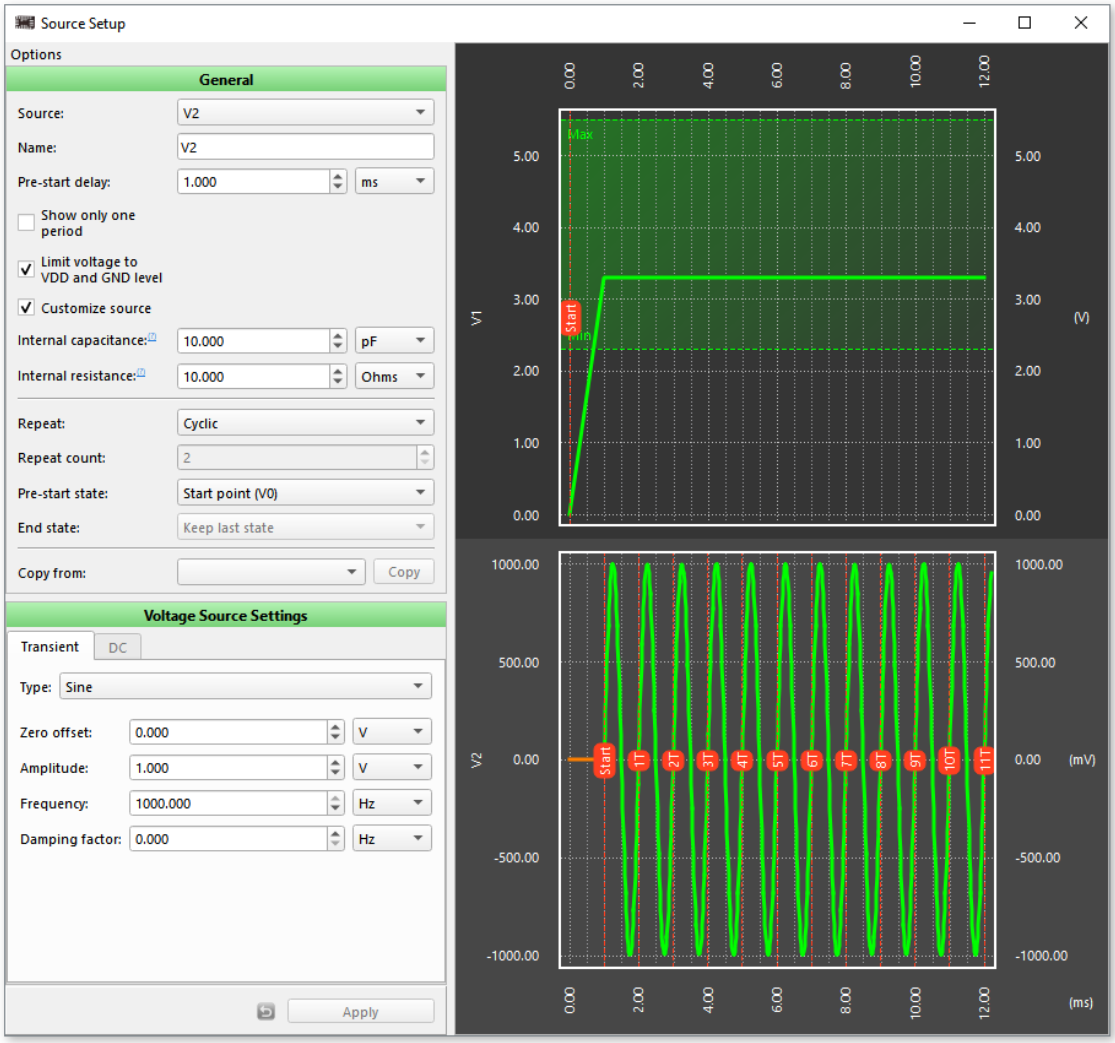


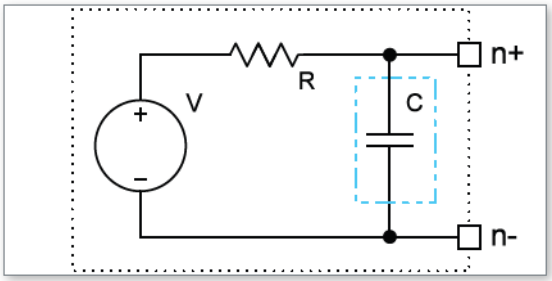
Figure 249. Source Setup Window

The set of settings and the window view differ depending on the simulation analysis type (read more about analysis types in section 2.4.7 Debugging Controls). In the **Options** section, general and waveform-specific parameters related to time intervals and periods can be set.

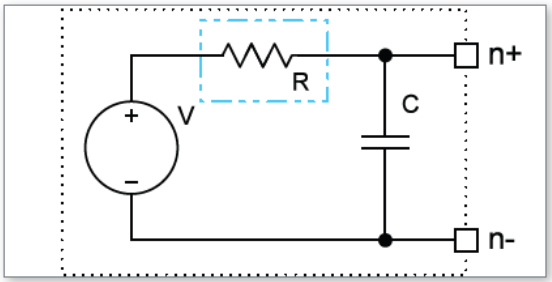
The **Customize source** property can be used to activate two additional parameters:

- **Internal capacitance** – The capacitance between the voltage source terminal and ground.

This value can be used to model a VDD bypass capacitor or IC pin parasitic capacitance.



- **Internal resistance** – The output resistance of the generator. A non-ideal (real) voltage source is modeled with an ideal voltage source and resistance connected in series.



If the configured settings need to be duplicated from one source to another, use the **Copy from** property.

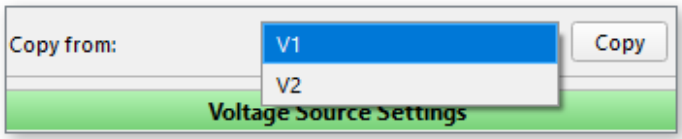


Figure 250. Copy from Property

2.4.4.1 Custom Signal Wizard

In addition to configuring specific waveform types, such as DC, sine, or trapeze, a custom waveform shape can also be set. **Custom Signal Wizard** allows creation, import, and editing of the signal for simulation voltage or current sources. The signal may be constructed by manually adding points or by importing a custom list of points from an external source.

To open the **Custom Signal Wizard**, select **Custom signal** in the **Type** dropdown and click **Set Signal**.

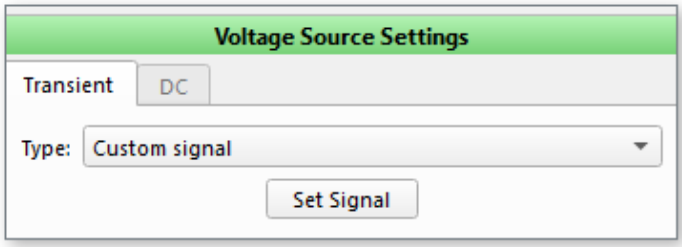


Figure 251. Custom Waveform Option

Examples of possible signals are shown in the pictures below.

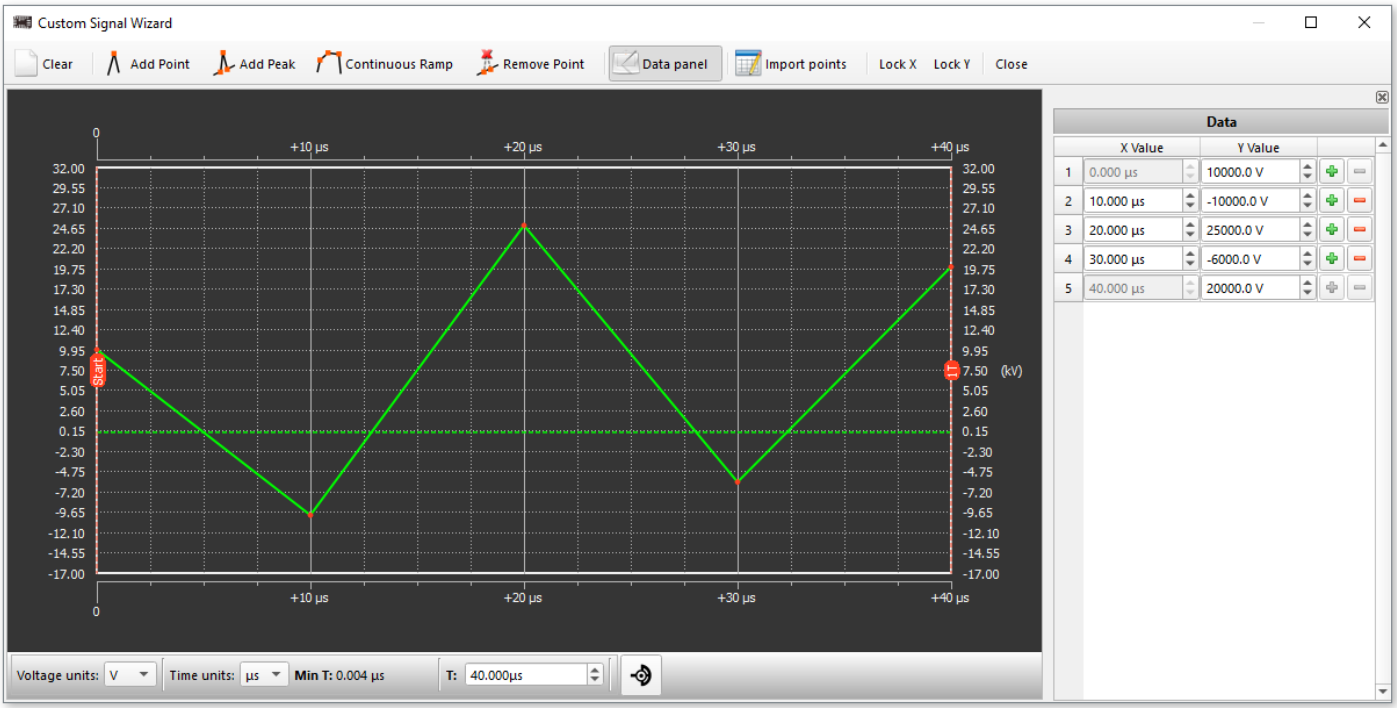


Figure 252. The Standard Editing Process in Simulation Custom Signal Wizard

The **Custom Signal Wizard** allows importing a set of points in different formats. Signals of various complexity, duration, and amplitude are accepted. For signals consisting of large amounts of data,

editing may be limited, and a simplified signal is shown. See the example below.

Figure 253. An Imported Waveform with a Large Number of Points (Above 1000)

2.4.5 Simulation Configurations

Simulation configuration snapshots can be saved and switched between as needed. The snapshot stores only the elements that belong to simulation. An asterisk next to its name indicates that the snapshot has been modified. A saved configuration state can also be imported from another project for the same part number.

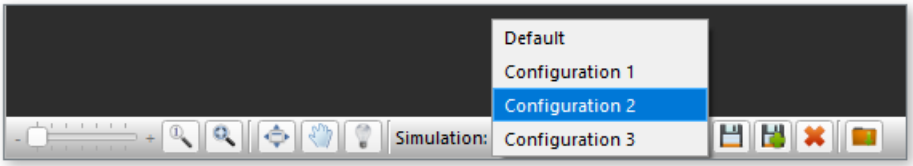


Figure 254. Simulation Configuration

Save the current configuration state.

Save a new configuration state.

Delete the selected configuration.

Import new configuration.

The list of saved configurations.

2.4.6 Probes

Probes provide the data that components yield during simulation. A probe can be attached to a pin to reflect component parameters. Active probes remain on hidden components, and their data is displayed in the simulation results.

Two types of probes are available: voltage probes and parametric probes.

2.4.6.1 Voltage Probes

The voltage probe is used to capture an output signal from a pin. A probe can be added to external component terminals and macrocell output pins. Click the respective buttons on the toolbar and then the pin to add or remove a probe. Probes can be added to all available pins or removed from all pins with a single click using the arrow next to the **Add Probes** or **Remove** button. An individual probe can also be deleted from its context menu.

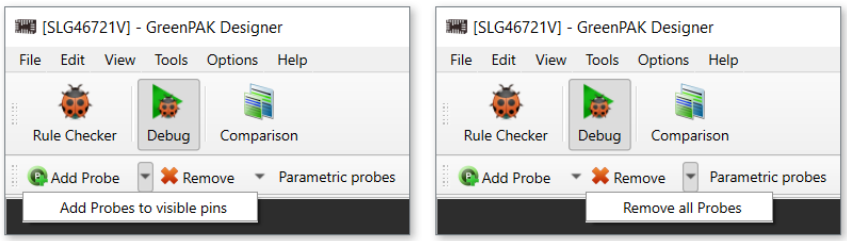


Figure 255. Add and Remove Voltage Probes

2.4.6.2 Parametric Probes

The parametric probe allows monitoring of macrocell parameters in simulation, such as the counted value in CNT/DLY blocks. To show the list of available probes, click the **Parametric probes** button on the toolbar.

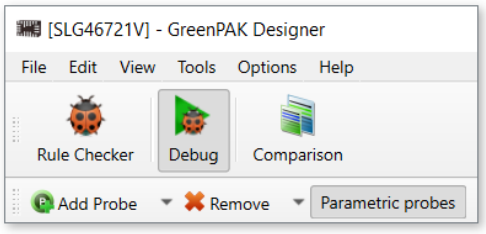


Figure 256. Active Parametric Probes Button

Add a probe by selecting a component and a property in the dropdowns on the respective panel.

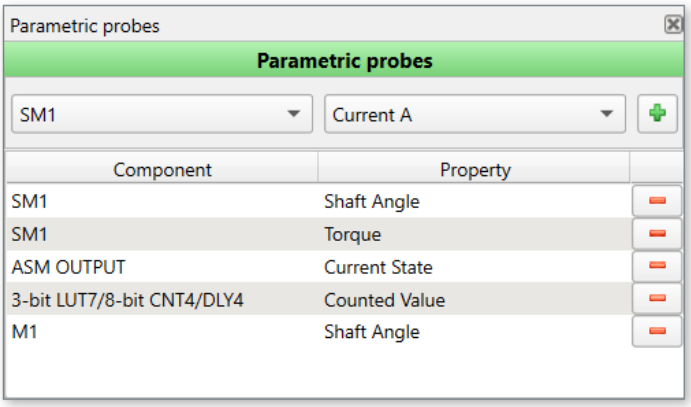


Figure 257. Parametric Probes Window

Right-click the parametric probe sticker to facilitate adding or removing a probe and editing the probe parameters.

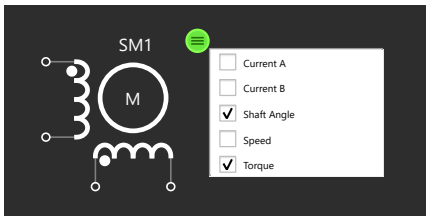


Figure 258. Parametric Probe Sticker

2.4.7 Debugging Controls

The **Debugging Controls** panel appears if **Software Simulation** is selected on the **Development Platform Selector**.

Choose between the analysis types before starting the simulation process. There are two types of simulation analysis: **Transient** and **Parametric DC**.

2.4.7.1 Transient Analysis

Transient Analysis settings define the simulation time span, maximum time step, source voltage, and temperature.

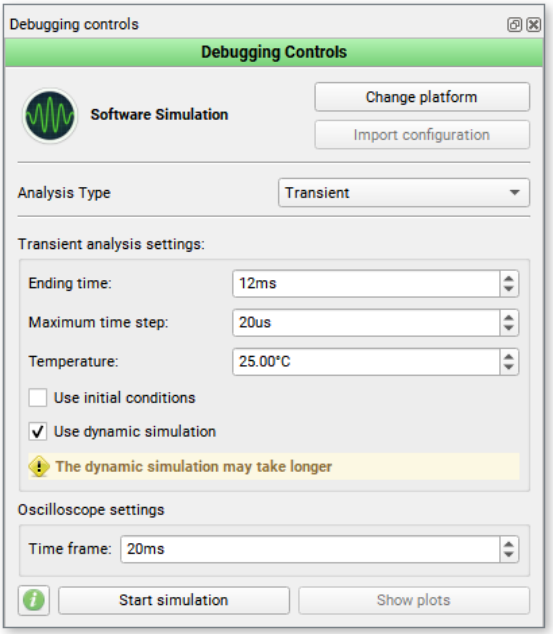


Figure 259. Transient Analysis Debugging Controls

Use initial conditions is an optional checkbox that allows an initial charge different from the default value (0) to be set for components such as a capacitor or inductor. This checkbox also enables all node voltages in a circuit to be set to 0V at the initial time point.

Note: In addition to regular simulation, the **Transient Analysis** type also supports the **Dynamic simulation** feature. It aims to process the data at runtime. Dynamic simulation preserves the same set of tools, as a regular one. Note, that this simulation type may take longer to process the data comparing to regular. Read more in section **Simulation Results window**).

2.4.7.2 Parametric DC Analysis

The properties of **Schematic library** objects can be customized by using **Parametric DC** analysis. The parameter sweep can be configured in the **Parametric DC analysis parametrization** window. Click **Open parametrization settings** in the **Debugging Controls** to activate it.

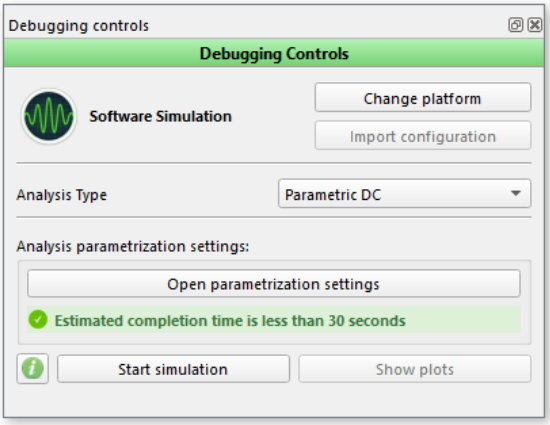


Figure 260. Parametric DC Analysis Debugging Controls

Analysis parametrization can set active parameters for external components, circuit temperature, and macrocell properties, such as **Resistance (initial data)** for **Digital Rheostats**.

Parametric DC data has two range types:

- **From/To** – Simulate data between the set **From** and **To**, incrementing by **Step**.
- **List** – Use data in a semicolon-separated list.

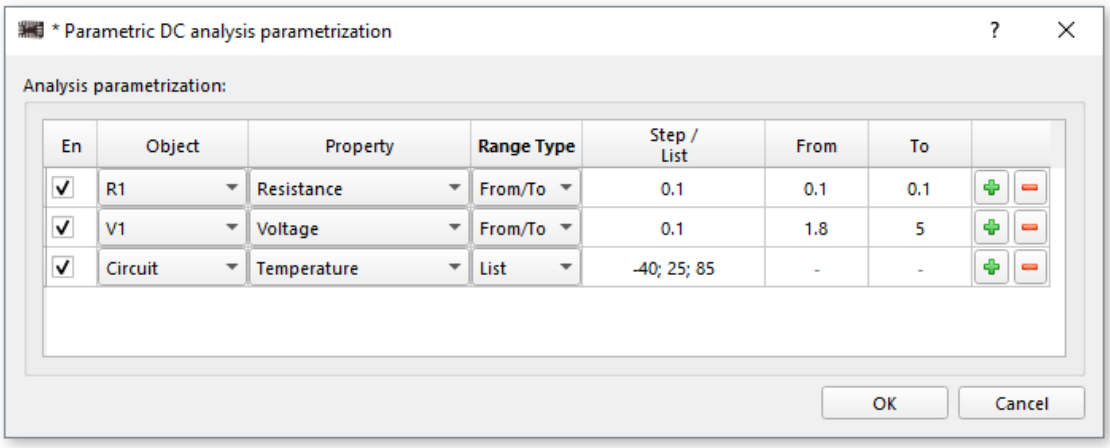


Figure 261. Parametric DC Analysis Window

Estimated Completion Time

It is possible to define simulation runs that may exceed the available resources of the computer, which can potentially make the system unstable. Longer runtimes require greater computer resources, including CPU and memory.

The software estimates runtime based on sample points in three broad categories: **green**, **yellow**, and **red**. The estimated runtime may vary depending on different factors, including computer performance.

2.4.8 Chip Current Consumption

The **Show chip current consumption** feature estimates the non-external current consumption of the selected chip during simulation. This estimation excludes current drawn by external components. The feature is available on the **Debugging Controls** in the **Transient analysis** mode. Here are instruments for current consumption analysis:

- The feature displays the calculated **Max**, **Min**, and **Average** current values based on the simulation data.

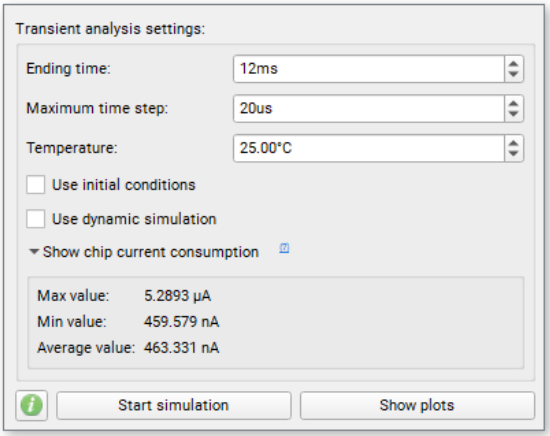


Figure 262. Current Consumption Estimation

- Placing a **Parametric probe** on the chip's VDD pin generates a current consumption waveform in the simulation plots.

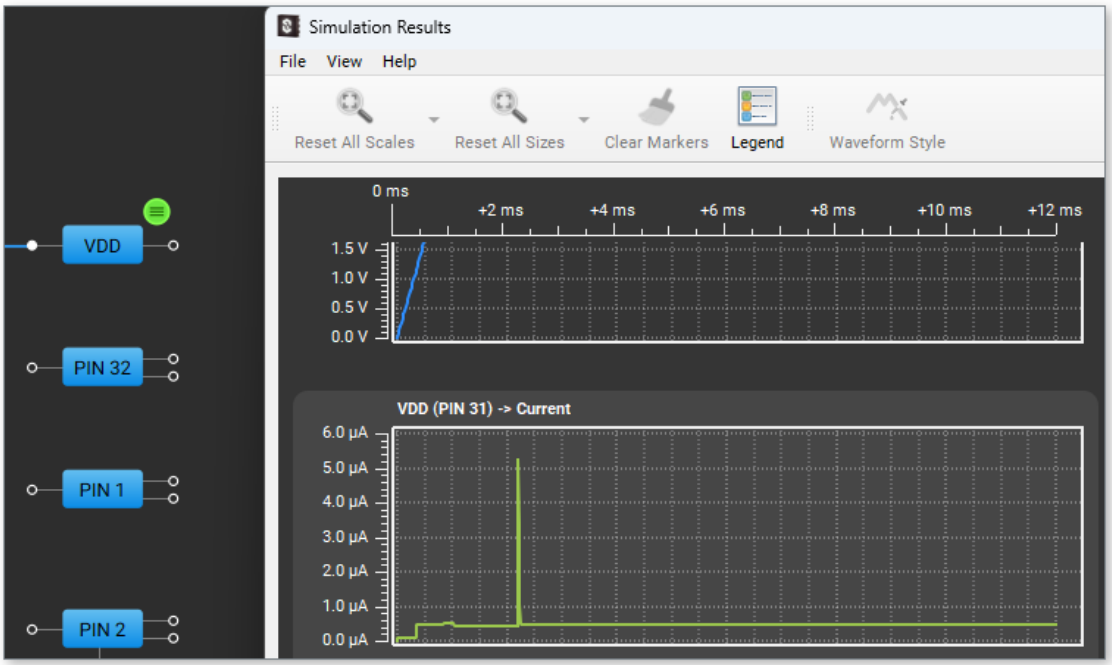


Figure 263. Probe Instrument for Current Consumption Analysis

The current consumption data is shown as a step waveform. See the example below:

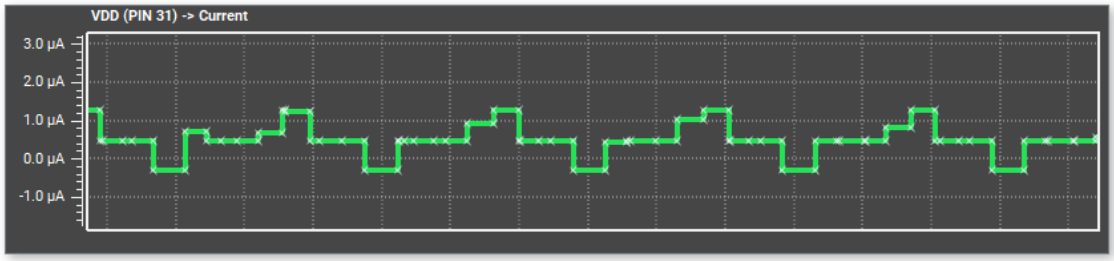


Figure 264. Step Waveform

Note: The measurement is taken only after the Power-On Reset (POR) or Logic 1 macrocell (depending on the selected Part Number) reaches a high logic level.

2.4.9 Simulation Progress

The **Simulation in progress** window appears while data is being processed. The remaining simulation time is estimated dynamically.

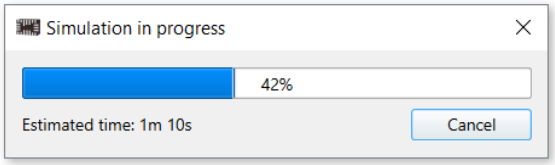


Figure 265. Simulation in Progress

The process can be interrupted by clicking **Cancel**. The simulation results are still displayed based on the data that the tool managed to process before cancellation.

2.4.9.1 Scheme Issue Resolution

If a simulation attempt encounters a known scheme convergence error, the internal simulation algorithm attempts to adjust the circuit by changing component properties or scheme parameters and reinitialize the simulation. In this case, the **Simulation in progress** window indicates the current attempt count.

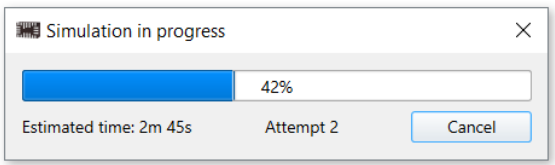


Figure 266. Simulation in Progress (2nd Attempt)

The simulation error is shown only when the algorithm cannot resolve the problem.

If the simulation process fails, the corresponding error window is shown. It may contain a brief description and details from the simulation engine about the cause.

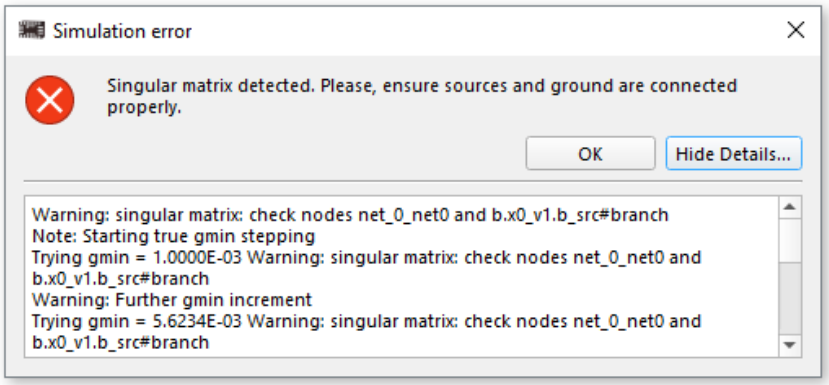


Figure 267. Simulation Error Window

After the issues are resolved and the simulation process is completed, the simulation report can be viewed by clicking the **Show report** button in the **Simulation Results** window.

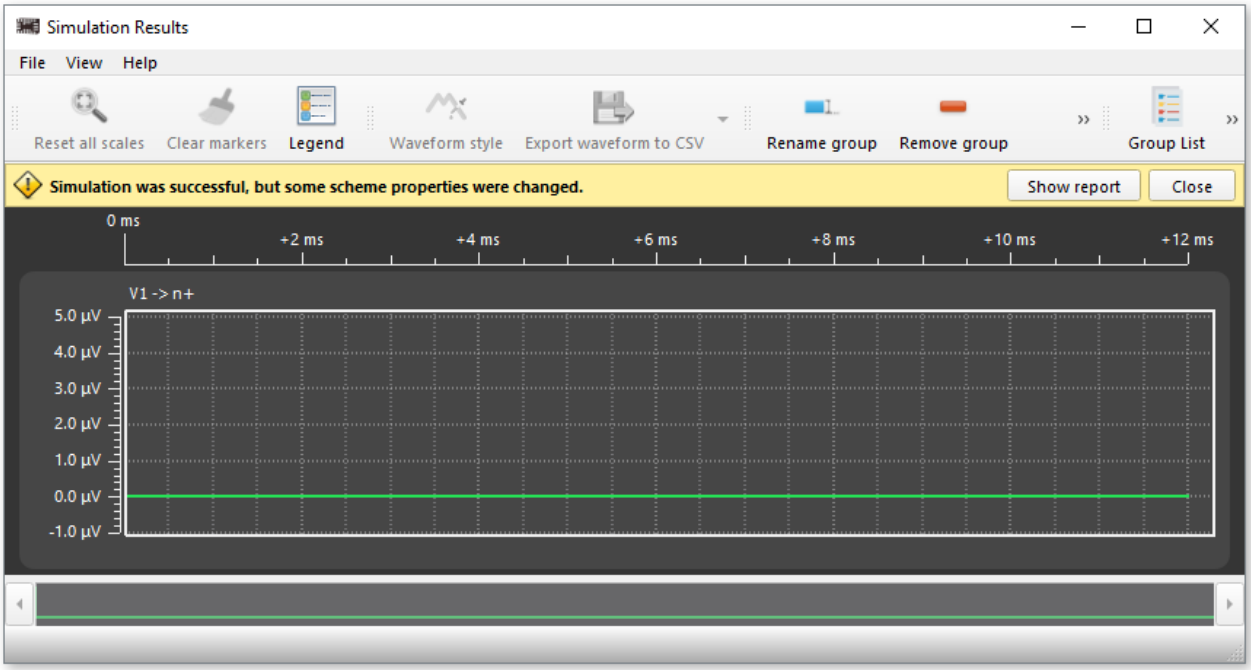


Figure 268. Show Report Button

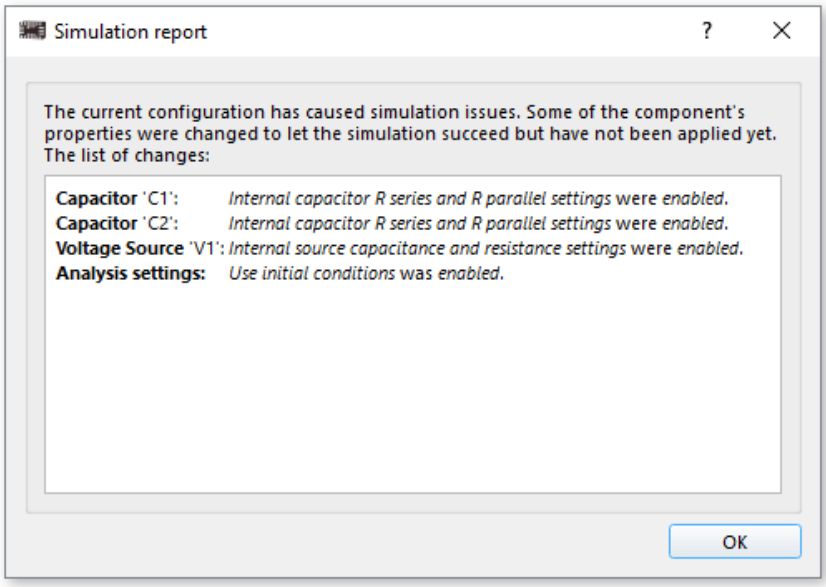


Figure 269. Simulation Report Window

2.4.10 Simulation Results Window

Depending on the selected simulation type (regular or dynamic) in the **Debugging Controls**, the **Simulation Results** window behaves differently.

When regular simulation is enabled, **Simulation Results** opens only after the simulation process is

completed or interrupted.

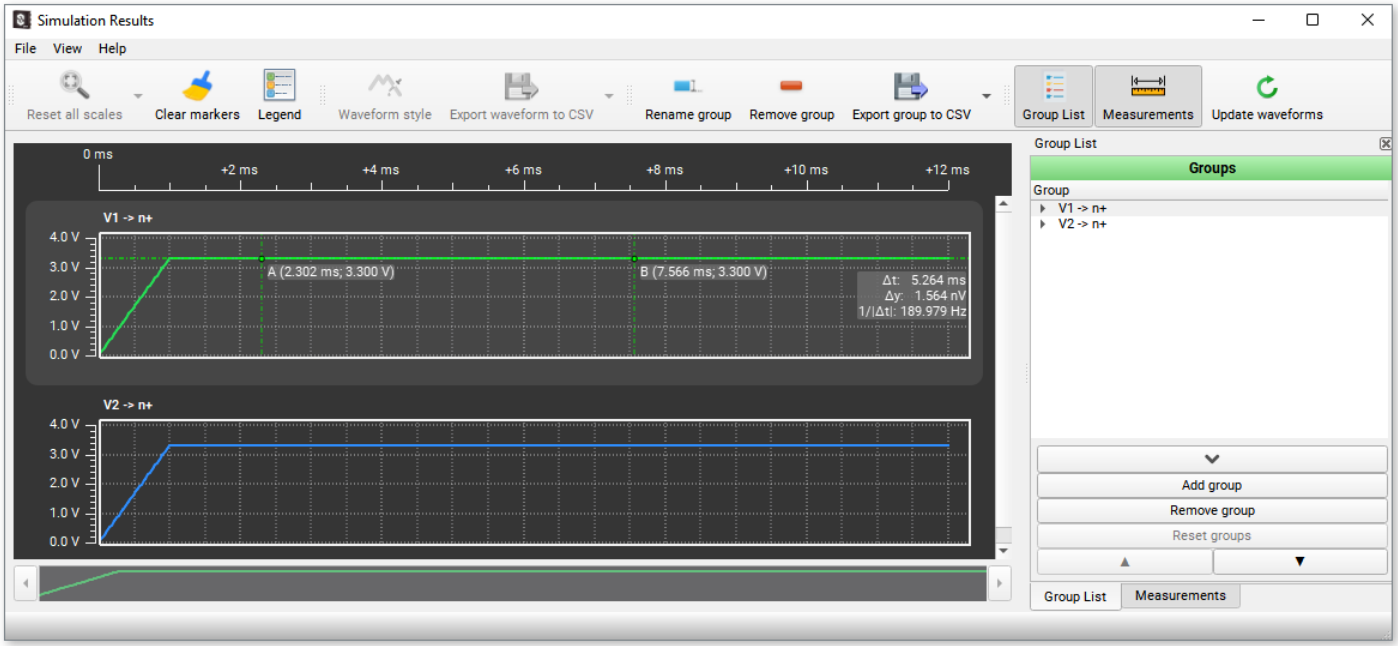


Figure 270. Simulation Results (Regular Simulation)

During dynamic simulation, the **Simulation Results** window can be in the following states:

- **Oscilloscope** – Displays the results at runtime. Interaction with waveforms on this tab is disabled. The progress bar with the estimated completion time is also shown here, along with the **Stop simulation** button for terminating the process. After the simulation run is completed, the **Analyzer** tab appears instead.

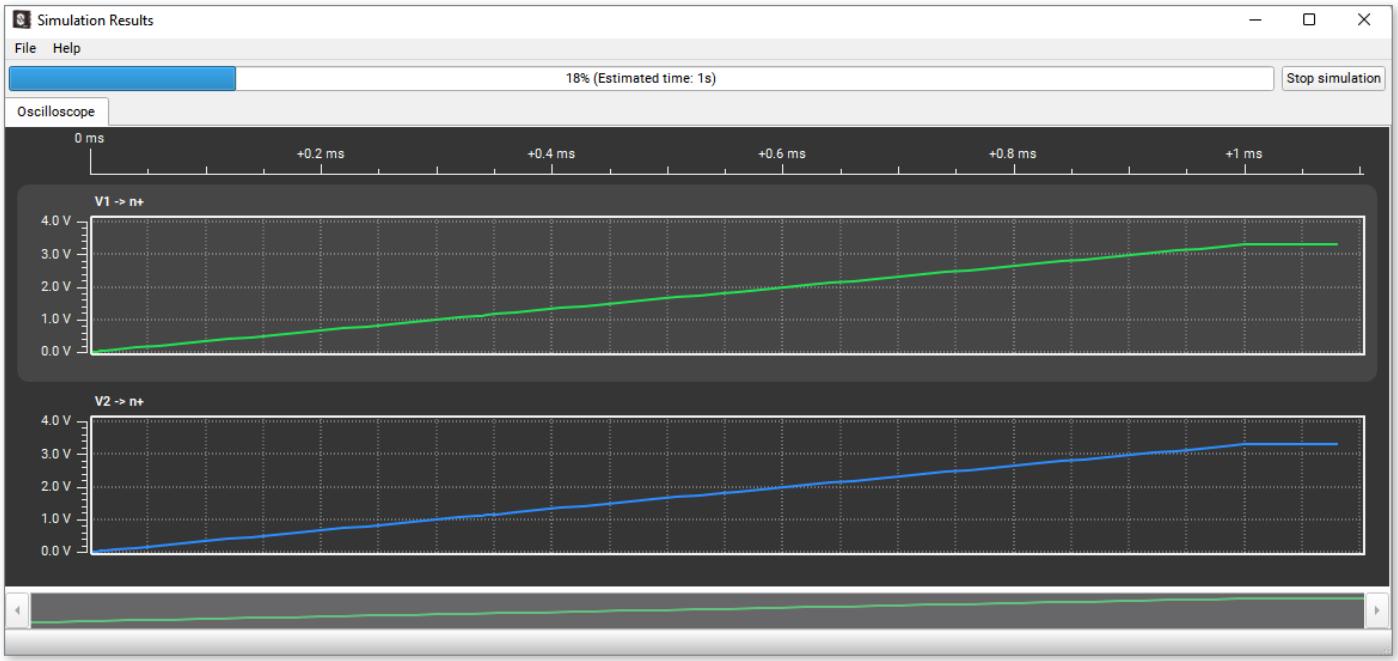


Figure 271. Simulation Results Oscilloscope Tab (Dynamic Simulation)

Note: For more efficient **Oscilloscope** tab usage, change the visible time range by adjusting (reducing) the **Time frame** value in the **Debugging Controls**. This allows more accurate changes to be seen in highly detailed signals.

- **Analyzer** – Shows the completed or interrupted simulation results, which are then available for interaction.

