Go Configure™ Software Hub

User Guide

v.2.12.0

Contents

1 Introduction

Renesas Go Configure™ *Software Hub* is a full-featured Integrated Development Environment (IDE) that enables a completely graphical design process, allowing you to configure, program and simulate custom circuits 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*, *AutomotivePAK*, and *ForgeFPGA*™ *Workshop*.

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 10/11, macOS 11 or higher, Ubuntu 22.04/24.04, Debian 13/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 your 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

In case you need more information about *Go Configure Software Hub*, online support is available at www.renesas.com.

You can also find *Go Configure Software Hub* on *Renesas* social media. To visit the page, go to *Help* → *Social* in the main menu.

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. You can learn more in section 4.1 Software update.

2 Software overview

The *Go Configure Software Hub* was designed to provide direct access to all *GreenPAK*, *SLG51000/1*, and *ForgeFPGA* device features and complete control over the routing and configuration options.

In this chapter, you'll find detailed explanations for various tools to simplify your workflow. These tools allow you to program a chip with your custom design, import data from a programmed part, and simulate with external components. This chapter will also become your guide to exploring the software interface and understanding essential tools like debug tools, asynchronous state machine, software simulation, and FPGA editor, etc.

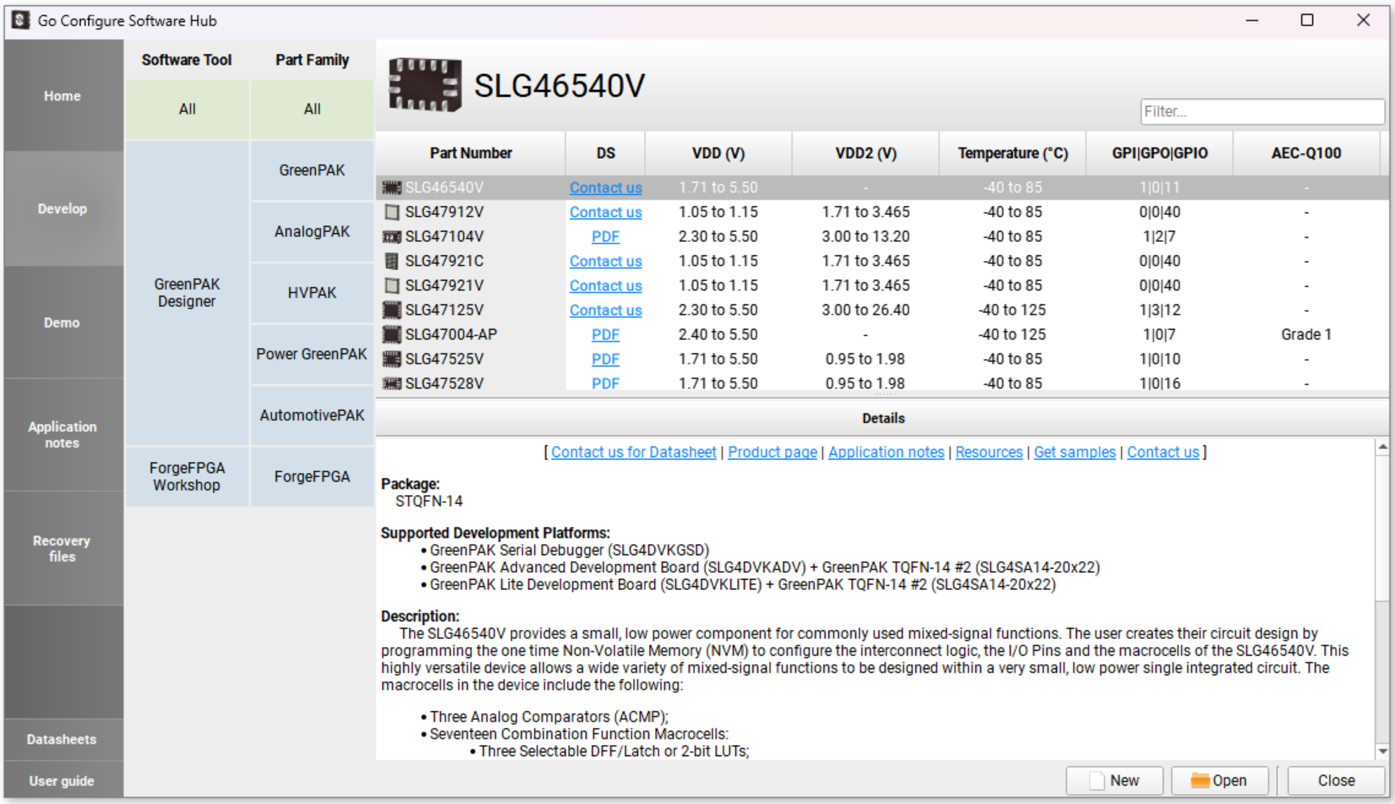
2.1 Design tools

Start your project from the *Hub* window with the following sections:

- *Home* — useful info and tips for new users. Also, find the list of the recently opened project files here
- *Develop* — the chip Part Number selection. See the *Details* section to learn more about the selected chip

At the bottom of the window, you can find the *New*, *Open*, and *Close* buttons, which allow you to start a new project for a selected Part Number, open an existing project or close the *Go Configure Software Hub*. The *Datasheets* and *User Guides* buttons redirect you to the *Renesas* website, where you can download the corresponding files.

- *Demo* — the list of *Demo* projects. You can use the specific *Demo Board* for the project debugging (read more in section 2.2.8 Demo Board and Demo mode)
- *Application notes* — design examples for different purposes. An application note includes a design description with various Integrated Circuits (ICs) and a preconfigured circuit project, where you can make customized changes
- *Recovery files* — restored project files after a crash or freeze. To read more refer to section 2.1.7 Settings



Part Number	DS	VDD (V)	VDD2 (V)	Temperature (°C)	GPI|GPO|GPIO	AEC-Q100
SLG46540V	Contact us	1.71 to 5.50	-	-40 to 85	1|0|11	-
SLG47912V	Contact us	1.05 to 1.15	1.71 to 3.465	-40 to 85	0|0|40	-
SLG47104V	PDF	2.30 to 5.50	3.00 to 13.20	-40 to 85	1|2|7	-
SLG47921C	Contact us	1.05 to 1.15	1.71 to 3.465	-40 to 85	0|0|40	-
SLG47921V	Contact us	1.05 to 1.15	1.71 to 3.465	-40 to 85	0|0|40	-
SLG47125V	Contact us	2.30 to 5.50	3.00 to 26.40	-40 to 125	1|3|12	-
SLG47004-AP	PDF	2.40 to 5.50	-	-40 to 125	1|0|7	Grade 1
SLG47525V	PDF	1.71 to 5.50	0.95 to 1.98	-40 to 85	1|0|10	-
SLG47528V	PDF	1.71 to 5.50	0.95 to 1.98	-40 to 85	1|0|16	-

Details

[Contact us for Datasheet | Product page | Application notes | Resources | Get samples | Contact us]

Package:
STQFN-14

Supported Development Platforms:
- GreenPAK Serial Debugger (SLG4DVKGSD)
- GreenPAK Advanced Development Board (SLG4DVKADV) + GreenPAK TQFN-14 #2 (SLG4SA14-20x22)
- GreenPAK Lite Development Board (SLG4DVKLITE) + GreenPAK TQFN-14 #2 (SLG4SA14-20x22)

Description:
The SLG46540V provides a small, low power component for commonly used mixed-signal functions. The user creates their circuit design by programming the one time Non-Volatile Memory (NVM) to configure the interconnect logic, the I/O Pins and the macrocells of the SLG46540V. This highly versatile device allows a wide variety of mixed-signal functions to be designed within a very small, low power single integrated circuit. The macrocells in the device include the following:

- Three Analog Comparators (ACMP);
- Seventeen Combination Function Macrocells:
  - Three Selectable DFF/Latch or 2-bit LUTs;

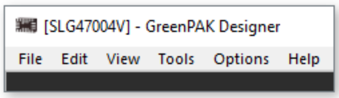
Go Configure Software Hub User Interface

2.1.1 Interface overview

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

Main menu

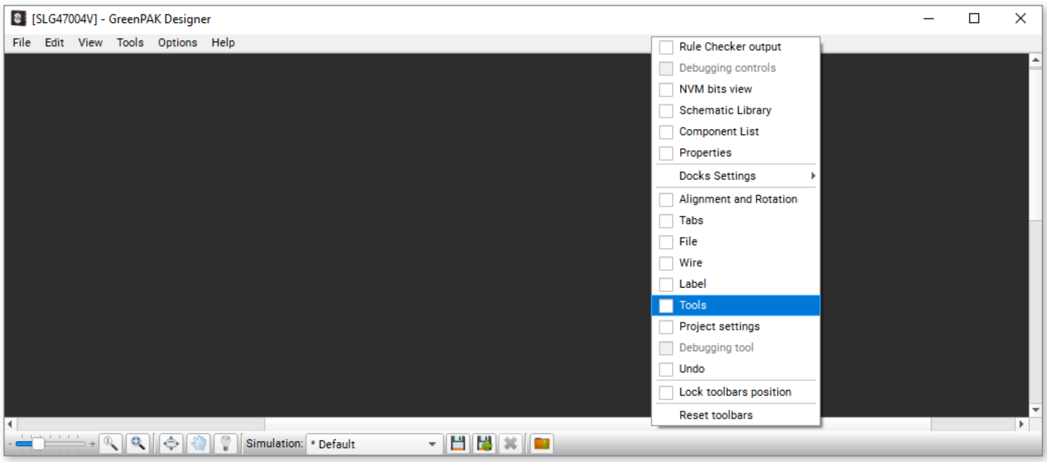
After you select the Part Number and start the project, you can see the menu bar with the following items: *File*, *Edit*, *View*, *Tools*, *Options* and *Help*. See the full description of all the menu commands in section 6.1 Main menu commands.

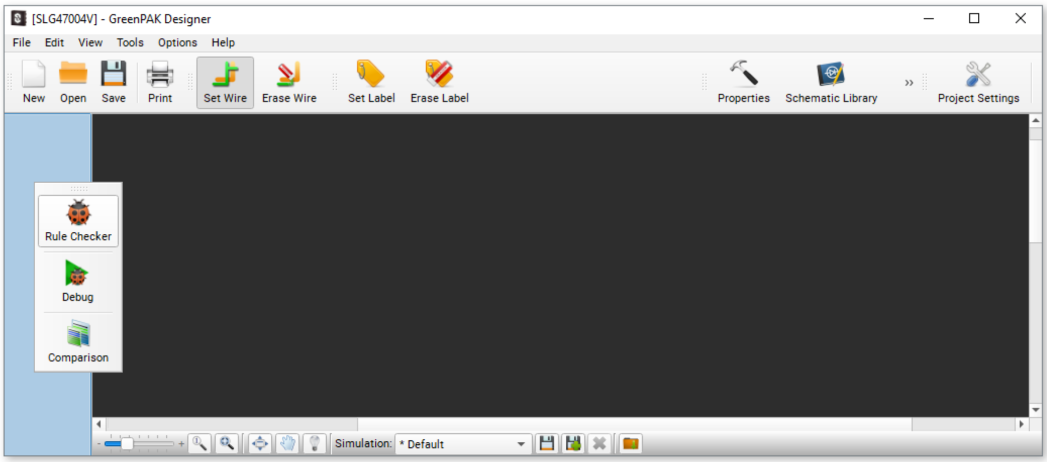


Main menu

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*.



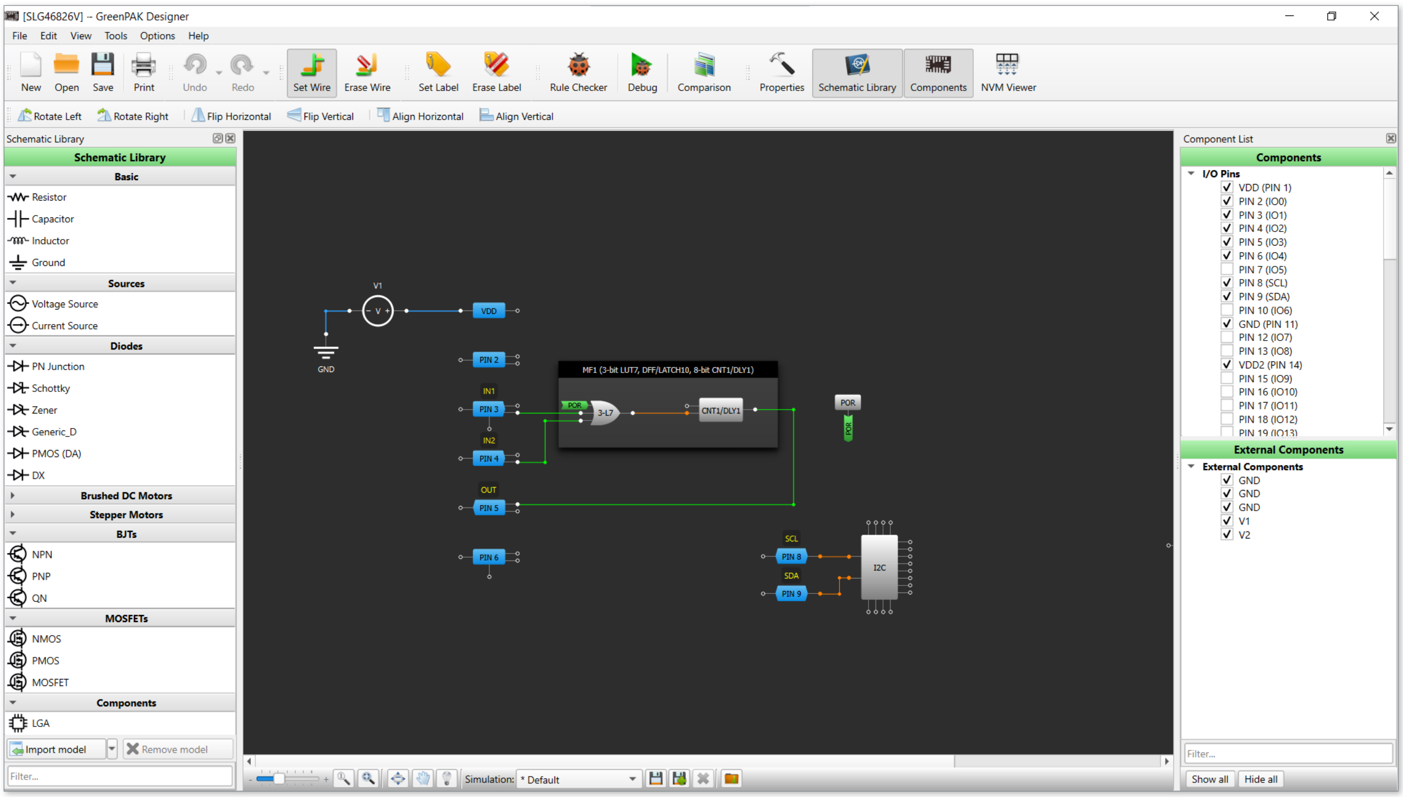


Toolbar customization

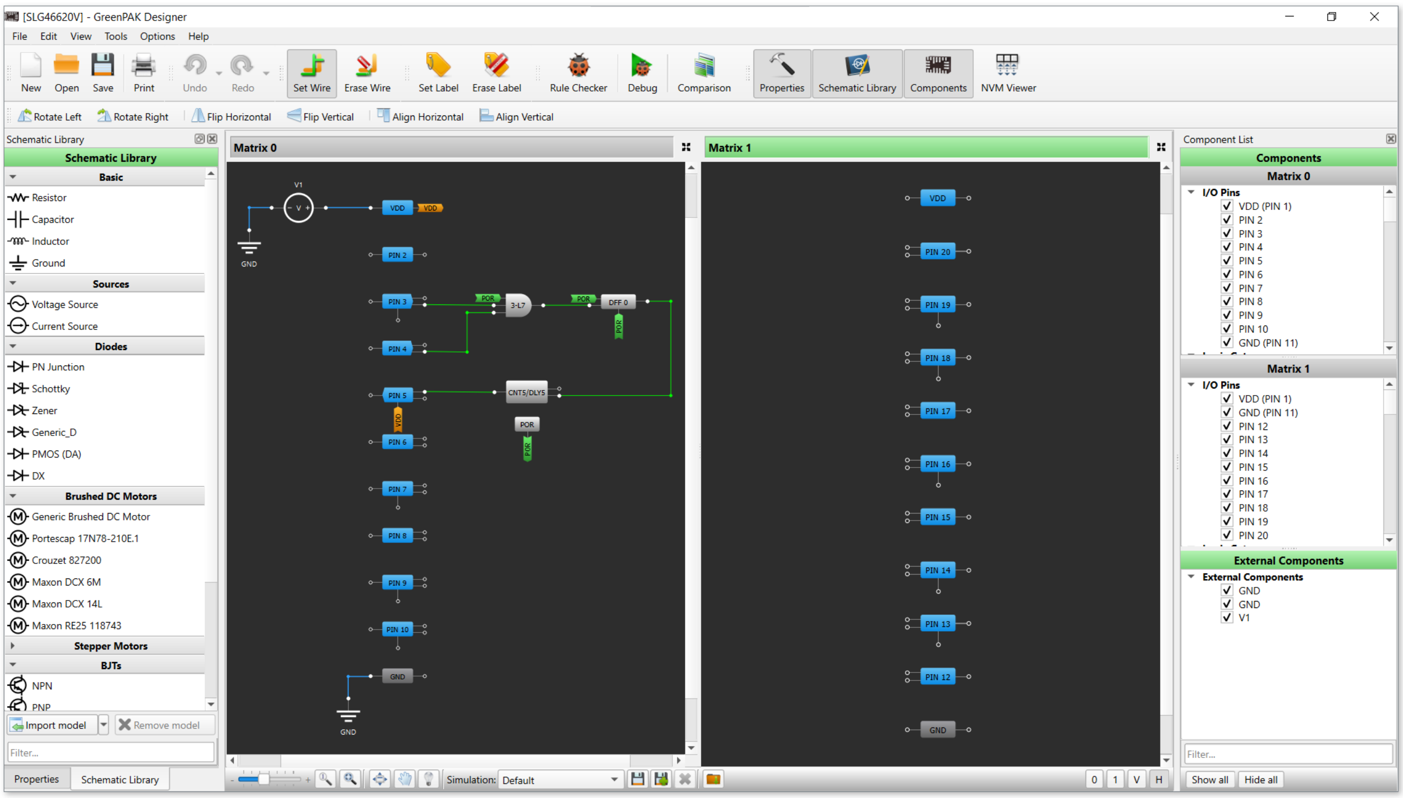
Work area

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).