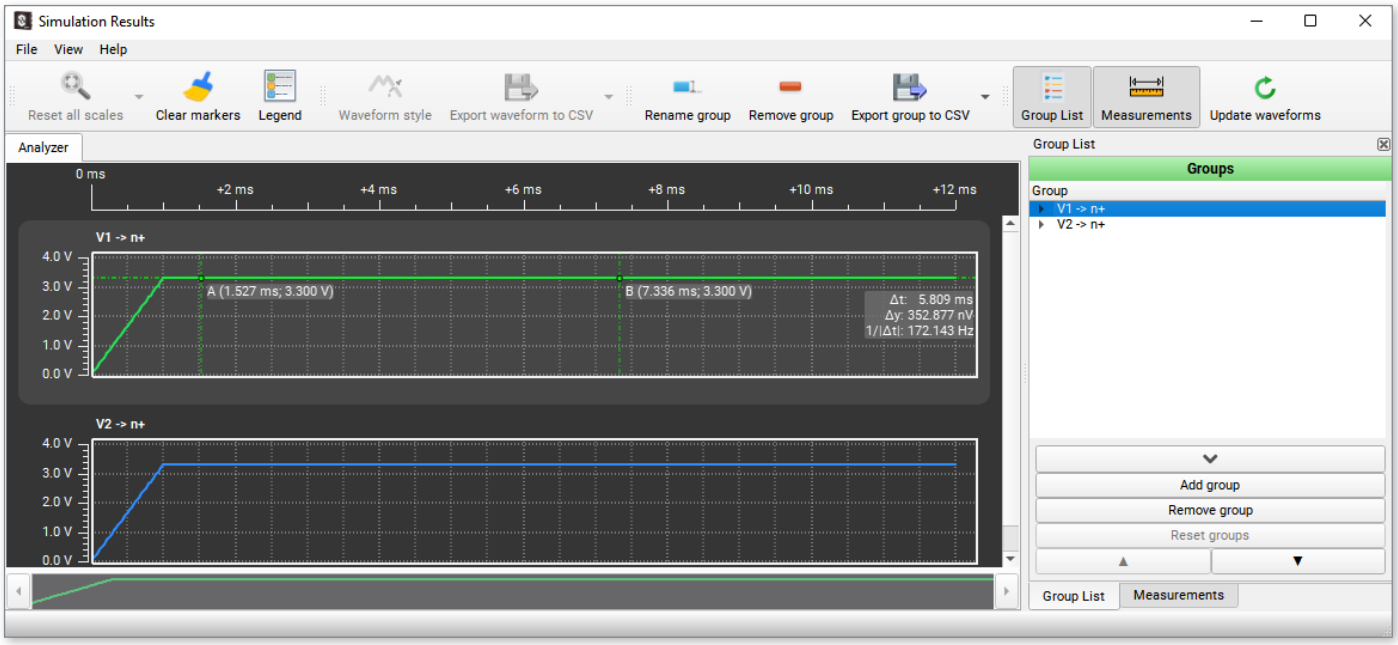


Figure 272. Simulation Results Analyzer Tab (Dynamic Simulation)

If the window is closed, the latest simulation results can be reopened by clicking **Show plots** on the **Debugging Controls** panel.

Regardless of the simulation type, simulation results can be used with the tools and features described below.

2.4.10.1 Toolbar

Toolbar provides operations for the plot area, waveform style, group management, and results data export. A single waveform or an entire group can be exported to a .csv, .vcd, or .png file. All waveforms can also be exported as a single **.png** file.

2.4.10.2 Plot widget

The **Plot** widget is the main area displaying a waveform or group of waveforms. The plot controls can be manipulated in several ways. In addition to the toolbar controls, right-click the plot to open the context menu and access the available actions. Also refer to the keyboard commands, which can be used instead.

Markers can be set with **Ctrl** + right and left click to view signal information at a particular point. Examples of markers and ΔT and ΔV parameters on the **Plot** widget are shown below:

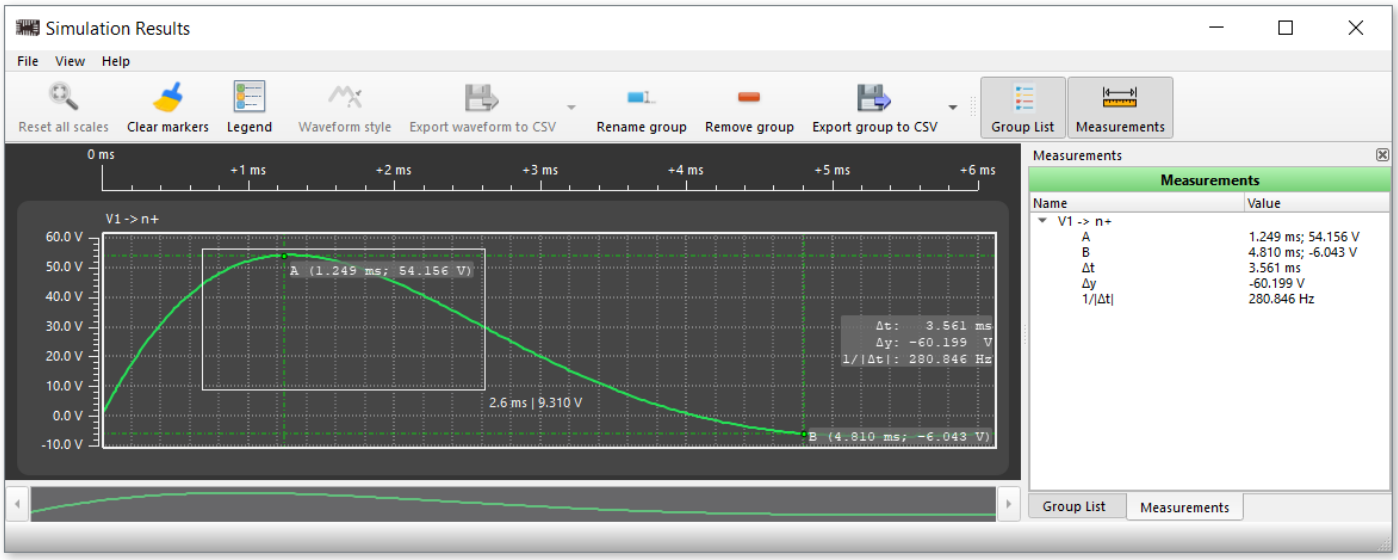


Figure 273. Plot Widget and Measurements

2.4.10.3 Group List

The **Group List** panel contains the list of all captured signals and source generators arranged in groups.

Click the plot on the panel to modify its style, color, or width by choosing the **Waveform style** option from the toolbar or context menu. The context menu can also be used to rename the group.

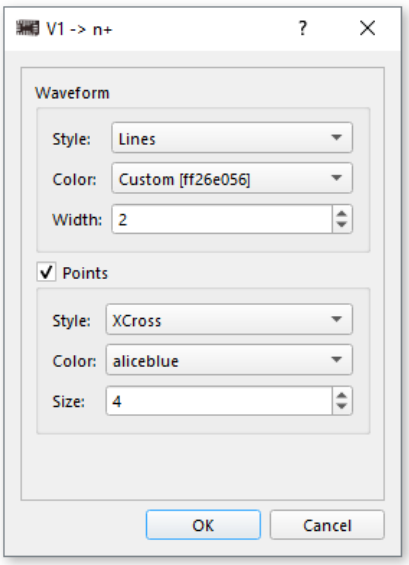


Figure 274. Waveform Plot Configuration Window

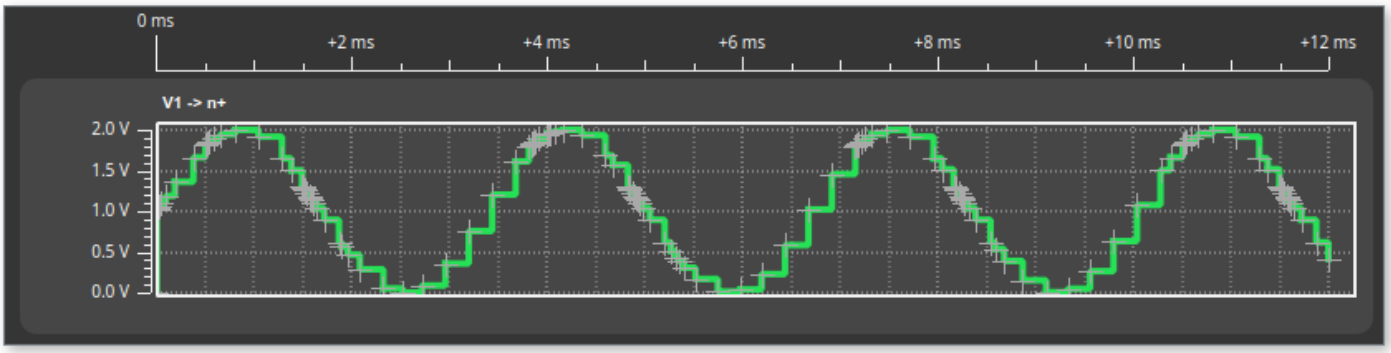


Figure 275. A Sine Simulated with Huge Interval Settings

The **Group List** bottom controls provide waveform management options.

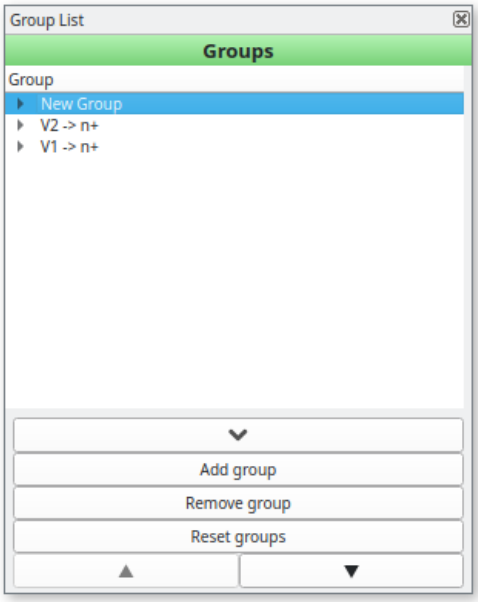


Figure 276. Group List

Group controls include:

- ⌄ / ⌃ – fold or unfold the controls.
- **Add group** – Create a group by selecting one or more signals from the list. The signal(s) can be combined in a group of **analog [A]**, **digital [D]**, or **parametric probe [B]** waveforms.

Multiple groups can be created using the same waveform.

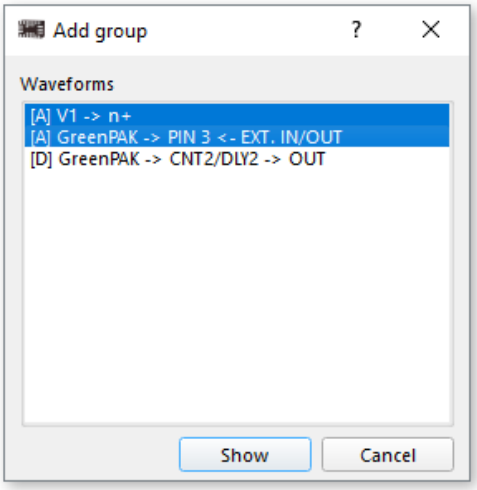


Figure 277. Add Group Menu

- **Remove group** – Delete the selected group.
- ▼ and ▲ – Move a selected group up or down.
- **Reset groups** – Restore the default configuration and remove all added groups.

2.4.10.4 Measurements

This panel displays measurement data for the added plot markers.

Figure 278. Measurements Panel

2.5 FPGA Editor Tool

The **FPGA Editor** is a complex software tool designed to create and configure FPGA logic. The editor allows designs for Field-Programmable Gate Arrays (FPGAs) to be created using Verilog, a Hardware Description Language (HDL). Before a design is programmed onto the FPGA, the editor provides an option to simulate it first and verify that it is correct.

The **FPGA Editor** also includes a macrocell constructor as an alternative to writing Verilog code. The required blocks can be placed manually on the work area, connected as needed, and the completed design can then be converted into its code representation.

In addition, **FPGA Editor** allows work with external designs by importing files containing created logic mapped to components. Descriptions of these and additional features are provided in the sections below.

2.5.1 Flowchart

We have created a flowchart to narrow down and follow the basic steps from start to finish. The flowchart shows all the important aspects of using the **ForgeFPGA** software.

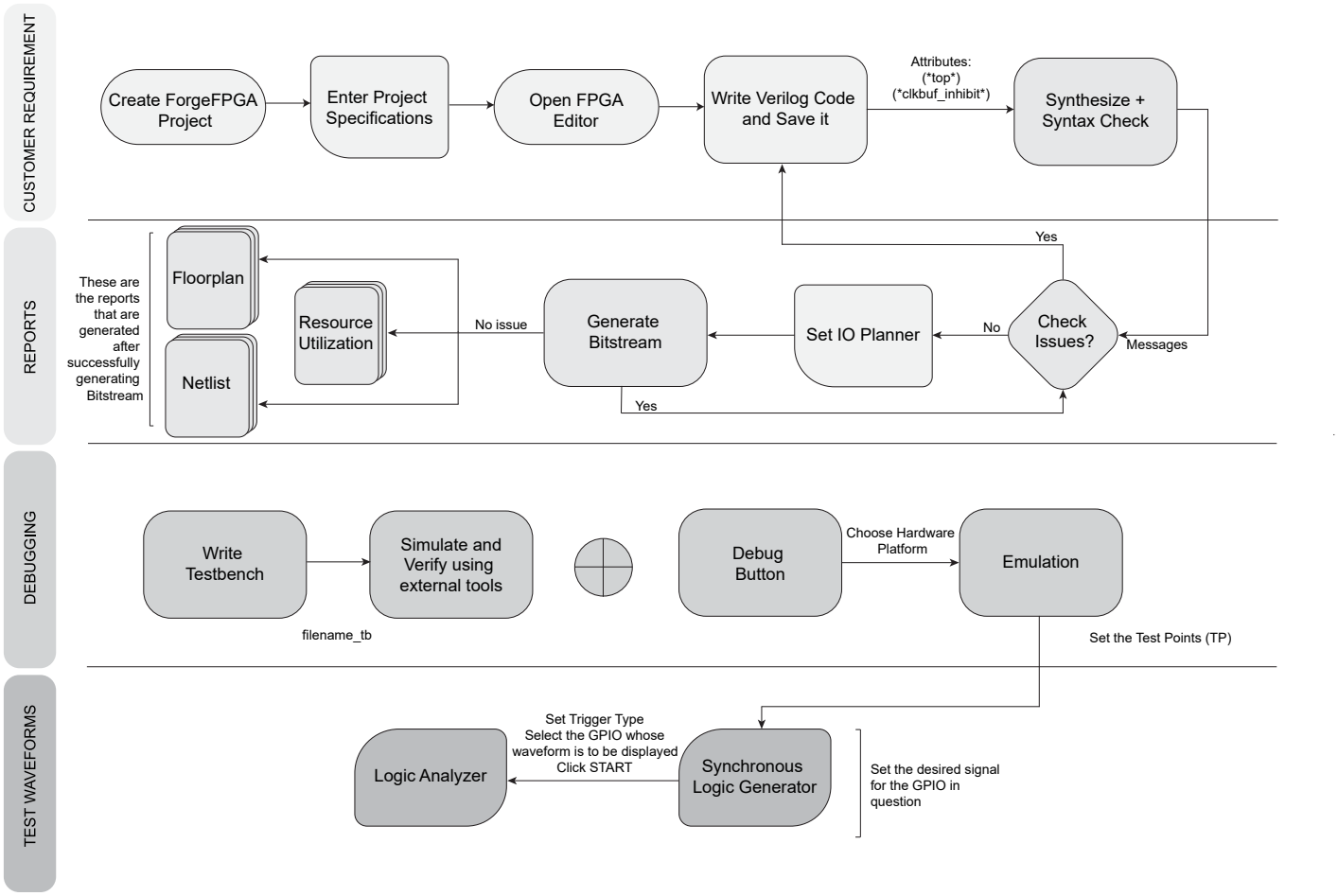


Figure 279. Software flow

2.5.2 Interface Overview

To open **FPGA Editor**, click the respective button on the toolbar or double-click the FPGA Core component on the work area.

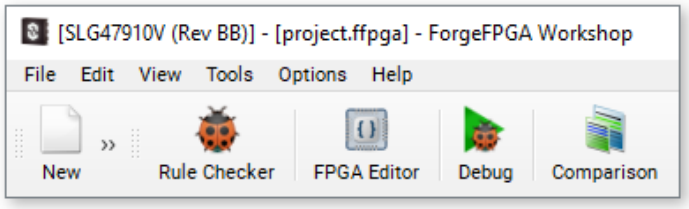


Figure 280. FPGA Editor Button on the Toolbar

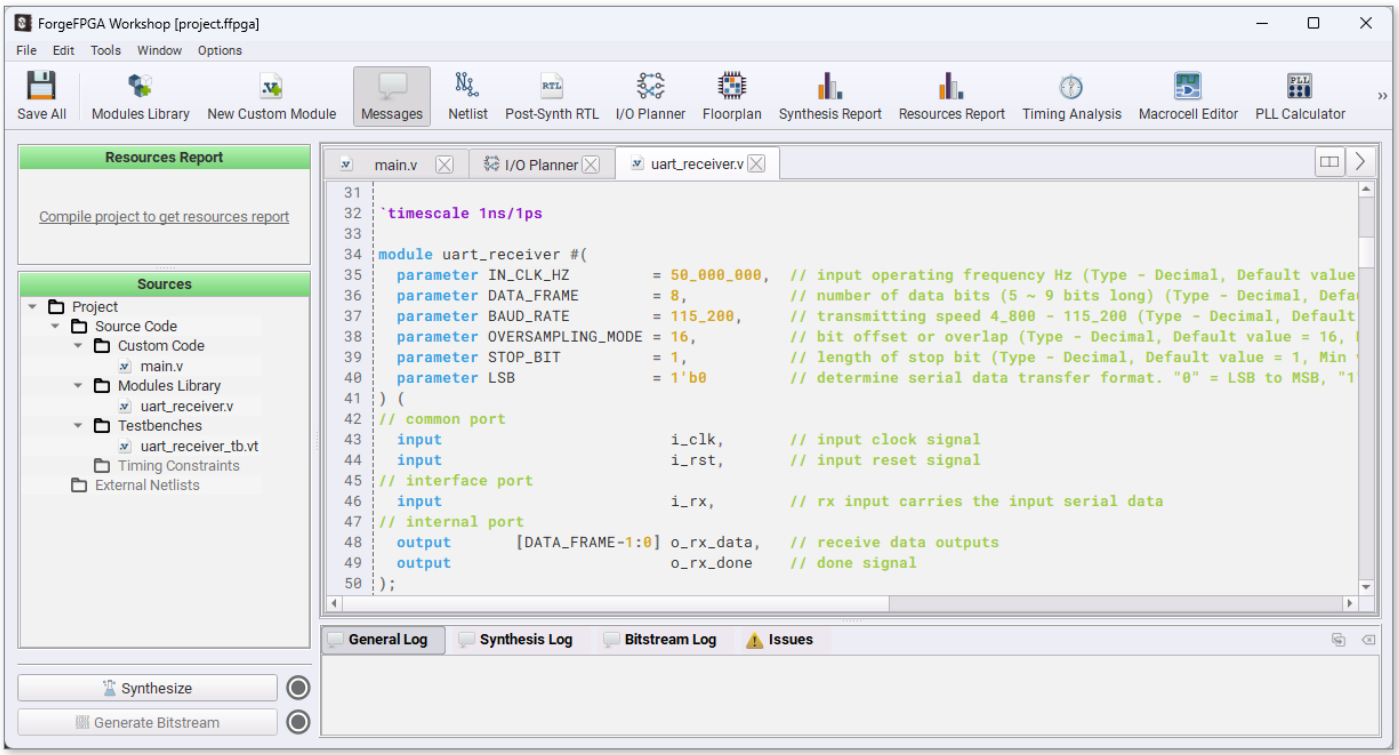


Figure 281. FPGA Editor

2.5.2.1 Toolbar

The top toolbar provides quick access to a set of tools and actions. Descriptions of all available tools are provided in the following sections.

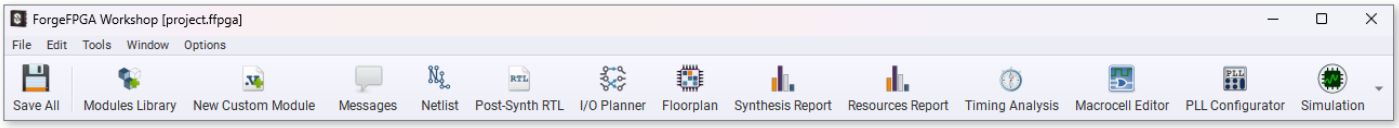


Figure 282. Toolbar

2.5.2.2 Control Panel

The left control panel provides access to:

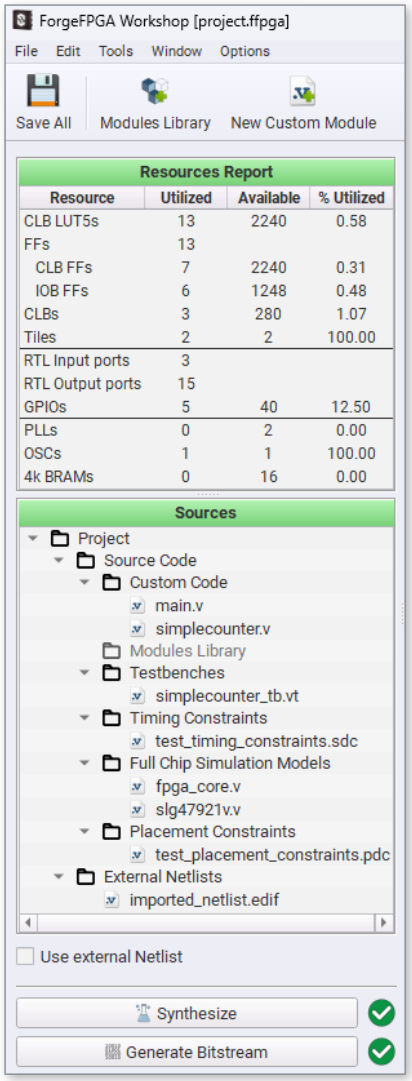


Figure 283. Control Panel

- **Resources Report** – An extract from the resource usage report. It shows the portion of available resources utilized by the design sources. Visually, it is divided into three sections. The upper part reflects the internal logic used in the project against the total available. The middle section shows connection-related information (RTL ports, GPIOs). The last part summarizes the utilization of internal components, such as OSCs, PLLs, and BRAMs.

- **Sources** – A list of all Verilog sources and other supported files that the project may contain. A module can be added using the toolbar by clicking **New Custom Module** or by selecting one from **Modules Library**. All supported sources can also be added from the main menu > **File**. In addition, external sources can be imported through **File** > **Import** (for more information about project structure, see here).

  The following subcategories are available here:

  - **Custom Code**.

- **IP Blocks**.
- **Testbenches**.
- **Full Chip Simulation models**.
- **Timing Constraints**.
- **Placement Constraints**.
- **External Netlists**.

The **Sources** list can be customized from the context menu with options to add, delete, or rename sources.

- **Synthesize and Generate Bitstream** – Main controls for running the toolchain on the design to produce chip configuration. Hover over the icon next to the procedure button to see status details, such as whether the procedure was successful, failed, or whether the Verilog file has been changed.

Note: Make sure all modified modules are saved before running procedures so that the most up-to-date data is processed. Configure the saving options in Settings, **Processing** tab.

2.5.2.3 Messages Panel

The generated messages that appear after running **Synthesize**, **Generate Bitstream**, or the **Simulation** procedure are shown here.

- **General Log** – Shows the information messages along with warnings and errors that were recorded while processing the design.
- **Synthesis Log** and **Bitstream Log** – Provide the output generated while the procedure is performed.
- **Issues** – Displays the warning and error messages that are automatically generated by the software when required. After a syntax error is reported, right-click it and select **Show in Editor** to highlight the line in the Verilog code where the mistake was detected.

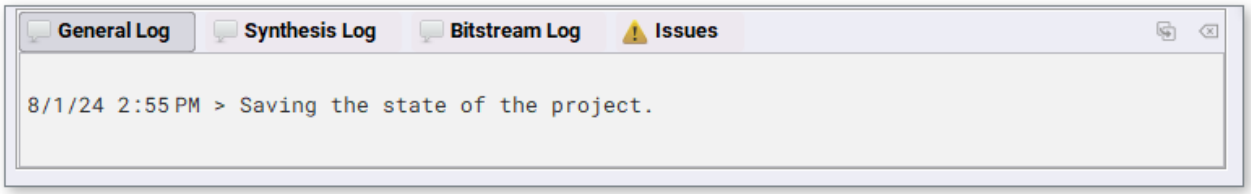


Figure 284. Messages Panel

2.5.2.4 Additional Controls

The following **FPGA Editor** controls can improve the overall user experience (availability may depend on the selected tool):

- **Split** – Divide the work area side-by-side to work with more than one tool simultaneously. Focus on the split area and click the tool to add it to the desired side. The same tool can be duplicated on both sides; this is supported for most tools, such as **I/O Planner**, **Netlist**, and **Post-Synth RTL**.

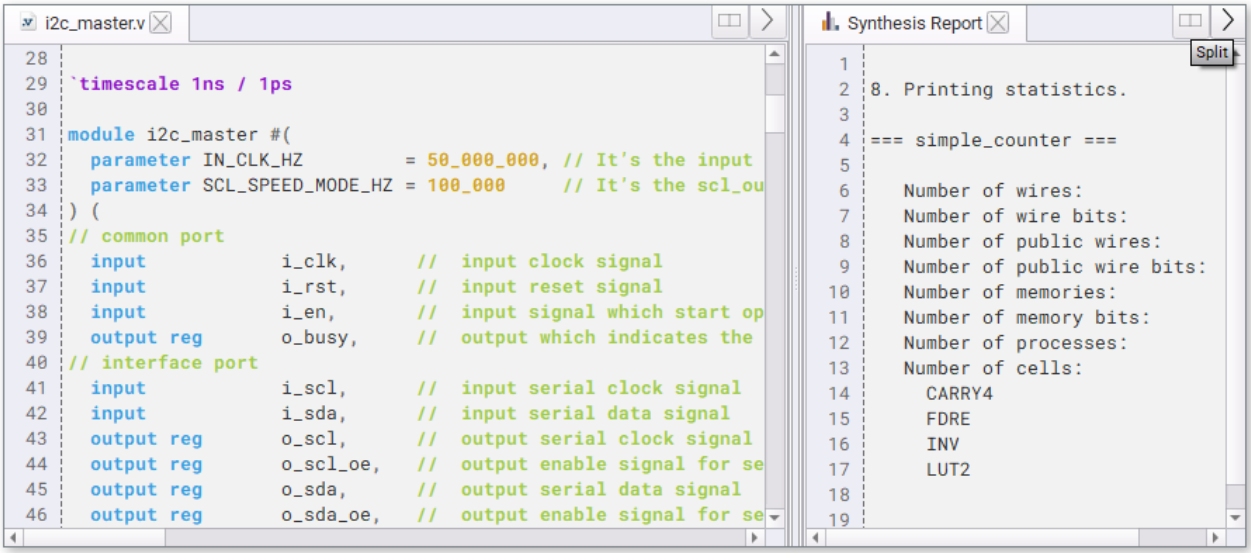


Figure 285. Split

- **Find** – Activate the search field in the main menu, **Edit** > **Find** (**Ctrl** + **F**). The feature can be enabled for most toolbar tools, including the **Messages** panel.

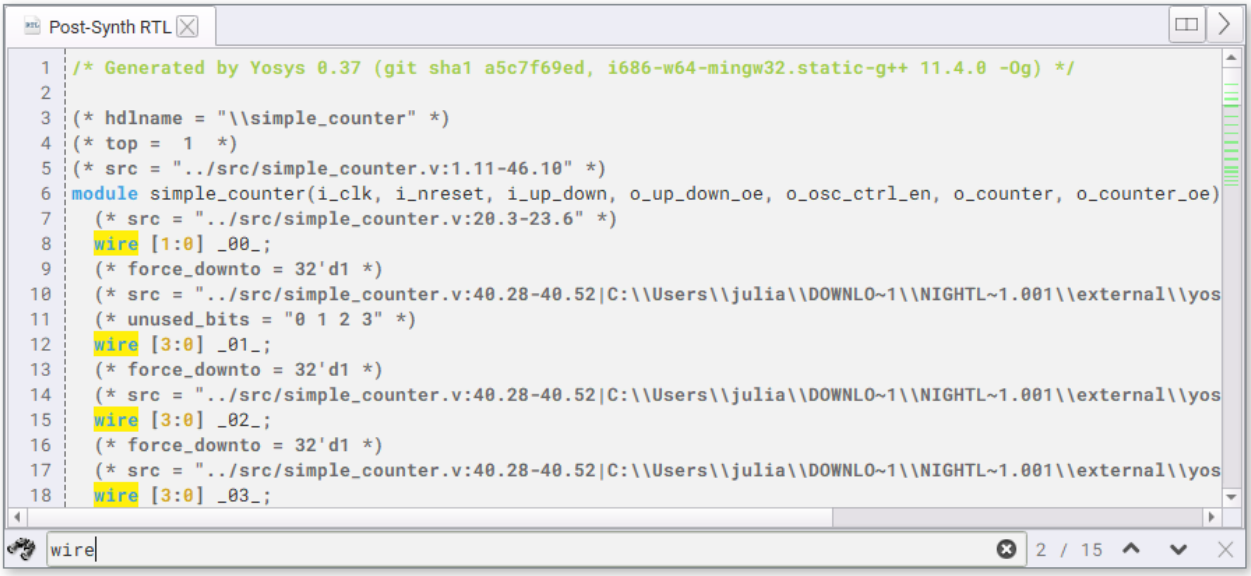


Figure 286. Find

- **Footer controls:**.

- **Zoom** – Zoom the work area in or out.
- **Pan mode** – Click, hold and drag the cursor to move the work area (use the middle mouse button as an alternative).
- **Zoom to selection** – Click, hold and drag a cross cursor over the element to zoom in on.
- **Align Vertical** and **Align Horizontal** – Align the macrocells relative to each other on the work area.
- **Snap to grid** – Use for precise graphic alignment of the macrocells.

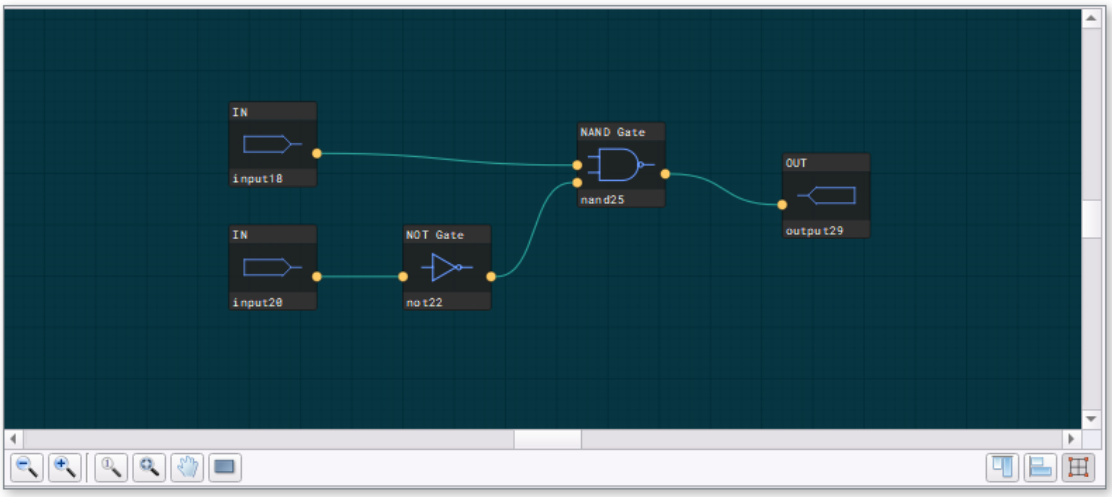


Figure 287. Footer Controls

2.5.3 Design Templates

Design Templates offer pre-built designs that can be integrated seamlessly into a project. They can be found in **FPGA Editor** > **File** > **Load Design Templates**.

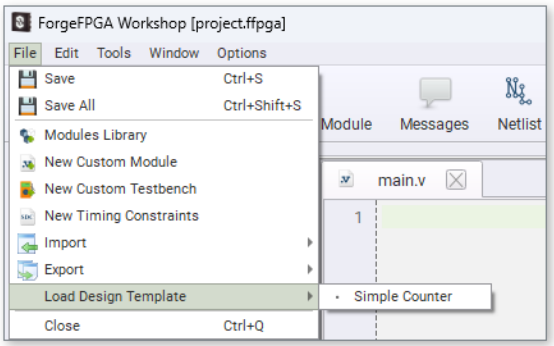


Figure 288. FPGA Design Templates

When a design template is selected, the **Design Template Wizard** appears. It guides the process through component selection, IO spec diff review, and module name assignment.

- **Component Selection** allows selection of the components to proceed with:
  - **Verilog Module**.
  - **Verilog Testbench**.
  - **IO Spec**.

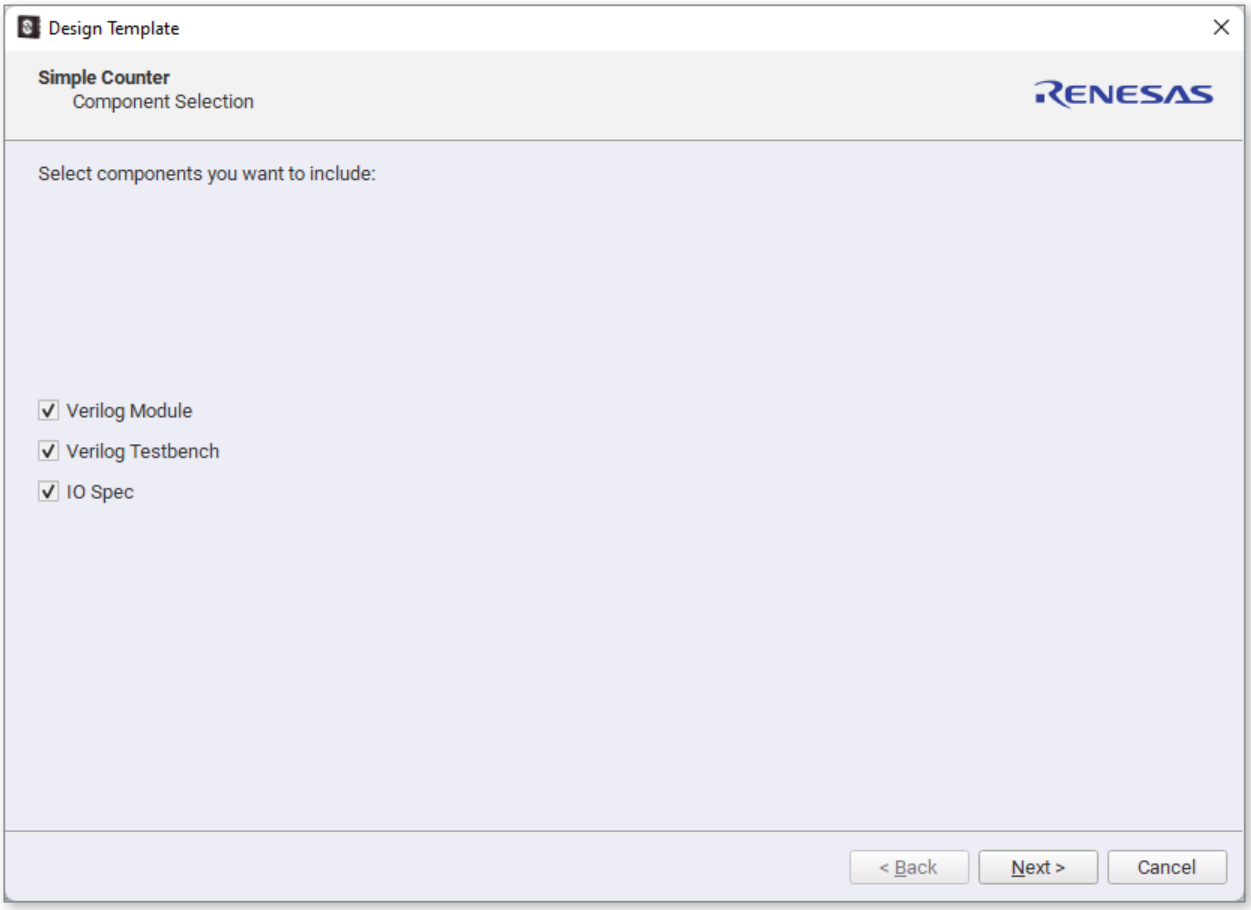


Figure 289. Component Selection

All components can be selected simultaneously or individually, depending on the desired outcome for the project. The choices made at this stage determine the next steps in the process.

- **IO Spec** generates an **IO Spec Diff**, presenting a comparison between the existing design port records and those associated with the selected template. This serves as a cautionary step, as the current port configurations could potentially be overwritten by those of the chosen template. The software highlights conflicting entries in yellow. It is possible to focus solely on conflicts by selecting the **Show Conflicts Only** option. However, if the **IO Specs** are not reviewed, the software streamlines the process by skipping the **IO Spec Diff**. In this scenario, the tool proceeds directly to **Module name**.

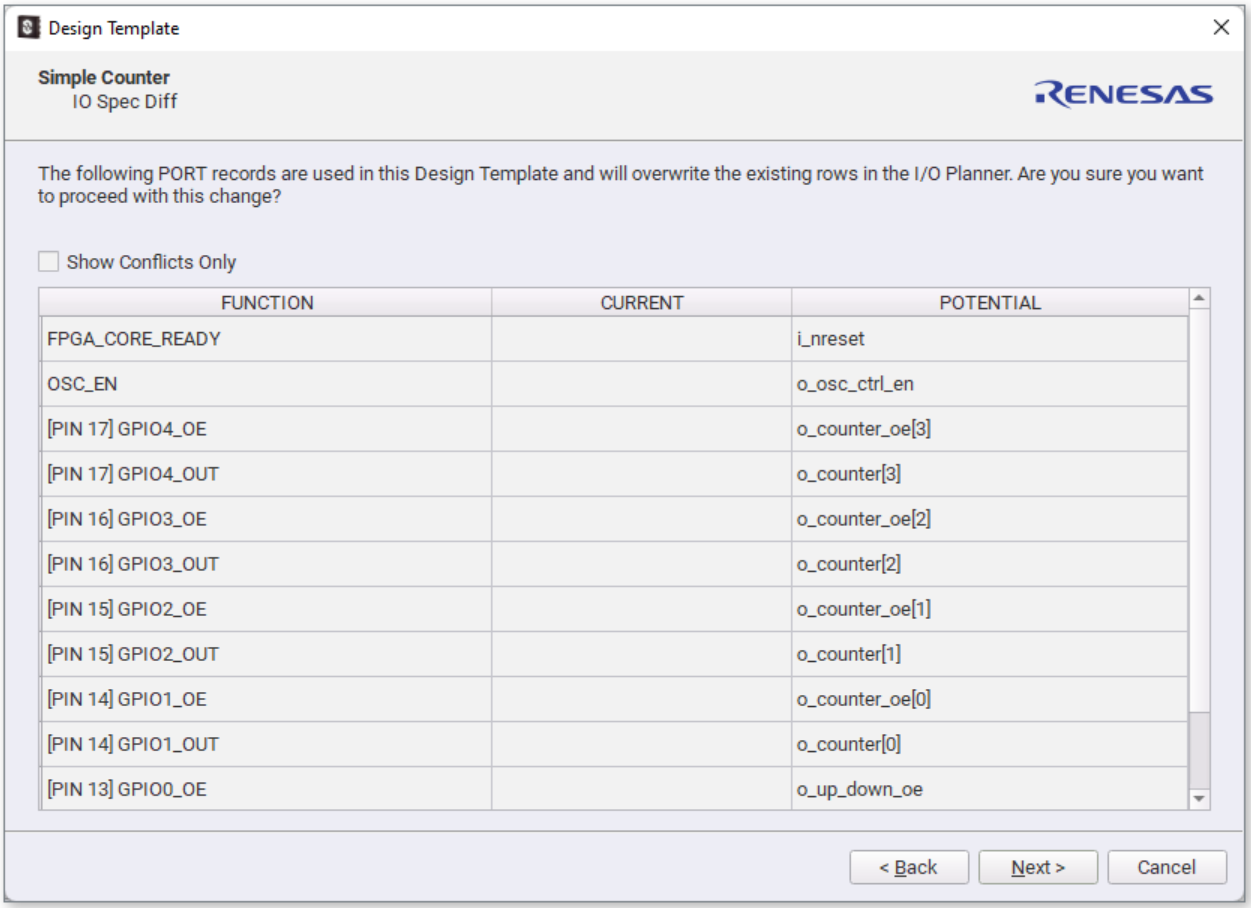


Figure 290. IO Spec Diff

- **Module name** is a component that assists in assigning names to the **Verilog Module** and **Testbench**. If the **Verilog Module** and **Verilog Testbench** options are skipped in the **Component Selection** window, default names are assigned.

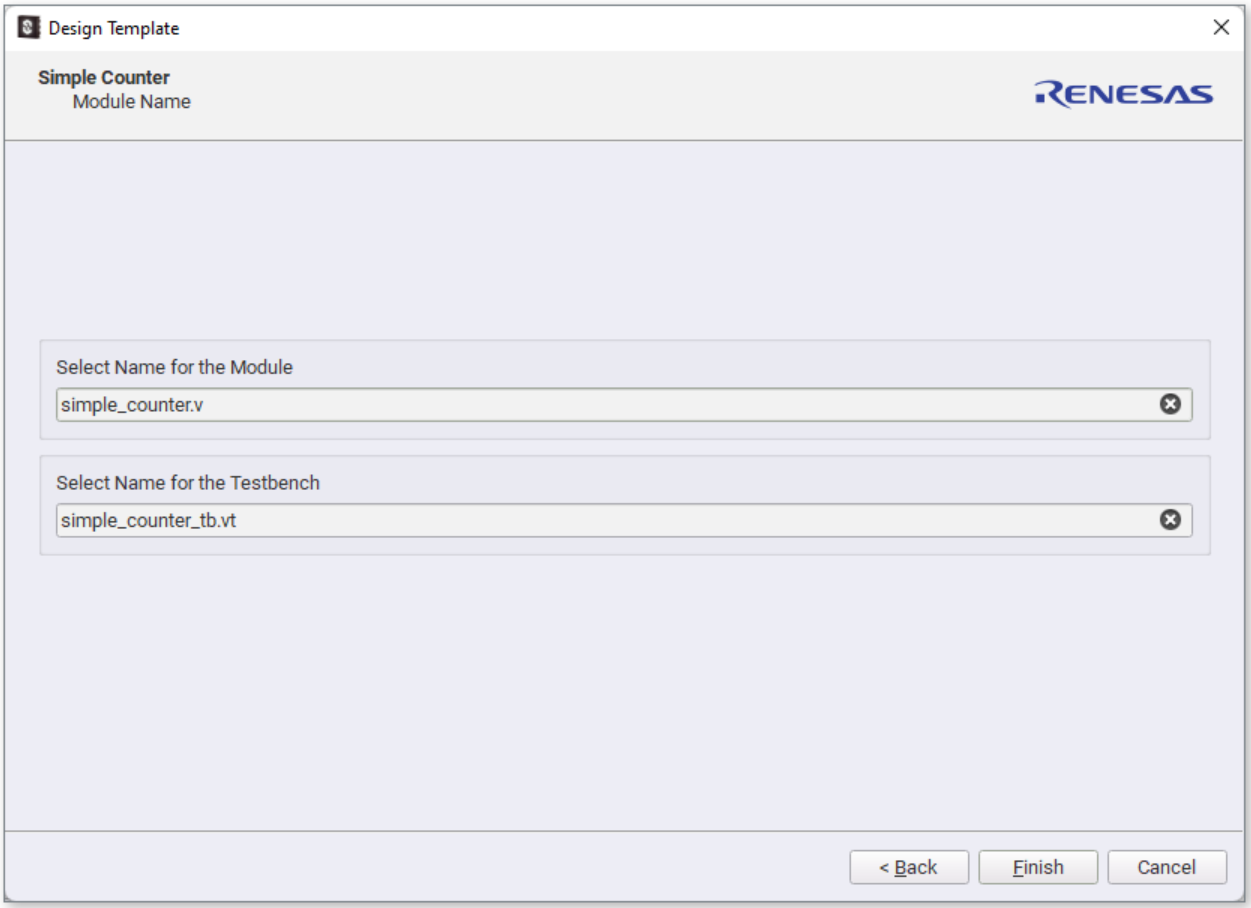


Figure 291. Module Name

After template setup is complete, new elements appear in the workspace. The **Design Template** Verilog code and **Testbench** names appear in the **Sources** list, with corresponding tabs displaying the actual code and **Testbench** details. In addition, the **I/O Planner** tab reflects the changes, showing the inclusion of port records from the selected design.

When integration of the design template is complete, its functionality can be assessed by running a simulation.

2.5.4 Writing Verilog Code

The toolchain of **ForgeFPGA** Workshop supports Verilog 2005 (IEEE Standard 1364-2005).

There are a few key code attributes that are important while working with the synthesis tool:

- `(* top *)` – The main module of the design should be marked with this attribute so that the toolchain can successfully recognize which module in the design is the top one.
- `(* clkbuf_inhibit *)` – Clock signals in the input list of the main module should be marked with this attribute. This prevents clock buffer insertion by the synthesis tool, which may lead to distortion of the clock signal name in the resulting netlist.

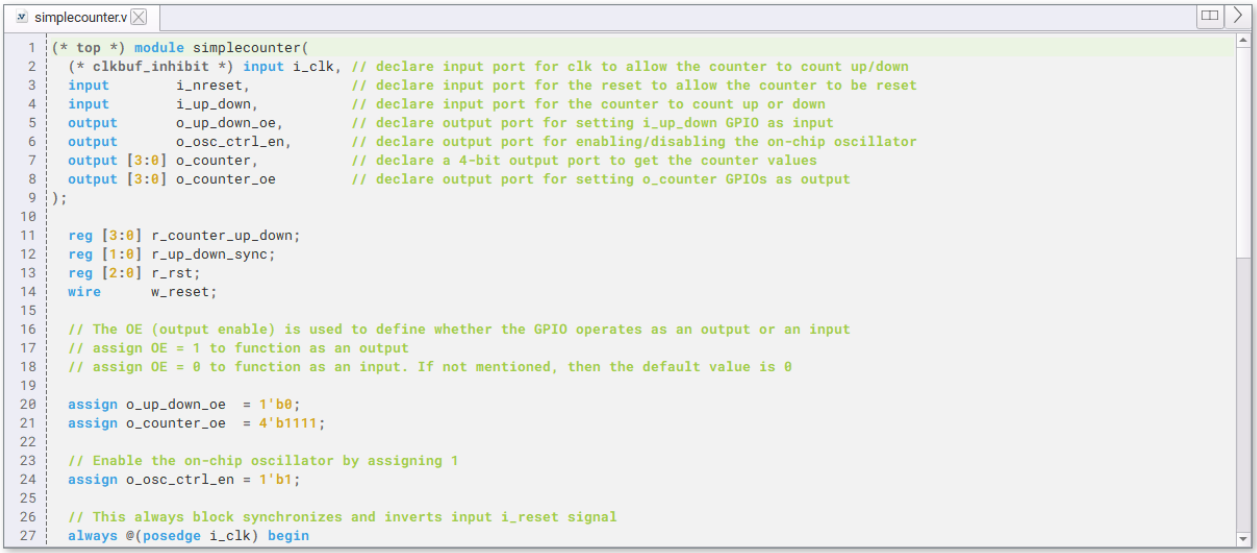


Figure 292. Example of Working with Verilog

The **ForgeFPGA** Workshop includes integrated linting support to detect syntax errors, coding issues, and design rule violations. The tool is compatible with Verilator v5.034.

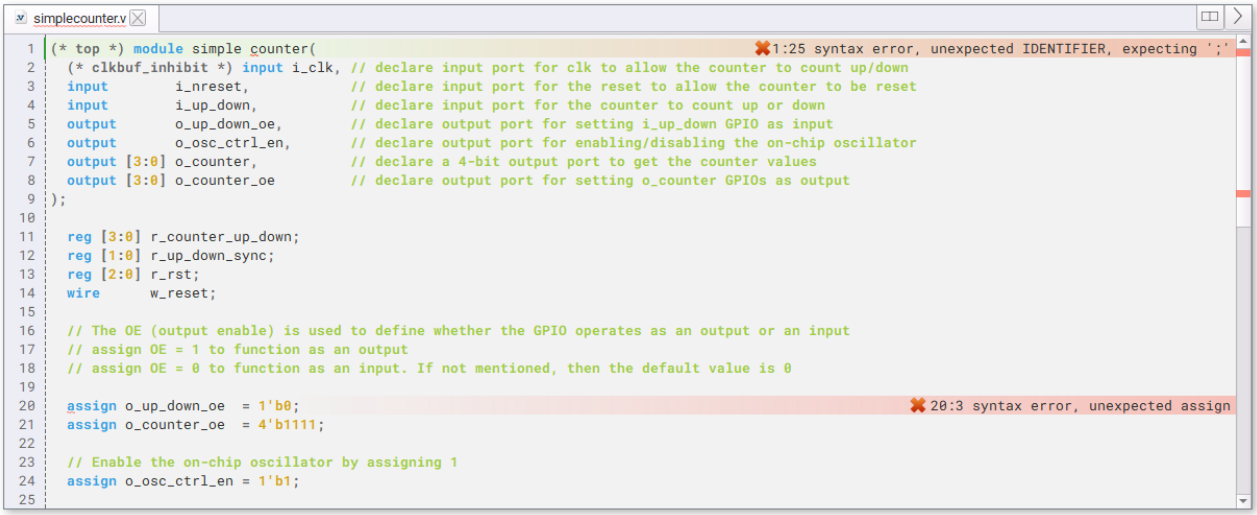


Figure 293. Linter Error Examples

2.5.5 Modules Library

Modules Library is a comprehensive repository of pre-designed and pre-verified modules that are easy to integrate. This tool provides Verilog code for various hardware modules, accompanied by testbenches for checking their functionality using simulation.

To open **Modules Library**, click the corresponding button on the toolbar or use the main menu, **File** > **Modules Library**. Choose the module, set the required configurations, and assign a name to complete module creation.

Inside the **Modules Library** GUI, the schematic, resource estimation, and description of the selected block are available. The GUI provides a detailed explanation of all input and output pins of

the block and allows the parameters to be changed as needed. This provides the flexibility to create the required Verilog code for the module and its associated testbench with just a few clicks.

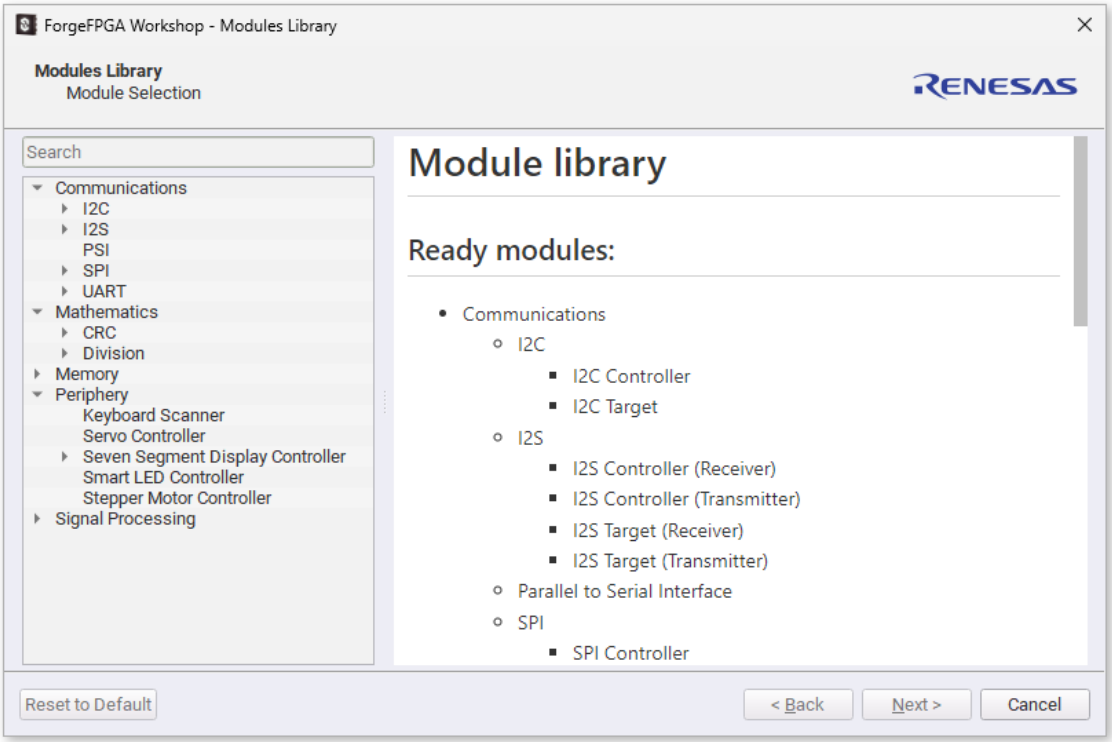


Figure 294. Modules Library GUI

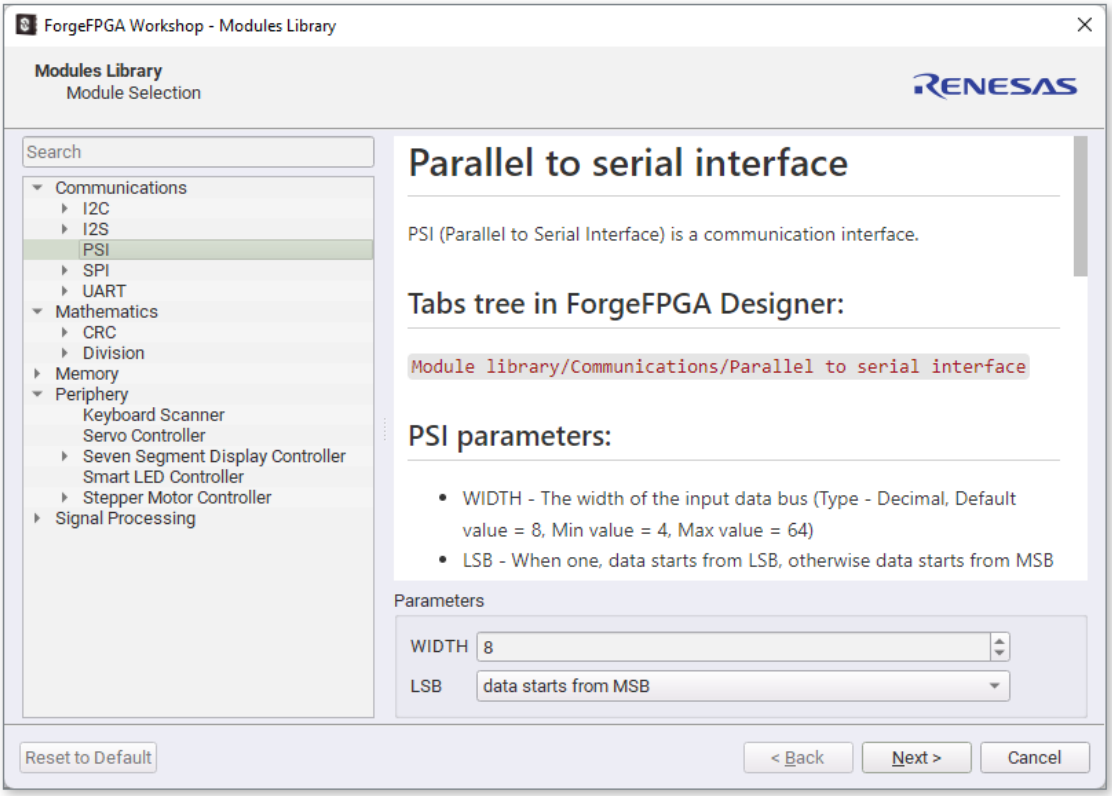


Figure 295. Modules Library GUI with Parameters

The added block can be found on the **Sources** control panel under the **IP Blocks** group. Along with the block, the testbench module is added and can be found under the **Testbenches** group. Multiple modules can also be added to the design and used simultaneously.

2.5.6 Synthesis

FPGA Editor has a built-in synthesis tool, Yosys (v0.37, v0.59), that takes an input design and produces a netlist file. While performing synthesis, the input design is analyzed and converted into gate-level representation. To run synthesis for the design, click the **Synthesize** button in the bottom left corner of **FPGA Editor** or from the main menu **Tools** > **Run Synthesis**. Choose between the Yosys versions in **Options** > **Settings**.

During synthesis, the software also checks the Verilog code for syntax errors. The log output can be checked on the **Messages** panel. Synthesis is successful only when the code is free of syntax errors.

2.5.6.1 Netlist

The system generates a netlist file after successful synthesis. It describes the components and connectivity of the source design and is required to perform the subsequent place-and-route procedure. The netlist can be accessed by clicking the toolbar **Netlist** button or from the main menu, **Window** > **Netlist**.

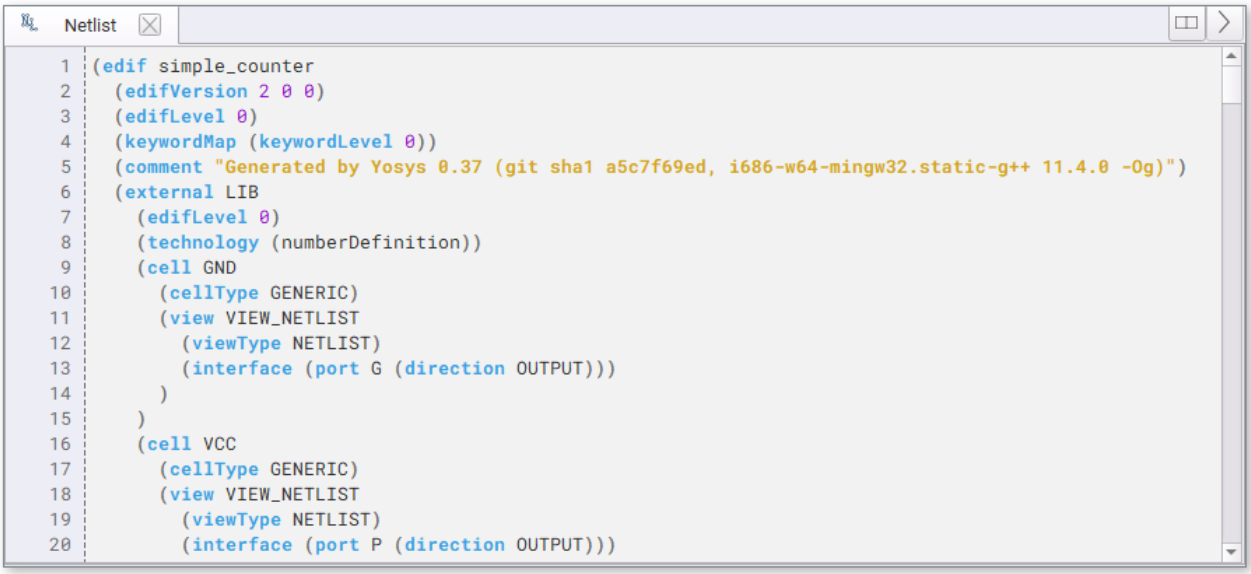


Figure 296. Netlist

An existing netlist can also be imported by clicking **File** > **Import** > **Netlist**. The external netlist appears in the **Sources** tree.

2.5.6.2 Post-Synth RTL

At the Register-Transfer Level, the design is represented by combinational data paths and registers. RTL synthesis is straightforward because each circuit node element in the netlist is replaced with an equivalent gate-level circuit. In Post-Synthesis RTL, the synthesized inputs are taken as a netlist. This helps provide information about the clock and other clock-related logic in the design, enabling additional I/O planning.

The Post-Synth RTL report can be found by clicking the respective toolbar button or using the main menu, **Window** → **Post-Synth RTL**. The report contains all connections made within the module between LUTs, FDREs, and Carry-Chain logic.

Figure 297. Post-Synth RTL

2.5.6.3 I/O Planner

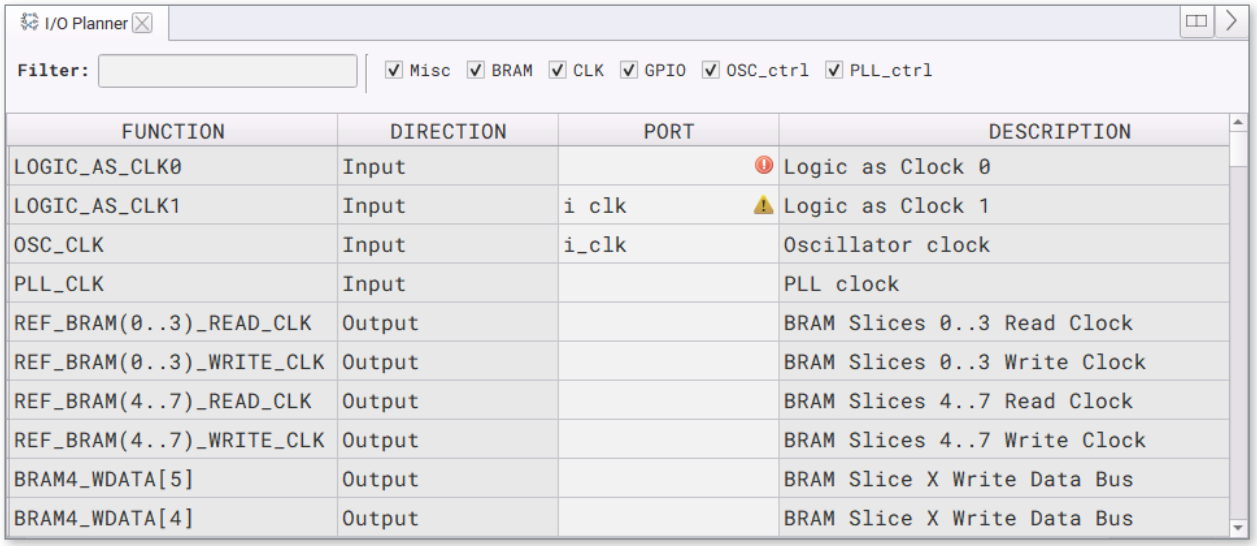
Each I/O port available on the FPGA has a dedicated function that can be mapped to the design using the **I/O Planner** tool. Click the corresponding button on the toolbar, or go to the main menu, **Window** > **I/O Planner** to launch the tool.

The **I/O Planner** tool is represented as a table with the following columns:

- **Function** – Dedicated function assigned to the port.
- **Direction** – Information about the IOB mode.
- **Port** – Editable column where ports from the Verilog design can be entered to connect them to the desired functionality. Double-click a cell to see the list of all ports defined in the Verilog code.
- **Description** – Data describing the port type and the function it fulfills.

Note: The cell indicates an invalid port name by showing a warning icon. Hover over the cell to

display the info hint with more details. Correct the name to perform the procedures successfully.



FUNCTION	DIRECTION	PORT	DESCRIPTION
LOGIC_AS_CLK0	Input		Logic as Clock 0
LOGIC_AS_CLK1	Input	i_clk	Logic as Clock 1
OSC_CLK	Input	i_clk	Oscillator clock
PLL_CLK	Input		PLL clock
REF_BRAM(0..3)_READ_CLK	Output		BRAM Slices 0..3 Read Clock
REF_BRAM(0..3)_WRITE_CLK	Output		BRAM Slices 0..3 Write Clock
REF_BRAM(4..7)_READ_CLK	Output		BRAM Slices 4..7 Read Clock
REF_BRAM(4..7)_WRITE_CLK	Output		BRAM Slices 4..7 Write Clock
BRAM4_WDATA[5]	Output		BRAM Slice X Write Data Bus
BRAM4_WDATA[4]	Output		BRAM Slice X Write Data Bus

Figure 298. I/O Planner

For easier navigation, use the additional controls to filter the desired data.

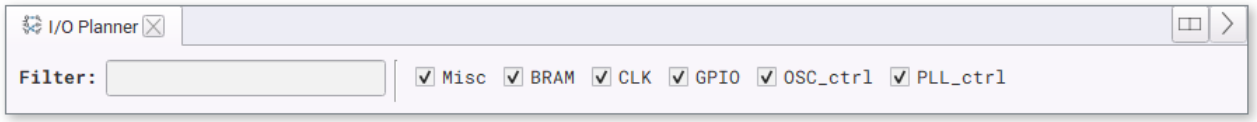


Figure 299. Filter Controls

Port data (.txt or .csv formats) can also be imported, exported, or cleared from the main menu, **Tools** > **I/O Planner**, or by using the table's context menu.

2.5.6.4 Synthesis Report

After synthesis of the current design is performed, a new report is generated. This report shows the number of primitive objects used to represent the design in the generated netlist.

To open the **Synthesis Report** window, click the corresponding button on the toolbar or use the

main menu **Window** > **Synthesis Report**.

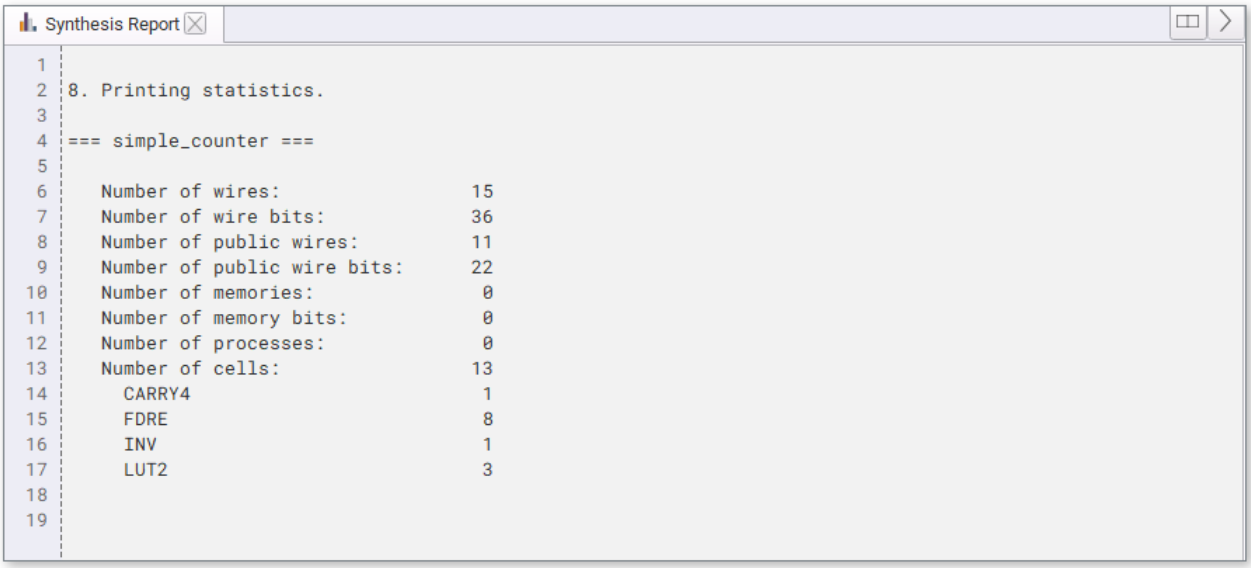


Figure 300. Synthesis Report

2.5.7 Generating Bitstream

To prepare a design to be sent to the device, the place-and-route and bitstream generation procedures must be performed. This can be done after the netlist and bitstream files are generated successfully. Place-and-route takes the elements of the synthesized netlist and maps its primitives to FPGA physical resources. To generate bitstream data, click the **Generate Bitstream** button in the bottom left corner of **FPGA Editor** or use the main menu, **Tools** > **Generate Bitstream**.

The **Bitstream Log** tab on the **Messages** panel can be checked to inspect the background steps after **Generate Bitstream** is clicked. In the background, the software automatically performs technology mapping, clustering and floor planning, placement and optimization, routing, and resource calculation. If an issue occurs in any of these steps, bitstream generation remains incomplete, and the outcome is shown on the **Messages** panel.

The generated bitstream is a hex file that, if generated correctly, can be used for further debugging of the design. For more information about the chip and flash procedures and their location, see section 2.2.6 Debugging Controls.

2.5.7.1 Bitstream Encryption

The software supports AES-128 encryption for generated bitstream files. When enabled, the compiler encrypts the resulting **FPGA_bitstream_FLASH_MEM** and **FPGA_bitstream_MCU** bitstreams to secure the design data. To configure bitstream encryption, enable the AES128 Config block (if available) on its **Properties** panel. Optionally, either manually enter the hexadecimal values for the **AES128 Key** and **AES128 init vec** (initialization vector), or use **Generate AES Keys**. The generated files will indicate **Bitstream type** in the file header.

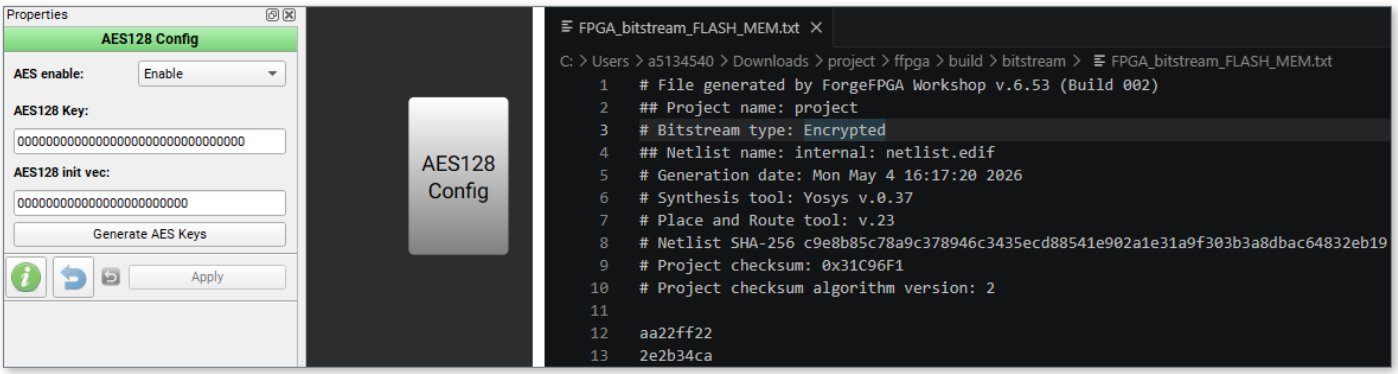


Figure 301. Bitstream Encryption

Note: Recompilation is not required after modifying encryption settings. The generated bitstream files are updated automatically.

2.5.7.2 Bitstream Content Management

Configure the contents of the generated **FPGA_bitstream_OTP** bitstream file using **OTP configuration file includes** setting:

- **Full FPGA configuration** – The generated OTP file will contain the complete FPGA design along with the boot configuration and user data.
- **Boot config + User OTP data only** – The generated OTP file will contain only the boot settings and specific User OTP data, excluding the main FPGA design.

The generated OTP bitstream file will indicate **Bitstream type** in the file header.

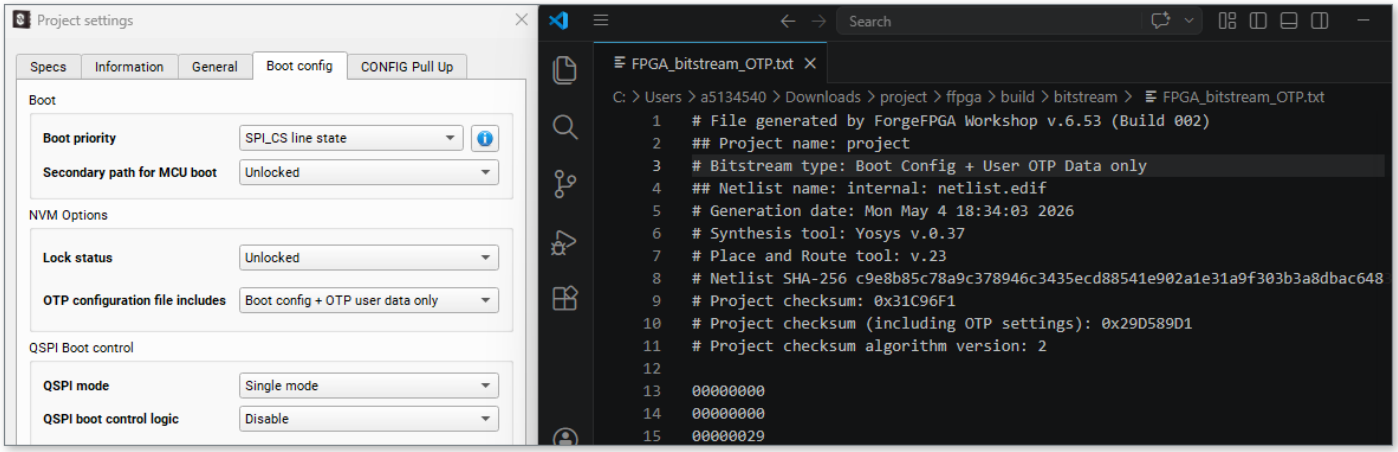


Figure 302. 'OTP Configuration File Includes' Setting

Note: Recompilation is not required after modifying **OTP configuration file includes** setting. The generated bitstream file is updated automatically.

2.5.7.3 Floorplan

The results of the place-and-route procedure are visualized in the **Floorplan** tool. Check how the primitives from the netlist are placed and interconnected, as well as how the I/O ports are mapped to the internal blocks and GPIOs. Launch the tool from the toolbar or the **Window** section in the

main menu. Use the bottom toolbar controls to navigate within the tool and inspect it more closely.

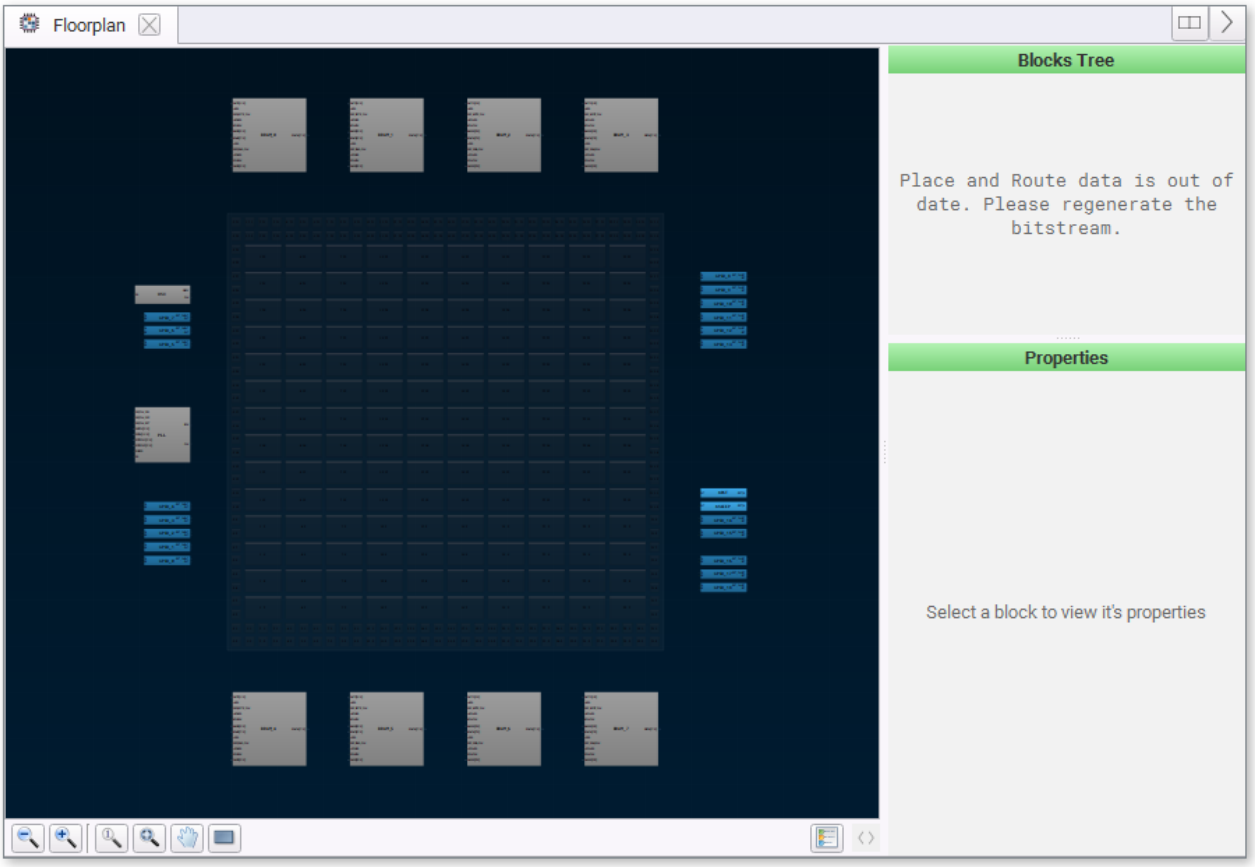


Figure 303. Empty Floorplan

Click the internal component or connection to see its details on the **Block Tree** and **Properties** info

panels. Filter out the required data in the respective field as needed.