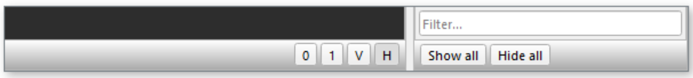


Part Number with one work area


[SLG46620V] - GreenPAK Designer
File Edit View Tools Options Help
New Open Save Print Undo Redo Set Wire Erase Wire Set Label Erase Label Rule Checker Debug Comparison Properties Schematic Library Components NVM Viewer
Rotate Left Rotate Right Flip Horizontal Flip Vertical Align Horizontal Align Vertical
Schematic Library
Matrix 0
Matrix 1
Component List
Simulation: Default
0 1 V H
Show all Hide all
Properties Schematic Library

Part Number with two work areas

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



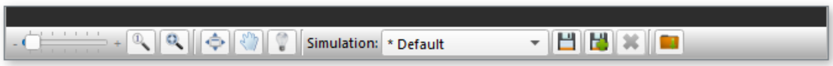
Matrices' window placement

- *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.

Work area controls

You can find the additional work area controls on the bottom toolbar.



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)

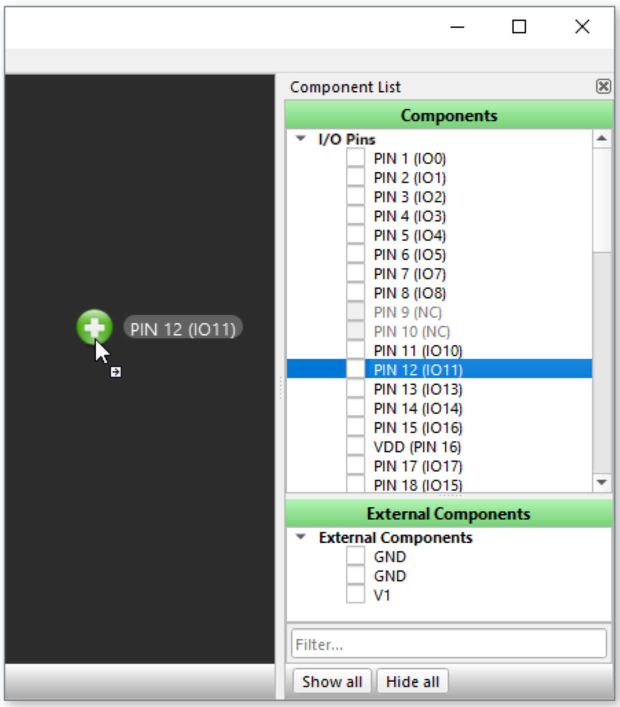
Activate the possibility to see the hint box with the block's property info upon hovering the mouse over a component

Component list

This panel contains a list of available macrocells in a chip. You can show, hide, or search for the components via the list.

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

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



Checkbox / Drag and Drop

You can use filter to easily search for the required component. Show/hide all the components by using *Show all* and *Hide all* buttons.

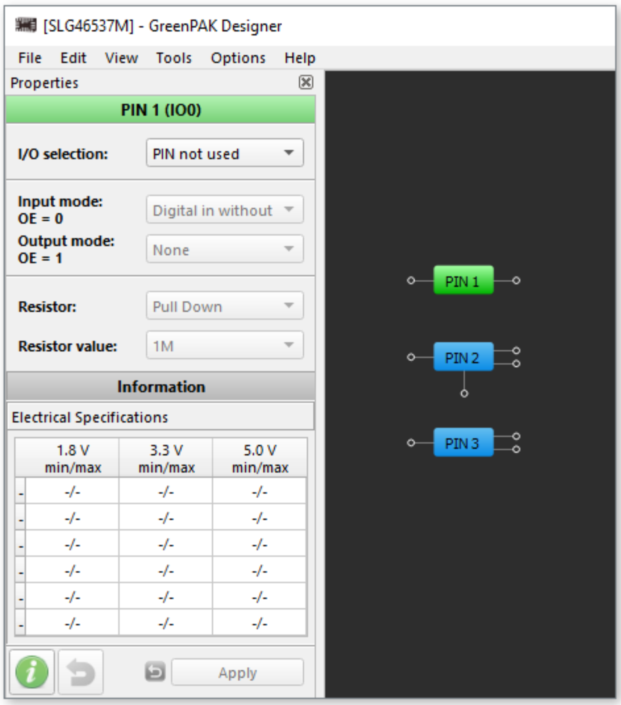
Note 1: A hidden component retains the configurations set on its properties panel (see more info about the *Properties* panel).
Note 2: It is not possible to hide connected blocks (see sections 2.1.4 Components and 2.1.5 Connections to find out more).

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


[SLG46537M] - GreenPAK Designer

File Edit View Tools Options Help

Properties

PIN 1 (IO0)

I/O selection: PIN not used

Input mode: OE = 0 Digital in without

Output mode: OE = 1 None

Resistor: Pull Down

Resistor value: 1M

Information

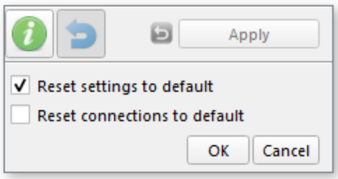
Electrical Specifications

	1.8 V min/max	3.3 V min/max	5.0 V min/max
-	-/-	-/-	-/-
-	-/-	-/-	-/-
-	-/-	-/-	-/-
-	-/-	-/-	-/-
-	-/-	-/-	-/-
-	-/-	-/-	-/-

Properties panel

Once you change the property values, you can *Apply* or *Revert* the modifications.

After changes are applied, it is possible to *Reset* the changes. 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, e.g. LUT).



✓ Reset settings to default

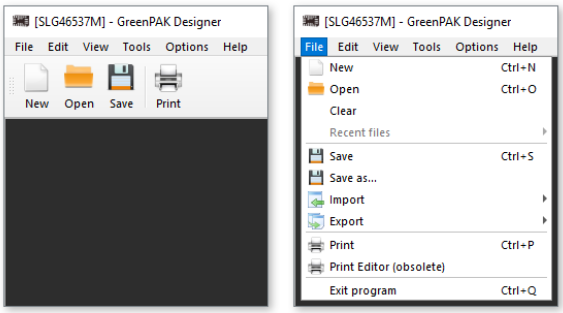
☐ Reset connections to default

Property reset feature

You can also *Reset* the block via 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, etc.

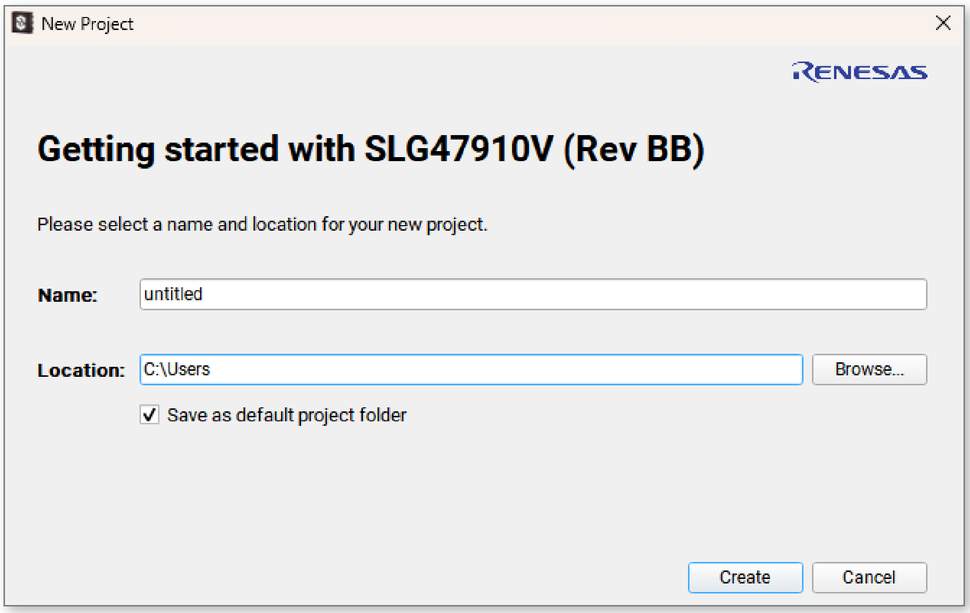


File operations

Create/Save/Open a project

You can start a new project from the *Hub* window (click the Part Number → *New*) or from an opened instance (click the *New* icon on the toolbar or *File* → *New*)

Note: For the *FPGA* Part Number (e.g. SLG47910), once you select it from the Part Numbers list in the *Hub* window, you will be prompted to create the project via the *New Project* window. Upon clicking *Create*, the *FPGA* project directory will appear in the specified location.



New Project

Getting started with SLG47910V (Rev BB)

Please select a name and location for your new project.

Name: untitled

Location: C:\Users

Browse...

☑ Save as default project folder

Create Cancel

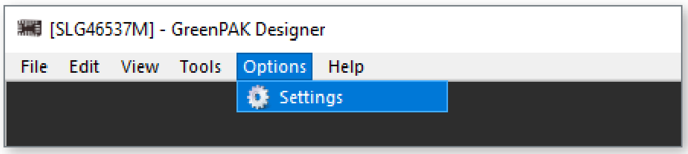
Window for FPGA project creation

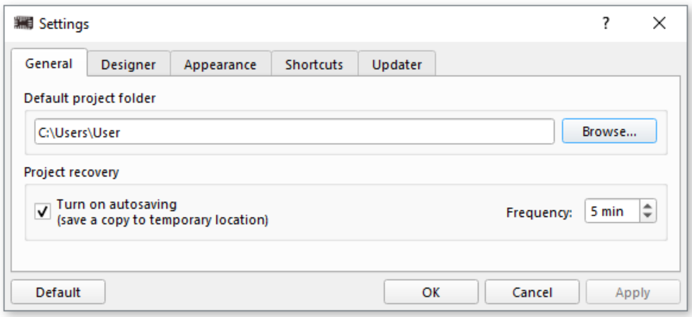
Once you start a new project, the *Project settings* window appears. You can skip this step, but once the data is needed, you'll be asked to fill it in again. Read more about *Project settings* window later in this section.

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

To save your 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*.





Default project folder: C:\Users\User

Project recovery

✓ Turn on autosaving (save a copy to temporary location)

Frequency: 5 min

Project file folder

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

There are several ways to open a project file:

- *Open* button on the *Hub* window
- *Open* icon on the toolbar (or *File* → *Open*)
- Drag and drop the project file to the *Hub* window
- Drag and drop the project file to the work area
- Double-click the project file

Note: It is highly recommended not to make manual changes in the file to prevent from its corruption.

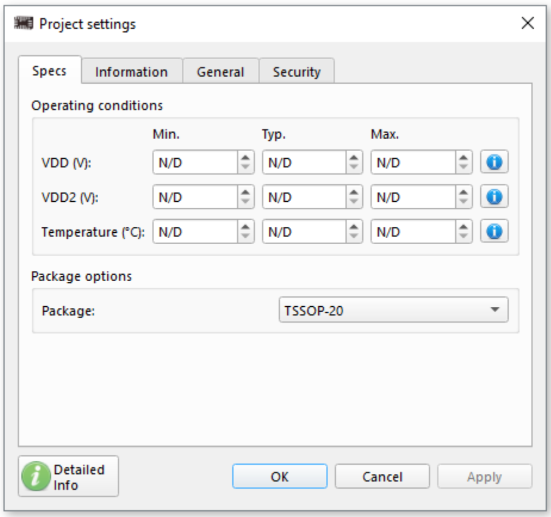
Project settings window

As mentioned earlier in this section, the *Project settings* window appears right after you start a new project. It contains four tabs: *Specs*, *Information*, *General*, and *Security*.

Specs

In *Specs*, you can see the fields to fill in the chip operating conditions, such as VDD and Temperature (Min., Typ. and Max). The Part Numbers may have one or multiple VDDs. Click the blue *information* button to see the recommended range within which the chip can operate safely.

The Part Numbers may support different packaging. The package can also be selected in the *Specs* tab.


Project settings

Specs | Information | General | Security

Operating conditions

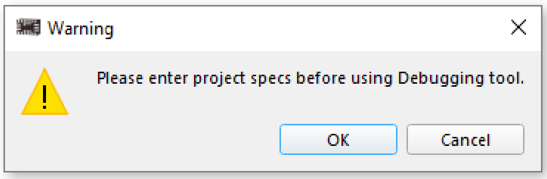
	Min.	Typ.	Max.
VDD (V):	N/D	N/D	N/D
VDD2 (V):	N/D	N/D	N/D
Temperature (°C):	N/D	N/D	N/D

Package options

Package: TSSOP-20

Project settings window examples

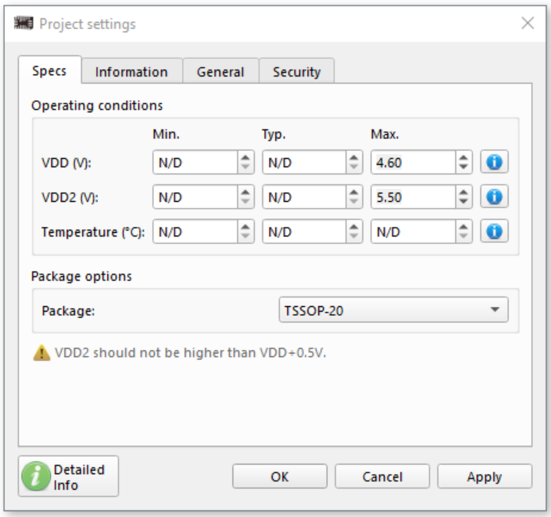
Note: You can close the *Project settings* window without entering the operating conditions. Working with *Simulation* or hardware development platforms in the *Debug tool* requires entering the specs. Once you click *Debug* on the toolbar while specs are not specified, a warning notification to add specs appears. Click the *OK* button on the warning window or the *Project Settings* icon on the toolbar to enter specs.


Warning

Please enter project specs before using Debugging tool.

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 you from applying the specs, while others require to fix them before the application. Here are the warning examples:



Operating conditions

	Min.	Typ.	Max.
VDD (V):	N/D	N/D	4.60
VDD2 (V):	N/D	N/D	5.50
Temperature (°C):	N/D	N/D	N/D

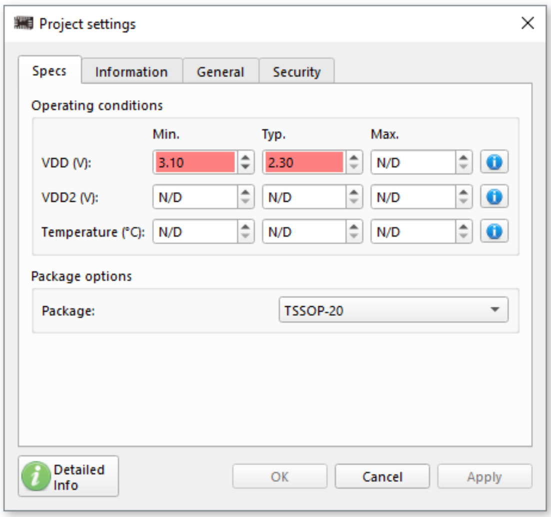
Package options

Package: TSSOP-20

⚠ VDD2 should not be higher than VDD+0.5V.

Specs-related warning examples

If e.g. 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.