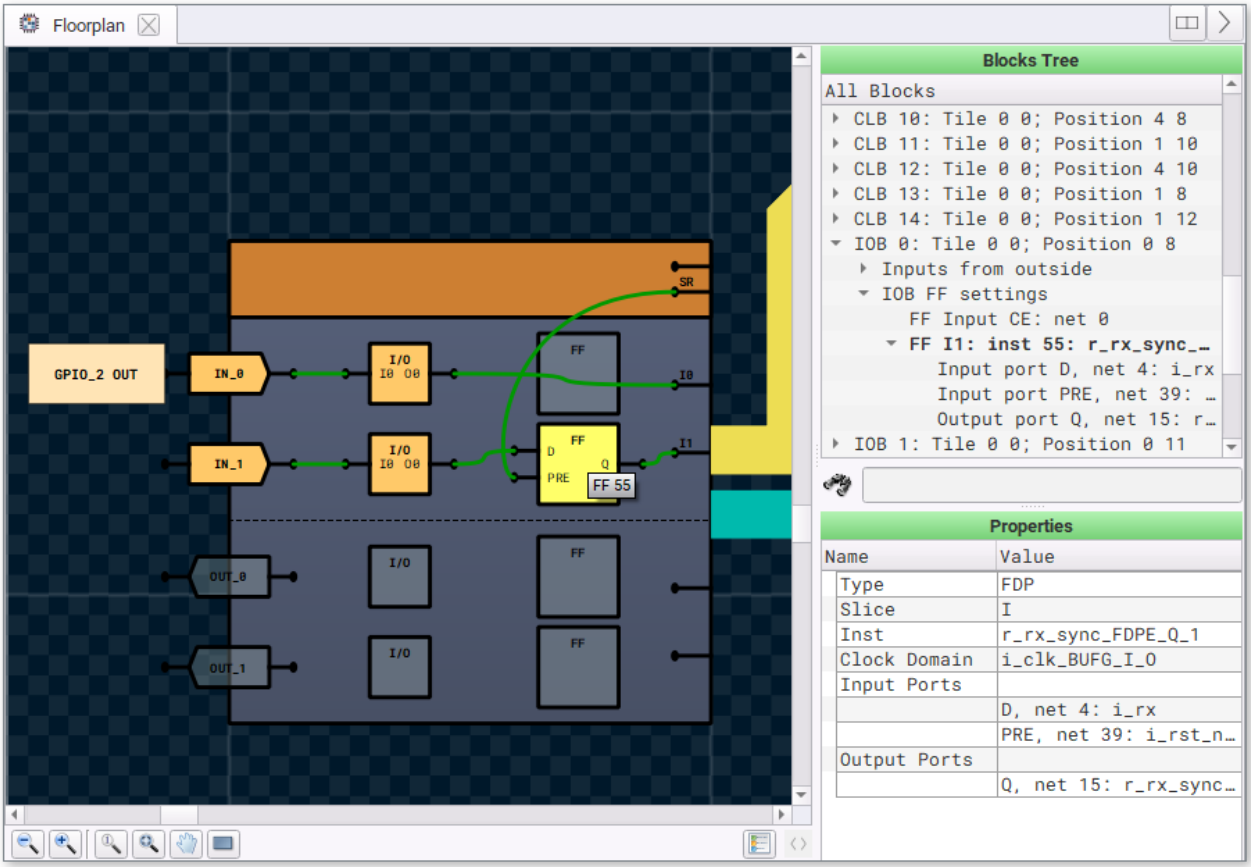


Figure 304. Floorplan Info Panels

Also, see the components and connections color scheme by clicking the **Legend box** icon at the right bottom.

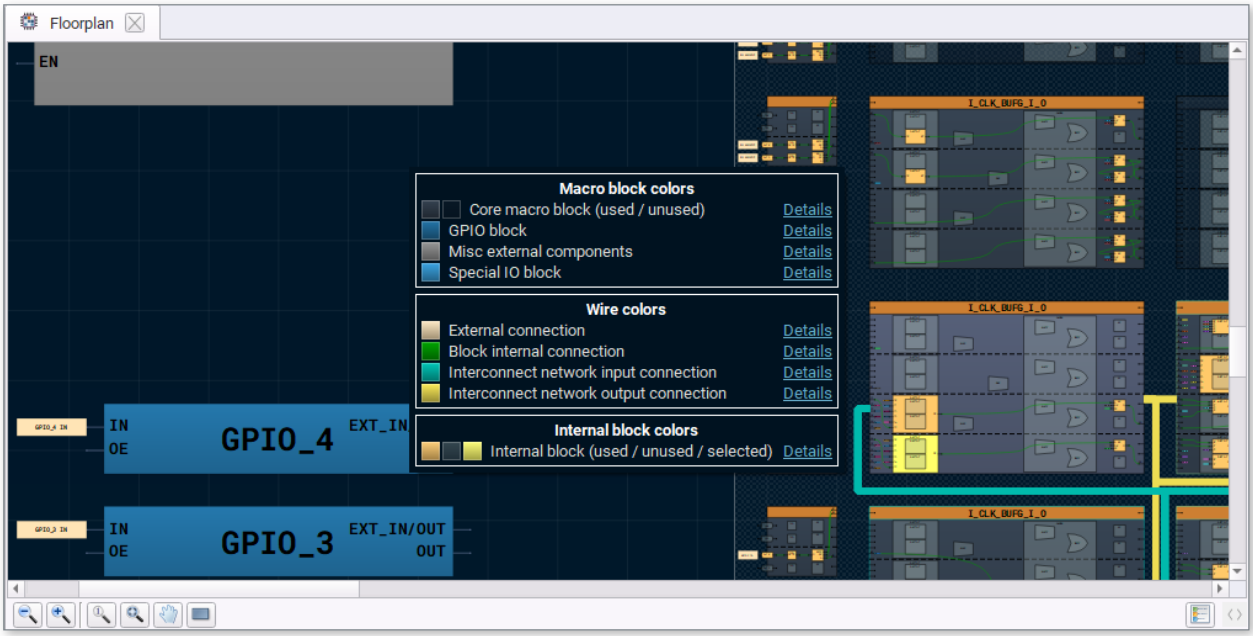


Figure 305. Informational Elements

In addition, configurations can be uploaded manually by accessing **Load custom PNR** from the main menu, **Tools** > **Floorplan**.

2.5.7.4 Resources Report

After the bitstream generation procedure is performed, a full report of the used resources is generated. This report shows the number of CLBs, FFs, I/Os, and LUTs used for design synthesis. The list of utilized resources can also be seen on the control panel. To open the **Resources Report** window, click the corresponding button on the toolbar or use the main menu, **Window** > **Resources Report**.

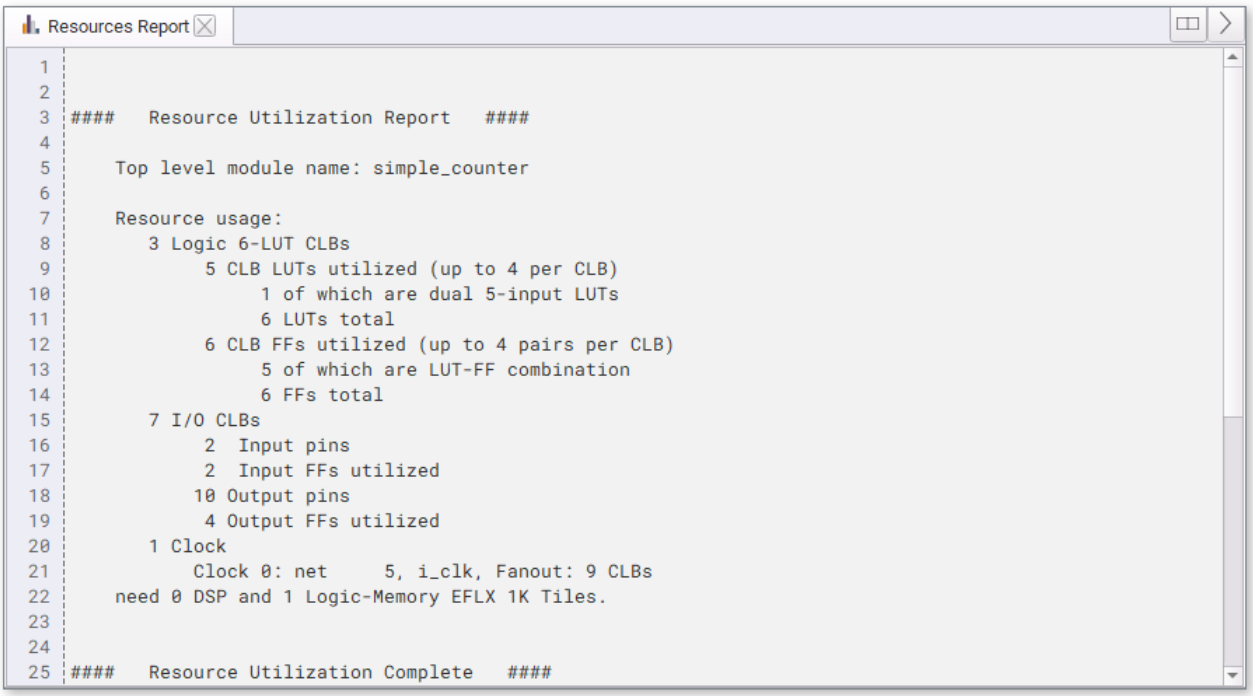


Figure 306. Resources Report

2.5.7.5 Timing Analysis

The tool is designed to extract timing information and check for any timing violations associated with internal registers. The results show whether all setup, hold, and pulse-width timing requirements are met.

The tool can be found on the toolbar or by clicking **Window** > **Timing Analysis** in the main menu.

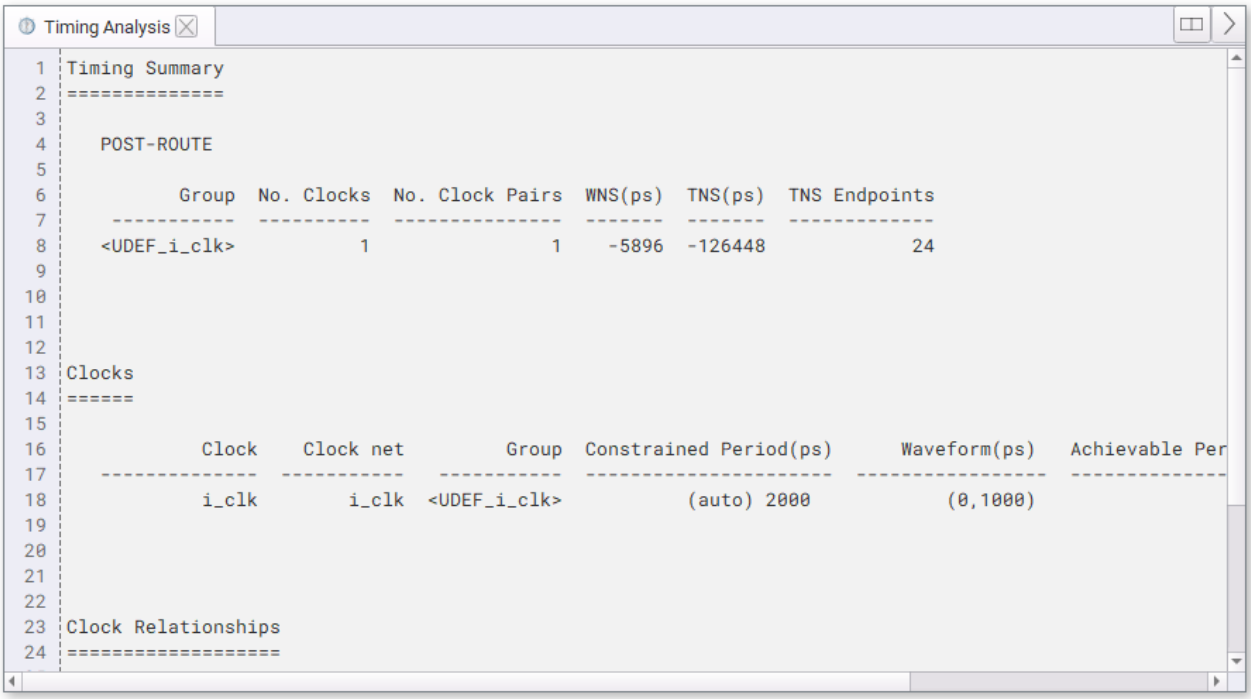


Figure 307. Timing Analysis

Add timing constraint files to define clock settings and path-specific requirements (add or import the file from the main menu). The constraints help validate timing and optimize performance to meet design timing goals.

See the list of supported SDC commands and arguments:

```
create_clock -name clockName [-add] {objectList} | -name clockName
[-add] [{objectList}] | [-name clockName [-add]] {objectList}
            -period value
            [-waveform {riseValue fallValue}]
            [-disable]
            [-comment commentString]
set_clock_groups
            -asynchronous | -physically_exclusive | -logically_exclusive
            [-name clockGroupname]
            -group {clockList} [-group {clockList} <85> ]
            -derive
            [-disable]
            [-comment commentString]
set_false_path
            [-setup | -hold]
            [-from {objectList}]
            [-through {objectList} [-through {objectList} ...] ]
```

```
            [-to {objectList}]
            [-forward_propagate]
            [-disable]
            [-comment commentString]
set_multicycle_path
            [-start | -end]
            [-setup | -hold]
            [-from {objectList}]
            [-through {objectList} [-through {objectList} ...] ]
            [-to {objectList}]
            pathMultiplier
            [-disable]
            [-comment commentString]
set_hierarchy_separator {sepchar} | sepchar
```

The commands below have basic support:

```
            get_cells
            get_ports
            get_nets
            get_pins
            get_clocks
```

See the short description of the supported SDC commands:

- `create_clock` – Creates a clock object and defines its characteristics.
- `set_clock_groups` – Defines the relationship between groups of clocks.
- `set_false_path` – Sets specific timing paths as false and excludes them from timing analysis.
- `set_multicycle_path` – Determines how many clock cycles a path can take.
- `set_hierarchy_separator` – Defines the character used to separate different levels of hierarchy in the design.

In case warnings appear during SDC parsing, they can be found in **build** > **ta** directory.

2.5.7.6 Placement Constraints

Add placement constraint file to control the location of the logic elements on the FPGA IC. Let's break down the supported commands:

- `create_placement_group name_of_the_group` – Defines a named placement region with specific coordinates (find the coordinates displayed on each block in the Floorplan). The first four numbers give the tile X/Y and X/Y within the tile, for the left-bottom corner. The next four numbers give the top-right corner. Note that the corner coordinates are included in the region. See the syntax example:

```
chip_tile_x=tilex, chip_tile_y=tiley, chip_x=x, chip_y=y,

chip_tile_x=tilex, chip_tile_y=tiley, chip_x=x, chip_y=y
```

Note: Find the blocks coordinates on the Floorplan.

- `create_cluster_only_placement_group name_of_the_group` – Creates a specialized placement group that is restricted to clustered placement only. During compilation, the cells (logic instances) in this group are packed near each other to minimize interconnect delay.
- `add_to_placement_group name_of_the_group [get_cells cells]` – Adds one or more cells (logic instances) to an existing placement group. During placement, the tool ensures these instances stay within the designated region or next to each other.
- `add_to_placement_group name_of_the_group [get_ports ports]` – Adds I/O ports (input or output pins) to an existing placement group, which may help to maintain routing efficiency between I/O and nearby logic.

Note: The ordering of the constraints matters. When an instance is assigned to more than one group by different constraints, the latter one will take effect.

To include all cells or ports from the design, use the * wildcard character instead of listing specific names. See the example:

- `add_to_placement_group name_of_the_group [get_cells *].`
- `add_to_placement_group name_of_the_group [get_ports *].`

Note: After placement constraints are added to the design, the compiler automatically applies them without requiring any additional action.

2.5.8 Project Directory Structure

After creating a project, the directory appears in the specified location. It contains the project file itself and several other subdirectories. Some appear depending on the added sources in the **Sources** tree. The changes made to the sources will synchronize between the **FPGA Editor** and the file on disk. The folders below store the following files:

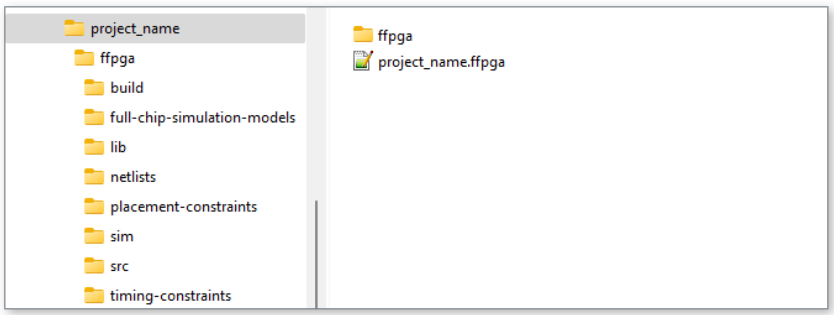


Figure 308. FPGA Project Structure

- **lib** – Modules from the **Modules Library**.

- **sim** – Testbench files.
- **src** – **main**, **mcm_generated** (created by clicking **Generate Verilog** in **Macrocell Editor**), and imported modules.
- **netlists** – Imported netlists.
- **full-chip-simulation-models** – Generated modules required for performing **Full Chip Simulation**.
- **timing-constraints** – Timing constraints files (the supported SDC commands can be found here).
- **placement-constraints** – Placement constraints files (the supported commands can be found here).

Use the **Open Containing Folder** feature in the context menu to quickly locate the file on disk.

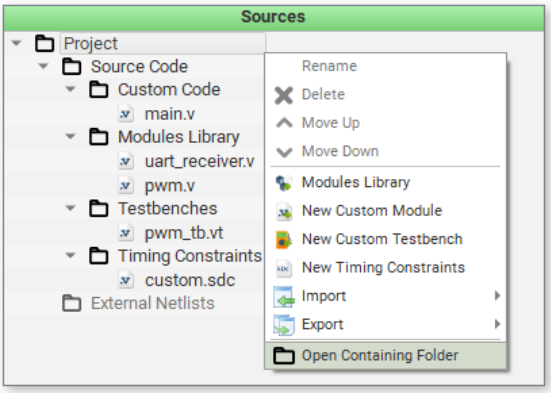


Figure 309. Check Sources File Location

As soon as we perform the procedures (synthesis or bitstream generation), the certain files are generated to the **build** folder. The list of generated files depends on the performed procedures. Below is a description of the most important files in the **build** folder:

- **bitstream** – The folder containing bitstream files in binary (.bin) and hexadecimal (.txt) formats:
  - **FPGA_bitstream_FLASH_MEM** – The bitstream sequence that can be used for on-board FLASH programming.
  - **FPGA_bitstream_MCU** – The bitstream sequence that can be used for MCU programming.
  - **FPGA_bitstream_OTP** – The bitstream sequence that can be used for OTP programming.
- **ta** – Stores log files containing potential warnings encountered during parsing of SDC commands in the **Timing Contraints** file (read more about supported commands here).

- **FPGA_bitstream_AXI.log** – Contains the configuration part of the bitstream that represents the logic of the FPGA core. This file is used while interacting with the development hardware when performing chip and flash procedures.
- **io_spec_in.txt** – Lists the I/O port specifications added in the **I/O Planner**, and is created immediately after clicking **Generate Bitstream**.
- **PNR_IO.log** – I/O port mapping file generated after successful bitstream generation.
- **netlist.edif** – The result of Yosys data processing, describing the components and connectivity within the source design. An external netlist from another software or synthesis tool can also be imported.
- **PNR_PACK_PLACE.log** – Source data for building a floorplan.
- **post_synth_results.v** – Yosys output data in Verilog code format.
- **resource-utilization-report.log** – Information about the amount of resources required for the design.
- **simulation** – Folder containing simulation results.

2.5.9 Simulation

Simulation allows verification of the overall functionality of the FPGA design and its response to different inputs without the need for physical hardware.

FPGA Editor works in conjunction with third-party software for testbench simulation, namely Icarus Verilog, and uses GTKWave to verify design functionality by viewing simulation results. Refer to the How to section for the installation and quick start guide for this additional software.

Run the process using the **Simulation** button on the toolbar or from **Tools** > **Simulation**.

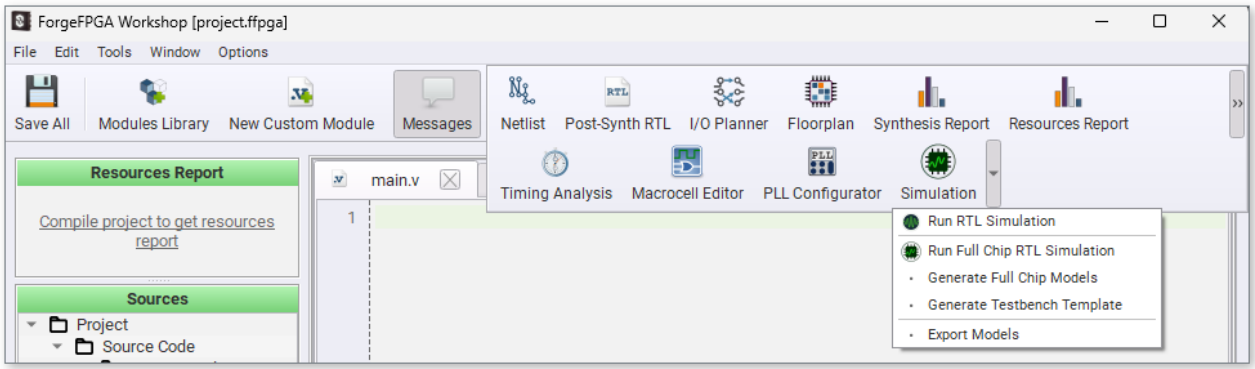


Figure 310. FPGA Simulation on the Toolbar

RTL Simulation executes the RTL only. **Full Chip RTL Simulation** allows extended behavioral simulation of the design, simulating not only the logic behavior of the code but also the behavior of the FPGA Core periphery, including GPIOs, BRAMs, LVDS, OSC, and PLLs. This simulation type also includes basic delays of peripheral circuits, producing results that more accurately reflect the

device's real behavior.

2.5.9.1 RTL Simulation

- Writing a testbench – To work with **RTL Simulation**, a testbench module for the FPGA design is required. A testbench can be added from the **Modules Library**, or a module can be imported from the main menu, **File** > **New Custom Testbench**.

uart_receiver_tb.vt

```
22  `timescale 1ns/1ps
23
24  module uart_receiver_tb;
25  //Parameters
26    parameter  BAUD_RATE         = 115_200;
27    parameter  IN_CLK_HZ         = 50_000_000;
28    parameter  OVERSAMPLING_MODE = 16;
29    parameter  DATA_FRAME        = 8;
30    parameter  STOP_BIT          = 1;
31    parameter  LSB               = 1'b0;
32    localparam CLK_PERIOD        = 20;
33    localparam BIT_PERIOD        = 8_600;
34  //Inputs
35    reg r_clk = 1'b0;
36    reg r_rst = 1'b1;
37    reg r_rx  = 1'b0;
38  //Outputs
39    wire [DATA_FRAME-1:0] w_rx_data;
40    wire                  w_rx_done;
41
42  // Module receive data from UART
43  uart_receiver #(
44    .BAUD_RATE         (BAUD_RATE),
45    .IN_CLK_HZ         (IN_CLK_HZ),
46    .OVERSAMPLING_MODE (OVERSAMPLING_MODE),
47    .DATA_FRAME        (DATA_FRAME),
48    .STOP_BIT          (STOP_BIT),
49    .LSB               (LSB)
50  ) dut (
51    .i_clk             (r_clk),
```

Figure 311. UART Receiver Testbench example from Modules Library

- Simulating a testbench – After the testbench is added, click **Simulation** > **Run RTL Simulation** on the toolbar to launch Icarus Verilog and GTKWave, or use the main menu: **Tools** > **Simulation** > **Run RTL Simulation**. The simulation stage handled by Icarus Verilog is performed in the background, while GTKWave provides the visual representation. If the selected testbench is correct and contains no syntax errors, GTKWave launches automatically.

2.5.9.2 Full Chip RTL Simulation

Performing a valid full chip simulation requires the following steps and conditions:

- Ensure the project contains error-free RTL code.
- Perform a successful synthesis stage. The generated netlist serves as the source for extracting the top-level module name and its port definitions.

- Verify that synthesis results are up to date with the design.
- Perform a successful bitstream generation stage. Post PnR results are utilized to assign correct delays in the generated models.
- Optional: perform testbench template generation (**Simulation** > **Generate Testbench Template**).
- Full chip models are automatically generated after you click **Run Full Chip RTL Simulation**, or you can generate them manually (**Simulation** > **Generate Full Chip Models**). Find them added in the **Sources** tree. To manage model generation behavior, reach **Options** > **Settings**. *Note:* Generated models are not part of the design and are not synthesized. They are used for full chip simulation only.

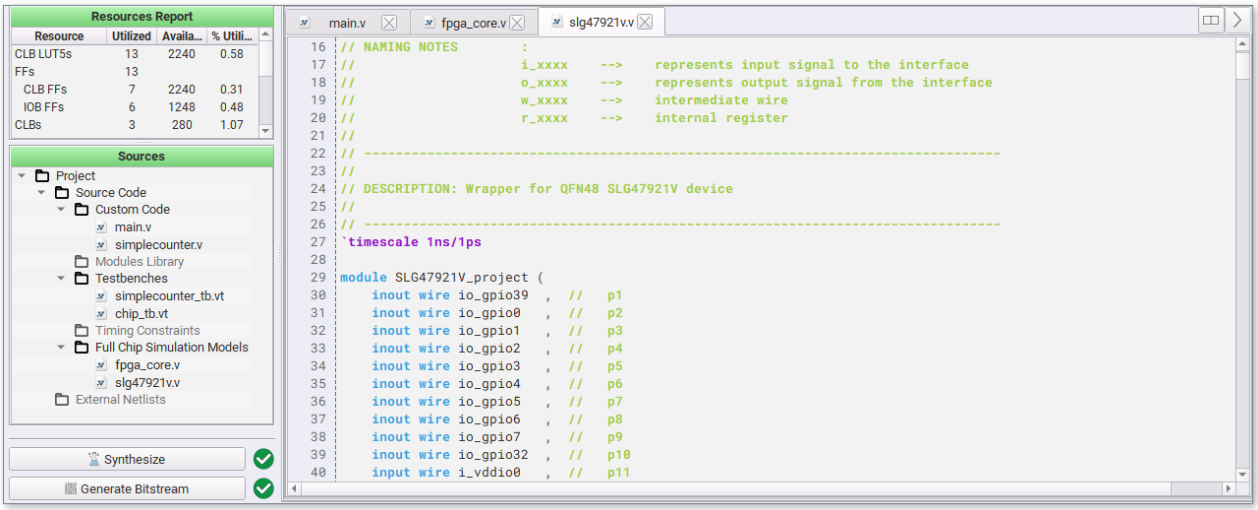


Figure 312. Full Chip Simulation Models in the Sources Tree

- Run simulation for selected testbench (**Simulation** > **Run Full Chip RTL Simulation**).

Models generated for the currently opened project can also be exported to a dump file. This file can be imported and used, for example, to perform two-chip simulation in another project (**Simulation** > **Export Models**). In case of simulation failure, check the **Messages** panel (**General Log** or **Issues** tab) and make the necessary changes.

FPGA Editor ensures that work can continue from the last saved state of the simulation results. After the necessary configurations are made, such as adding signals or adjusting their graphical representation, save the progress in GTKWave. The next time the same testbench is simulated, the previous state is restored.

2.5.10 Macrocell Editor

Macrocell Editor is a tool that allows creation of the desired circuit in a schematic view. Launch the tool by clicking the corresponding button on the toolbar, or go to the main menu, **Window**

> **Macrocell Editor**.

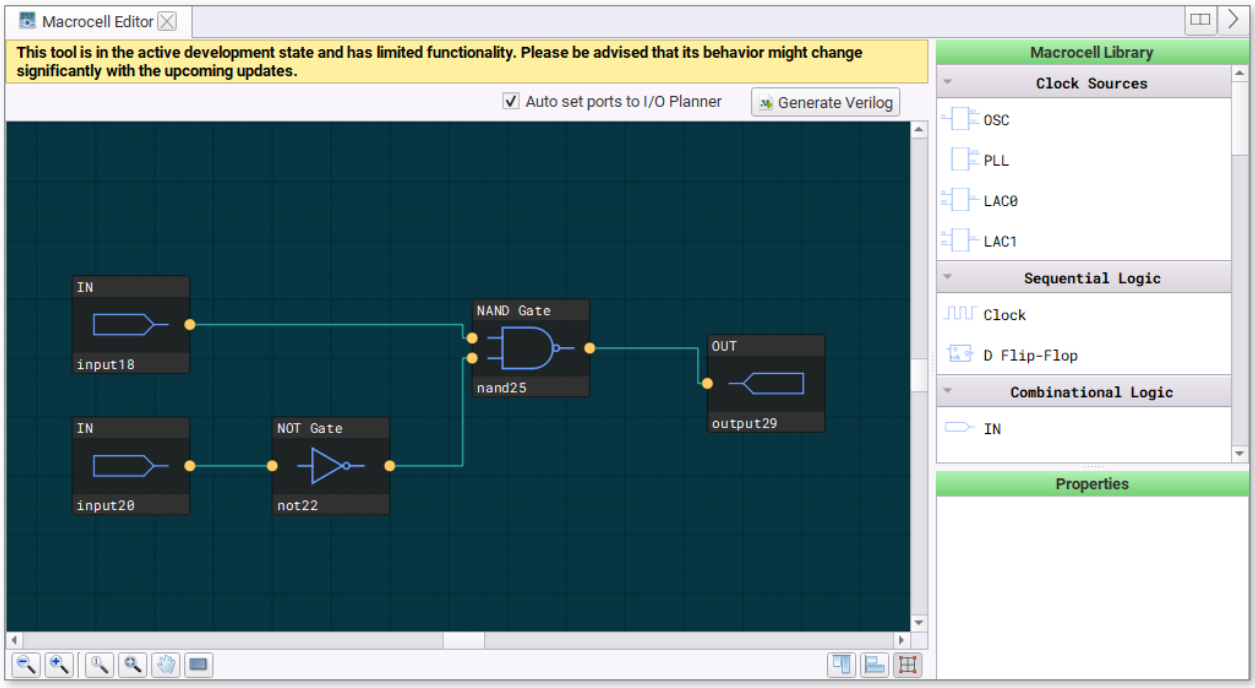


Figure 313. Macrocell Editor

The **Macrocells Library** panel contains the list of components that can be used for circuit design. The components may be grouped as follows (grouping depends on the selected PN): **Clock Sources**, **Sequential Logic**, **Combinational Logic**, **Blocks**, and **IP Modules**. Click the desired component from the library and then the work area to add it to the circuit.

The **Properties** panel shows the details for one selected macrocell or connection. The **Name** field is editable (double-click the field or the macrocell).

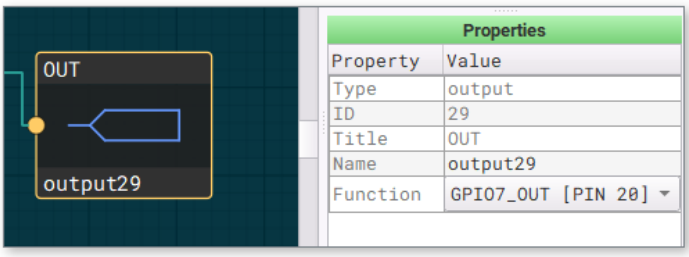


Figure 314. Ports Configuration in Properties Panel

For IN and OUT **Combinational Logic** blocks, map the ports directly on the **Properties** panel > **Function** dropdown. Tick **Auto set ports to I/O Planner** before clicking **Generate Verilog** to

display the desired mapping in the I/O Planner tool.

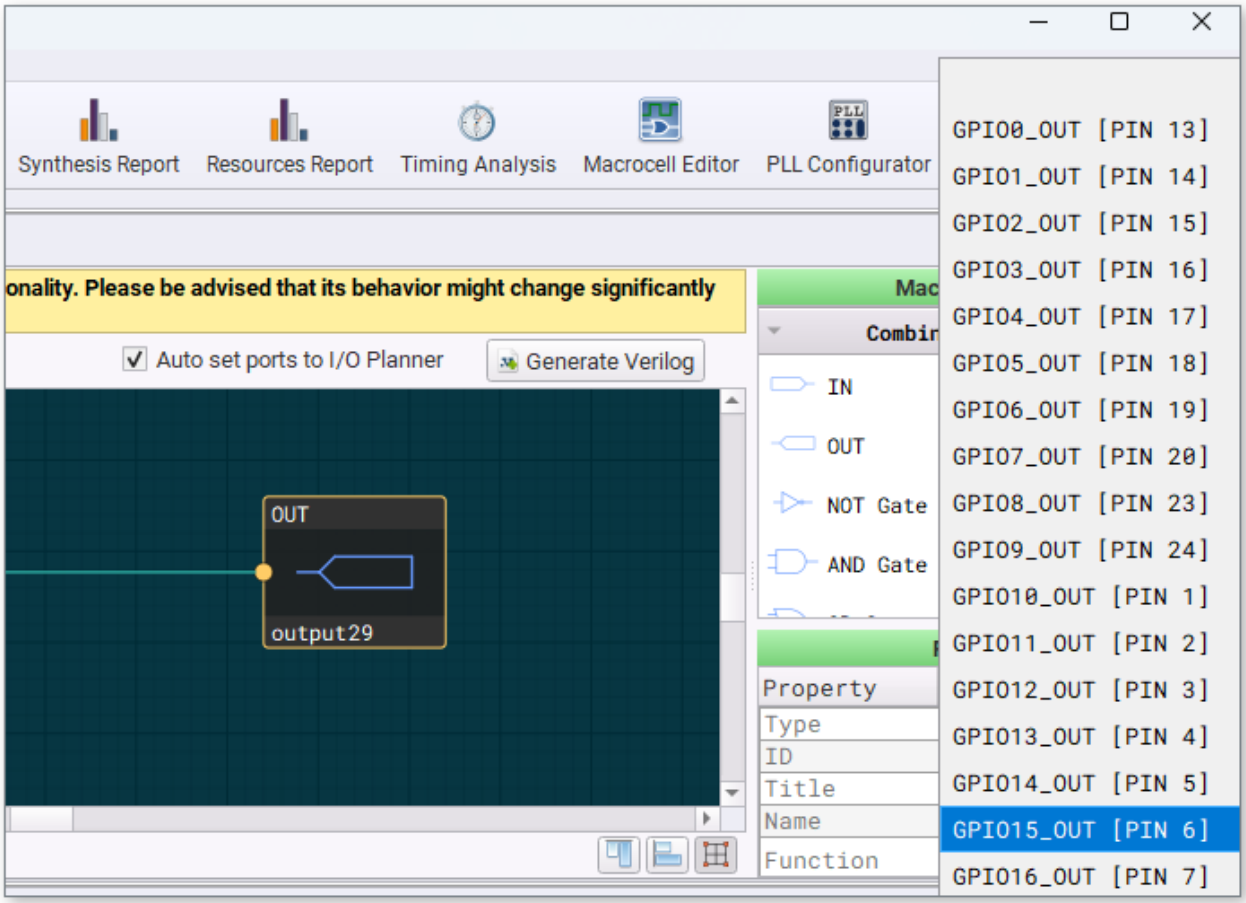


Figure 315. Port Autoset Checkbox

The macrocells may have clock and logic port types, which can be distinguished by color. Clicking two ports of the same type creates the connection between the macrocells.

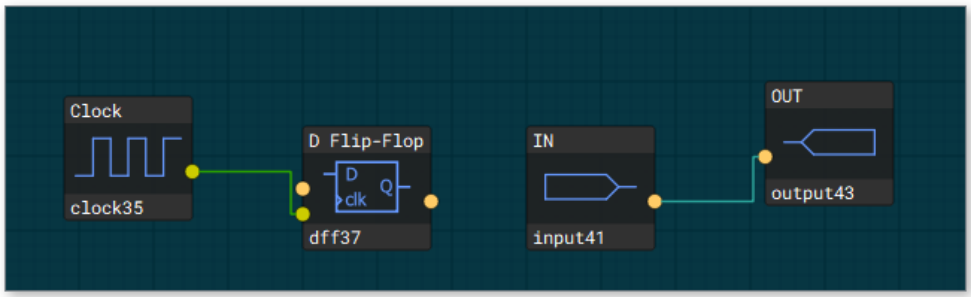


Figure 316. Connections Between Different Port Types

To generate Verilog code based on the created design, click the **Generate Verilog** button. The created Verilog code is listed as **mcm_generated** under the **Custom Code** section of the control panel. This code can be used for synthesis along with other modules added to the design.

2.5.11 PLL Configurator

The **PLL Configurator** helps determine the parameters for the desired output signal or dividers. It can be accessed through **FPGA Editor** > **PLL Configurator**.

2.5.11.1 PLL Calculator Tab

The main tab allows parameter configuration and application to the Verilog template, I/O Planner, and registers. For part numbers with multiple PLLs, tick the checkbox to enable the component and proceed with configuration.

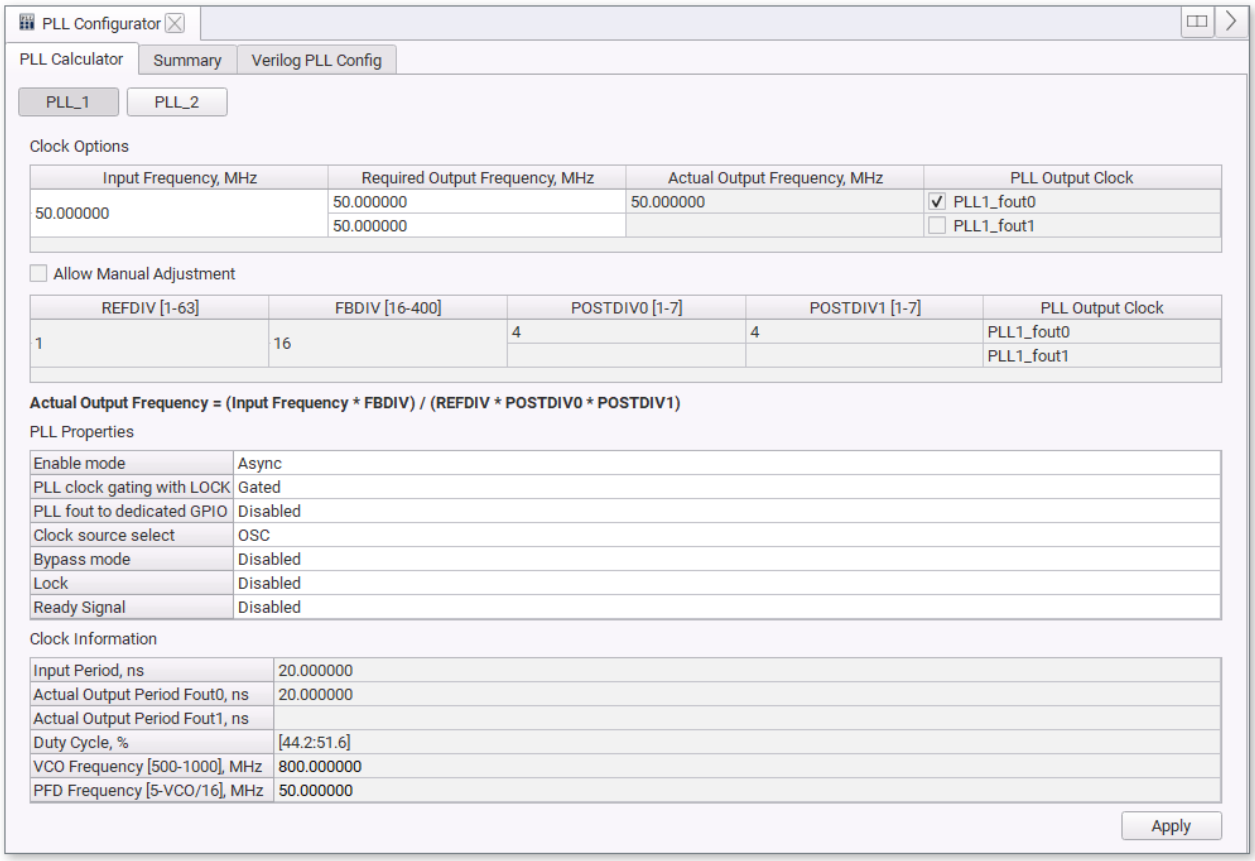


Figure 317. PLL Calculator Tab

The **PLL Configurator** tool has the following calculation modes:

- Dividers auto calculation – Set the **Input Frequency** and **Required Output Frequency** parameters to achieve the desired divider values (**REFDIV**, **FBDIV**, **POSTDIV1**, and **POSTDIV2**).
- Dividers manual adjustment – Enter **Input Frequency** along with all four dividers to receive the **Actual Output Frequency** value.

  Note: Check **Allow Manual Adjustment** to activate the mode.

Configure the parameters in the **PLL Properties** table, and pass the data to registers by clicking **Apply**. The **Apply** button saves changes for each PLL separately (applicable for part numbers with

multiple PLLs). Manage the behavior of the confirmation window before applying the changes in **Settings**.

Modifying settings may remap specific ports in I/O Planner. Conflicts trigger a resolution window.

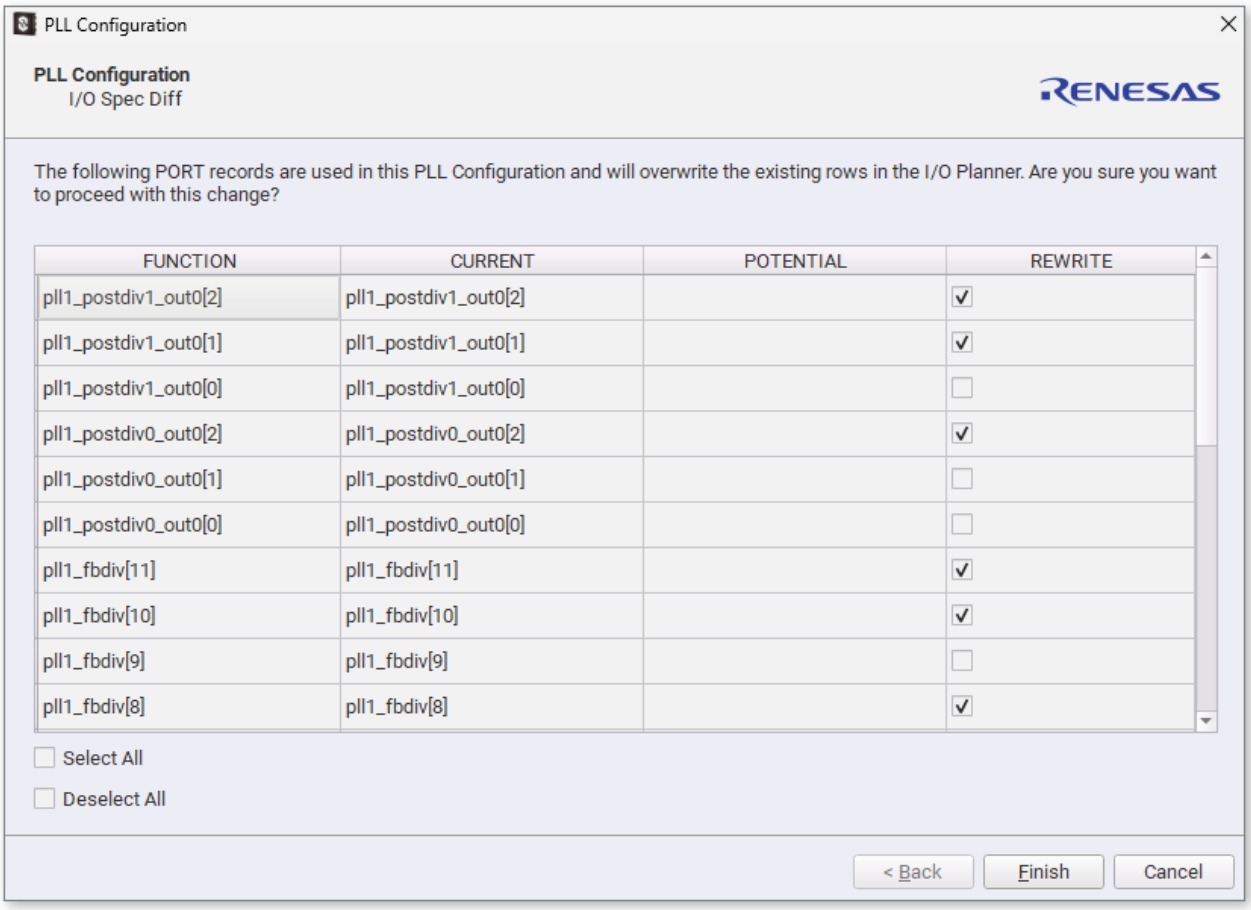


Figure 318. Conflict Resolution Window

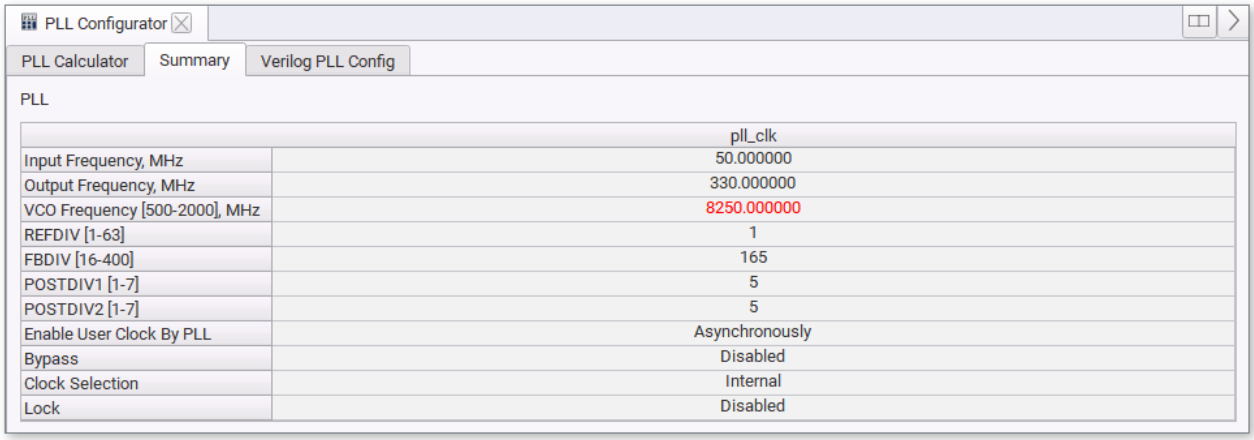
PLL data can also be configured in the component's **Properties** panel. The configurations are synchronized between the panel and **FPGA Editor**.

The **Clock Information** table values are based on the configured data. Results that exceed the valid range are highlighted in red.

2.5.11.2 Summary Tab

The current page displays the list of all configured parameters from the **PLL Calculator** tab in one

place.



	pll_clk
Input Frequency, MHz	50.000000
Output Frequency, MHz	330.000000
VCO Frequency [500-2000], MHz	8250.000000
REFDIV [1-63]	1
FBDIV [16-400]	165
POSTDIV1 [1-7]	5
POSTDIV2 [1-7]	5
Enable User Clock By PLL	Asynchronously
Bypass	Disabled
Clock Selection	Internal
Lock	Disabled

Figure 319. Summary Tab

2.5.11.3 Verilog PLL Config Tab

This tab shows dynamically generated Verilog code based on the predefined settings in the **PLL Configurator** tool, which can be used to configure the PLL component.

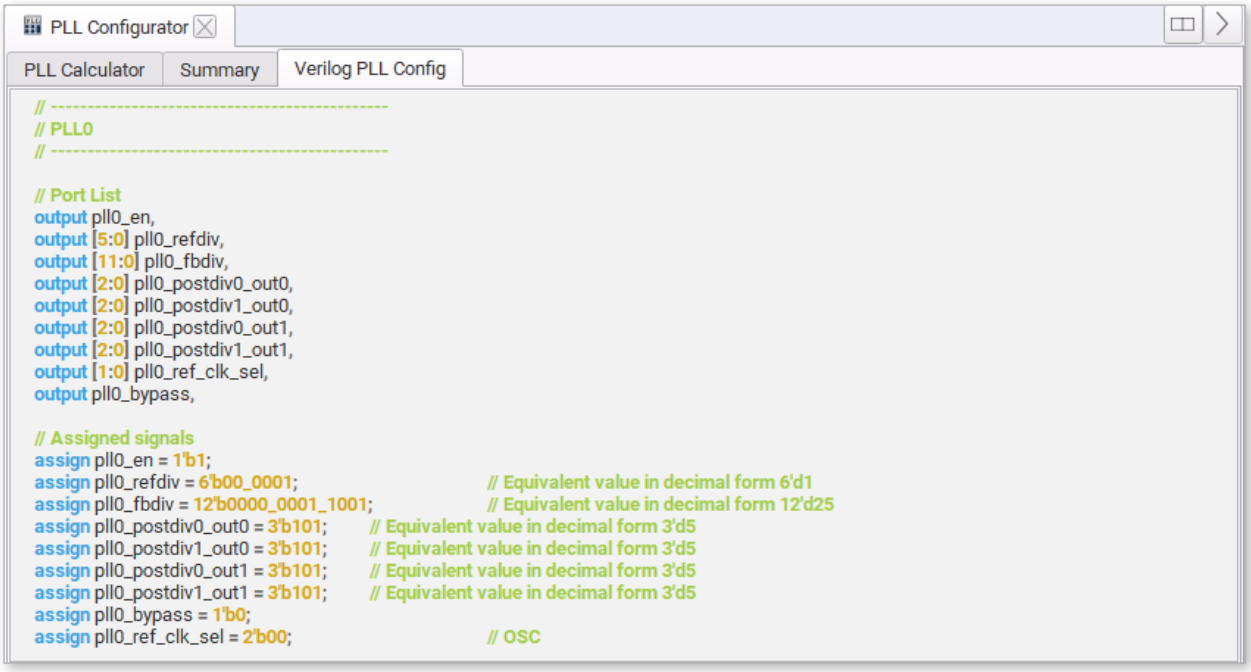


Figure 320. Verilog PLL Config Tab

2.5.12 Settings

The **ForgeFPGA Workshop Settings** window allows adjustment of multiple project and user settings.

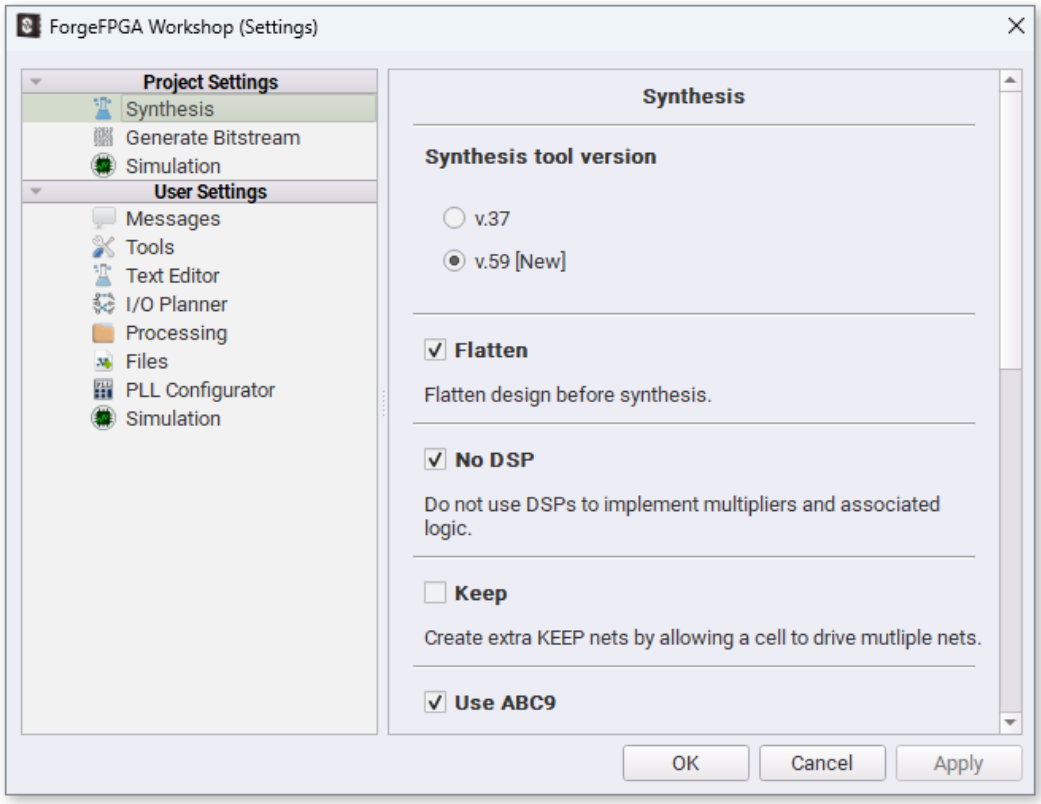


Figure 321. ForgeFPGA Workshop Settings

Note: Changes made in the **Project Settings** category apply to the project currently being worked on. Adjustments made in the **User Settings** section have a global impact, ensuring consistency across all FPGA-related projects on a particular device. If work is continued on another computer, these settings are either reset to default values or adapted to the configurations of that specific computer.

2.5.12.1 Project Settings

Within the **Project Settings** section, a range of options is available to optimize workflow. These options include:

- **Synthesize** – Enables flattening of the design before synthesis, management of DSP usage, generation of additional KEEP nets, and control of ABC9 usage. Additional arguments can also be added to the synthesis procedure.
- **Generate Bitstream** – Involves several settings and processes. These include the option to choose between Place-and-Route compiler versions (where applicable). Other settings include **High-Density IO Packing**, **High-Density Packing Logic**, **Clock-Concurrent Optimization (CCopt)**, **Place-and-Trial routing**, adjustments to the **Place-and-Trial routing iteration count**, and the **Maximum Routing Iterations** parameter. The number of CPU cores

used for the routing stage can also be set here. Additional arguments can also be added to the Generate Bitstream procedure.

- **Simulation** – Allows configuration of the Wavedump output format to either .vcd or .fst.

2.5.12.2 User Settings

In the **User Settings** category, the following can be customized:

- **Messages Panel** – Offers the flexibility to decide whether to clear **Synthesis Log** or **Bitstream Log** each time a new procedure is started.

- **Tools** – Provides the ability to modify the accessibility settings for the **Icarus Verilog tool** and the **GTKWave tool**. The path to the folders can either be specified for detection, or **Autodetect** can be chosen to let the software locate the files automatically. It is also possible to switch to using the system environment.

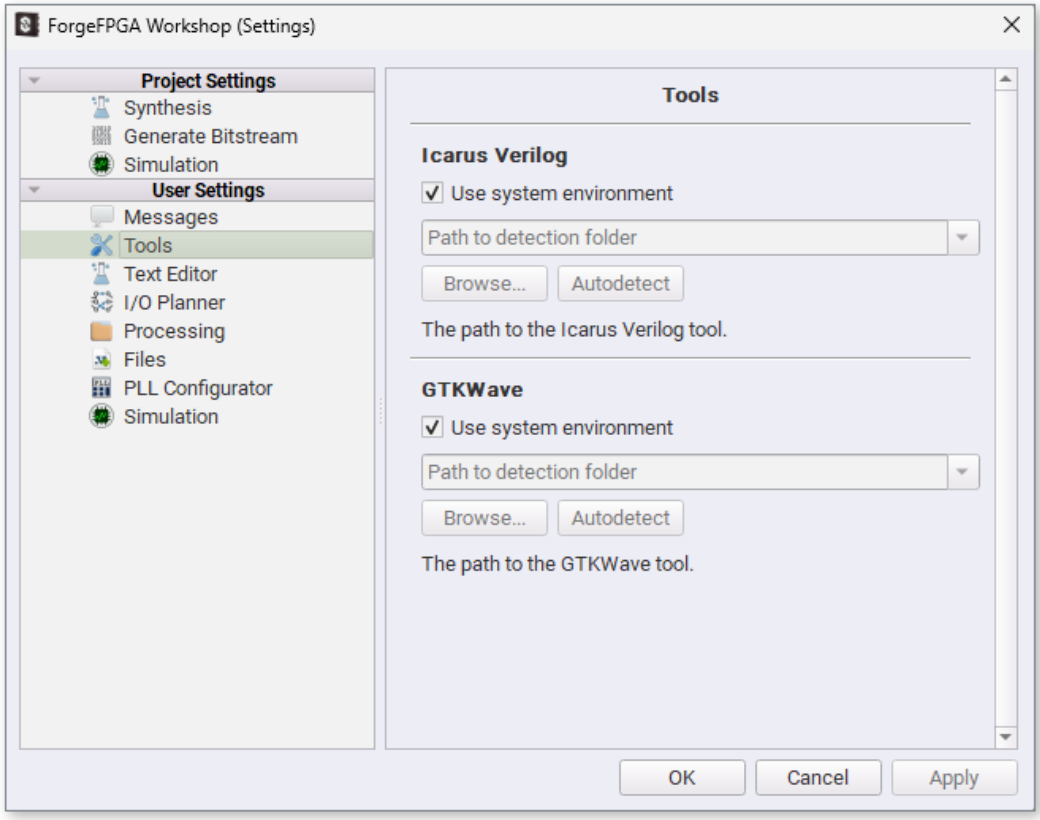


Figure 322. ForgeFPGA Workshop Settings Tools Tab

- **Text Editor** – Allows definition of the number of spaces in a tab for all text editors.

- **I/O Planner** – Serves to manage the display of information in the **I/O Planner** table.

- **Processing** – Provides saving options related to modified modules or configurations. Choose whether the latest changes should be saved before starting procedures.

- **Files** – Allows selection of whether to add the **_tb** suffix by default while creating new custom

testbenches.

- **PLL Configurator** – Provides the possibility to manage the behavior of the confirmation window before applying changes in the PLL Configurator.

Follow the instructions provided in the option descriptions to avoid errors and improve the design.

3 Devices

The Go Configure™ Software Hub allows project debugging using diverse hardware platforms to provide communication between a specific chip and the software. Depending on the device selected for a project, the software adapts to support different platforms.

In this chapter, the supported development platforms for GreenPAK™, ForgeFPGA™, and Power GreenPAK™ devices are introduced. Each platform is presented with an overview, functional descriptions, and insights into its software representation.

3.1 GreenPAK Devices

3.1.1 GreenPAK Advanced Development Board (SLG4DVKADV)

- Programming and emulation for GreenPAK devices.
- 18 individually configurable Test Points (TPs):
  - Analog, digital and I2C generators support.
  - Onboard LED state indication.
  - Pull-up, Pull-down, GND, VDD, Hi-Z.
  - Programmable software button.
  - Floating hooks on each Test Point for oscilloscope connection.
- Dual VDD IC's support.
- 20-pin socket adapter support.
- USB interface for power and communication (mini-B connector).
- In-System Debugging (ISD) and In-System Programming (ISP) functionality.
- Software controlled configurable pin header for user schematic integration and signal monitoring (expansion connectors).

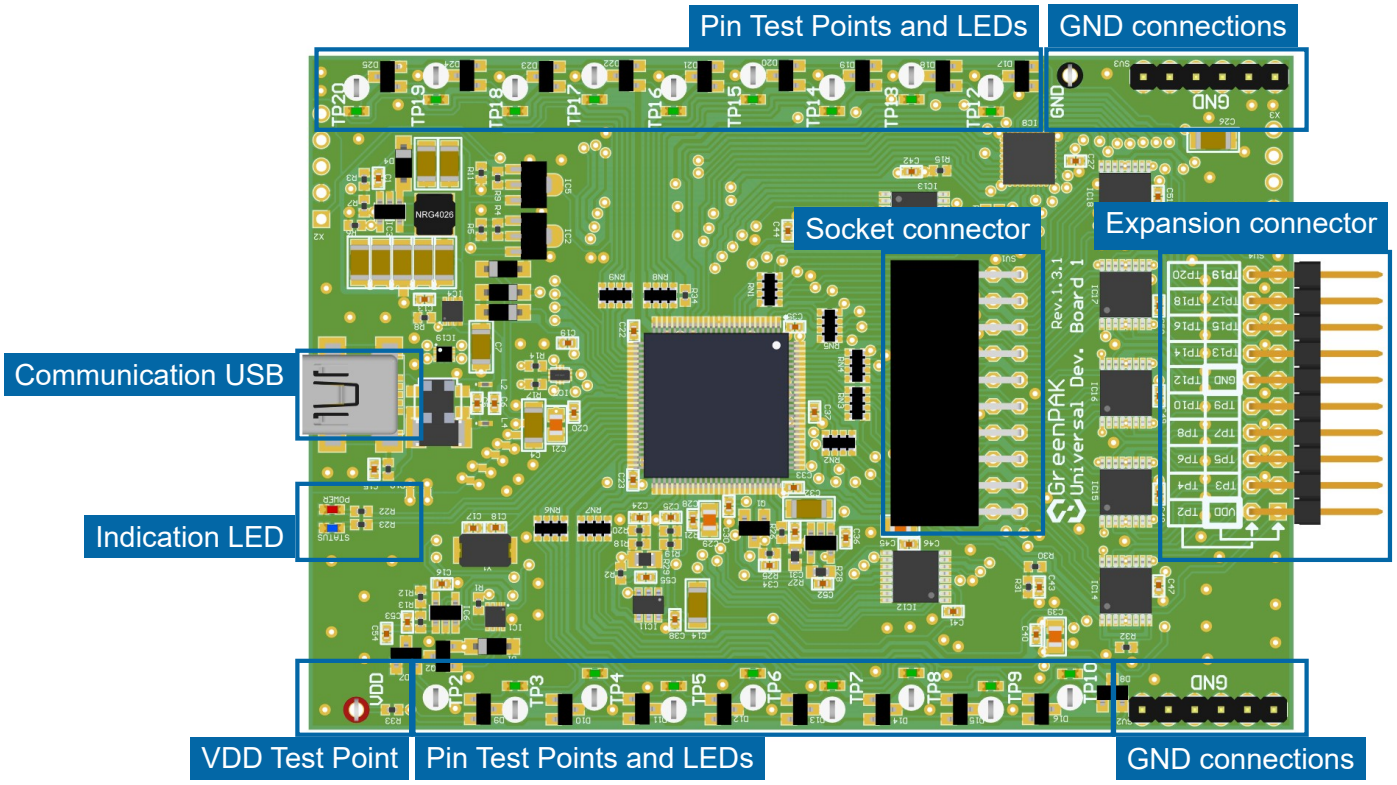


Figure 323. GreenPAK Advanced Development Board

The GreenPAK Advanced Development Board **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Device selector
- Emulation
- Emulation(sync)
- Test Mode
- Read
- Program
- NVM Programmer window
- Generator controls
- Expansion connectors
- Power source selector
- TP map
- LEDs ON and LEDs OFF
- Info details

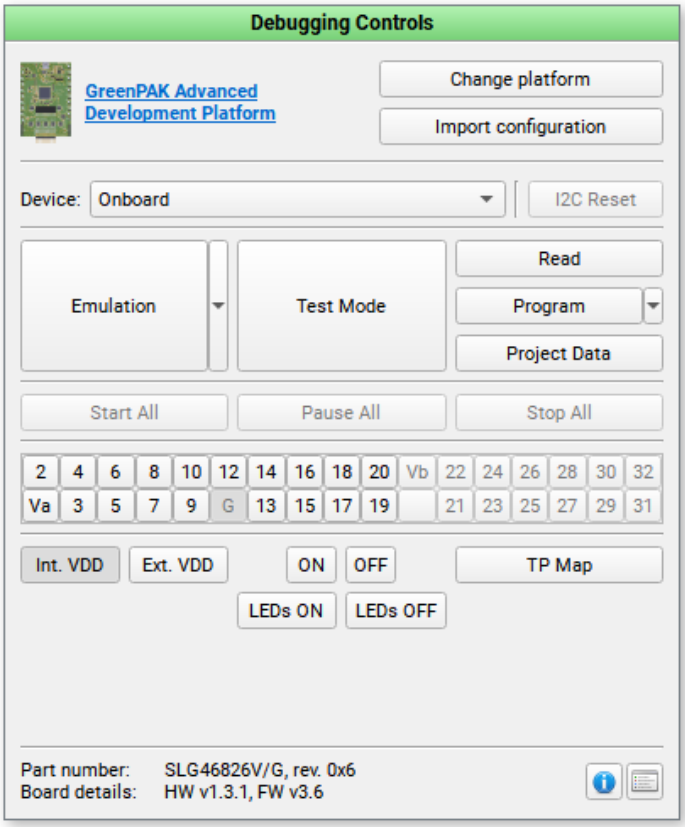


Figure 324. GreenPAK Advanced Development Board controls

Find a comparison of the controls available on different GreenPAK boards in the appendix, section 6.3 Debugging Controls Feature Availability.

3.1.2 GreenPAK DIP Development Board (SLG4DVKDIP)

- Programming and emulation for GreenPAK devices.
- 18 individually configurable Test Points (TP):
  - Pull-up, Pull-down, GND, VDD, Hi-Z.
  - Programmable software button.
- Dual VDD IC's support.
- DIP adapter support.
- USB interface for power and communication (micro-B connector).
- In-System Debugging (ISD) and In-System Programming (ISP) functionality.
- Software controlled configurable dual pin header for user schematic integration and signal monitoring (expansion connectors).

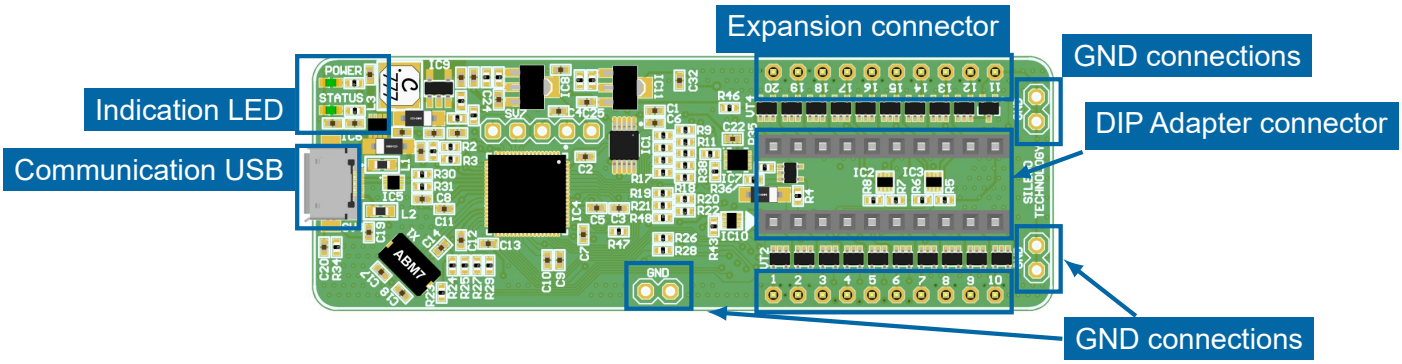


Figure 325. GreenPAK DIP (Dual In-Line Package) Development Board

The GreenPAK DIP Development Board **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Device selector
- I2C Reset
- Emulation
- Emulation(sync)
- Test Mode
- Read
- Program
- NVM Programmer window
- Generator controls
- Expansion connectors
- Power source selector
- TP map
- LEDs ON and LEDs OFF
- Info details

Figure 326. GreenPAK DIP Development Board controls

Find a comparison of the controls available on different GreenPAK boards in the appendix, section 6.3 Debugging Controls Feature Availability.

3.1.3 GreenPAK Lite Development Board (SLG4DVKLITE)

- Programming and emulation for GreenPAK devices.
- 18 individually configurable Test Points (TP):
  - Onboard LED state indication.
  - Pull-up, Pull-down, GND, VDD, Hi-Z.
  - Programmable software button.
  - Four floating hooks for oscilloscope connection.
- Dual VDD IC’s support.
- DIP and socket adapters support.
- USB interface for power and communication (USB-C connector).
- In-System Debugging (ISD) and In-System Programming (ISP) functionality.
- Software controlled configurable dual pin header for user schematic integration and signal monitoring (expansion connectors).

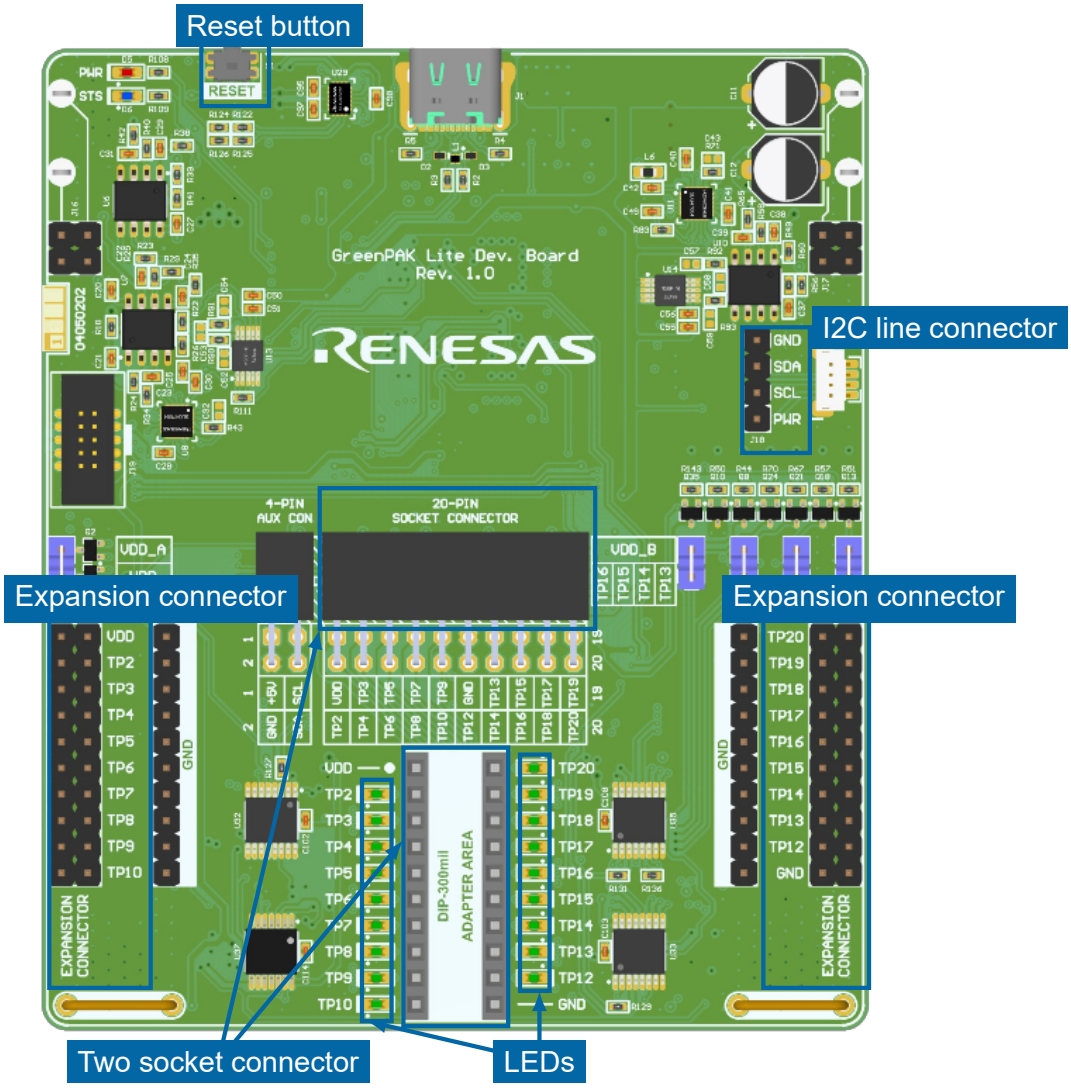


Figure 327. GreenPAK Lite Development Board

The GreenPAK Lite Development Board **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Device selector
- External device modes
- I2C Reset
- Emulation
- Emulation(sync)
- Test Mode
- Read
- Program
- NVM Programmer window
- Expansion connectors
- Power source selector
- TP map
- LEDs ON and LEDs OFF
- Voltage level controls
- Info details

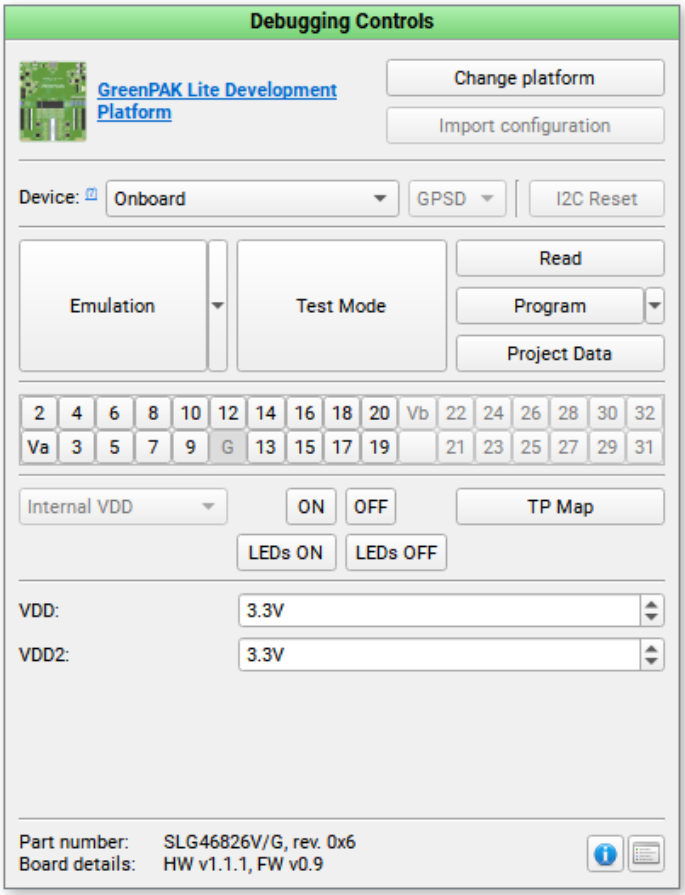


Figure 328. GreenPAK Lite Development Board Controls

Find a comparison of the controls available on different GreenPAK boards in the appendix, section 6.3 Debugging Controls Feature Availability.

3.1.4 GreenPAK Serial Debugger Board (SLG4DVKGSD)

- In-System Programming for Multiple-Time Programmable GreenPAK devices (e.g. SLG46824, SLG46826, SLG47004).
- In-System Debugging for reading, emulating and debugging GreenPAK devices.
- USB interface for power and communication (micro-B connector).
- 4-pin connector with I2C interface.
- Internal and external power supply for I2C.

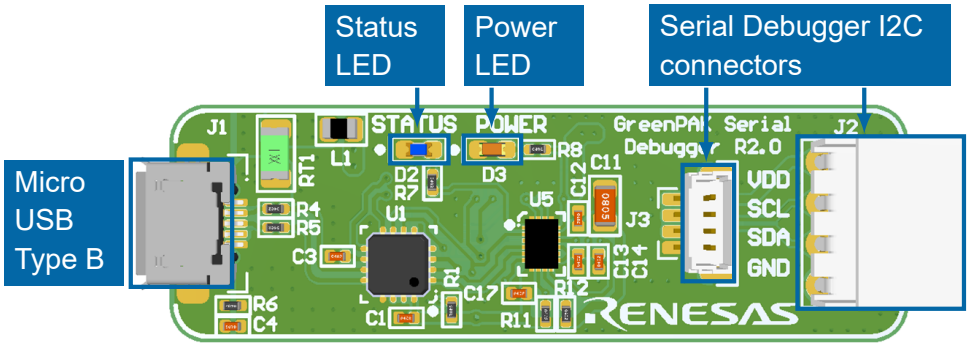


Figure 329. GreenPAK Serial Debugger Board

The GreenPAK Serial Debugger Board **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Device selector
- I2C Reset
- Emulation
- Emulation(sync)
- Test Mode
- Read
- Program
- NVM Programmer window
- Power source selector
- Voltage level controls
- Info details

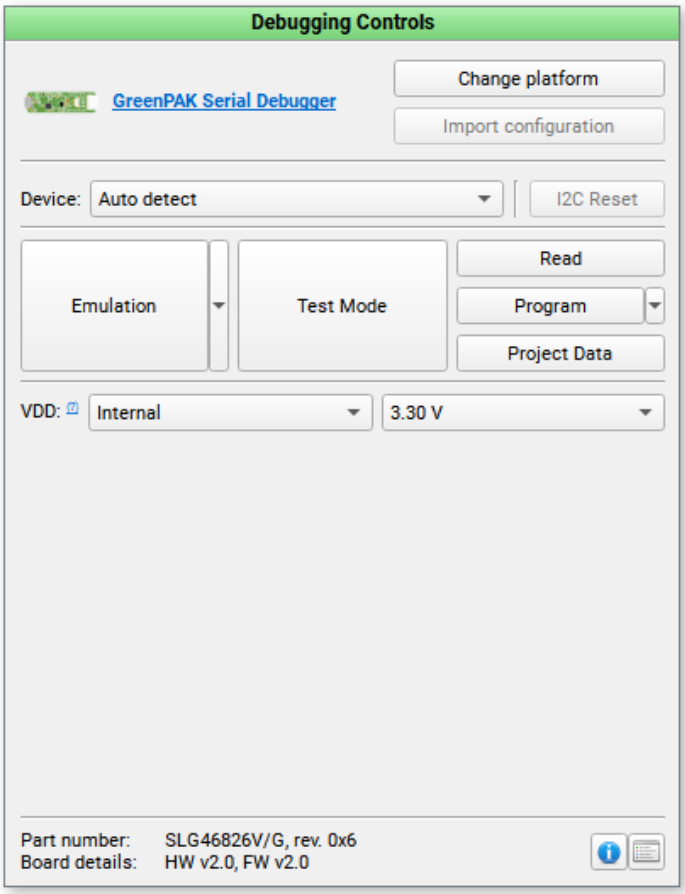


Figure 330. GreenPAK Serial Debugger Board Controls

Find a comparison of the controls available on different GreenPAK boards in the appendix, section 6.3 Debugging Controls Feature Availability.