Operating conditions

	Min.	Typ.	Max.
VDD (V):	3.10	2.30	N/D
VDD2 (V):	N/D	N/D	N/D
Temperature (°C):	N/D	N/D	N/D

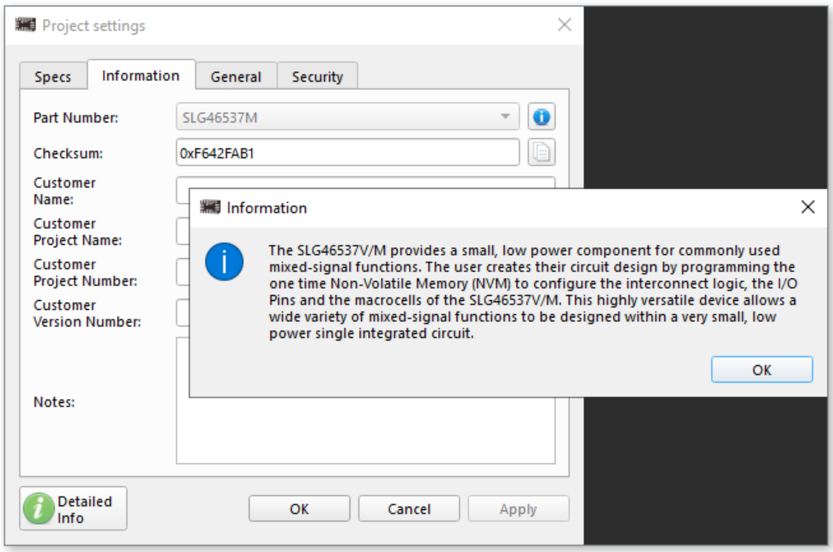
Package options

Package: TSSOP-20

Improper specs values

Information

The *Information* tab contains the short piece of information about the Part Number. To see the info, click the *Info* button.



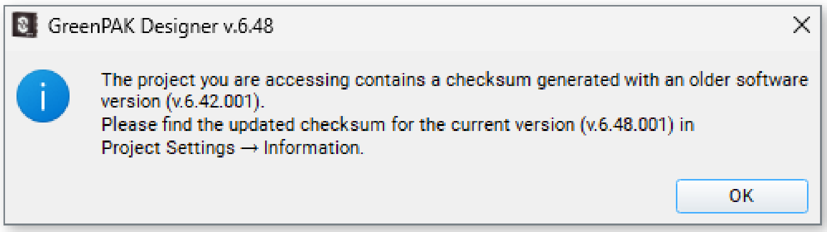
The SLG46537V/M provides a small, low power component for commonly used mixed-signal functions. The user creates their circuit design by programming the one time Non-Volatile Memory (NVM) to configure the interconnect logic, the I/O Pins and the macrocells of the SLG46537V/M. This highly versatile device allows a wide variety of mixed-signal functions to be designed within a very small, low power single integrated circuit.

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 (e.g. SLG47910), the OTP registers [0:191] are ignored during calculation. The FPGA core configuration bitstream is also taken into account upon computing the checksum.

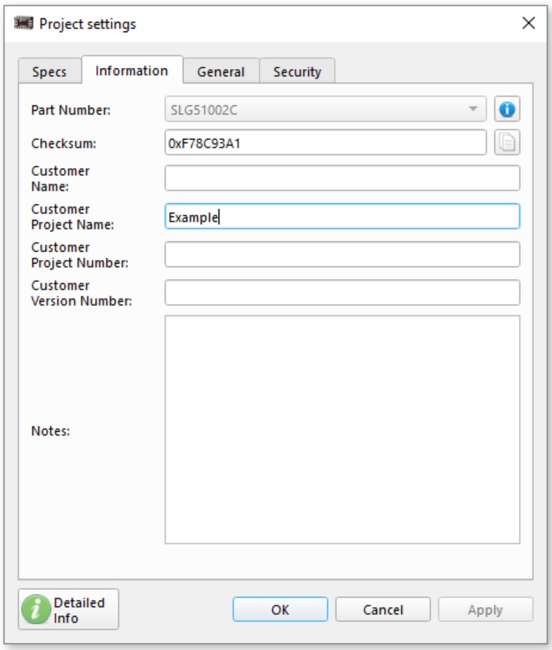
Note: The same project may show different checksums when opened in different software versions. The discrepancy occurs because the set of project bits used to compute the checksum, may vary between software versions. In this case, the warning will appear upon the project load. The checksum will be updated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



The project you are accessing contains a checksum generated with an older software version (v.6.42.001).
Please find the updated checksum for the current version (v.6.48.001) in Project Settings → Information.

Project checksum update warning

You can also see the input fields to fill in the information about the project. The entry in *Customer* *project name* field is also displayed in *Print Preview*.


Project settings

Specs | Information | General | Security

Part Number: SLG51002C

Checksum: 0xF78C93A1

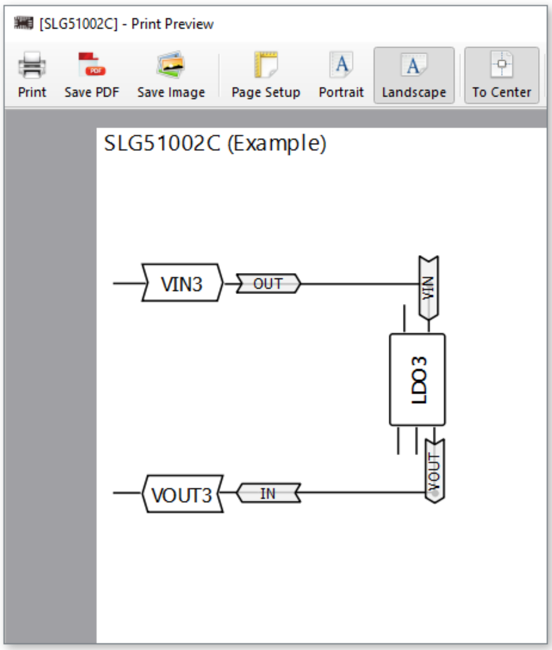
Customer Name:

Customer Project Name: Example

Customer Project Number:

Customer Version Number:

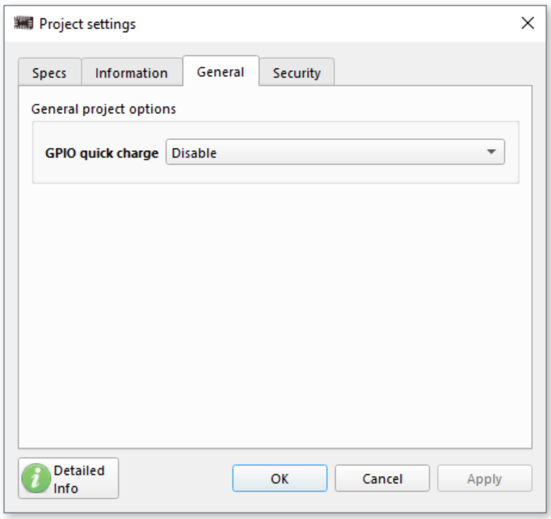
Notes:



Customer project name in Print Preview

General

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


Project settings

Specs | Information | General | Security

General project options

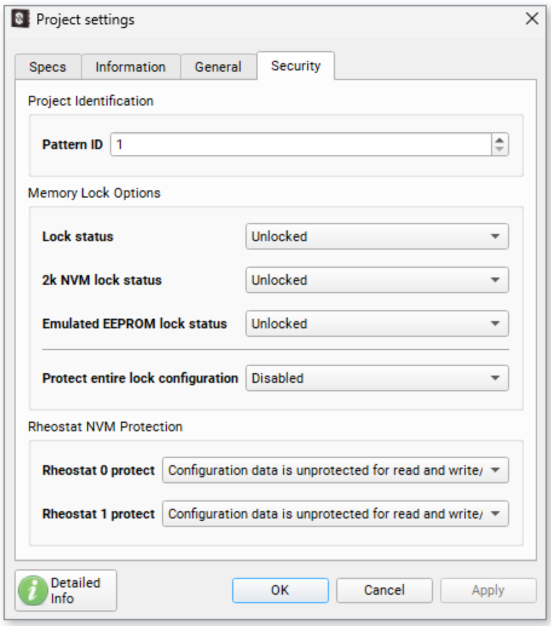
GPIO quick charge: Disable

General tab

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 e.g. available memory type (OTP or MTP). The common and important *Security* tab configurations are project identification and memory lock/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/Protection Options — allow to protect read/write chip operations (manage configurations for specific memory type if applicable)



Project Identification

Pattern ID: 1

Memory Lock Options

Lock status: Unlocked

2k NVM lock status: Unlocked

Emulated EEPROM lock status: Unlocked

Protect entire lock configuration: Disabled

Rheostat NVM Protection

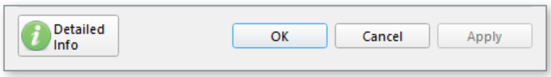
Rheostat 0 protect: Configuration data is unprotected for read and write/

Rheostat 1 protect: Configuration data is unprotected for read and write/

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 your project's memory.

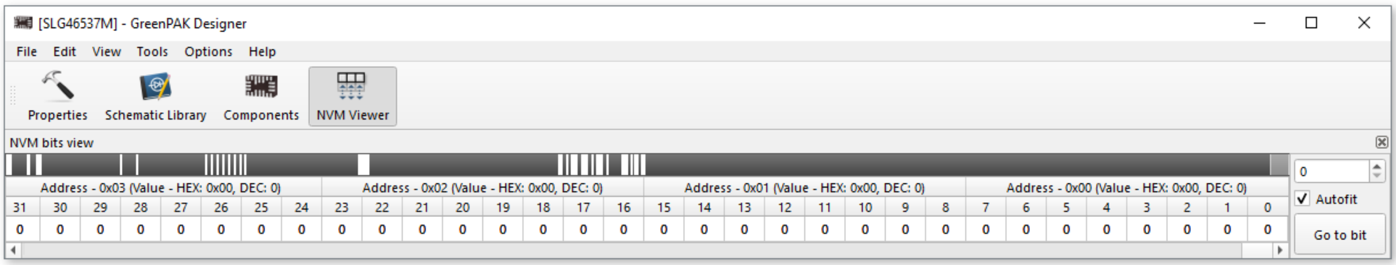
To find out more about the particular chip configurations, click the *Detailed info* button on the *Project settings* window or access the Datasheets through the *Hub* window.



Detailed info button

2.1.3 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, you can see the information about the byte's address 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.



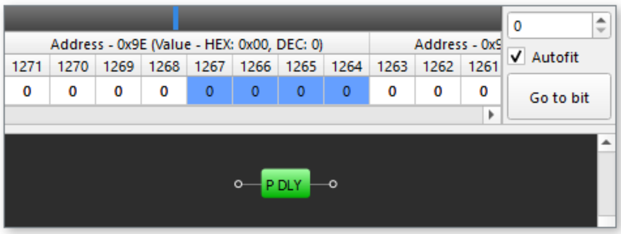
Address - 0x03 (Value - HEX: 0x00, DEC: 0)								Address - 0x02 (Value - HEX: 0x00, DEC: 0)								Address - 0x01 (Value - HEX: 0x00, DEC: 0)								Address - 0x00 (Value - HEX: 0x00, DEC: 0)							
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

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

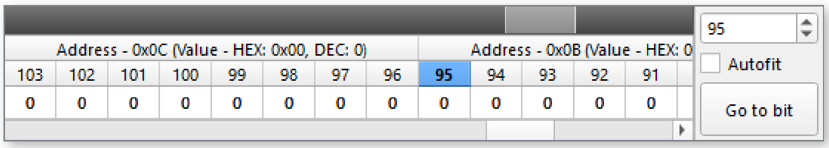
You can also use the *NVM viewer* controls for quick navigation through the bit table. Use the *Autofit* feature to jump directly to the bit range of the selected component.



Address - 0x9E (Value - HEX: 0x00, DEC: 0)								Address - 0x9		
1271	1270	1269	1268	1267	1266	1265	1264	1263	1262	1261
0	0	0	0	0	0	0	0	0	0	0

Autofit

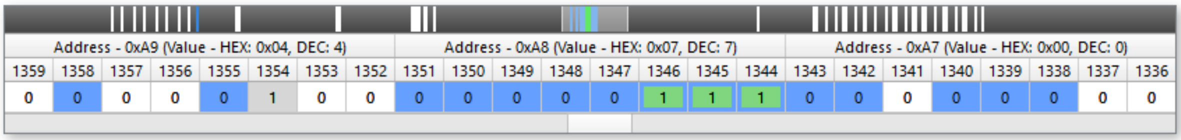
Click *Go to bit* once you enter a bit number (in the decimal format) to find the bit you need. The bit order number becomes highlighted after clicking a bit or using the *Go to bit* button.



Address - 0x0C (Value - HEX: 0x00, DEC: 0)								Address - 0x0B (Value - HEX: 0				
103	102	101	100	99	98	97	96	**95**	94	93	92	91
0	0	0	0	0	0	0	0	0	0	0	0	0

Bit order number highlight

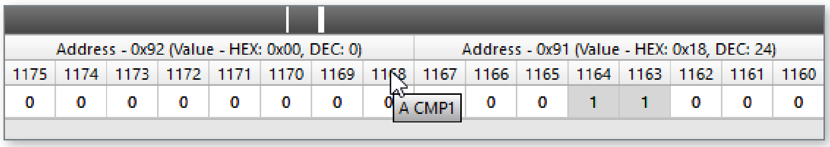
The *NVM viewer* navigation bar can also orient you to the location of selected or changed bits in the table (the color on the bar corresponds the cells color described above). In addition, black color represents `0` and white shows `1` values.



Address - 0xA9 (Value - HEX: 0x04, DEC: 4)								Address - 0xA8 (Value - HEX: 0x07, DEC: 7)								Address - 0xA7 (Value - HEX: 0x00, DEC: 0)							
1359	1358	1357	1356	1355	1354	1353	1352	1351	1350	1349	1348	1347	1346	1345	1344	1343	1342	1341	1340	1339	1338	1337	1336
0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

NVM viewer top bar

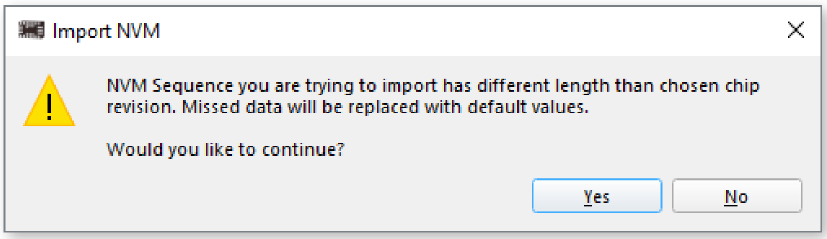
Hover the mouse cursor over the bit to see which macrocell this bit belongs to.



Address - 0x92 (Value - HEX: 0x00, DEC: 0)								Address - 0x91 (Value - HEX: 0x18, DEC: 24)							
1175	1174	1173	1172	1171	1170	1169	1168	1167	1166	1165	1164	1163	1162	1161	1160
0	0	0	0	0	0	0	0		0	0	1	1	0	0	0

NVM hint

You can save or load the NVM sequence by clicking *File* → *Import/Export* in the main menu (see the description of all main menu items in section 6.1 Main menu commands). If the imported file contains a different bit sequence length than the selected Part Number, the corresponding pop-up appears and it is possible to proceed with loading the data.



Import NVM

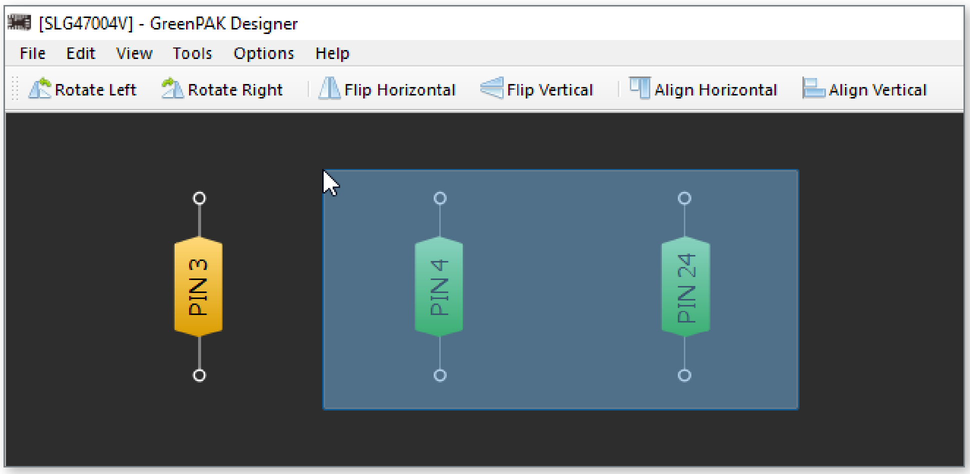
NVM Sequence you are trying to import has different length than chosen chip revision. Missed data will be replaced with default values.

Would you like to continue?

Import different NVM sequence length

2.1.4 Components

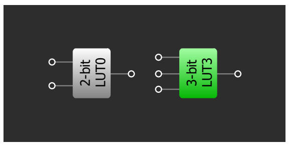
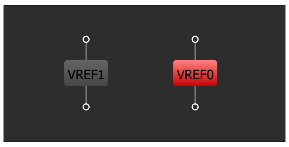
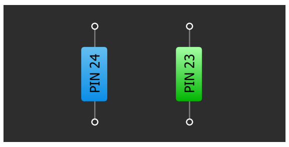
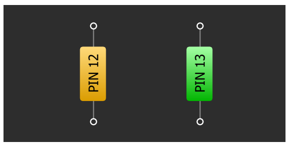
After you add a macrocell from the component list to the work area, you can choose how to interact with it. Move the component(s) using a mouse or a keyboard (see section 6.2 Keyboard and mouse controls). You can also use the toolbar widgets to rotate, flip, and align the selected block(s). To select more than one block click, hold, and drag over the components you want to select.



Selected block adjustment

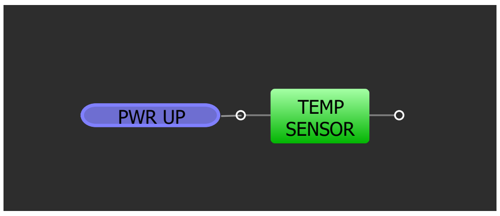
GreenPAK chips components highlight

The macrocell color gives you information about its mode (enabled/disabled) and state (deselected/selected). Input/Output (I/O) Pin block's color informs about its relation to a particular VDD.

	enabled, deselected/selected
	disabled, deselected/selected
	Pin, deselected/selected
	Pin for connection with VDD2 (analog), deselected/selected

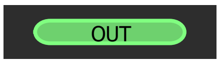
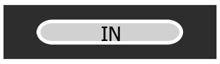
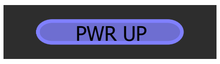
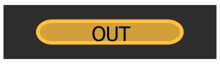
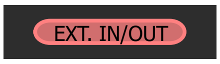
You can connect macrocells via the pins.

Hover the cursor over the block to see the pin hints. For more info about managing pin hints behavior, refer to section 2.1.7 Settings.



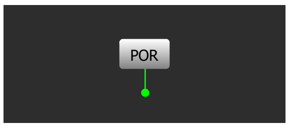
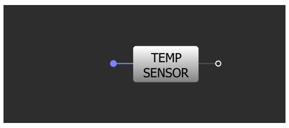
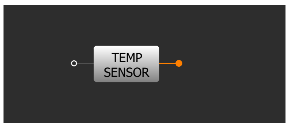
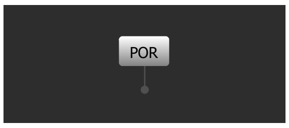
Pin hint

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 (you can change macrocells properties to make it available for connection)
	used for hard-wired connections
	external IN/OUT

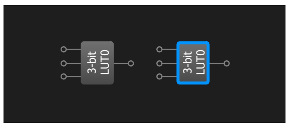
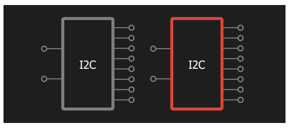
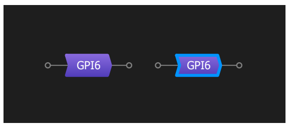
Read about connection types in section 2.1.5 Connections.

When you are in the process of setting a connection, you can see the pin highlight. Just like pin hint color, the pin color indicates the connection possibility.

	open for connection
	temporarily closed for connection (you can change macrocell settings on its *Properties* panel to make it available for connection)
	used for hard-wired connection
	connection is not allowed

SLG51000/1 chip components highlight

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

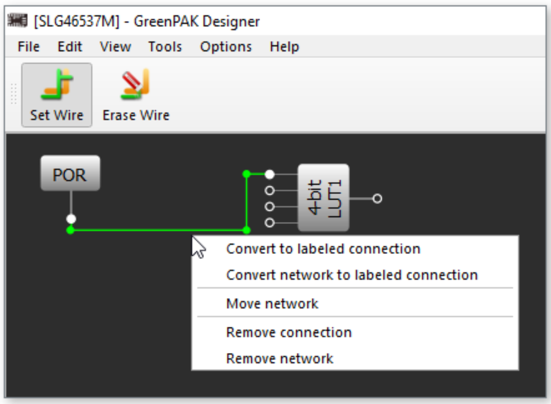
	enabled, deselected/selected
	disabled,deselected/selected
	I/O Pin, deselected/selected

Pin and pin tip colors are the same as for chips described above.

2.1.5 Connections

The *Set Wire* widget helps you to set a connection between the blocks. To add a wire, click two pins between which you are setting a connection. More than one connection set from one OUT pin is a network. You can dismiss connection creation by clicking the right mouse button.

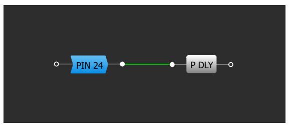
To remove the connection, enable the *Erase Wire* tool and click the wire (you can also delete the connection or network via the context menu, triggered by right-clicking the wire).

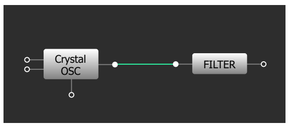
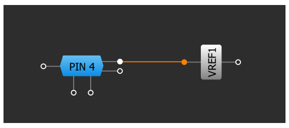
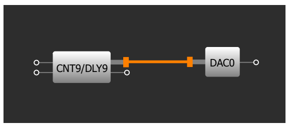
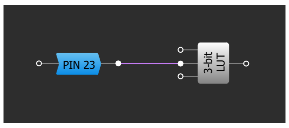
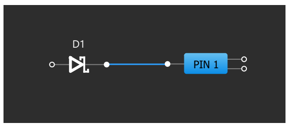

[SLG46537M] - GreenPAK Designer
File Edit View Tools Options Help
Set Wire
Erase Wire
POR
4-bit LUT1
Convert to labeled connection
Convert network to labeled connection
Move network
Remove connection
Remove network

Set/Erase wire/network

GreenPAK chips connections highlight

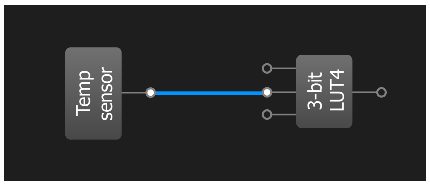
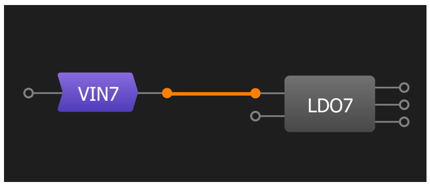
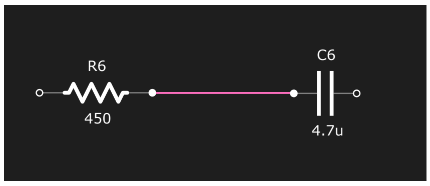
While working with different Part Numbers you may encounter the following component connection types:

	*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 (e.g. *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 from/to external components (read more in section 2.4.2 External components)

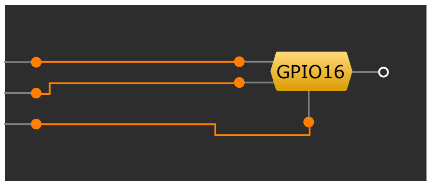
SLG51000/1 chip connection highlight

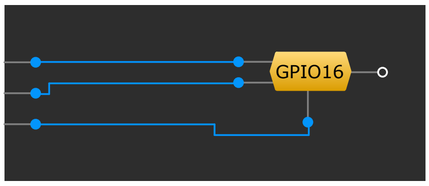
In SLG51000/1 Part Number the connection types are as follows:

	*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 from/to external components (applicable only for SLG51001), read more in section 2.4.2 External components

FPGA chip connection highlight

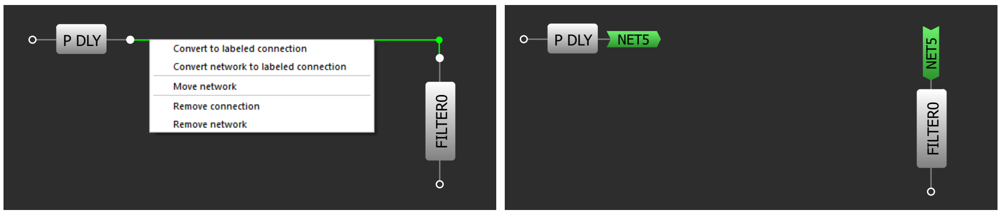
Two connection types for FPGA Part Number (SLG47910V (Rev BB)) are as follows:

	*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

Common connection-related behavior for all Part Numbers

A wired connection/network can be converted to labeled one. Trigger the context menu upon right-clicking the wire to see such option.

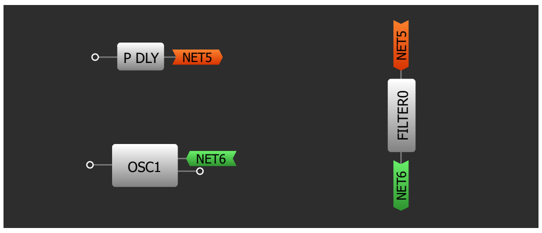


Convert a wire to label

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.

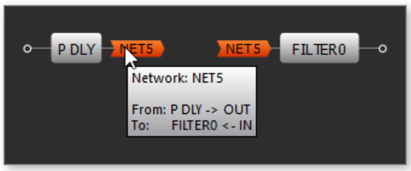
The connections can be labeled by default, but you can change them to wired upon need. You can also change the labeled connection’s name. Double-click the label and enter a new name in a pop-up window.

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



Highlighted label

You can also see the hint with the connection or network info upon hovering the cursor over the wire or label.

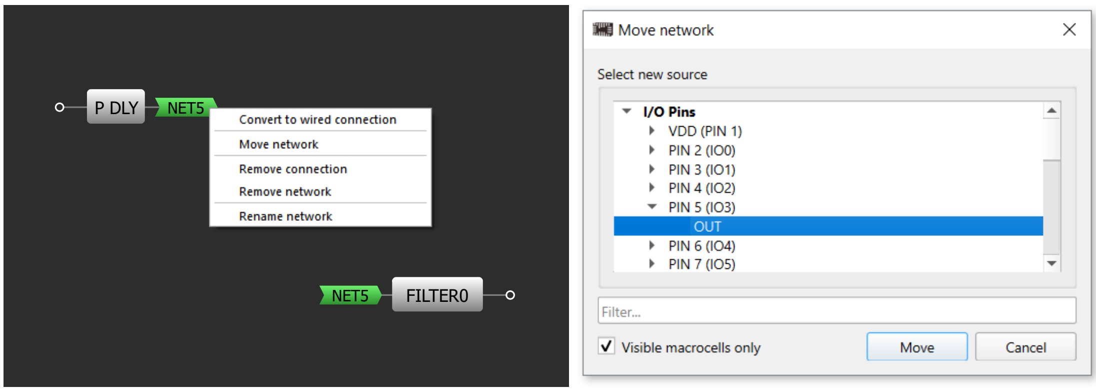


Network: NET5

From: P DLY -> OUT
To: FILTER0 <- IN

Connection/Network hint

You can move the connection/network to another OUT. Select *Move network* from the context menu. Choose the new component in a *Move network* window and click *Move*.

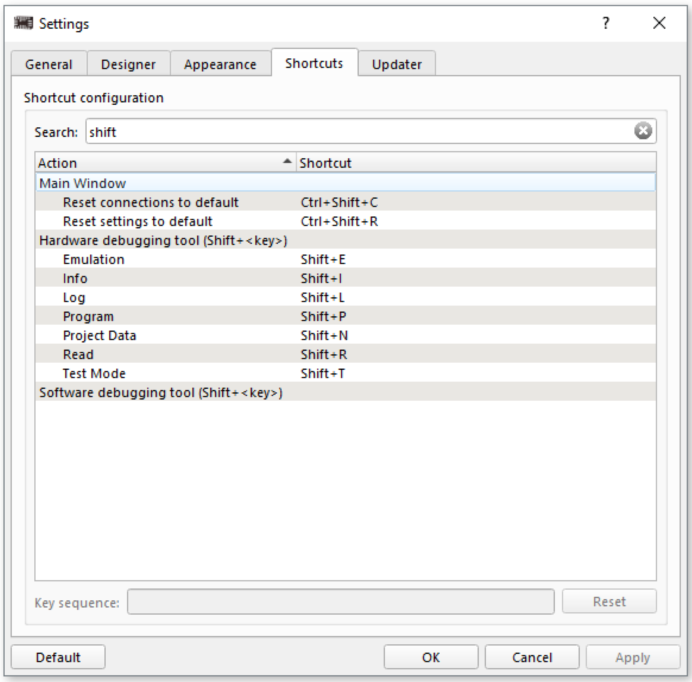


Move connection/network

The information about the components, pins, connections colors and their description is also present in *Go Configure Software Hub*, in *Legend box*. You can read more is section 2.1.8 Legend box.

2.1.6 Keyboard commands

There are configurable and non-configurable keyboard commands. The configurable commands can be managed in *Settings* (for more info, see section 2.1.7 Settings). To find the command easily, use the search field.

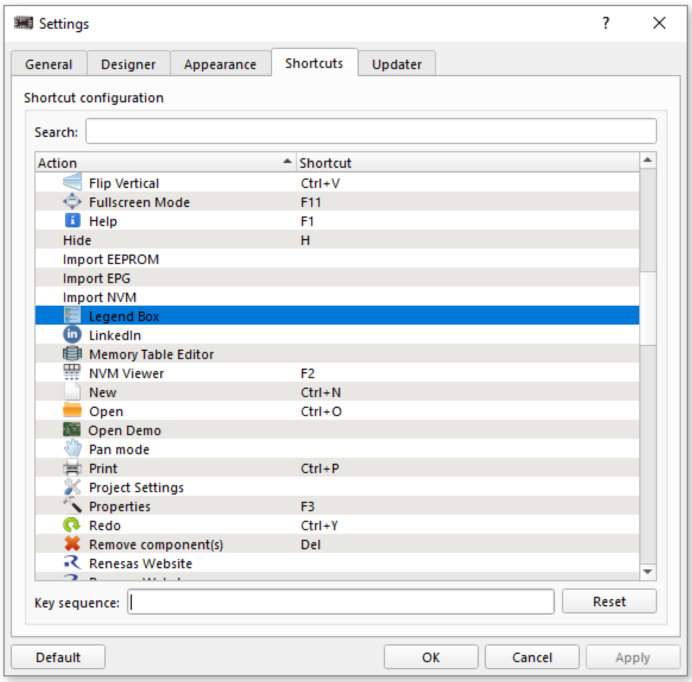


Search: shift

Action	Shortcut
Main Window	
Reset connections to default	Ctrl+Shift+C
Reset settings to default	Ctrl+Shift+R
Hardware debugging tool (Shift+<key>)	
Emulation	Shift+E
Info	Shift+I
Log	Shift+L
Program	Shift+P
Project Data	Shift+N
Read	Shift+R
Test Mode	Shift+T
Software debugging tool (Shift+<key>)	

Shortcuts table

To change or add a shortcut, double-click the action and press the key sequence on your keyboard to add it to the corresponding field. To discard changes, use the *Reset* button.


Settings

General | Designer | Appearance | Shortcuts | Updater

Shortcut configuration

Search:

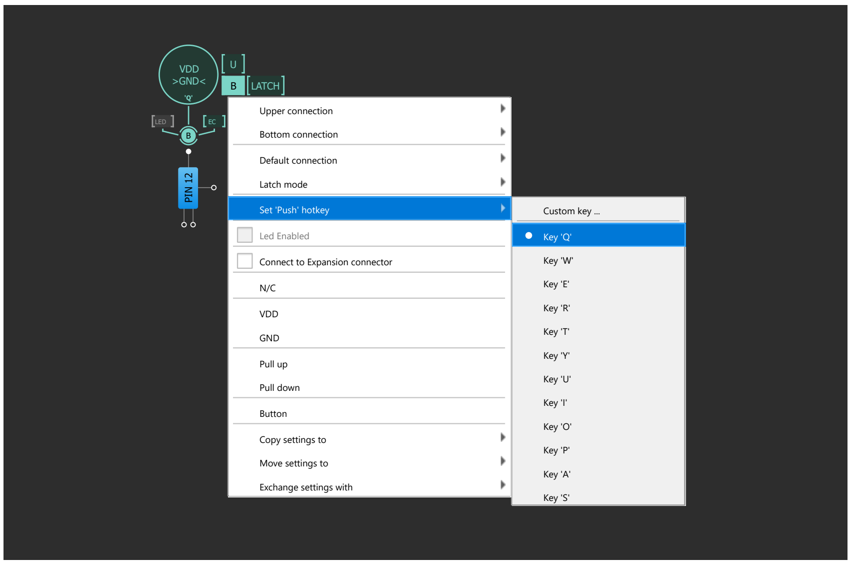
Action	Shortcut
Flip Vertical	Ctrl+V
Fullscreen Mode	F11
Help	F1
Hide	H
Import EEPROM	
Import EPG	
Import NVM	
Legend Box	
LinkedIn	
Memory Table Editor	
NVM Viewer	F2
New	Ctrl+N
Open	Ctrl+O
Open Demo	
Pan mode	
Print	Ctrl+P
Project Settings	
Properties	F3
Redo	Ctrl+Y
Remove component(s)	Del
Renesas Website	

Key sequence: | Reset

Default | OK | Cancel | Apply

Assign/Change shortcut

In *Debug tools*, you can assign a hotkey to some hardware sources via 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.