3.1.5 Connecting External GreenPAK

3.1.5.1 GreenPAK Advanced, DIP, GreenPAK Lite and Go Configure Development platforms

- Connect a socket with an external chip to the expansion connector pins:
  - Corresponding **I2C** pins (**SCL**, **SDA**).
  - **VDD** pin.
  - **GND** pin.
- Disconnect the Onboard chip to proceed with the external chip.
- Start the Go Configure Software Hub and select the part number of the connected external chip.
- Start the **Debug tool** and select the development platform.
- Select the **Device address** (used by default for an empty chip) or the corresponding address programmed to the chip.

Note 1: If an external socket is connected to a development board correctly and the proper **Device address** is selected, a chip is detected either after clicking **Update chip info** or automatically after starting any chip operation

Note 2: **Ext. VDD (Va)** expansion can be disabled if the voltage is applied to an external **VDD** port. The expansion connector will automatically disconnect, and a warning message will be displayed if **Emulation** or **Test mode** operations for a chip with applied external voltage are running.

After all steps are completed, the debugging controls become active.

3.1.5.2 GreenPAK Serial Debugger Board

- Connect a socket with an external chip to a development board through the ECs:
  - Corresponding **I2C** pins (**SCL**, **SDA**).
  - **VDD** pin.
  - **GND** pin.
- Connect a chip with wires to the **Serial Debugger** pins (**VDD**, **CLK**, **SDA**, **GND**):
- Start Go Configure Software Hub and select the part number of the connected external chip.
- Start the **Debug tool** and select the GreenPAK Serial Debugger Board.
- Select `0001b` for the **Device address** (used by default for an empty chip) or a corresponding address programmed to the chip.

- Specify **VDD** configuration:
  - **Internal** – Chip is powered from a GreenPAK Serial Debugger Board.
  - **External** – Chip is powered from an external power source.

When all steps are completed, the debugging controls become active.

3.2 ForgeFPGA Devices

3.2.1 ForgeFPGA Deluxe Development Boards (SLG7DVKFORGE)

- Programming and emulation for ForgeFPGA devices.
- 80 digital configurable Test Points (TPs):
  - 64-channel high-speed digital generators (up to 200 MS/s).
  - Programmable software button (VDD, GND, Hi-Z).
- 32 configurable constant voltage sources (8 in Rev. 1.0).
- 64-channel digital Logic Analyzer (up to 200 MS/s).
- Dual PCIe connector for socket adapter.
- USB 3.0 interface for communication (USB-C connector).
- External 12V power supply.

Note: The ForgeFPGA Deluxe Development Board Rev. 2.0 is now accessible alongside the platform revision described above. Be sure to select the appropriate platform revision when working with **Debug tools**. Preset configurations can be loaded between the ForgeFPGA Deluxe Development Board and ForgeFPGA Deluxe Development Board Rev. 2.0 (currently excluding **Logic Analyzer** settings) using the **Import configuration** option in the **Debugging Controls**.

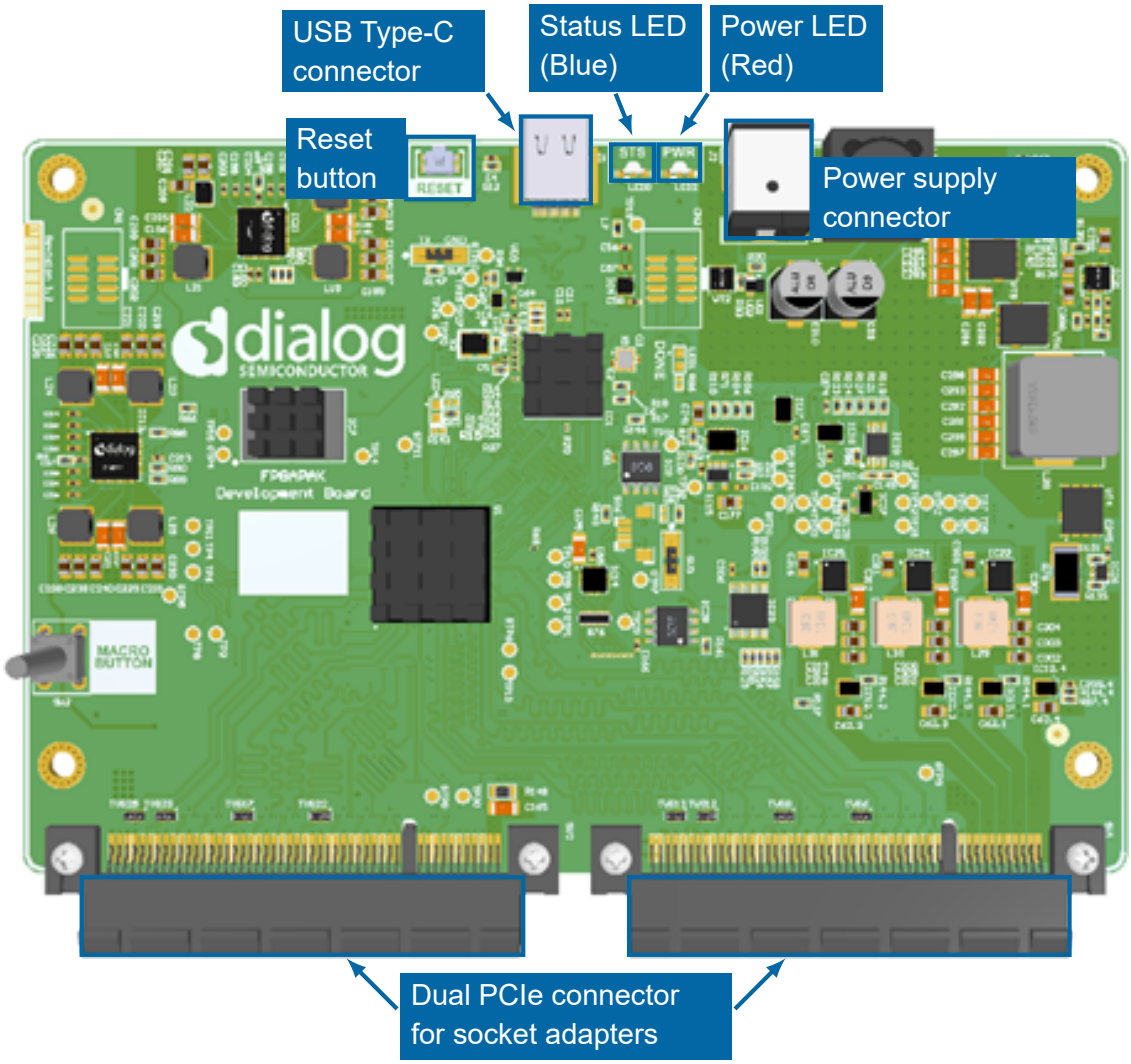


Figure 331. ForgeFPGA Deluxe Development Board

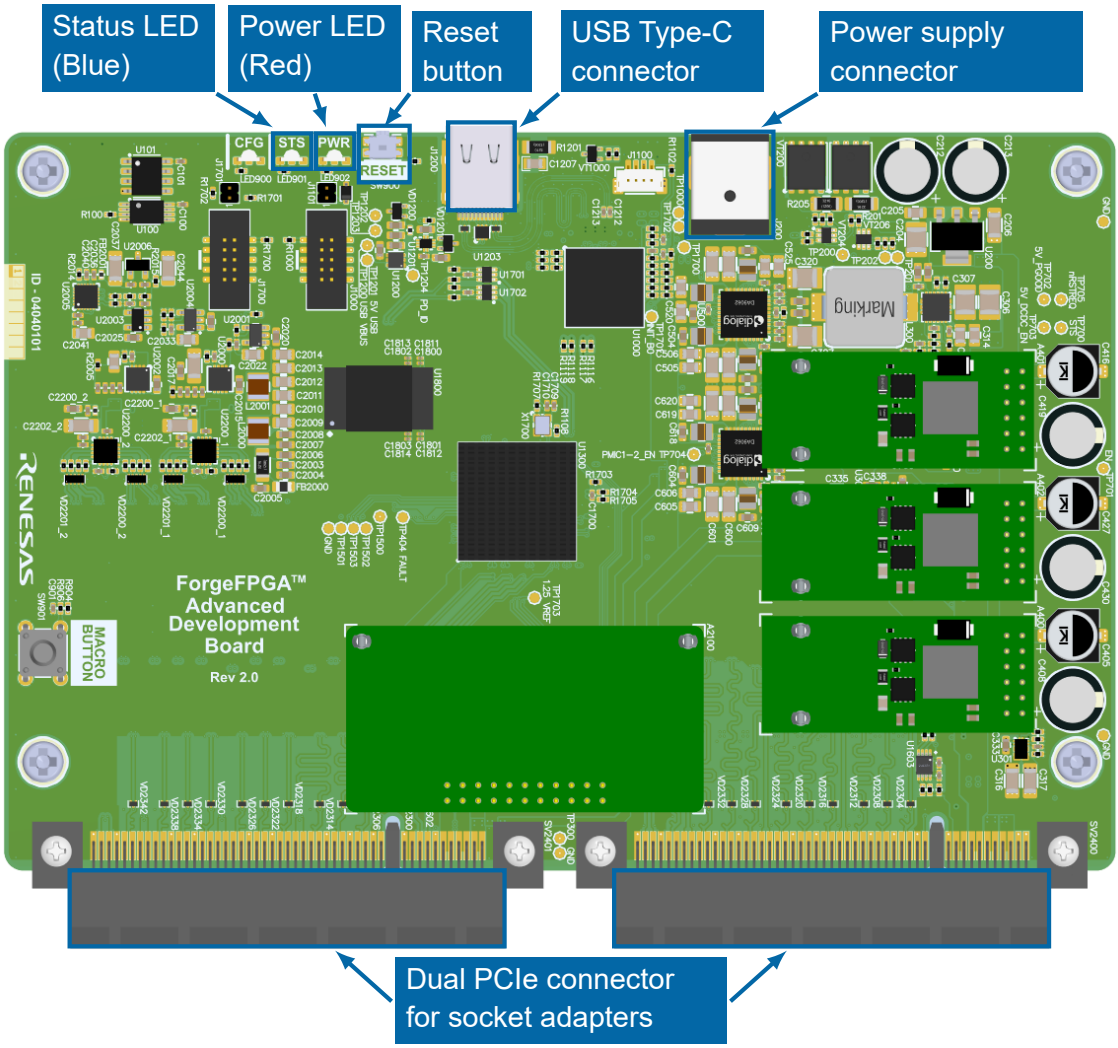


Figure 332. ForgeFPGA Deluxe Development Board Rev. 2.0

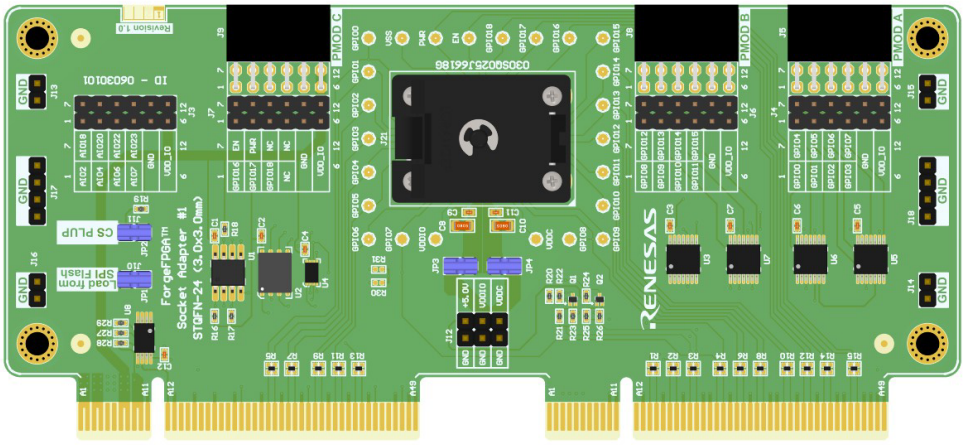


Figure 333. ForgeFPGA Socket Adapter

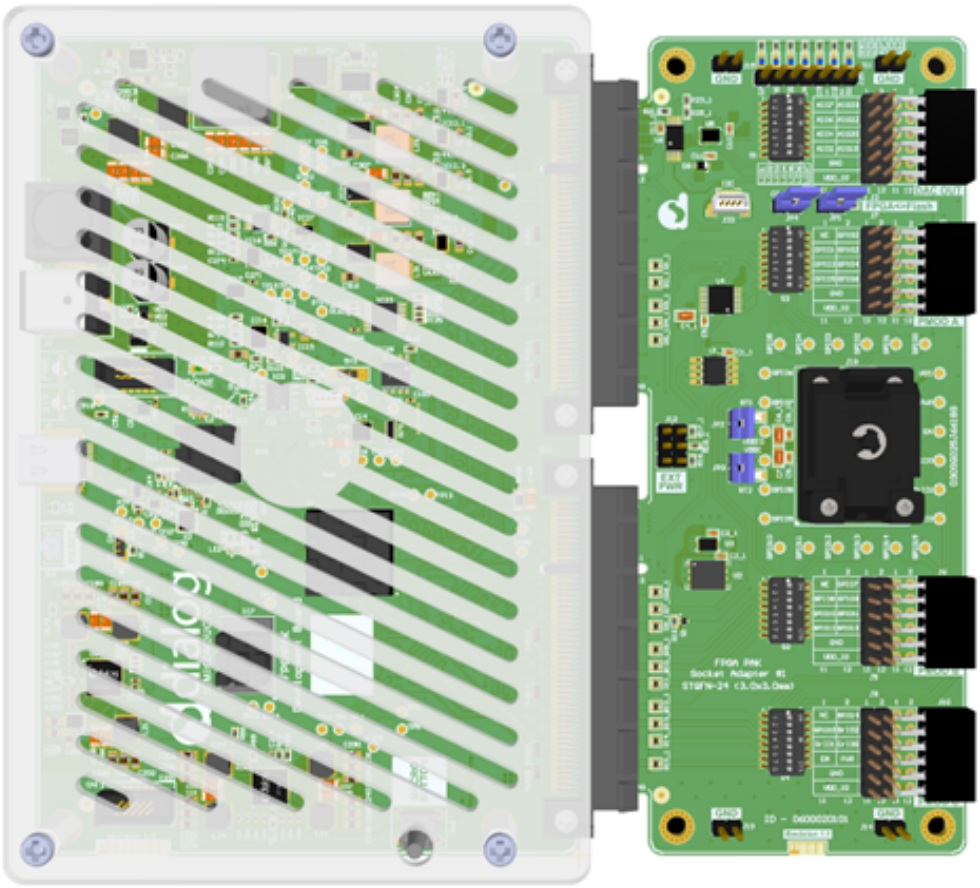


Figure 334. Assembled Equipment for Working with a Chip

The ForgeFPGA Deluxe Development Board **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Emulation
- Test Mode
- Read
- Program
- Test Mode* (flash)
- Read (flash)
- Program (flash)
- Erase (flash)
- Generator controls
- TP map
- Socket controls
- External Bitstream
- Info details

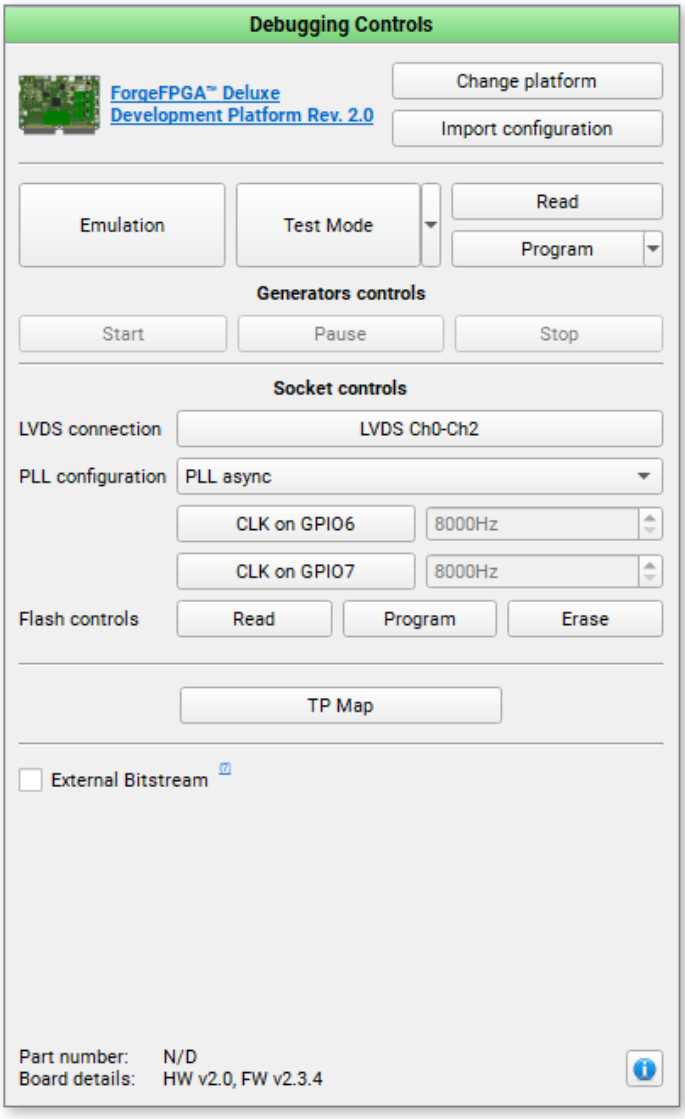


Figure 335. ForgeFPGA Deluxe Development Board Controls

Find a comparison of the controls available on different FPGA boards in the appendix, section 6.3 Debugging Controls Feature Availability.

3.2.2 ForgeFPGA Evaluation Board (SLG7EVBFORGE)

- Programming and emulation for ForgeFPGA devices.
- PMOD connectors.
- UART connector for USB UART bridge mode.
- USB interface for power and communication (USB-C connector).

Note: There are two variations of ForgeFPGA Evaluation Board: version 2.0 and 1.0. ForgeFPGA Evaluation Board v.1.0 is a simplified version of the platform and does not support chip programming. Since the socket is absent, the SLG47910 IC is soldered directly on the board.

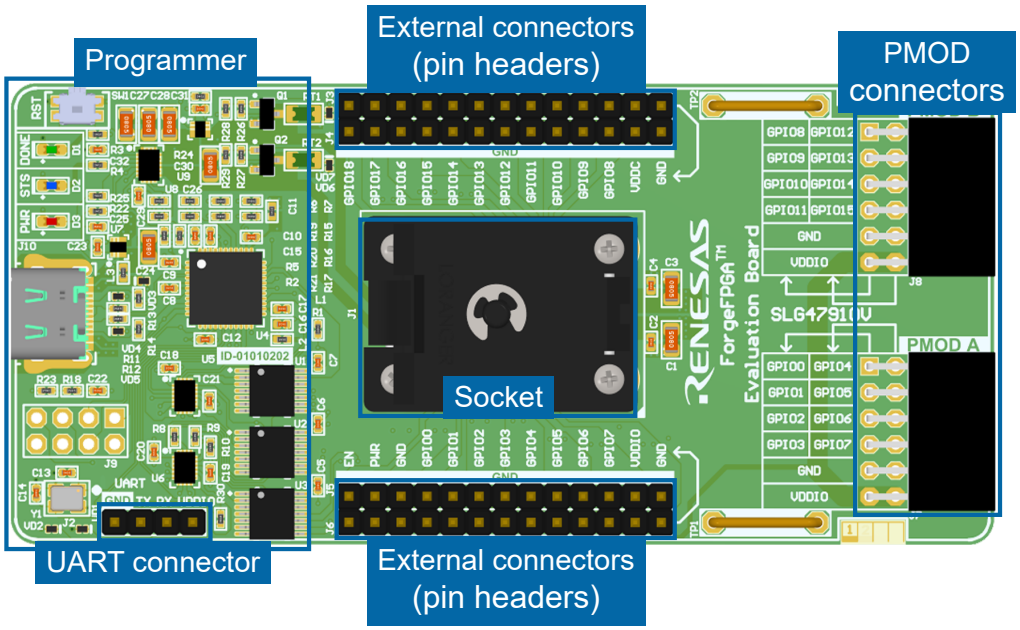


Figure 336. ForgeFPGA Evaluation Board 2.0

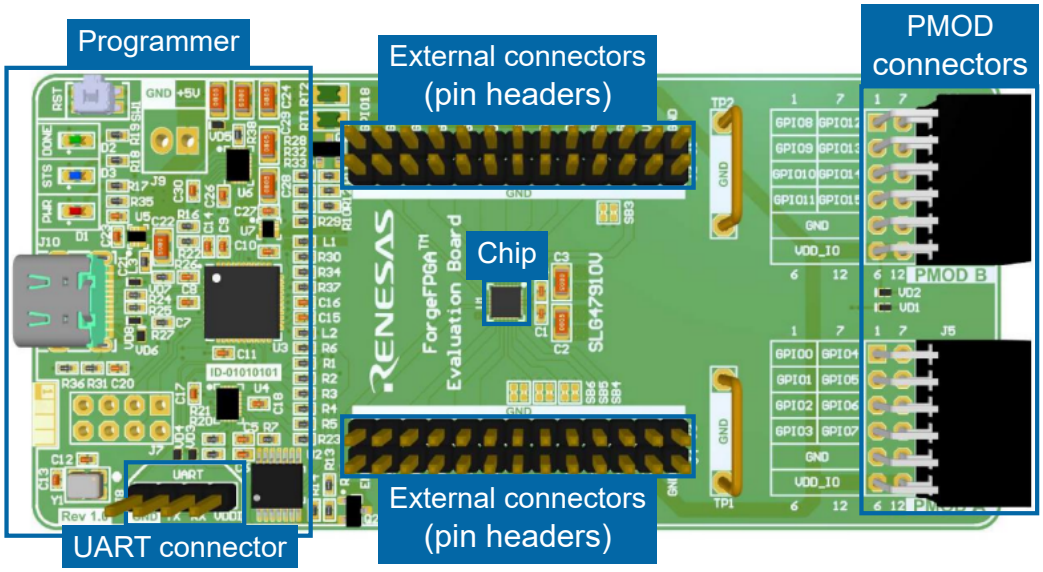


Figure 337. ForgeFPGA Evaluation Board 1.0

The ForgeFPGA Evaluation Board **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Emulation
- Test Mode
- Read
- Program
- Voltage level controls
- Info details

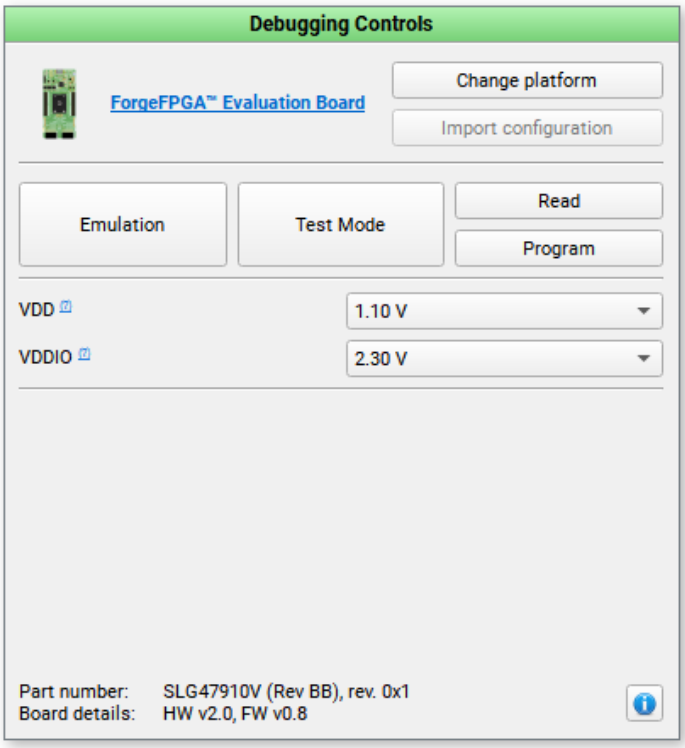


Figure 338. ForgeFPGA Evaluation Board Controls

Find a comparison of the controls available on different FPGA boards in the appendix, section 6.3 Debugging Controls Feature Availability.

3.3 Power GreenPAK devices

3.3.1 Power GreenPAK Development Motherboard (SLG5PGPDM)

- Programming and emulation for SLG51000, SLG51001, SLG51002, and SLG51003 Power GreenPAK devices.
- Individually configurable Test Points (TPs):
  - Onboard LED state indication.
  - Programmable software button (VDDIO, GND, Hi-Z).
  - Pull Up and Pull down jumpers.
  - Floating hooks on each TP for oscilloscope connection.
- Internal (onboard DC modules) and external power supply (TIP connectors) for IC's LDOs Vin; VDD.
- Voltage measurement for chip's LDOs Vin/Vout channels.
- USB interface for communication (USB-C connector).
- External 12V power supply.
- In-System Debugging (ISD) functionality.

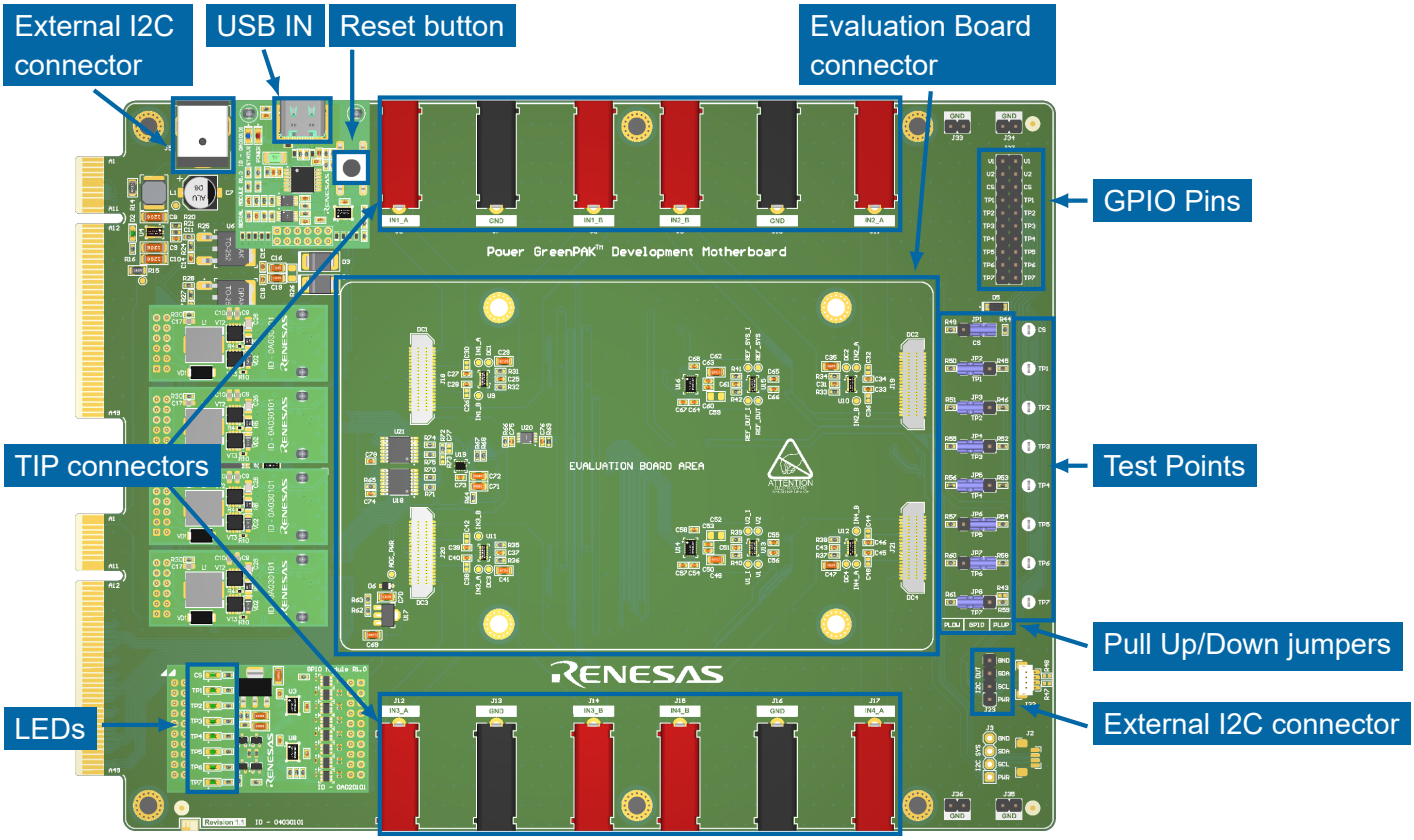


Figure 339. Power GreenPAK Development Motherboard

The Power GreenPAK Development Motherboard **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Device selector
- Emulation
- Sync
- Test Mode
- Voltage controls
- LEDs ON and LEDs OFF
- TP map
- Info details

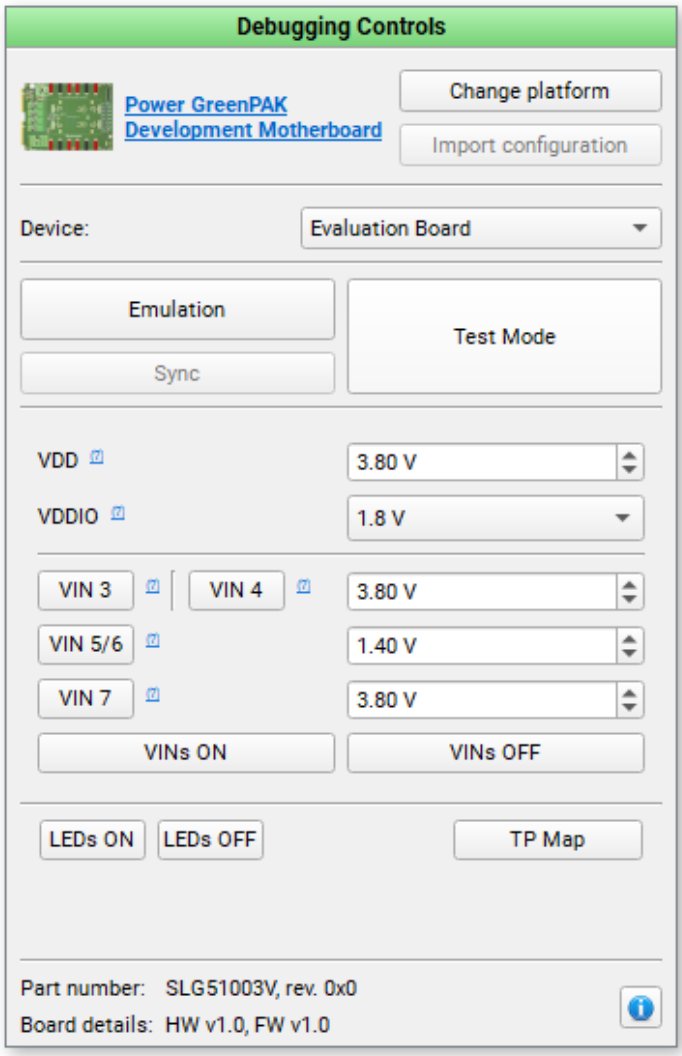


Figure 340. Power GreenPAK Development Motherboard Controls

Find a comparison of the controls available on different **Power GreenPAK** boards in the appendix, section 6.3 Debugging Controls Feature Availability.

3.3.2 Power GreenPAK SLG51002C Demo Board

- Programming and emulation for SLG51002.
- Individually configurable Test Points (TPs):
  - Programmable software button (VDDIO, GND, Hi-Z).
  - Pull Up jumpers.
  - Floating hooks on each TP for oscilloscope connection.
- SMB connectors for measurement LDO parameters.
- USB interface for communication (USB-C connector).
- External 12v power supply.

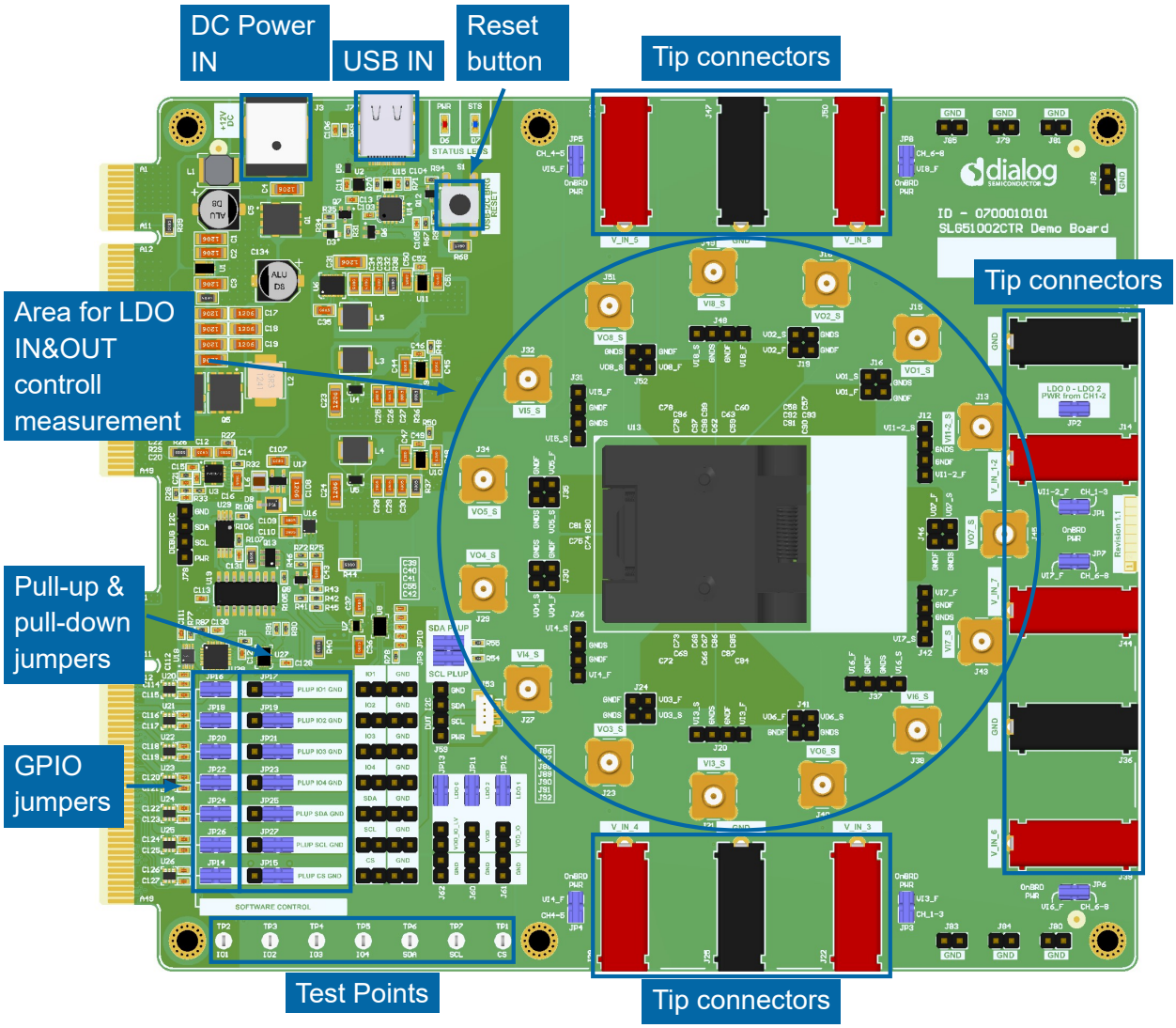


Figure 341. Power GreenPAK SLG51002C Demo Board

The Power GreenPAK SLG51002CTR Demo Board **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Emulation
- Sync
- Test Mode
- GPIO SW Control
- Voltage controls
- Info details

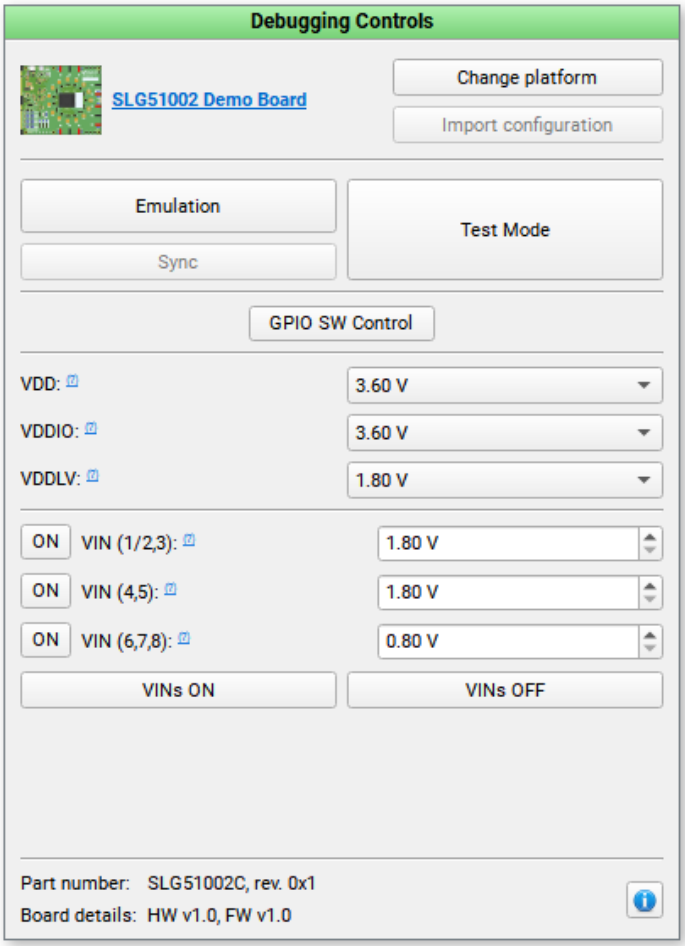


Figure 342. Power GreenPAK 002 Demo Board Controls

Find a comparison of the controls available on different **Power GreenPAK** boards in the appendix, section 6.3 Debugging Controls Feature Availability.