Hardware sources hotkeys

See the list of configurable and non-configurable hotkeys in section 6.2 Keyboard and mouse controls.

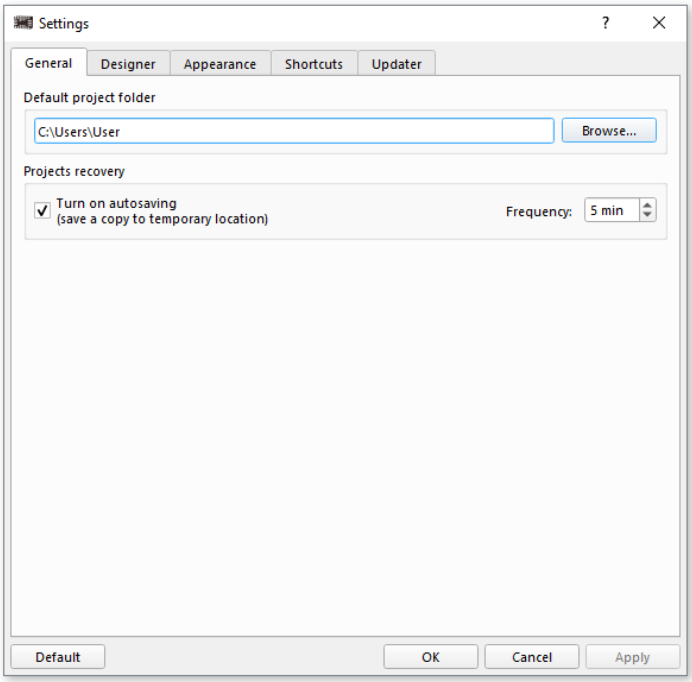
2.1.7 Settings

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*.

General

- *Default projects folder* — define the path to your 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).



Default project folder: C:\Users\User

Projects recovery

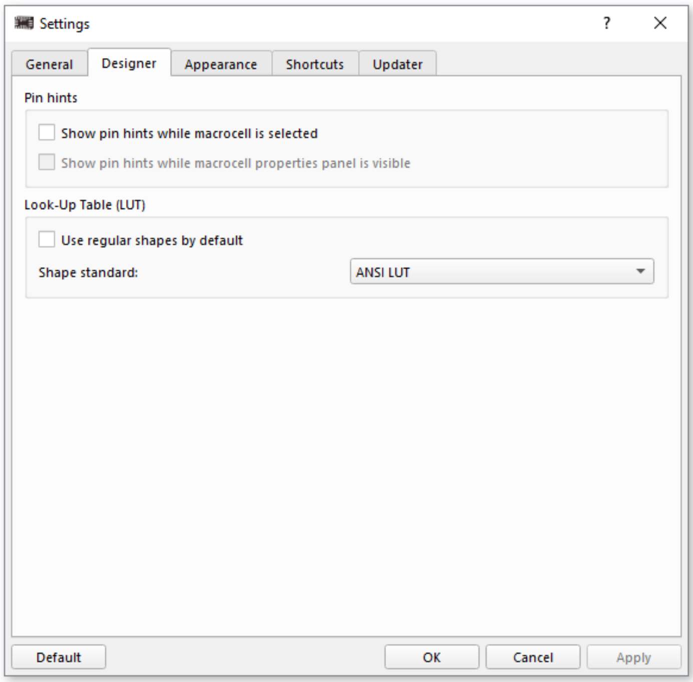
[✓] Turn on autosaving (save a copy to temporary location)

Frequency: 5 min

General tab

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 2.1.4 Components)
- *Look-Up Table (LUT)* — select the preferred LUT shape (regular, ANSI or IEC) using different standard gates



Settings

General | Designer | Appearance | Shortcuts | Updater

Pin hints

☐ Show pin hints while macrocell is selected

☐ Show pin hints while macrocell properties panel is visible

Look-Up Table (LUT)

☐ Use regular shapes by default

Shape standard: ANSI LUT

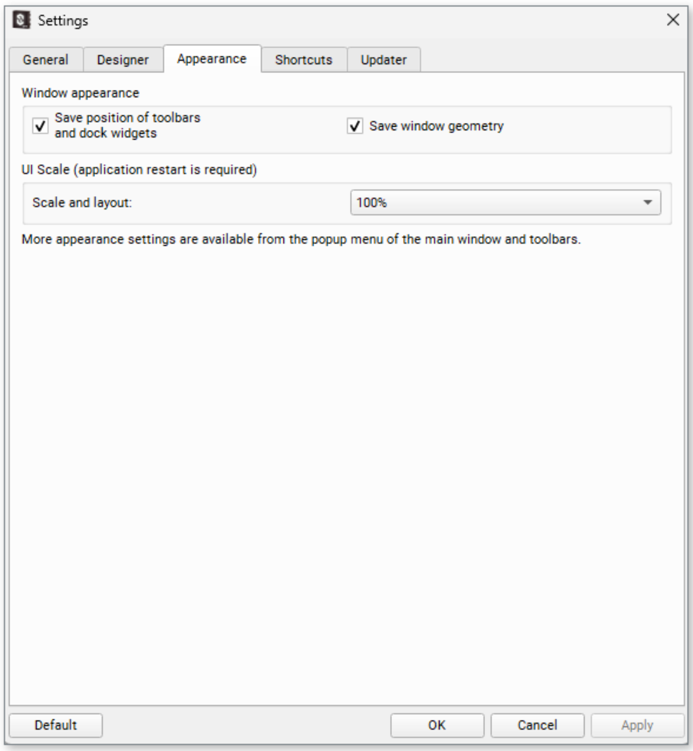
Designer tab

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


Settings

General | Designer | Appearance | Shortcuts | Updater

Window appearance

✓ Save position of toolbars and dock widgets

✓ Save window geometry

UI Scale (application restart is required)

Scale and layout: 100%

More appearance settings are available from the popup menu of the main window and toolbars.

Appearance tab

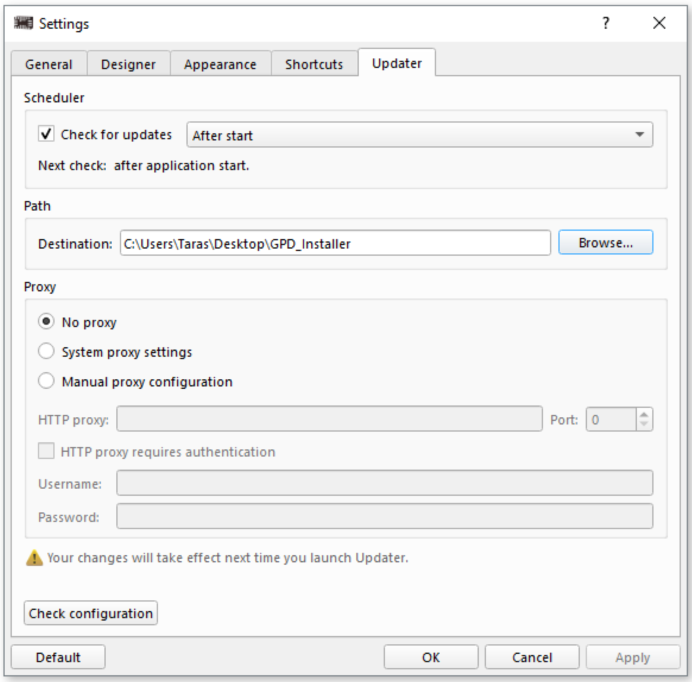
Shortcuts

- *Shortcuts configuration* — assign/change the shortcut for the available actions

Read more about shortcuts in section 2.1.6 Keyboard commands.

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



Scheduler

☑ Check for updates: After start

Next check: after application start.

Path

Destination: C:\Users\Taras\Desktop\GPD_Installer

Proxy

● No proxy
○ System proxy settings
○ Manual proxy configuration

HTTP proxy: Port: 0

☐ HTTP proxy requires authentication

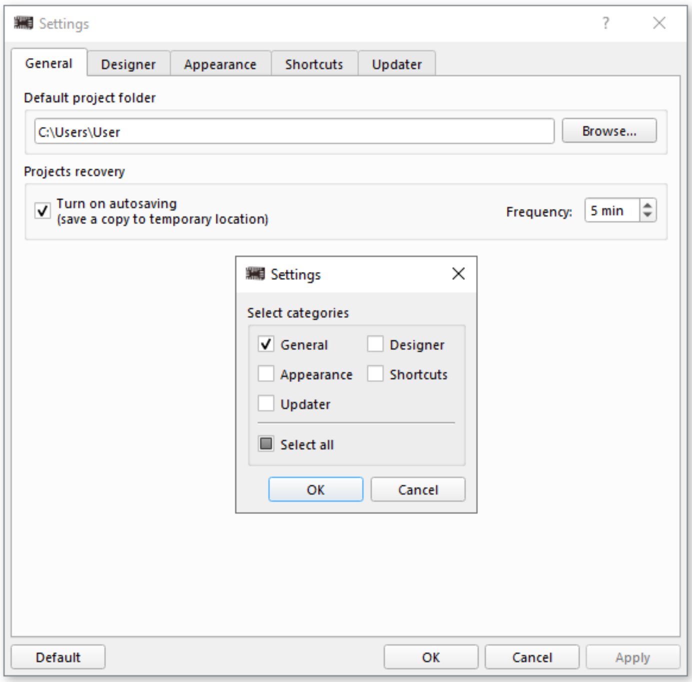
Username:

Password:

Your changes will take effect next time you launch Updater.

Updater tab

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



Default project folder

C:\Users\User

Projects recovery

☑ Turn on autosaving (save a copy to temporary location)

Frequency: 5 min

Select categories

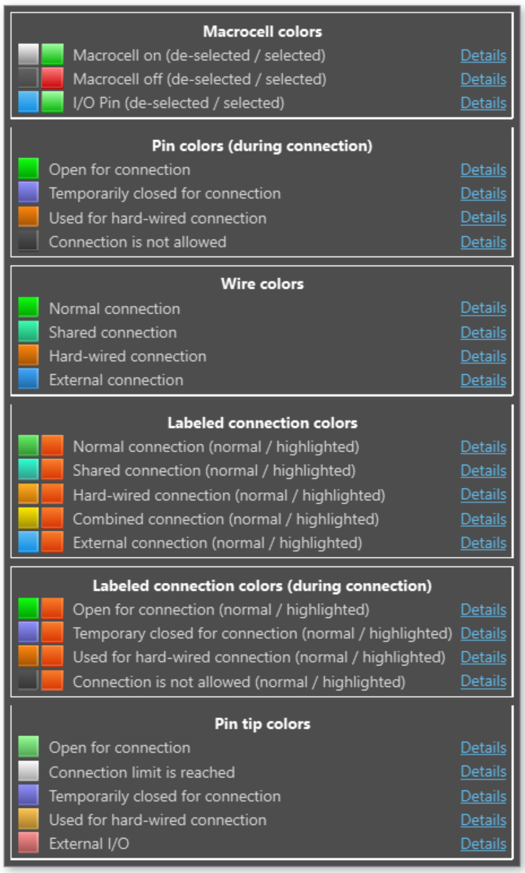
☑ General ☐ Designer
☐ Appearance ☐ Shortcuts
☐ Updater

▪ Select all

Default button

2.1.8 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. You can find it in the main menu, *Help* → *Legend box*.



Macrocell colors

⬜🟩	Macrocell on (de-selected / selected)	Details
⬛🟥	Macrocell off (de-selected / selected)	Details
🟦🟩	I/O Pin (de-selected / selected)	Details

Pin colors (during connection)

🟩	Open for connection	Details
🟪	Temporarily closed for connection	Details
🟧	Used for hard-wired connection	Details
⬛	Connection is not allowed	Details

Wire colors

🟩	Normal connection	Details
🟩	Shared connection	Details
🟧	Hard-wired connection	Details
🟦	External connection	Details

Labeled connection colors

🟩🟧	Normal connection (normal / highlighted)	Details
🟩🟧	Shared connection (normal / highlighted)	Details
🟧🟧	Hard-wired connection (normal / highlighted)	Details
🟨🟧	Combined connection (normal / highlighted)	Details
🟦🟧	External connection (normal / highlighted)	Details

Labeled connection colors (during connection)

🟩🟧	Open for connection (normal / highlighted)	Details
🟪🟧	Temporary closed for connection (normal / highlighted)	Details
🟧🟧	Used for hard-wired connection (normal / highlighted)	Details
⬛🟧	Connection is not allowed (normal / highlighted)	Details

Pin tip colors

🟩	Open for connection	Details
⬜	Connection limit is reached	Details
🟪	Temporarily closed for connection	Details
🟨	Used for hard-wired connection	Details
🟥	External I/O	Details

Legend box

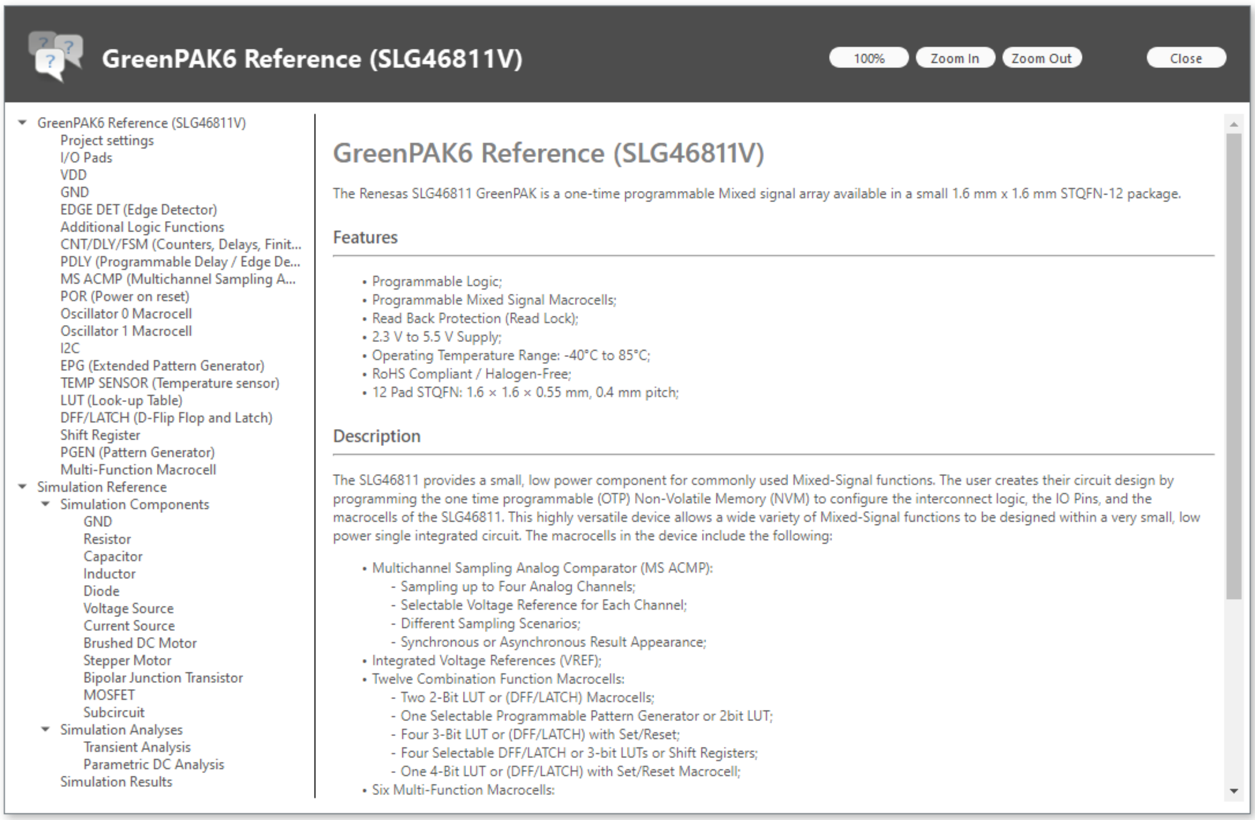
Find out more in sections 2.1.4 Components and 2.1.5 Connections.

2.1.9 Help window

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

If you prefer 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 information you need.


GreenPAK6 Reference (SLG46811V)

100% | Zoom In | Zoom Out | Close

- GreenPAK6 Reference (SLG46811V)
  - Project settings
  - I/O Pads
  - VDD
  - GND
  - EDGE DET (Edge Detector)
  - Additional Logic Functions
  - CNT/DLY/FSM (Counters, Delays, Finit...
  - PDLY (Programmable Delay / Edge De...
  - MS ACMP (Multichannel Sampling A...
  - POR (Power on reset)
  - Oscillator 0 Macrocell
  - Oscillator 1 Macrocell
  - I2C
  - EPG (Extended Pattern Generator)
  - TEMP SENSOR (Temperature sensor)
  - LUT (Look-up Table)
  - DFF/LATCH (D-Flip Flop and Latch)
  - Shift Register
  - PGEN (Pattern Generator)
  - Multi-Function Macrocell
- Simulation Reference
  - Simulation Components
    - GND
    - Resistor
    - Capacitor
    - Inductor
    - Diode
    - Voltage Source
    - Current Source
    - Brushed DC Motor
    - Stepper Motor
    - Bipolar Junction Transistor
    - MOSFET
    - Subcircuit
  - Simulation Analyses
    - Transient Analysis
    - Parametric DC Analysis
  - Simulation Results

GreenPAK6 Reference (SLG46811V)

The Renesas SLG46811 GreenPAK is a one-time programmable Mixed signal array available in a small 1.6 mm x 1.6 mm STQFN-12 package.

Features

- Programmable Logic;
- Programmable Mixed Signal Macrocells;
- Read Back Protection (Read Lock);
- 2.3 V to 5.5 V Supply;
- Operating Temperature Range: -40°C to 85°C;
- RoHS Compliant / Halogen-Free;
- 12 Pad STQFN: 1.6 × 1.6 × 0.55 mm, 0.4 mm pitch;

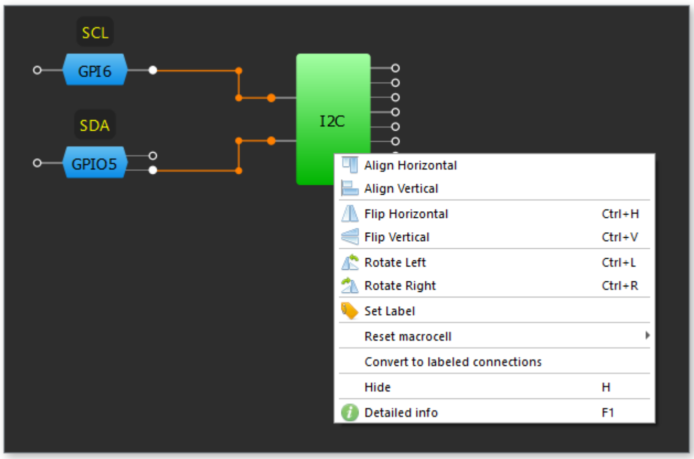
Description

The SLG46811 provides a small, low power component for commonly used Mixed-Signal functions. The user creates their circuit design by programming the one time programmable (OTP) Non-Volatile Memory (NVM) to configure the interconnect logic, the IO Pins, and the macrocells of the SLG46811. This highly versatile device allows a wide variety of Mixed-Signal functions to be designed within a very small, low power single integrated circuit. The macrocells in the device include the following:

- Multichannel Sampling Analog Comparator (MS ACMP):
  - Sampling up to Four Analog Channels;
  - Selectable Voltage Reference for Each Channel;
  - Different Sampling Scenarios;
  - Synchronous or Asynchronous Result Appearance;
- Integrated Voltage References (VREF);
- Twelve Combination Function Macrocells:
  - Two 2-Bit LUT or (DFF/LATCH) Macrocells;
  - One Selectable Programmable Pattern Generator or 2bit LUT;
  - Four 3-Bit LUT or (DFF/LATCH) with Set/Reset;
  - Four Selectable DFF/LATCH or 3-bit LUTs or Shift Registers;
  - One 4-Bit LUT or (DFF/LATCH) with Set/Reset Macrocell;
- Six Multi-Function Macrocells:

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.



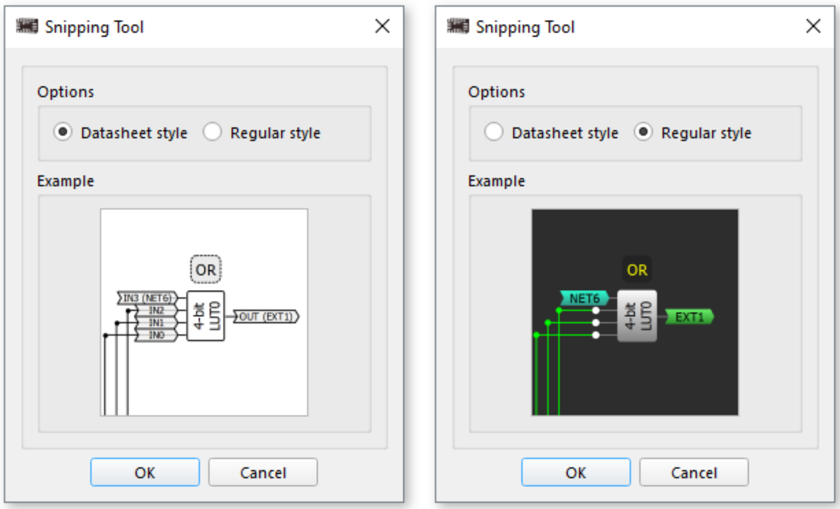
Detailed info

You may also see the information buttons (ⓘ) in different workspace locations, e.g., on a component's *Properties* panel or *Project settings* window. Clicking the button also opens the *Help* window.

2.1.10 Snipping tool

Snipping tool is a work area screenshot-maker. You can find the tool in the main menu, *Tools* → *Snipping tool*.

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


Snipping Tool

Options

[✓ Datasheet style] Regular style

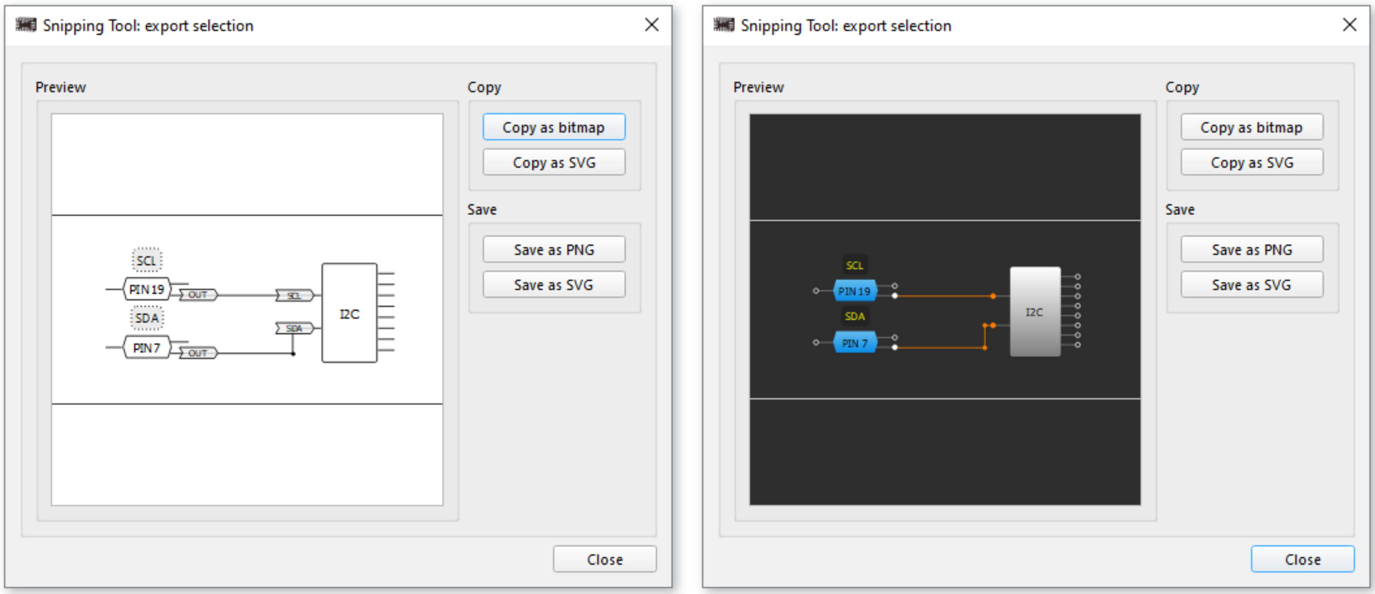
Snipping Tool

Options

Datasheet style [✓ Regular style]

Screenshot style

Once you select the area, copy/save the file in the supported format.



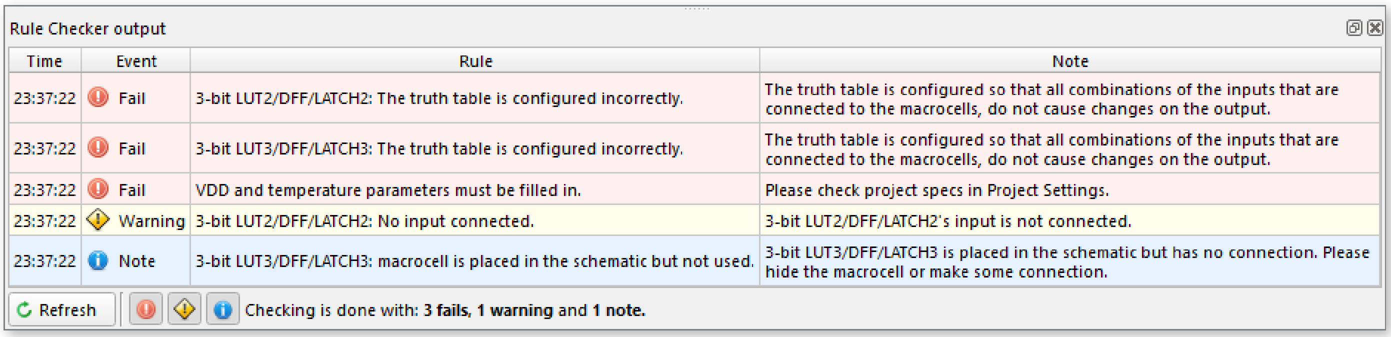
Supported formats

2.1.11 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 (also, find the tool in the main menu → *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



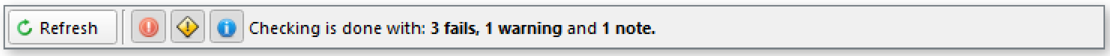
Rule Checker output

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.

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

Rule checker window

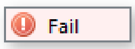
You can use controls to sort the messages by their type. Click the *Refresh* button to see the latest events.



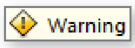
Checking is done with: **3 fails, 1 warning** and **1 note.**

Rule checker controls

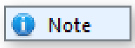
Rule Checker window shows three message types:



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



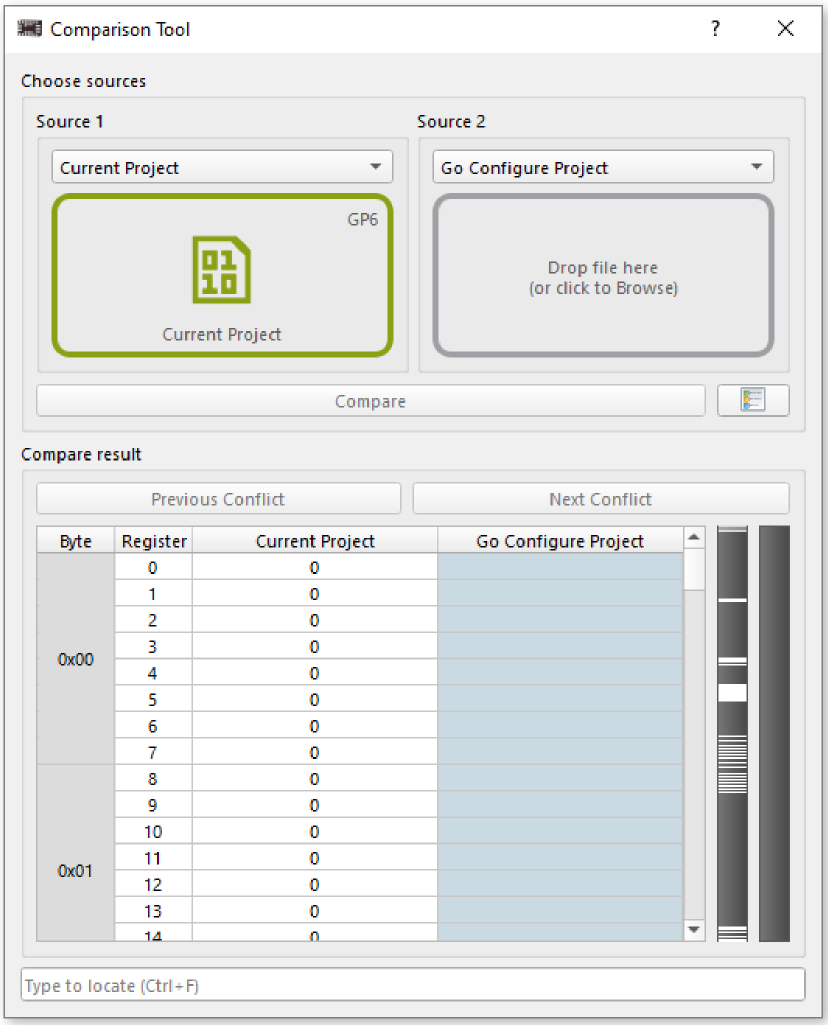
The project contains improper block(s) connections or settings. This message does not necessarily imply an error, but it notifies about a potential problem and urges checking the block(s) connections and settings.



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

2.1.12 Comparison tool

The *Comparison tool* allows you to compare the NVM state of two projects. You can open the tool by clicking the *Comparison tool* widget on the toolbar or by reaching the main menu → *Tools* → *Comparison tool*.



Comparison Tool

Choose sources

Source 1: Current Project — GP6 — Current Project

Source 2: Go Configure Project — Drop file here (or click to Browse)

Compare

Compare result

Previous Conflict | Next Conflict

Byte	Register	Current Project	Go Configure Project
0x00	0	0	
	1	0	
	2	0	
	3	0	
	4	0	
	5	0	
	6	0	
	7	0	
0x01	8	0	
	9	0	
	10	0	
	11	0	
	12	0	
	13	0	
	14	0	

Type to locate (Ctrl+F)

Comparison tool

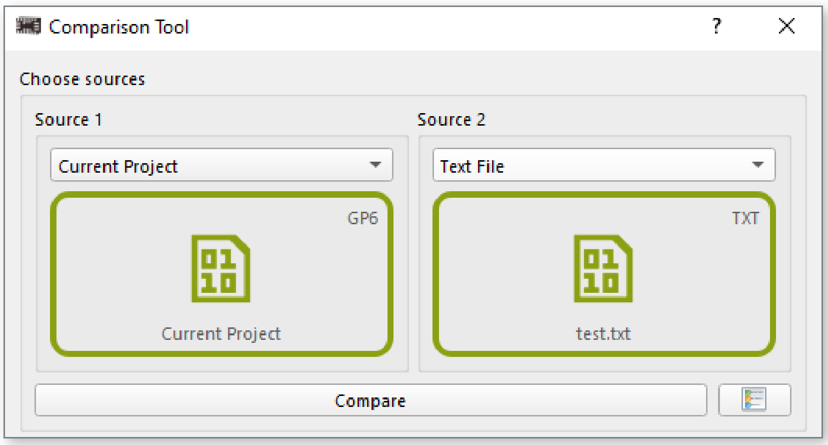
The tool consists of two main parts: *Choose sources* and *Compare result*.

Choose sources

Here you can select the sources for the NVM comparison. 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:

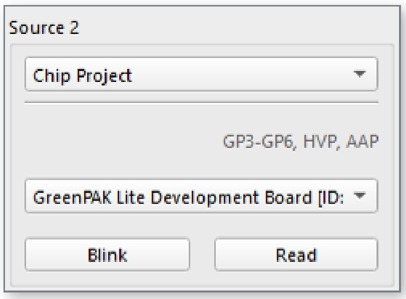
- *Current Project* — current state of your design
- *Go Configure Project* — *Go Configure Software Hub* project file with the following extensions: `.aap`, `.can`, `.gp3`, `.gp4`, `.gp5`, `.gp6`, `.hvp`, and `.ppak`
- *Text File* — exported NVM file in .txt format

➤ *Chip Project* — configurations programmed on a chip



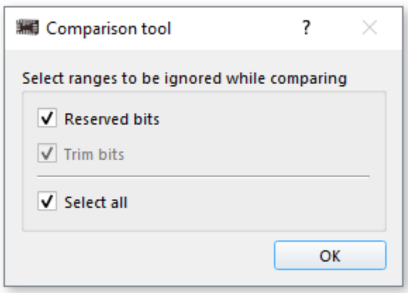
Choose sources

If you select *Chip Project* as a source, make sure you connect the supported platform with the inserted IC to your 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 to check 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.