3.3.3 Connecting External Power GreenPAK

3.3.3.1 GreenPAK Serial Debugger Board

GreenPAK Serial Debugger Board can be used for external chip debugging of Power GreenPAK part numbers.

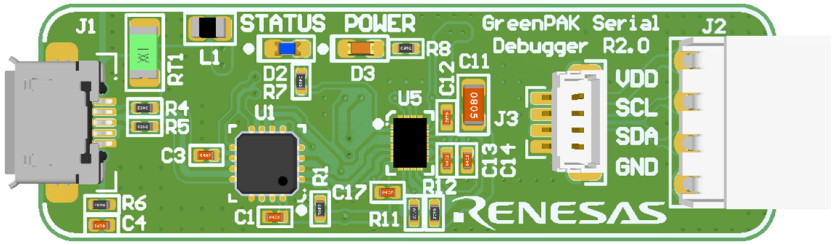


Figure 343. GreenPAK Serial Debugger

To start working with an external chip on the GreenPAK Serial Debugger Board:

- Connect a chip with wires to the **Serial Debugger** pins (**CLK**, **SDA**, **GND**). When using SLG5100xCRT Socket Eval., use the DUT I2C connector, as shown in the picture GreenPAK Serial Debugger Board with Power GreenPAK socket.
- Ensure the Evaluation board's CS, VDD, and VDDIO pins are connected to the external power supply.
- Start the Go Configure Software Hub and select the part number of the connected external chip.
- Start the **Debug tool** and select the GreenPAK Serial Debugger Board.
- Select **I2C Device address** or the corresponding address programmed to the chip.
- Specify the **I2C Voltage Level** configuration.
- Power up the external power supply inputs.

After all steps have been completed, the debugging controls become active.

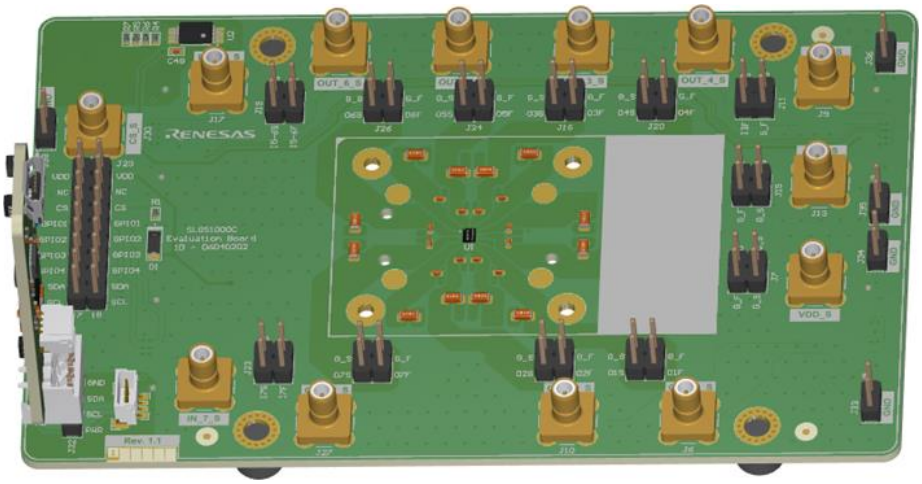


Figure 344. GreenPAK Serial Debugger Board with Power GreenPAK Socket

Debugging Controls interface for the GreenPAK Serial Debugger Platform (with SLG5100x device) includes:

- Debug Configuration
- Device selector
- Emulation
- Sync
- Test Mode
- Voltage level controls
- Info details

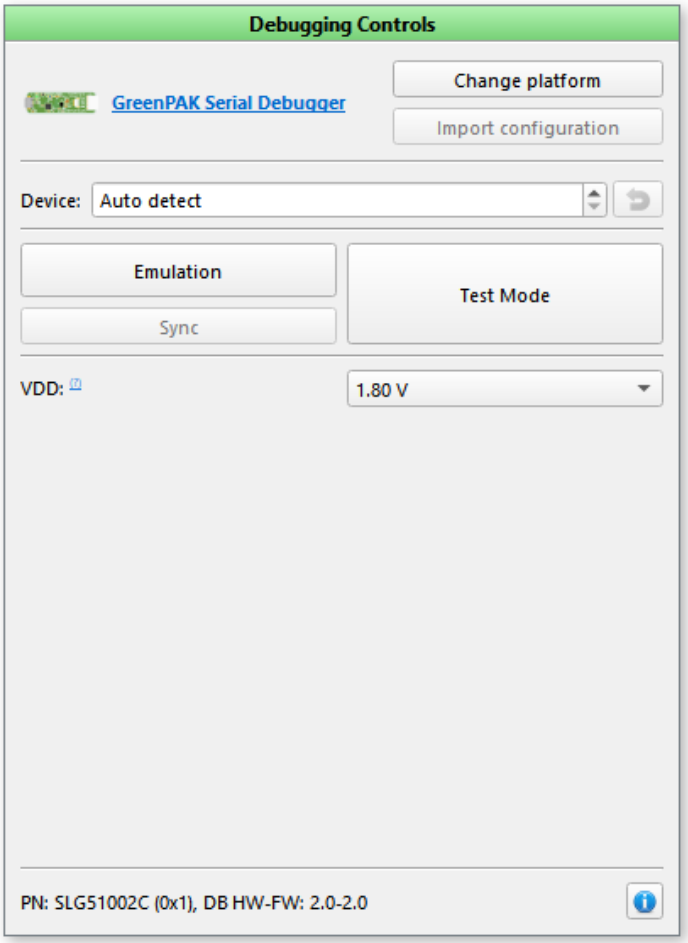


Figure 345. Power GreenPAK Serial Debugger Controls

Find a comparison of the controls available on different Power GreenPAK boards in the appendix, section Debugging Controls feature availability.

3.4 Go Configure Development Board (SLG4DVKGOCONF)

- Programming and emulation.
- 80 multi-function configurable I/O lines Test Points (TPs):
  - 32 channels Digital Pattern Generator.
  - 16 channels Analog Waveform Generator (AWG).
- Dual PCIe connector.
- Flexible power management with software-configurable protections (OVP, OCP).
- USB 2.0 interface for communication (USB-C connector).

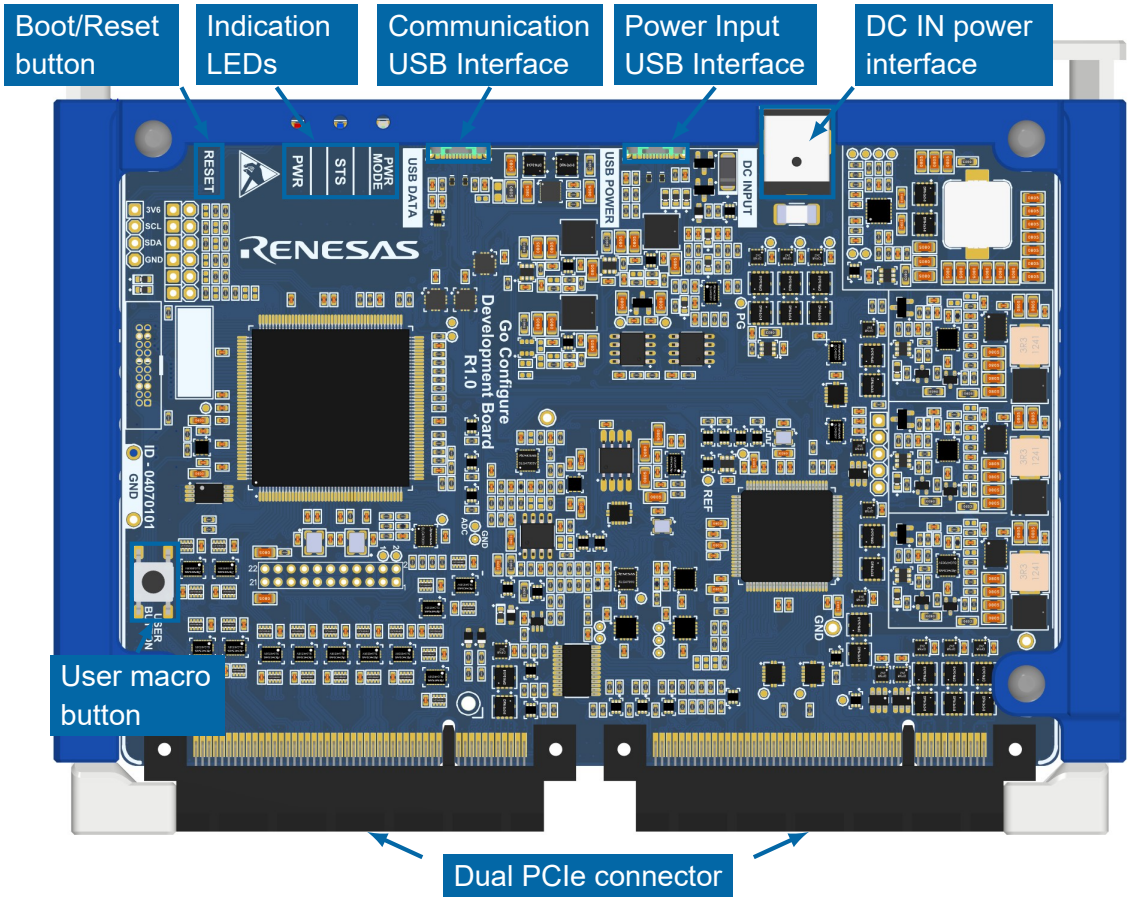


Figure 346. Go Configure Development Board

The **Go Configure Development Board** supports external power supply connection. Read more in section 3.1.5 Connecting External GreenPAK.

When used with a GreenPAK device, the Go Configure Development Board **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Device selector
- I2C Reset
- Emulation
- Emulation(sync)
- Test Mode
- Read
- Program
- NVM Programmer window
- Generator controls
- Power source selector
- TP map
- LEDs ON and LEDs OFF
- Info details

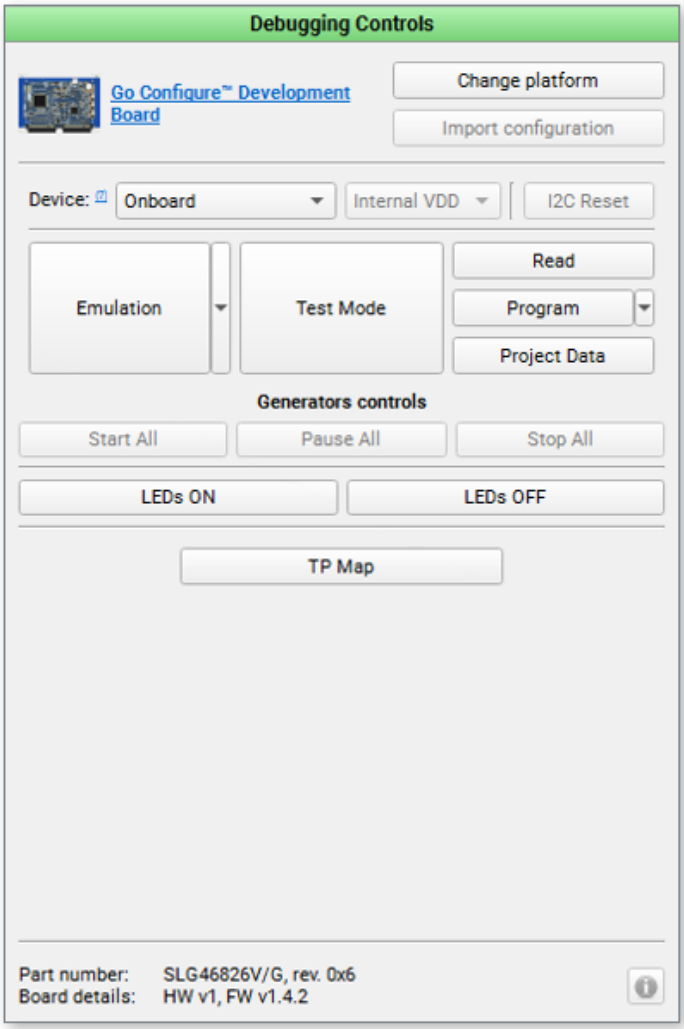


Figure 347. Go Configure Development Board with GreenPAK Device

When used with ForgeFPGA device, the Go Configure Development Board **Debugging Controls** User Interface includes the following sections:

- Debug Configuration
- Emulation
- Test Mode
- Read
- Program
- Test Mode* (flash)
- Read (flash)
- Program (flash)
- Erase (flash)
- Generator controls
- TP map
- Socket controls
- External Bitstream
- Info details

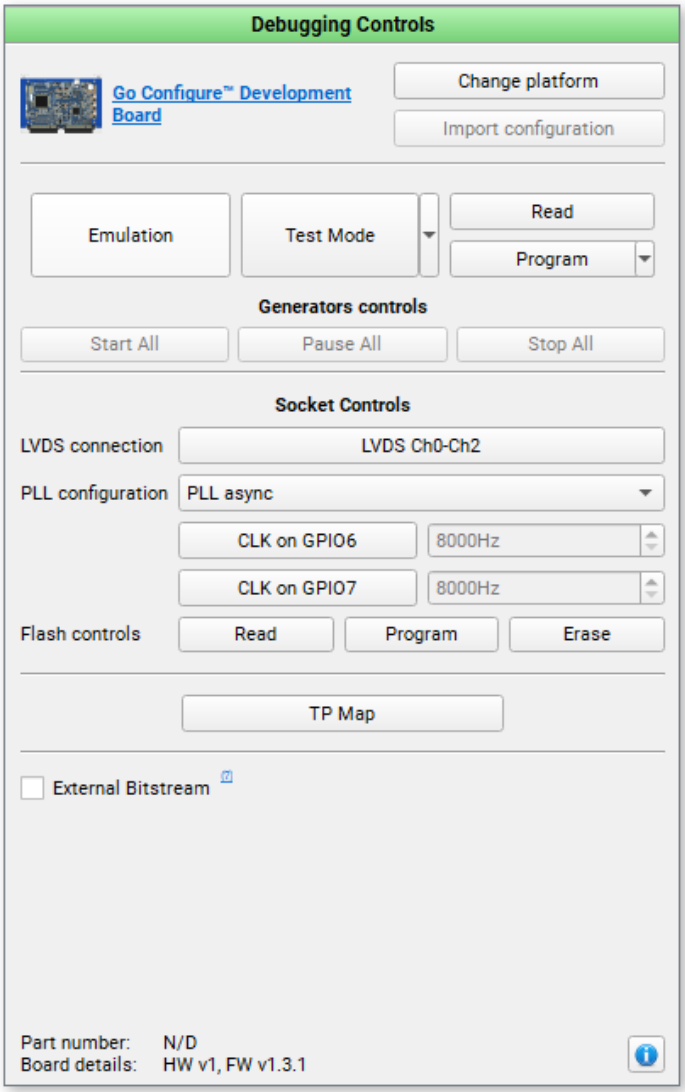


Figure 348. Go Configure Development Board with ForgeFPGA Device

3.5 Demo devices

Demo boards are hardware platforms intended for evaluating demonstration projects (read more in section 2.2.18 Demo Board and Demo Mode). Each board includes a soldered IC preprogrammed with a demo design, enabling interaction with the device's functionality. Certain demo boards can also operate as development platforms, providing capabilities similar to those described earlier in this chapter. This section provides an overview of this platform type.

3.5.1 HVPAK SLG47125 Demo Board

- Soldered SLG47125V device.
- Regulated DC-DC converter (IC3), which powers the high voltage part (VDD2).
- Multipurpose encoder and OLED display for board settings highlight.
- BLDC Motor load.
- Power RGB LED load.
- USB interface for power and communication (type-C connector).

Note: The **HVPAK SLG47125 Demo Board** supports two operating configurations: **BLDC motor load** and **RGB LED load**. Switch between the modes on the **Development Platform Selector**.

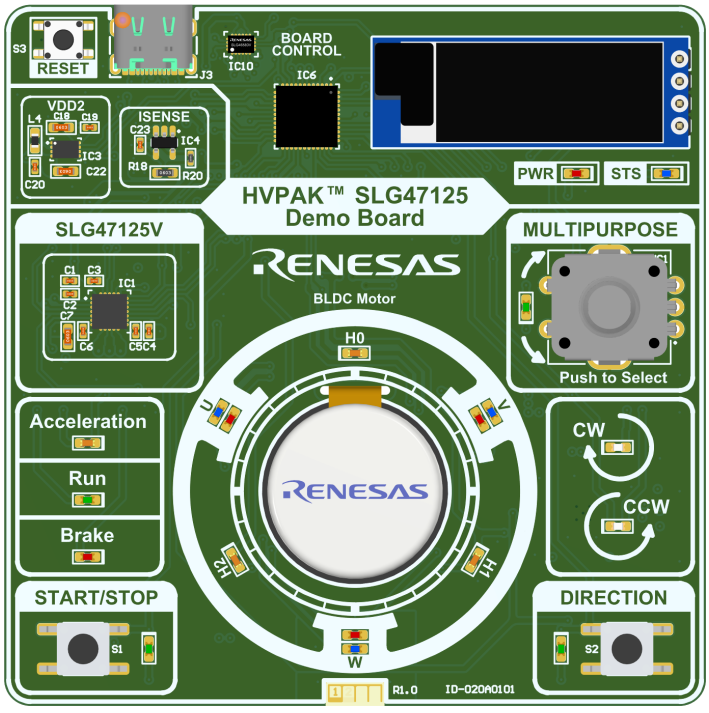


Figure 349. HVPAK SLG47125 Demo Board

The HVPAK SLG47125 Demo Board (BLDC Motor load) **Debugging Controls** User Interface includes the following sections:

Debug Configuration	Write (sync)	Initial design
VDD2	Voltage level controls	Info details
Write	Motor measurements	

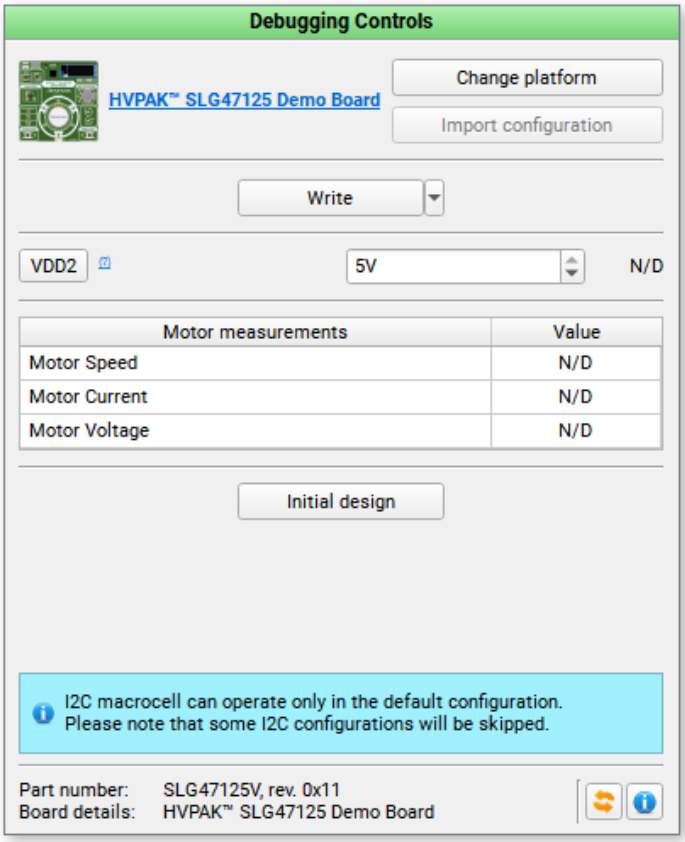


Figure 350. HVPAK SLG47125 Demo Board (BLDC Motor Load)

The HVPAK SLG47125 Demo Board (RGB LED load) **Debugging Controls** User Interface includes the following sections:

Debug Configuration	Write (sync)	Initial design
VDD2	Voltage level controls	Info details
Write	Color picker	

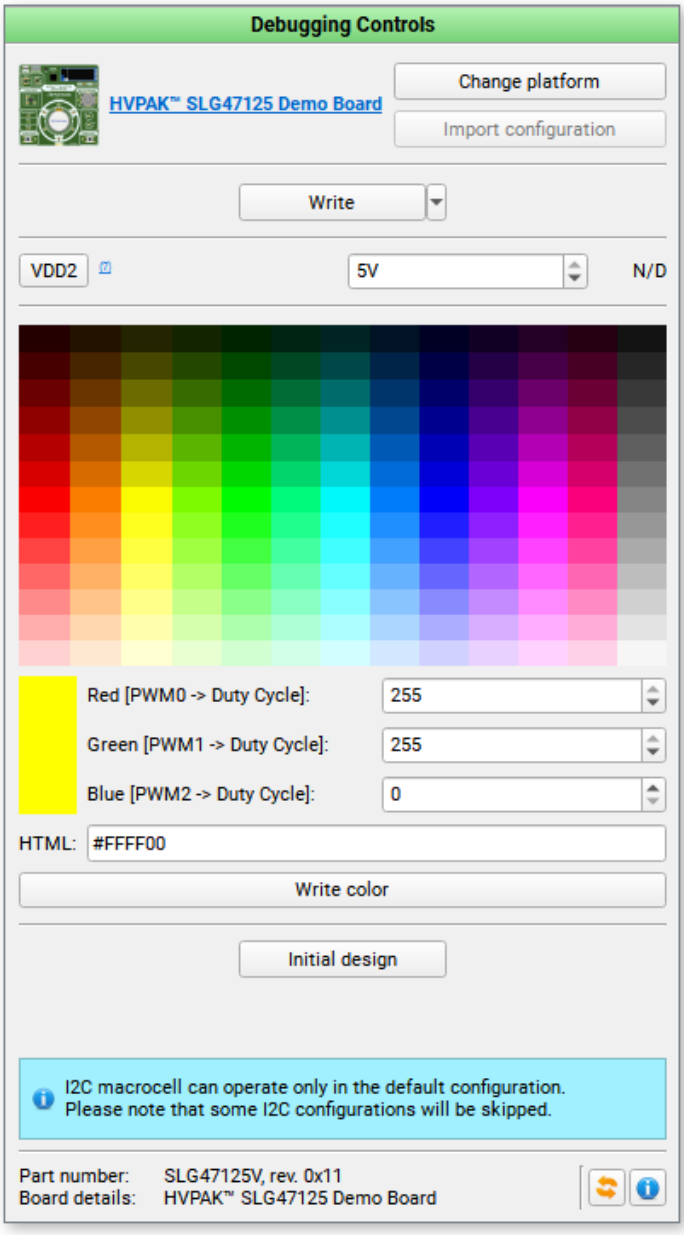


Figure 351. HVPAK SLG47125 Demo Board (RGB LED Load)

4 How To

In addition to facilitating chip configuration, the Go Configure Software Hub offers a range of features designed to improve workflow efficiency. While these features may not be directly tied to a specific project, they can help streamline various aspects of the work process.

This chapter presents methods for keeping the software up to date and introduces various approaches to updating it. It also explains how to create and save block diagrams, provides debug tool-related instructions, and describes the integration of external software into a project.

4.1 Software Update

Software updates improve the user experience by addressing issues and offering new features. Updates can provide access to new templates, resources, or content libraries that enhance the creative process.

There are two ways to update the Go Configure Software Hub:

- When updates are available, a notification appears. The latest version can either be downloaded immediately or the update can be postponed until the program restarts. After the download is complete, the Go Configure Software Hub handles the automatic installation of the software. *Note:* It is important to restart the software after the installation process is complete.

- The latest version of the Go Configure Software Hub can also be found on the Software page on the Renesas website. To ensure the best user experience, keep the software up to date.

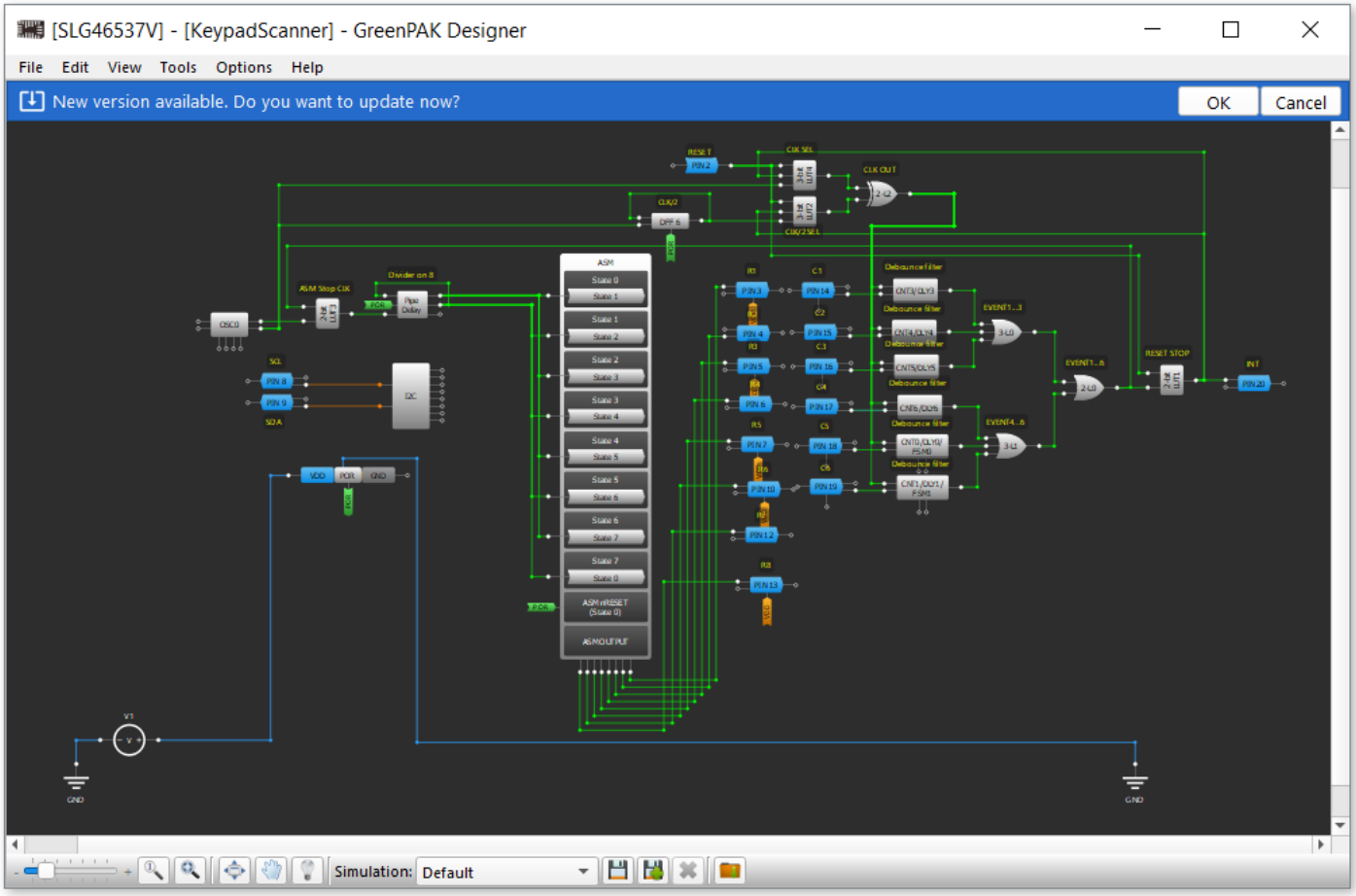


Figure 352. Software Update Notification

The **Updater** preferences can be adjusted in the **Designer Settings** window, **Updater** section (refer to Section 2.1.15 Settings).

Suggestions for updates can be emailed to the developer team to help improve the software (click **Help** > **About Go Configure Software Hub**).

4.2 Printing

4.2.1 Print

The **Print tool** can be used to present and save a block diagram in a graphic format for inclusion in datasheets or other project documentation showing the interconnections between blocks and their configurations. The **Print** tool is available on the toolbar.

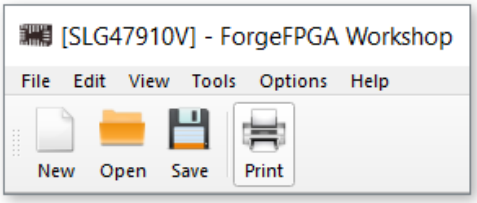


Figure 353. Print Tool on the Toolbar

The **Print tool** can also be opened by clicking **File** > **Print** in the main menu.

To open the **Print Preview** window, click the **Print** button in the top toolbar. This window shows a ready-to-print diagram and tables with block configurations. At this point, the positions of elements or lines on the diagram cannot be changed. To reposition components or wires, return to the working area, make the required changes, and then launch the **Print Tool** again.

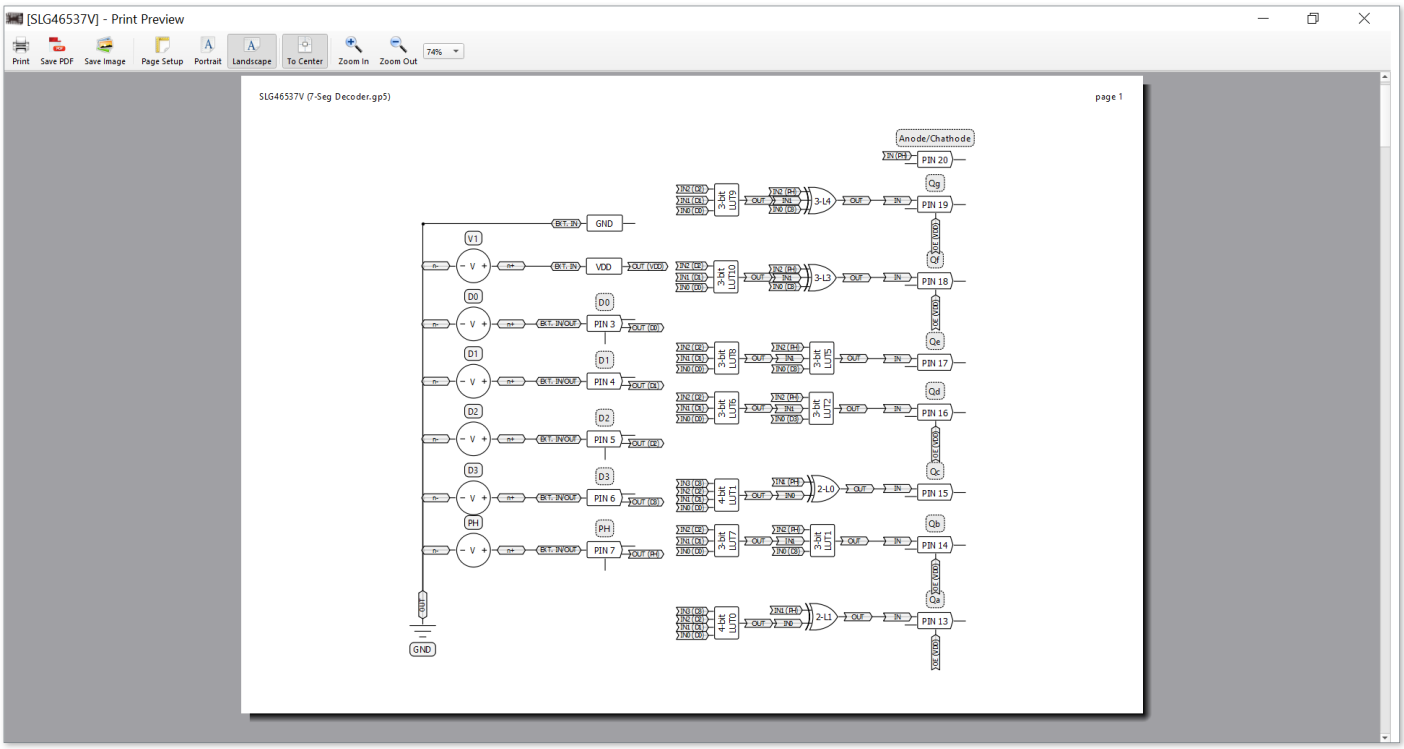


Figure 354. Print Preview Window

Additionally, the **Print Preview** window contains a table with a list of all blocks and their configurations.

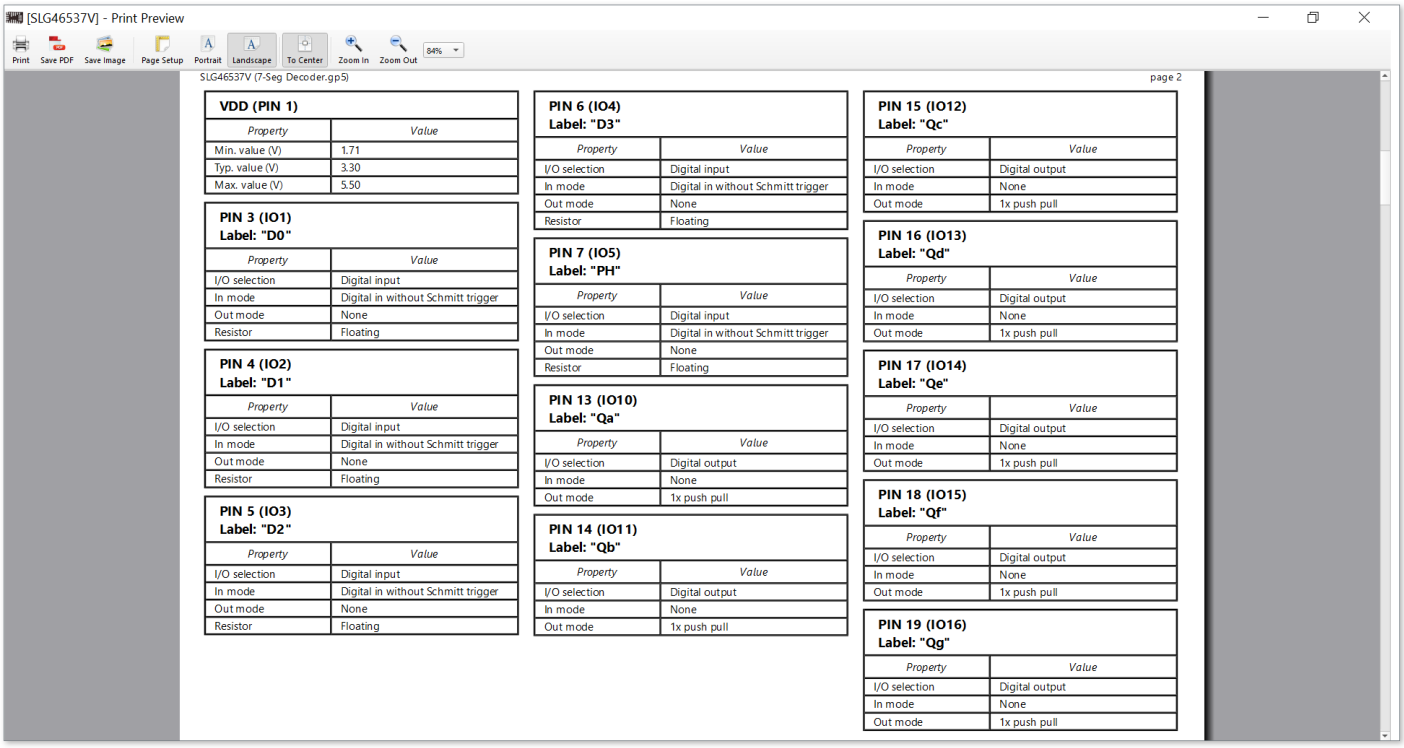


Figure 355. Table with Block Properties and Values in Print Preview Window

4.2.2 Print Editor (Obsolete)

In older versions of the software, the **Print editor (obsolete)** is available. This tool can be found by clicking **File** > **Print Editor (obsolete)**.

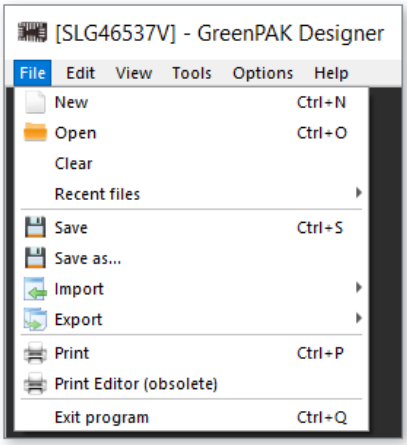


Figure 356. Print Editor (Obsolete) in Main Menu

The **Print Editor** allows pre-print configuration such as rotating, flipping, and hiding selected components or adding text labels.

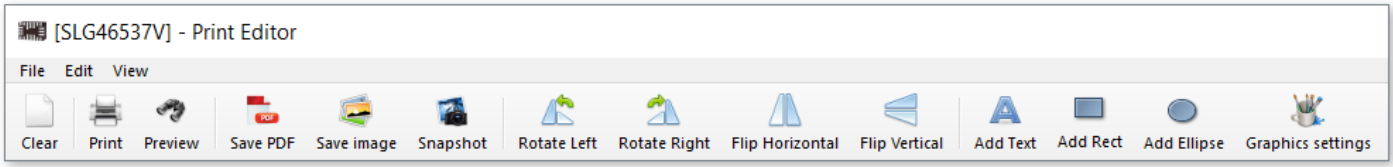


Figure 357. Print Editor Toolbar

The editable working area of the **Print Editor (obsolete)** contains all components used in the design. It allows customization of component positions, component views, and wires.