Chip source

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

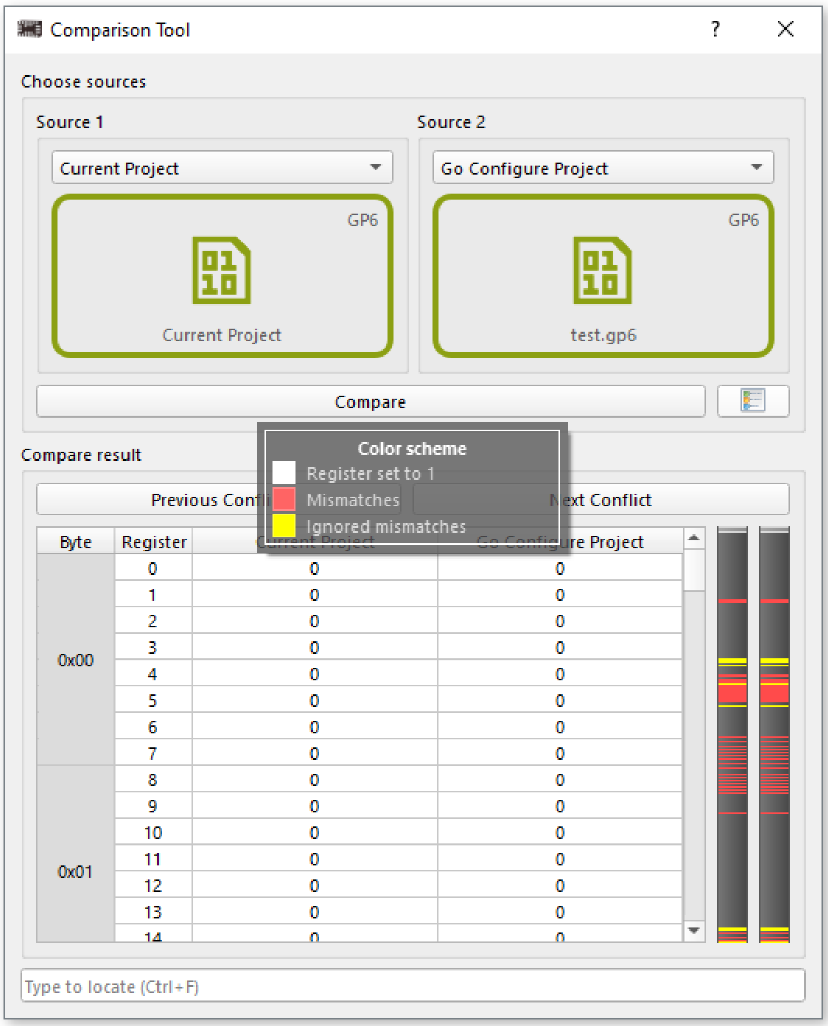


Range selection

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

Compare results

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



Comparison Tool

Choose sources

Source 1: Current Project — GP6 — Current Project

Source 2: Go Configure Project — GP6 — test.gp6

Compare

Compare result

Color scheme
- ⬜ Register set to 1
- 🟥 Mismatches
- 🟨 Ignored mismatches

Previous Conflict | Next Conflict

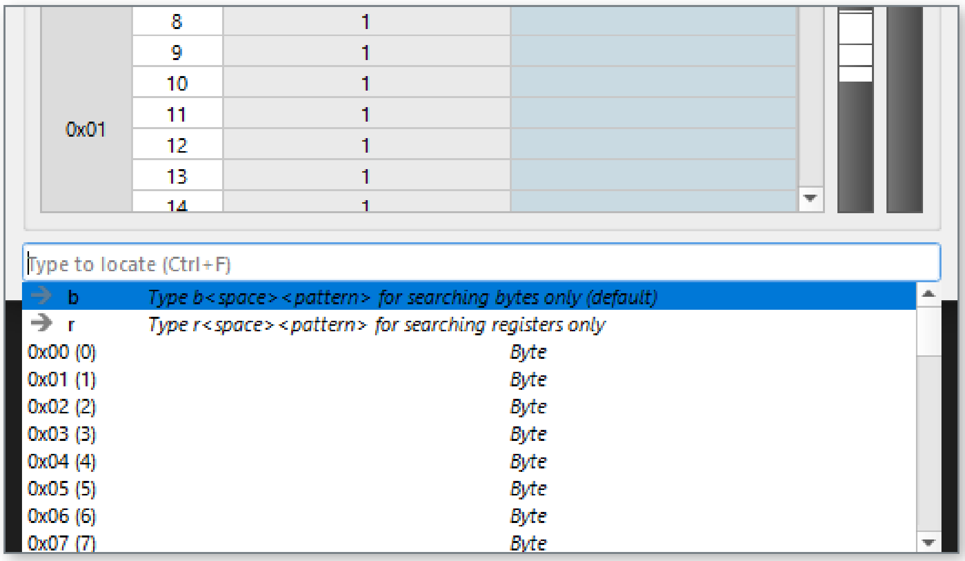
Byte	Register	Current Project	Go Configure Project
0x00	0	0	0
	1	0	0
	2	0	0
	3	0	0
	4	0	0
	5	0	0
	6	0	0
	7	0	0
0x01	8	0	0
	9	0	0
	10	0	0
	11	0	0
	12	0	0
	13	0	0
	14	0	0

Type to locate (Ctrl+F)

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.

You can also use the search field to find the particular byte or register.



0x01	8	1
	9	1
	10	1
	11	1
	12	1
	13	1
	14	1

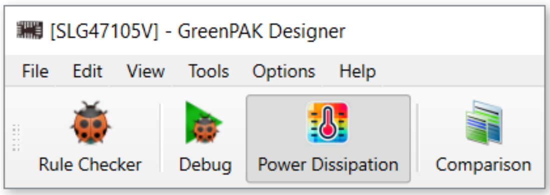
Type to locate (Ctrl+F)

b	*Type b<space><pattern> for searching bytes only (default)*	
r	*Type r<space><pattern> for searching registers only*	
0x00 (0)		*Byte*
0x01 (1)		*Byte*
0x02 (2)		*Byte*
0x03 (3)		*Byte*
0x04 (4)		*Byte*
0x05 (5)		*Byte*
0x06 (6)		*Byte*
0x07 (7)		*Byte*

Search field

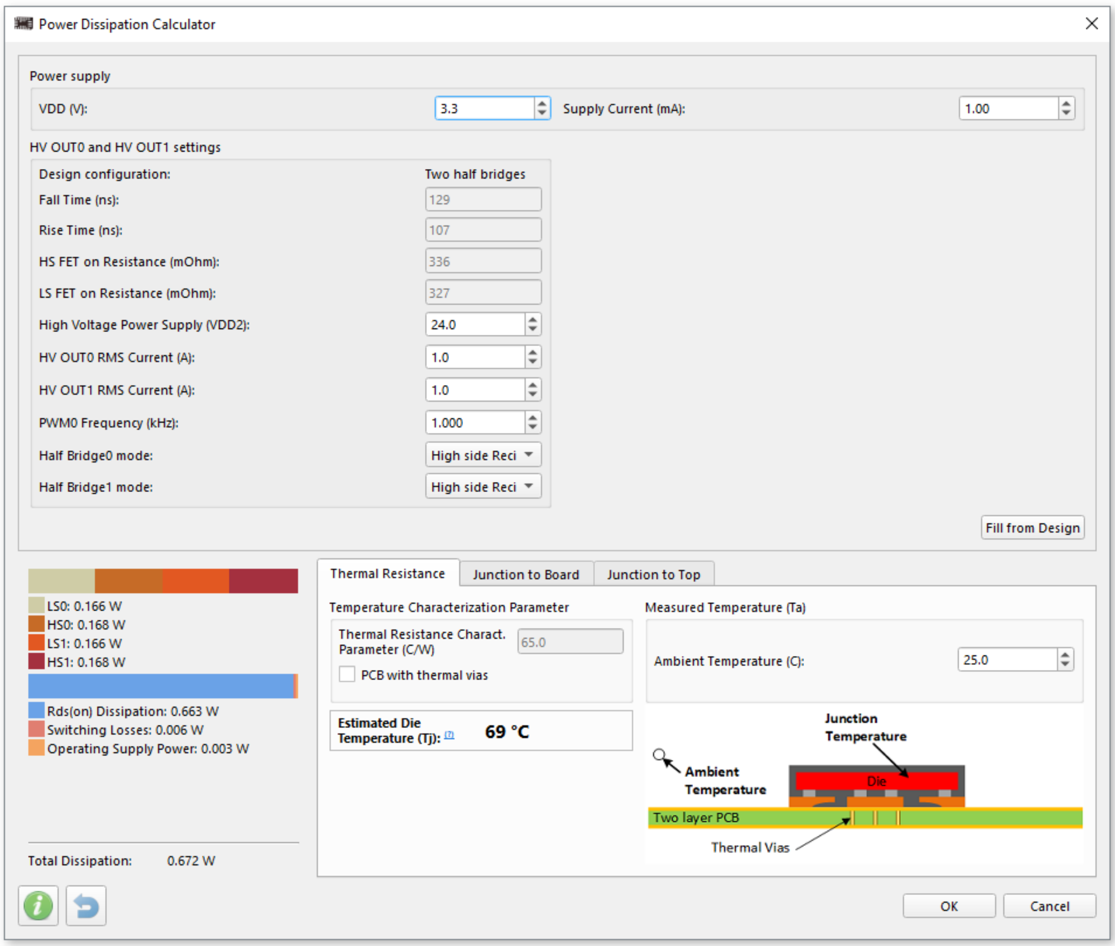
2.1.13 Power Dissipation Calculator

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.



Power Dissipation Calculator on the toolbar

To launch the tool, click the *Power Dissipation* button on the toolbar or find it in the main menu, *Tools*.



Power Dissipation Calculator

Power supply

VDD (V): 3.3 Supply Current (mA): 1.00

HV OUT0 and HV OUT1 settings

Design configuration:	Two half bridges
Fall Time (ns):	129
Rise Time (ns):	107
HS FET on Resistance (mOhm):	336
LS FET on Resistance (mOhm):	327
High Voltage Power Supply (VDD2):	24.0
HV OUT0 RMS Current (A):	1.0
HV OUT1 RMS Current (A):	1.0
PWM0 Frequency (kHz):	1.000
Half Bridge0 mode:	High side Reci
Half Bridge1 mode:	High side Reci

Fill from Design

LS0: 0.166 W
HS0: 0.168 W
LS1: 0.166 W
HS1: 0.168 W

Rds(on) Dissipation: 0.663 W
Switching Losses: 0.006 W
Operating Supply Power: 0.003 W

Total Dissipation: 0.672 W

Thermal Resistance | Junction to Board | Junction to Top

Temperature Characterization Parameter

Thermal Resistance Charact. Parameter (C/W) 65.0

PCB with thermal vias

Estimated Die Temperature (Tj): 69 °C

Measured Temperature (Ta)

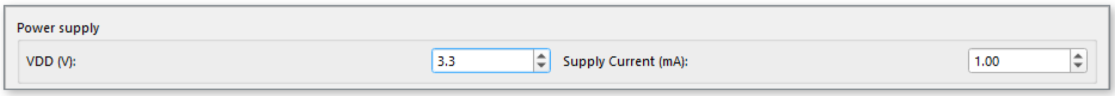
Ambient Temperature (C): 25.0

OK Cancel

Power Dissipation Calculator preview window

The tool consists of four main parts:

- *Power supply* — voltage applied to VDD pin and estimated current consumption parameters



Power supply

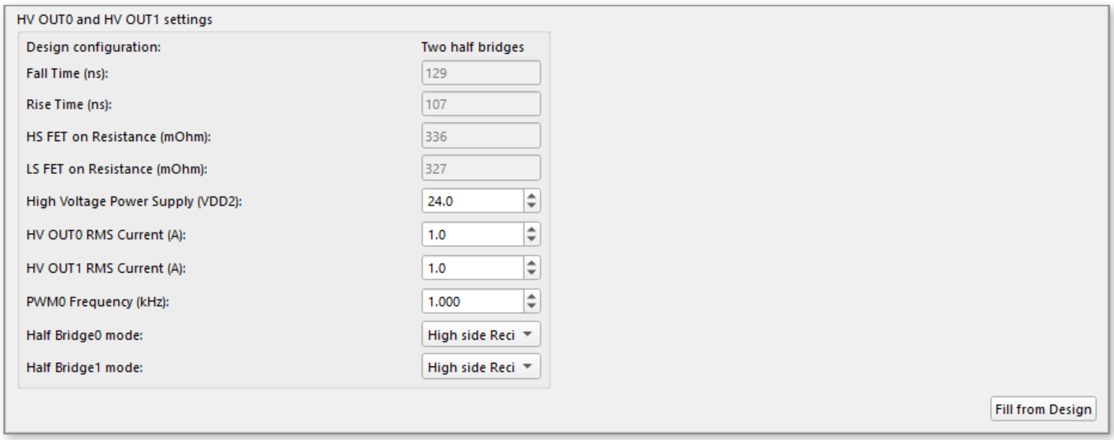
VDD (V): 3.3 Supply Current (mA): 1.00

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 CTRL

– *HV OUT RMS* — 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 CTRL operates. You can set the frequency value using the PWM 0 macrocell. Additionally, you can automatically transfer the frequency value using the *Fill from Design* button in the lower right corner of the *HV OUT 0 and HV OUT 1 settings* section

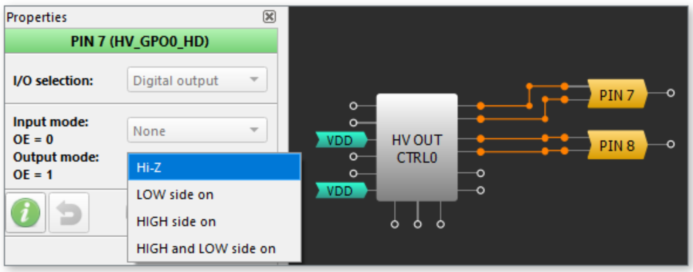


HV OUT0 and HV OUT1 settings

Design configuration:	Two half bridges
Fall Time (ns):	129
Rise Time (ns):	107
HS FET on Resistance (mOhm):	336
LS FET on Resistance (mOhm):	327
High Voltage Power Supply (VDD2):	24.0
HV OUT0 RMS Current (A):	1.0
HV OUT1 RMS Current (A):	1.0
PWM0 Frequency (kHz):	1.000
Half Bridge0 mode:	High side Reci
Half Bridge1 mode:	High side Reci

HV OUT 0 and HV OUT 1 settings

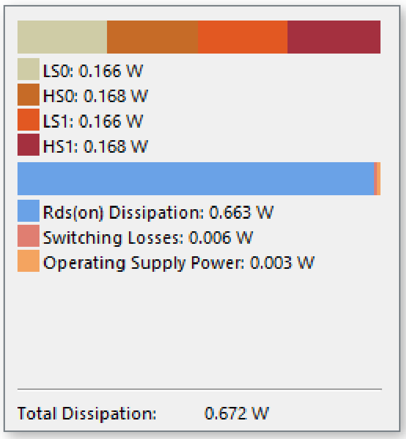
Note: When the output pins connected to the HV OUT CTRL macrocell are in the *Hi-Z* state, *HV OUT 0 and HV OUT 1 settings* are displayed as locked. Change the HV Pin's *Output mode* on its *Properties* panel to unlock the settings.



Properties
PIN 7 (HV_GPO0_HD)
I/O selection: Digital output
Input mode: OE = 0: None
Output mode: OE = 1: Hi-Z

HV OUT properties panel

➤ *Power dissipation bar* — indication of the power dissipation for each transistor: *LSx (low side)* and *HSx (high side)*. Below, you can see the total power dissipation, including open transistor resistance Rds(on), transistor switching losses, and the power dissipated by the chip even when the output pins of HV OUT CTRL are in *Hi-Z*.

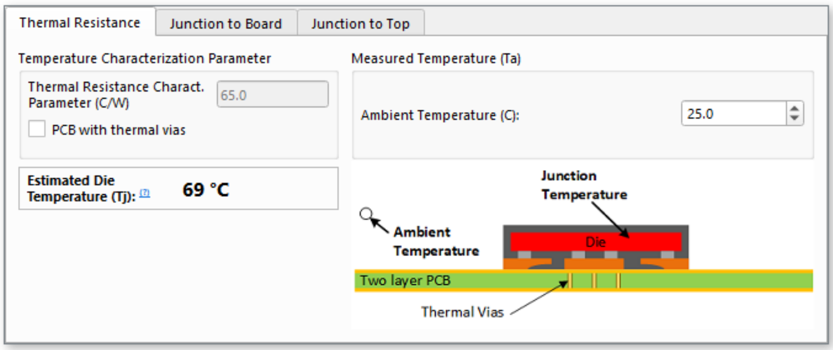

LS0: 0.166 W
HS0: 0.168 W
LS1: 0.166 W
HS1: 0.168 W
Rds(on) Dissipation: 0.663 W
Switching Losses: 0.006 W
Operating Supply Power: 0.003 W

Total Dissipation: 0.672 W

Power dissipation bar

➤ *Temperature Characterization* — provides three options for calculating the device *Die* temperature:

– *Thermal Resistance* — you need only the ambient temperature of the device for calculations. This method is the least accurate. If the platform you are using contains thermal vias, you can tick the respective checkbox to take it into account


Thermal Resistance | Junction to Board | Junction to Top

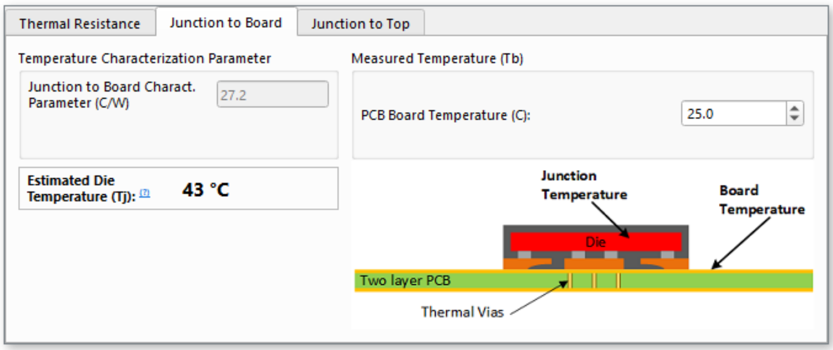
Temperature Characterization Parameter
Thermal Resistance Charact. Parameter (C/W): 65.0
PCB with thermal vias

Estimated Die Temperature (Tj): 69 °C

Measured Temperature (Ta)
Ambient Temperature (C): 25.0

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*


Thermal Resistance | Junction to Board | Junction to Top

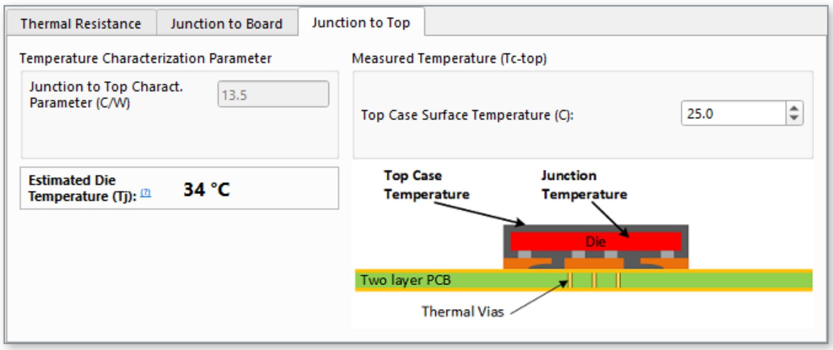
Temperature Characterization Parameter
Junction to Board Charact. Parameter (C/W): 27.2

Estimated Die Temperature (Tj): 43 °C

Measured Temperature (Tb)
PCB Board Temperature (C): 25.0

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.