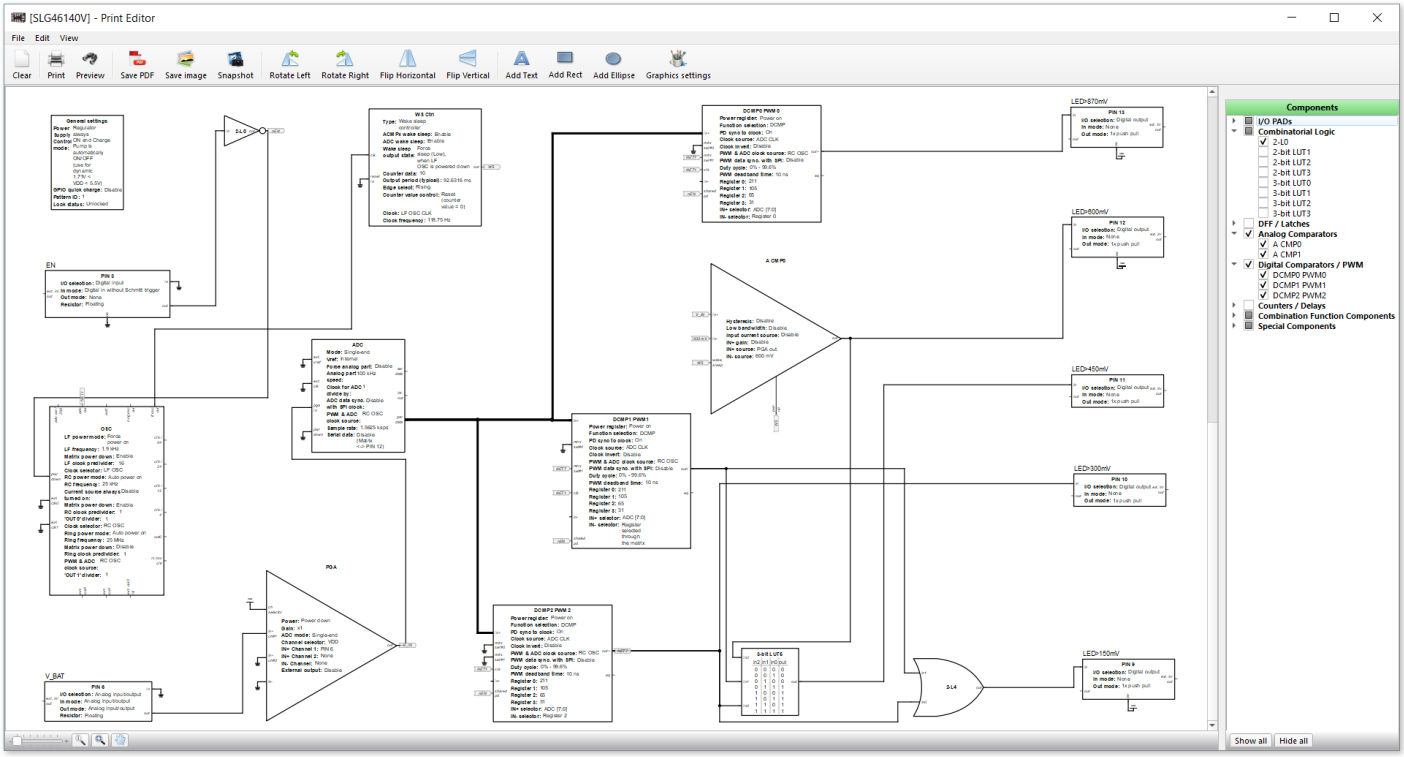


Figure 358. Print Editor (Obsolete) Interface

4.3 Icarus Verilog and GTKWave Quick Start

The Icarus Verilog and GTKWave software serve as simulation and simulation result visualization tools for validating Verilog design code using a testbench. Within the testbench, the **Unit Under Test (UUT)** or **Device Under Test (DUT)** can be instantiated and input combinations can be applied systematically. The resulting outputs can then be observed and analyzed using GTKWave.

4.3.1 Installation

Install the latest version of Icarus Verilog (iVerilog) from here. The download package includes both Icarus Verilog for simulation and GTKWave for visualizing simulation results. It is important to install both components to ensure proper functionality. Add iVerilog and GTKWave to the system paths during the installation process. If this step is overlooked, the paths can be added later by navigating to **FPGA Editor** > **Options** > **Settings** > **Tools**.

Note: If the simulation process fails to launch, this may occur due to an incorrect tool path specified in **Settings**. Try using the **Autodetect** feature to resolve the issue.

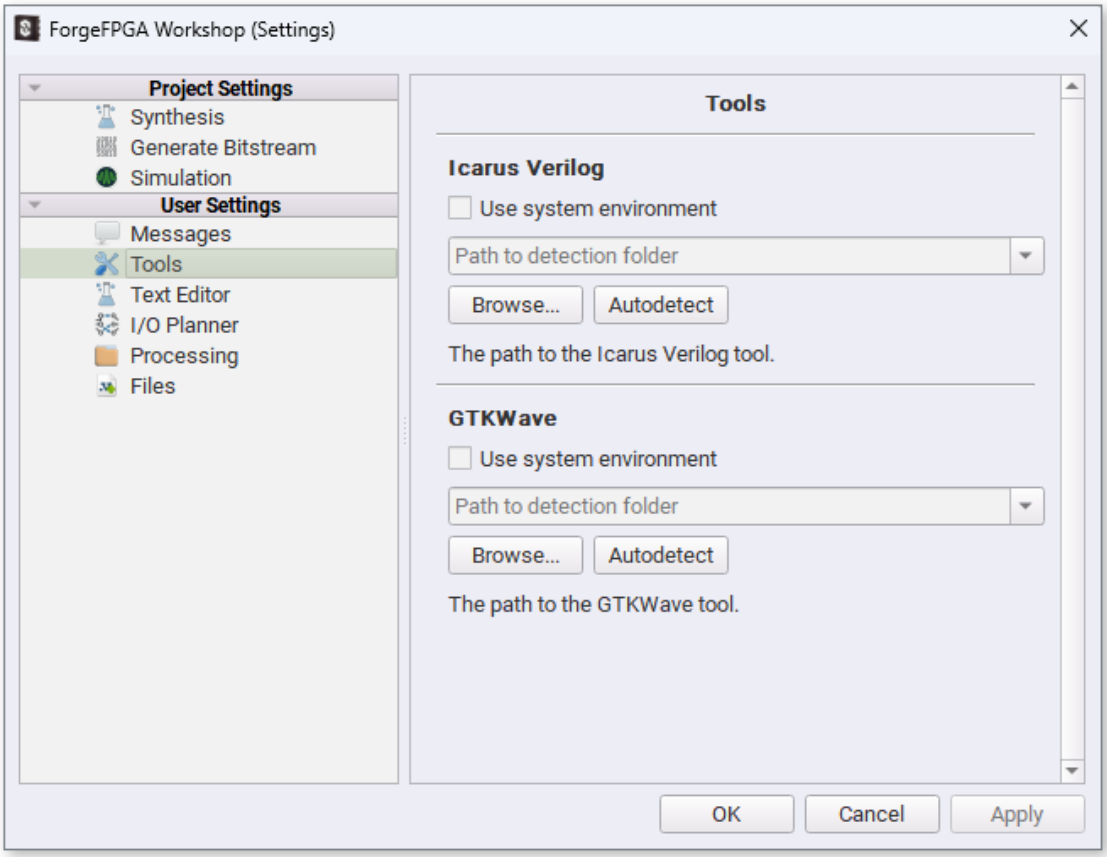


Figure 359. FPGA Editor Settings

4.3.2 Quick Start

Below are some useful tips for using the Icarus Verilog and GTKWave software within a ForgeFPGA Workshop workflow.

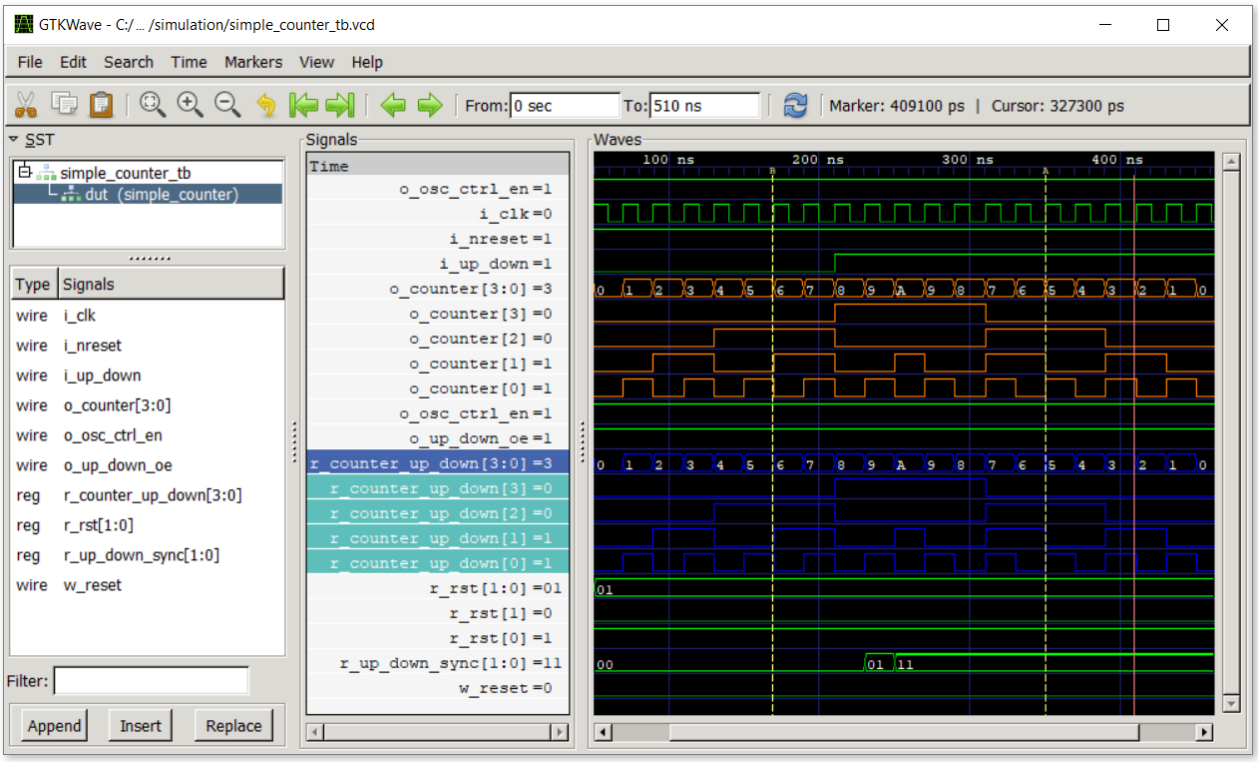


Figure 360. Simple Counter in GTKWave Software

- Access GTKWave either by clicking **Simulate Testbench** in ForgeFPGA Workshop or by entering `gtkwave` in the terminal. If the software is launched externally, select the .vcd files. These files serve as dump files generated by Icarus Verilog launched by ForgeFPGA Workshop when running a testbench (simulation).

- Generate a .vcd file. When creating a custom testbench, ForgeFPGA Workshop automatically inserts two lines for `$dumpfile` (creates dump files) and `$dumpvars` (assigns various values to signals in the design) in the `initial` block of the template code. These lines are tailored to the module's name.

```
1  // Custom testbench
2
3  `timescale 1ns / 1ps
4
5  module ALU16_tb;
6
7    initial begin
8
9      $dumpfile ("ALU16_tb.vcd");
10     $dumpvars (0, ALU16_tb);
11
12   end
13
```

```
14  endmodule
```

The `initial` block is executed only once. Above the `initial` block are the declarations of input and output signals, along with the module instantiation.

- Start the simulation by clicking the **Simulate Testbench** button in ForgeFPGA Workshop. This action triggers GTKWave to automatically open the corresponding .vcd file generated by Icarus Verilog. When the **Wave** window launches, no waveforms are initially displayed, allowing selection of the specific signals whose waveforms are to be viewed.
- Select the desired signal by navigating to the **SST** window, which shows the hardware hierarchy. The signals associated with a specific instance can be revealed in the bottom section. Either drag and drop the preferred signal or double-click it to display it in the **Signals** window. Alternatively, all signals can be selected (**Ctrl** + **A**) and inserted. To observe the corresponding signal values, click the **Reload** button on the toolbar.
- Customize signal properties in the context menu of a signal by selecting **Data Format** or **Color Format**. By default, signal values are shown in hexadecimal format, and all waves are colored green (if the simulation is running correctly). Additionally, a **Blank** signal can be inserted to create sections between groups of signals.
- After the desired visual configuration has been achieved, save the settings by navigating to **File** > **Write Save File**. Saving the configuration ensures that when the same testbench is simulated again, most previous adjustments are restored automatically.

For additional details and features, please consult the complete *GTKWave and Icarus Verilog user guides*.

4.4 Chip Memory Lock Configuration

Follow the steps below to lock the project data:

1. Open the **Project Settings** window from the toolbar. For projects, where all operating conditions in specs are not filled in, the **Project Settings** window appears automatically when the project launches.

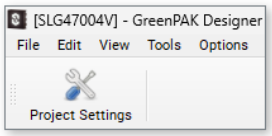


Figure 361. Project Settings on the Toolbar

2. Reach **Security** tab in the **Project Settings** window.

3. The settings may vary between different part numbers based on their specific characteristics, for example, available memory type (OTP, MTP, or EEPROM). See the main locking options based on the SLG47004V:

   - **Lock status** – Enables to lock RAM registers. See more information about **Lock status** modes clicking the **Detailed info** button on the **Project settings** window or access the Datasheets through the Hub window.

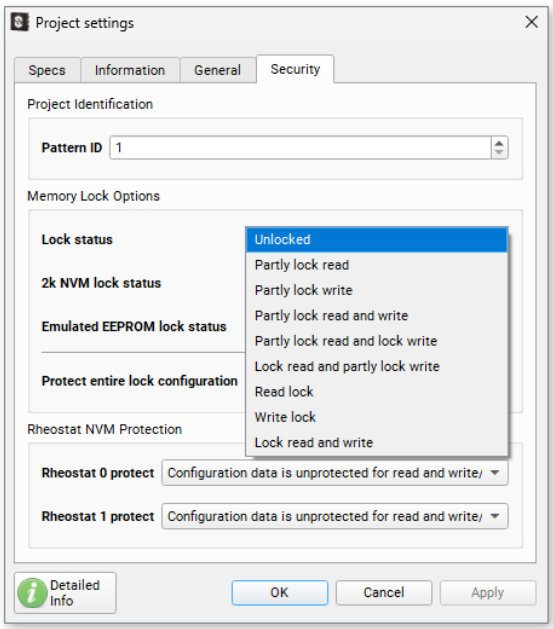


Figure 362. SLG47004V Lock Status Options

- **2k NVM lock status** – Allows to lock MTP (Multiple-Time Programmable) NVM.

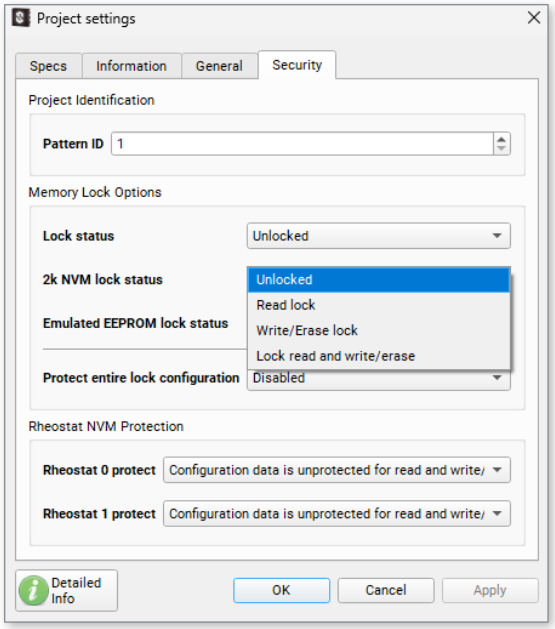


Figure 363. SLG47004V 2k NVM Lock Status Options

- **Emulated EEPROM lock status** – Locks EEPROM (Electrically Erasable Programmable Read-only Memory) memory type.

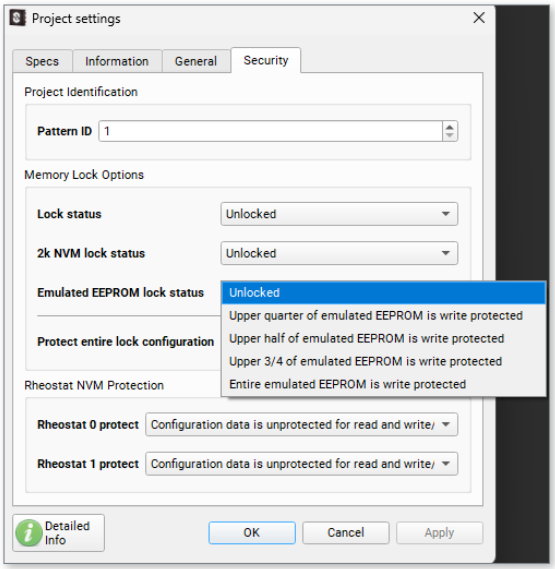


Figure 364. SLG47004V Emulated EEPROM Lock Status Options

- **Protect entire lock configuration** – Choose if all lock modes described above can be modified.

4. After the configurations are applied, all selected lock status modes will take effect during chip programming.

5 Troubleshooting

Challenges are an essential part of any creative process. Providing solutions is important in helping users overcome obstacles that may arise while using the software.

This chapter addresses common problems that may occur while using the software. It covers troubleshooting for issues such as technical glitches, error messages, and unexpected behavior. By following these steps, problems can be resolved more easily and the software can be used more effectively.

5.1 Failed Socket Test

- Ensure the contacts connecting a socket and chip pins are clean and undamaged.
- Disconnect any external circuits or signal sources (such as a generator or voltage supply) connected to a board or a socket itself. *Note:* External devices connected to expansions connectors do not affect the **Socket Test** procedure.
- Reconnect a board to a USB port.
- Use an empty chip, if possible.

  Note: Certain combinations of chip memory protection bits or specific I2C configurations on already programmed silicons may cause **Socket Test** to fail.

5.1.0.1 SLG47011V Evaluation Board

- **Socket Test failed on Test Point 10** > Check the PIN15/VREF jumper position. This jumper should connect PIN15 to Test Point 10. To pass **Socket Test**, please try the following configuration:

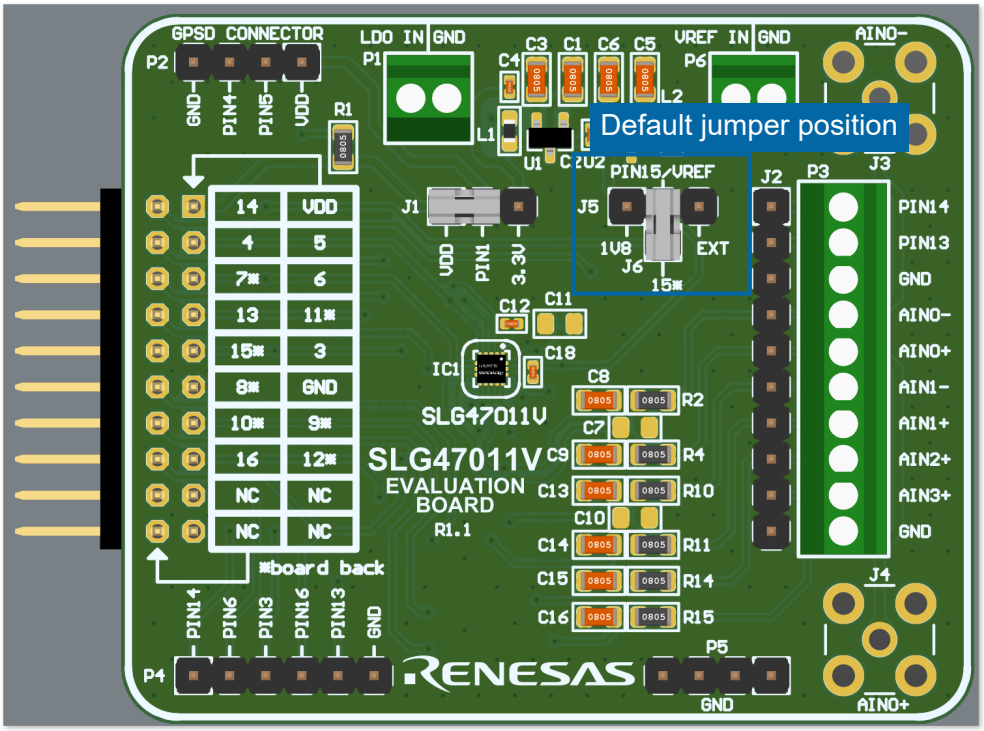


Figure 365. SLG47011V Evaluation Board, Default Pin15/Vref Jumper Position

6 Appendices

6.1 Main Menu Commands

File

New	Start a new or open existing project from Go Configure Software Hub.
Open	Open an existing project in software tools.
Save	Save current project.
Save as	Save the current project in a specified location.
Import NVM bits	Load configuration bits from a text file.
Export NVM bits	Save configuration bits to a text file.
Print	A simple print feature without detailed block information.
Print Editor (Obsolete)	Start the Print Editor.
Exit program	Close software tools.

Edit

Rotate Left	Rotate a selected block counterclockwise.
Rotate Right	Rotate a selected block clockwise.
Flip Horizontal	A horizontal reflection of a selected block.
Flip Vertical	A vertical reflection of a selected block.
Align Horizontal	Horizontal alignment of selected blocks.
Align Vertical	Vertical alignment of selected blocks.
Set Label	Creating a text label for selected blocks.
Erase Label	Erasing text labels near selected blocks.
Set Wire	Enable wire creating mode.
Erase Wire	Enable wire erase mode.

View

Zoom in	Increase the work area scale.
Zoom out	Decrease the work area scale.
Fit work area	Tune scale to show all blocks visible in the project.
Zoom 1:1	Set default scale.
Full-screen mode	Switch to full-screen mode.
Pan mode	Enable or disable the work area move in pan mode.
Show hints	Enable or disable hints for blocks on the work area.
Properties	Show or hide Properties panel.
Schematic Library	List of external components for Software Simulation.
Components	Show or hide software tools blocks list.
NVM Viewer	Show or hide NVM bits viewer.
Rule Checker Output	Scan the project for design errors

Tools

Debug	Convenient project testing.
Rule Checker	Check current design for correct settings.
Comparison	Compare bits of two projects.
ASM Editor	Configure the State Machine using state diagram and set the output configuration for SM Output block.
I2C Tools	Enhanced I2C Tools with I2C snapshot Reconfigurator.

Options

Settings	Set the default projects folder, autosave interval, toolbars position, recovery, shortcuts, and update options.

Help

Help	Show help window.

User Guides	Open User guides on the web.
Legend box	Show the color legend box.
Renesas web site	Open the Renesas official website.
Software and documentation	Open Software and Documentation web page.
Renesas web store	Open Renesas chip store.
Design support	Web page with training courses and videos.
Contact Us	Web form with request.
Social	Renesas in social networks.
Application Notes	Open examples web page.
Datasheet	Open documentation web page.
Updater	Open software update tool.
About Go Configure Software Hub	Show information about software tools version modification.

6.2 Keyboard and Mouse Controls

Basic tools		
Action	**Windows/Linux controls**	**macOS controls**
NVM Viewer	**F2**	**F2**
Component properties	**F3**	**F3**
Apply changes	**Ctrl + A**	**Command + A**
Revert changes	**Ctrl + U**	**Command + U**
Reset settings to default	**Ctrl + Shift + R**	**Command + Shift + R**
Reset connections to default	**Ctrl + Shift + C**	**Command + Shift + C**
Component List	**F4**	**F4**
Filter on the Component List	**Ctrl + F**	**Command + F**
Rule Checker	**F5**	**F5**
Debug tool	**F9**	**F9**

Components and connections		
Action	**Windows/Linux controls**	**macOS controls**
Move selected block(s) by 1 pixel*	**Alt + Left/Right**	**Alt + Left/Right**
Move selected block(s) by 10 pixels*	**Ctrl + Left/Right**	**Command + Left/Right**
Rotate selected component(s) left	**Ctrl + L**	**Command + L**
Rotate selected component(s) right	**Ctrl + R**	**Command + R**
Flip selected component(s) horizontally	**Ctrl + Shift + H**	**Command + Shift + H**
Flip selected component(s) vertically	**Ctrl + Shift + V**	**Command + Shift + V**
Hide selected component(s)	**H**	**H**
Remove selected external component(s)	**Del**	**Backspace**
Set wire	**Ctrl + W**	**Command + W**
Erase wire	**Ctrl + E**	**Command + E**
Discard adding a wire*	**RMB**	**RMB**
Force **Set wire** while **Erase wire** is enabled*	Hold **Shift**	Hold **Shift**
Force **Erase wire** while **Set wire** is enabled*	Hold **Alt**	Hold **Alt**
Add multiple wires from the same source*	Hold **Ctrl**	Hold **Command**
Add multiple wires from the same source while **Erase Wire** is enabled*	Hold **Ctrl + Shift**	Hold **Command + Shift**
Force remove network while **Set wire** is enabled*	Hold **Ctrl + Alt**	Hold **Command + Alt**
Copy external components	**Ctrl + C**	**Command + C**

Paste external components	**Ctrl + V**	**Command + V**

Simulation results		
Action	**Windows/Linux controls**	**macOS controls**
Move the plot*	Hold **MMB**	Hold **MMB**
Add one marker on the plot*	**Ctrl + LMB/RMB**	**Command + LMB/RMB**
Add two markers on the plot*	**Ctrl + LMB + RMB**	**Command + LMB + RMB**
Clear all markers*	**Esc**	**Esc**
Zoom in/out x-axis*	**Ctrl + mouse wheel**	**Command + mouse wheel**
Zoom in/out y-axis*	**Shift + mouse wheel**	**Shift + mouse wheel**
Help*	**F1**	**F1**

Import model/subcircuit		
Action	**Windows/Linux controls**	**macOS controls**
Open search field*	**Ctrl + F**	**Command + F**
Close search field*	**Esc**	**Esc**

Debug		
Action	**Windows/Linux controls**	**macOS controls**
Emulation	**Shift + E**	**Shift + E**
Program	**Shift + P**	**Shift + P**
Read	**Shift + R**	**Shift + R**
Test Mode	**Shift + T**	**Shift + T**
NVM Data	**Shift + N**	**Shift + N**
Info	**Shift + I**	**Shift + I**
Log	**Shift + L**	**Shift + L**

I2C reconfigurator		
Action	**Windows/Linux controls**	**macOS controls**
Create snapshot*	**Shift + A**	**Shift + A**
Add delay*	**Shift + D**	**Shift + D**
Move up*	**Shift** + up	**Shift** + up
Move down*	**Shift** + down	**Shift + E**
Send one snapshot*	**F7**	**F7**
Send all snapshots*	**F6**	**F6**
Stop sending snapshots*	**Shift + F7**	**Shift + F7**

ASM editor		
Action	**Windows/Linux controls**	**macOS controls**
Set link	**Ctrl + W**	**Command + W**
Erase link	**Ctrl + E**	**Command + E**
Clear	**Ctrl + N**	**Command + N**
Undo	**Ctrl + Z**	**Command + Z**
Redo	**Ctrl + Y**	**Command + Y**

Help	**F1**	**F1**

General		
Action	**Windows/Linux controls**	**macOS controls**
New project	**Ctrl + N**	**Command + N**
Open project	**Ctrl + O**	**Command + O**
Save project	**Ctrl + S**	**Command + S**
Undo	**Ctrl + Z**	**Command + Z**
Redo	**Ctrl + Y**	**Command + Y**
Zoom in work area	**+**	**+**
Zoom out work area	**-**	**-**
Fullscreen mode	**F11**	**F11**
Print	**Ctrl + P**	**Command + P**
Help	**F1**	**F1**
Exit program	**Ctrl + Q**	**Command + Q**

Note: Non-configurable controls are marked with an asterisk (*).

6.3 Debugging Controls Feature Availability

6.3.0.1 GreenPAK Boards

Feature	GreenPAK Advanced Development Board	GreenPAK DIP Development Board	GreenPAK Lite Development Board	GreenPAK Serial Debugger Board	Go Configure Development Board
Debug Configuration	✓	✓	✓	✓	✓
Device selector	✓	✓	✓	✓	✓
External device mode			✓	✓	✓
Chip procedures	✓	✓	✓	✓	✓
NVM Programmer window	✓	✓	✓	✓	✓
Generator controls	✓	✓			✓
Expansion connector	✓	✓	✓		
TP map	✓	✓	✓		✓
Power source selector	✓	✓	✓	✓	✓
Voltage level controls			✓	✓	
LEDs ON and LEDs OFF	✓	✓	✓		✓
Info details	✓	✓	✓	✓	✓

6.3.0.2 ForgeFPGA Boards

Feature	ForgeFPGA Evaluation Board	ForgeFPGA Deluxe Development Board	Go Configure Development Board
Debug Configuration	✓	✓	✓
Chip procedures	✓	✓	✓
Flash procedures		✓	✓
Generator controls		✓	✓
TP map		✓	✓
Socket controls		✓	✓
External bitstream		✓	✓
Voltage level controls	✓		
Info details	✓	✓	✓

6.3.0.3 Power GreenPAK Boards

Feature	Power GreenPAK Development Motherboard	SLG51002CTR Demo Board	GreenPAK Serial Debugger Board (with SLG5100x)
Debug Configuration	✓	✓	✓
Device selector	✓		✓
Chip procedures	✓	✓	✓
GPIO SW Control		✓	
TP map	✓		
Voltage level controls			✓
Voltage controls	✓	✓	
Info details	✓	✓	✓

6.4 Abbreviations

Abbreviation	Description
ASM	Asynchronous State Machine
CLK	Clock
CS	Circuit Switched
DC	Direct Current
DIP	Dual In-Line Package
EEPROM	Electrically Erasable Programmable Read-only Memory
EPG	Extended Pattern Generator
FPGA	Field-Programmable Gate Array
GND	Ground
GPI	General-Purpose Input
GPIO	General-Purpose Input/Output
I/O	Input/Output
I2C	Inter-Integrated Circuit
IC	Integrated Circuit
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
LDO	Low-Dropout Regulator
LED	Light Emitting Diode
LUT	Look-Up Table
MTP	Multiple-Time Programmable
NC	Not Connected
NVM	Non-Volatile Memory
OE	Output Enable
OTP	One-Time Programmable
POR	Power-On Reset
PWM	Pulse Width Modulation
SCL	Serial Clock
SDA	Serial Data
SPI	Serial Peripheral Interface
SPICE	Simulation Program with Integrated Circuit Emphasis
TP	Test Point
UART	Universal Asynchronous Receiver/Transmitter

6.5 Changelog

Version	Date	Changes
2.13.0	2026-06-05	■ Introduced the redesigned Hub window. ■ Documented the Draw Frame tool. ■ Included information about component pin indicators. ■ Added the description of chip current consumption functionality. ■ Included information about Go Configure Development Board support for GreenPAK devices. ■ Documented demo boards. ■ Added the Current Limits tool description. ■ Included a note about emulation bypass of standard chip startup. ■ Refreshed the debugging controls feature availability table for GreenPAK boards with Go Configure Development Board support. ■ Refreshed the debugging controls feature availability table for ForgeFPGA boards with Go Configure Development Board support. ■ Updated the full chip simulation functionality description (FPGA Editor). ■ Documented bitstream encryption. ■ Included information about generating a bitstream with the Boot config + User OTP data only setting configured. ■ Described how to map I/O Planner ports in Macrocell Editor (FPGA Editor). ■ Included a note that placement constraints are applied after being added (FPGA Editor). ■ Updated information about supported Yosys versions (FPGA Editor). ■ Refreshed the Floorplan description according to tool UI changes (FPGA Editor). ■ Improved guide formatting. ■ Corrected typos.

2.12.0	2025-11-07	■ Added Go Configure Development Board description. ■ Added Placement Constraints description (FPGA Editor). ■ Mentioned External Bitstream debugging controls feature. ■ Updated Flash Programmer tool description. ■ Added info about Yosys version (FPGA Editor). ■ Added info about Linter support (FPGA Editor). ■ Updated FPGA project Sources description (FPGA Editor). ■ Updated the supported OS list.
2.11.1	2025-09-23	■ Extended Simulation description with information about Full Chip RTL Simulation (FPGA Editor). ■ Updated NVM Programmer (Project Data) tool description. ■ Updated Hub structure description. ■ Noted Window Appearance Settings saving behavior. ■ Listed groups of supported blocks in Macrocell Editor (FPGA Editor).
2.11.0	2025-07-18	■ Added new Chip memory lock configuration instruction.
2.10.0	2025-06-20	■ Described new Flash Programmer tool. ■ Updated information for Security tab of the Project Settings window. ■ Extended the Generate Bitstream settings section with new configurations description (FPGA Editor). ■ Updated list of SDC commands with supported arguments (FPGA Editor). ■ Updated screenshots to match minor functionality changes.
2.9.0	2025-04-11	■ Updated Settings description with info about high DPI configurations. ■ Added description for new NVM and EEPROM eraser tools. ■ Described new Socket controls on the Expert debugging controls panel. ■ Updated chip procedure Validation checks list. ■ Updated description for PLL Configurator (FPGA Editor).
2.8.0	2025-03-06	■ Added new HBRAM OTP Data Editor description. ■ Added distinction between Standard and Expert Debugging Controls panel types.

2.7.0	2025-01-30	■ Added new Go Configure Driver Tool description. ■ Updated Connections description with new external connection type for SLG51001. ■ Mentioned possibility to copy and paste external components between different software instances. ■ Added info about new Description column to I/O Planner description (FPGA Editor). ■ Updated description for PLL Calculator (FPGA Editor).
2.6.3	2024-12-18	■ Added list of supported SDC commands in Timing Constraints file (FPGA Editor). ■ Added possible solution for successful simulation launch (FPGA Editor).
2.6.2	2024-11-22	■ Added Dynamic simulation description. ■ Described new FPGA connection types (SLG47910V (Rev.BB)). ■ Extended Project directory structure section with information about project subfolders. ■ Extended description of the PLL Calculator tool (FPGA Editor). ■ Fixed typos.
2.6.1	2024-09-26	■ Added information about Manual generator type in Memory Table Editor. ■ Added Math Core Table description to I2C Virtual Outputs section. ■ Extended Project directory structure section with information about new bitstream folder. ■ Updated description about PLL Calculator (FPGA Editor). ■ Renamed PowerPAK Development Platform to Power GreenPAK Development Motherboard. ■ Renamed ForgeFPGA Advanced Development Board to ForgeFPGA Deluxe Development Board. ■ Improved screenshots style.
2.6.0	2024-07-31	■ Added new troubleshooting instruction for the ForgeFPGA Evaluation Board driver issue. ■ Added information about Timer macrocells support in the Acceleration Profile Configurator tool. ■ Described the reason for possible project checksum modification. ■ Mentioned about macrocells' port types in the Macrocell Editor (FPGA Editor). ■ Extended the Settings section with info about new Processing and Files tabs (FPGA Editor).

2.5.1	2024-07-04	■ Added information about the new FPGA project creation approach in Working with project files section. ■ Updated the Project directory structure section with information about quick source file localization (FPGA Editor). ■ Added note about the requirement to save the modified modules before performing procedures (FPGA Editor). ■ Fixed typos.
2.5.0	2024-06-20	■ Added information about the Speed control tool in the I2C Tools section. ■ Updated the I2C Virtual Outputs description with new macrocells support. ■ Extended the resource meter description in the Synchronous Logic Generator section. ■ Added the description of new FPGA project structure. ■ Updated the Messages panel description with new way of logs displaying (FPGA Editor). ■ Added information about new status indication of the Synthesis/Generate bitstream procedures (FPGA Editor). ■ Updated the CPU and suppported OS information in the System requirements section.
2.4.0	2024-04-26	■ Added the new Acceleration Profile Configurator section. ■ Added the new Reg File Configurator section. ■ Added the new DAC Tool section. ■ Added the new Power Monitor section. ■ Added the new PLL Calculator section (FPGA Editor). ■ Updated the Memory Table Editor section with information about Signal generator type. ■ Extended the Synchronous Logic Generator content with new resource meter description. ■ Extended the Logic Analyzer section with the description of export feature in Protocol Analyzer. ■ Updated the I/O Planner content with the import and export formats information (FPGA Editor). ■ Extended the I/O Planner content with warning icons for invalid port names description (FPGA Editor). ■ Extended the Macrocell Editor content with Properties panel description (FPGA Editor). ■ Added information about project checksum in Working with project files section. ■ Substituted the Devices section content with a feature list for each platform. ■ Updated the supported OS list. ■ General content improvements.

2.3.0	2024-02-12	■ Added the new Synthesis Report section (FPGA Editor). ■ Added the new Design Templates section (FPGA Editor). ■ Added the new Settings section (FPGA Editor). ■ Added the *Icarus Verilog and GTKWave quick start* guide in the How to section. ■ Extended the Additional controls section with a new option for the Split tool (FPGA Editor). ■ Extended the Simulating a testbench section with information about ensuring persistence for simulation results (FPGA Editor). ■ Extended the Project settings window section with the Security tab description. ■ Improved and extended the ForgeFPGA devices section. ■ Improved the Logic Analyzer section. ■ Improved the I2C I/O Tool section. ■ Improved the I2C Tools section. ■ Updated the I/O Planner section (FPGA Editor). ■ Fixed typos.
2.2.0	2023-12-22	■ Added a new Memory Table Editor section.
2.1.0	2023-12-08	■ Added a new FPGA Editor section. ■ Added a new Troubleshooting section with the Failed Socket Test solutions. ■ Added a new section with the Voltage Monitor tool. ■ Extended the I2C Tools section with the new Memory Table tool. ■ Extended the I2C Tools section with the new Data Buffers tool. ■ Extended Debugging Controls for the PowerPAK Dev. Platform with the LEDs ON and LEDs OFF feature description. ■ Extended Debugging Controls for the PowerPAK Dev. Platform with the Device selector (external chip support) feature description. ■ Updated the Board Selector in the Debugging Controls with the new information about the Socket Test. ■ Updated the notification of the processing data time for Transient Simulation Analysis. ■ Updated the System Requirements section. ■ Improved the Demo Board and Demo mode section. ■ Improved the Hardware source image list. ■ Fixed typos.
2.0.1	2023-10-25	■ Fixed typos. ■ Added Changelog.

2.0.0	2023-10-13	■ New revised structure. ■ Significantly improved navigation. ■ Carefully selected materials. ■ Improved instructions. ■ Improved graphic material.

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.01)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.