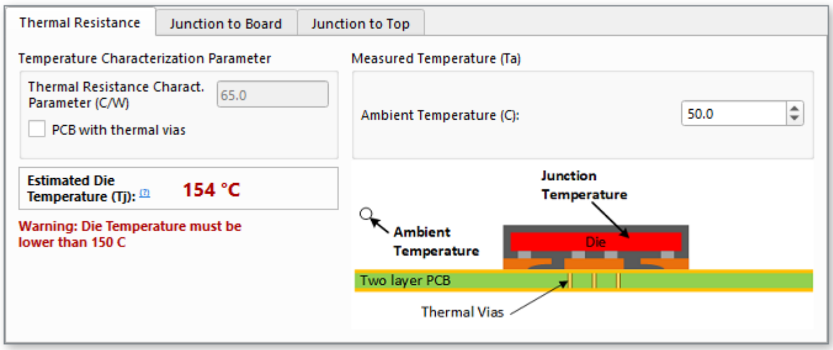

Thermal Resistance | Junction to Board | Junction to Top

Temperature Characterization Parameter
Junction to Top Charact. Parameter (C/W): 13.5
Estimated Die Temperature (Tj): 34 °C

Measured Temperature (Tc-top)
Top Case Surface Temperature (C): 25.0

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.


Thermal Resistance | Junction to Board | Junction to Top

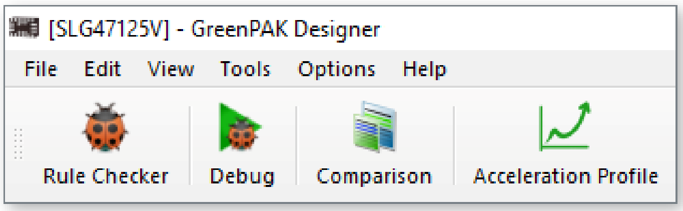
Temperature Characterization Parameter
Thermal Resistance Charact. Parameter (C/W): 65.0
☐ PCB with thermal vias
Estimated Die Temperature (Tj): 154 °C
Warning: Die Temperature must be lower than 150 C

Measured Temperature (Ta)
Ambient Temperature (C): 50.0

Overheating warning

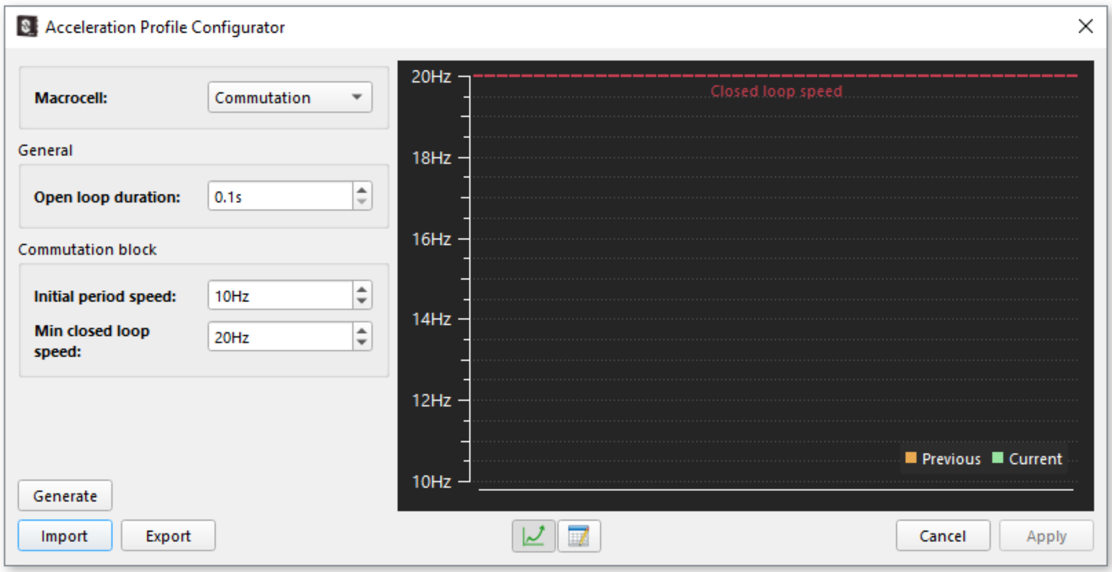
2.1.14 Acceleration Profile Configurator

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 find it in the main menu, *Tools*.


[SLG47125V] - GreenPAK Designer
File Edit View Tools Options Help
Rule Checker
Debug
Comparison
Acceleration Profile

Acceleration Profile on the toolbar

Once you select the desired macrocell, set the parameters for calculations.



Macrocell: Commutation

General

Open loop duration: 0.1s

Commutation block

Initial period speed: 10Hz

Min closed loop speed: 20Hz

Acceleration Profile Configurator tool

Commutation

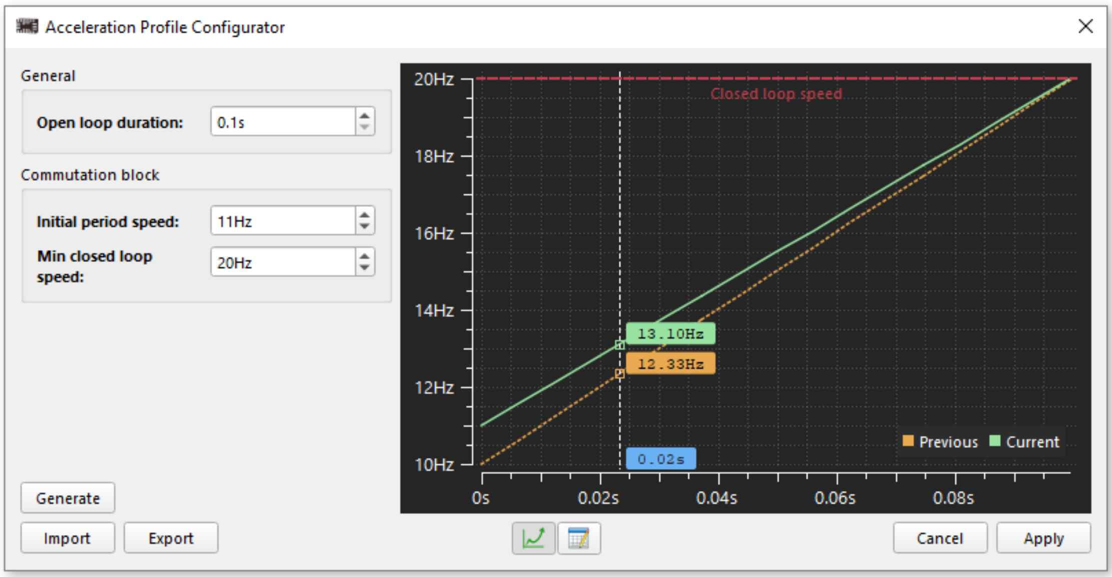
Below, you can see the parameters available for the Commutation macrocell.

- *Open loop duration* — time interval of motor operation (x-axis on the graph)
- *Initial period speed* and *Min closed loop speed* — motor rotation frequency range (y-axis on the graph). Both parameters can be configured in the tool directly or on the Commutation block *Properties* panel, since they are syncronized

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

The results are represented in the following ways:

- *Waveform* — graphical visualization of motor operation. It provides the following information and possibilities:
  - x-axis (time interval set as *Open loop duration*) and y-axis (frequency range set as *Initial period speed* and *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`


Acceleration Profile Configurator

General

Open loop duration: 0.1s

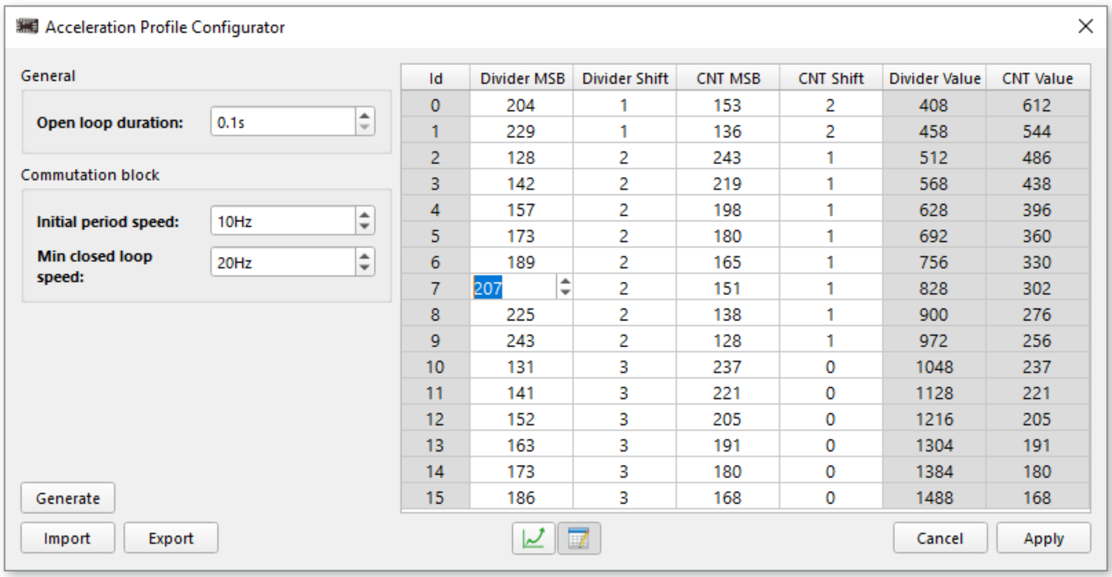
Commutation block

Initial period speed: 11Hz

Min closed loop speed: 20Hz

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 manually modify the *MSB* and *Shift* cells' data by double-clicking the cell


Acceleration Profile Configurator

General

Open loop duration: 0.1s

Commutation block

Initial period speed: 10Hz

Min closed loop speed: 20Hz

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

Manual data editing for Commutation table

Timers

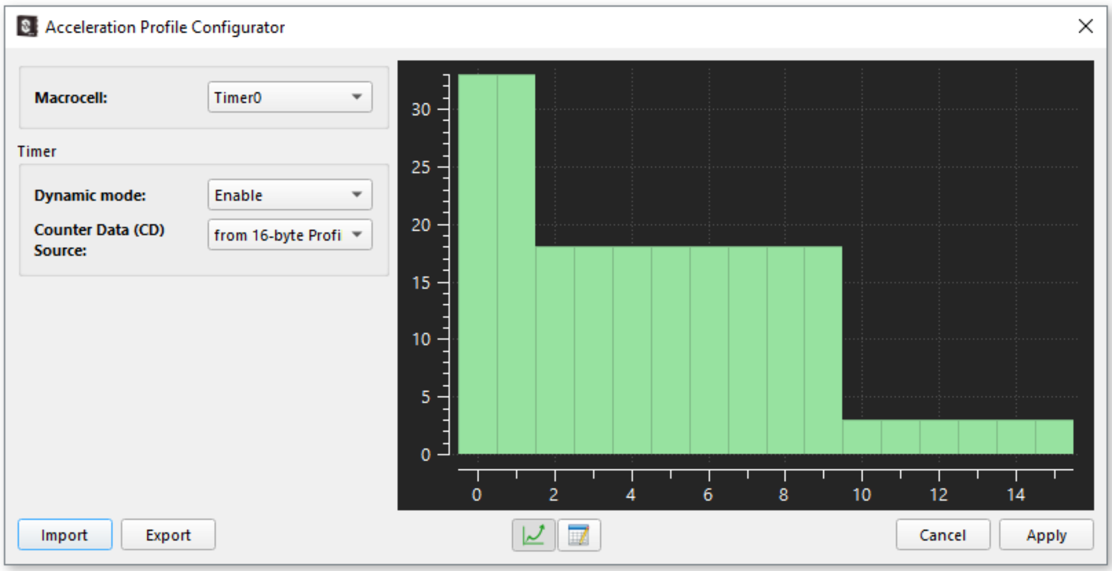
The Timer macrocell parameters in the tool are also synchronized with the ones on the Timers *Properties* panel.

➤ *Dynamic mode* — enable the mode to activate the *Counter Data (CD) Source* parameter

➤ *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:

- *Waveform* — visualizes the counter data values



Macrocell: Timer0

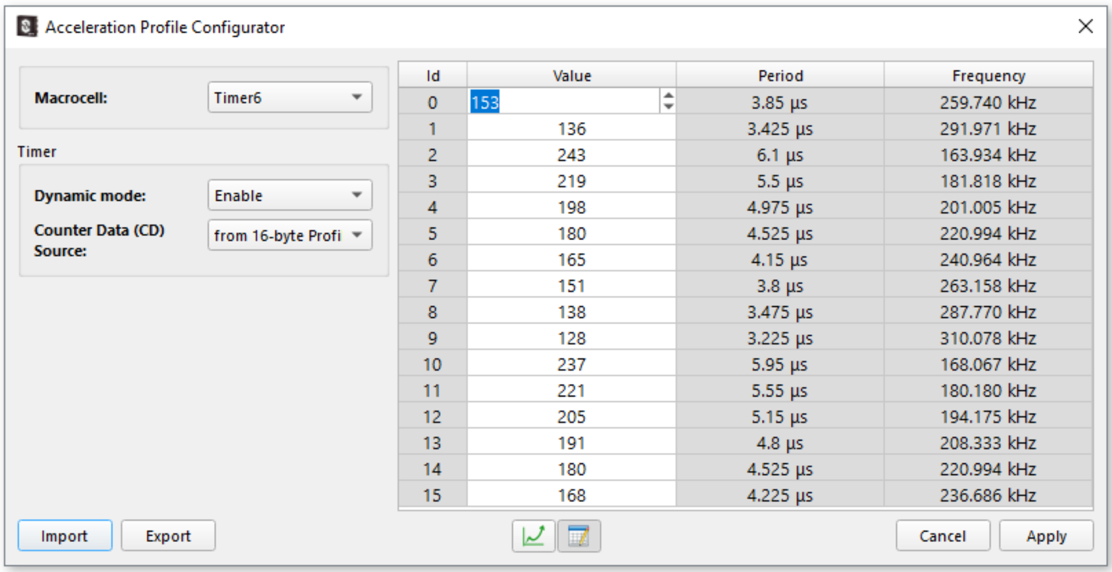
Timer

Dynamic mode: Enable

Counter Data (CD) Source: from 16-byte Profi

Waveform

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



Macrocell: Timer6

Timer

Dynamic mode: Enable

Counter Data (CD) Source: from 16-byte Profi

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

Timer macrocell table

Import/Export the calculation results in .csv or .txt format using the respective bottom buttons.

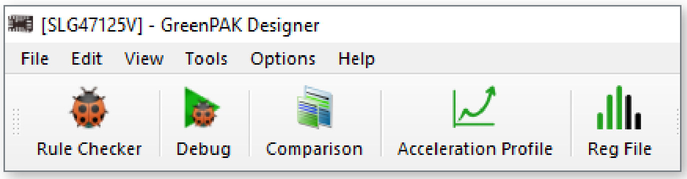
Note 1: If you wish to write the table data to NVM click the *Apply* button.

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

2.1.15 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*.


[SLG47125V] - GreenPAK Designer
File Edit View Tools Options Help
Rule Checker
Debug
Comparison
Acceleration Profile
Reg File

Reg File Configurator on the toolbar

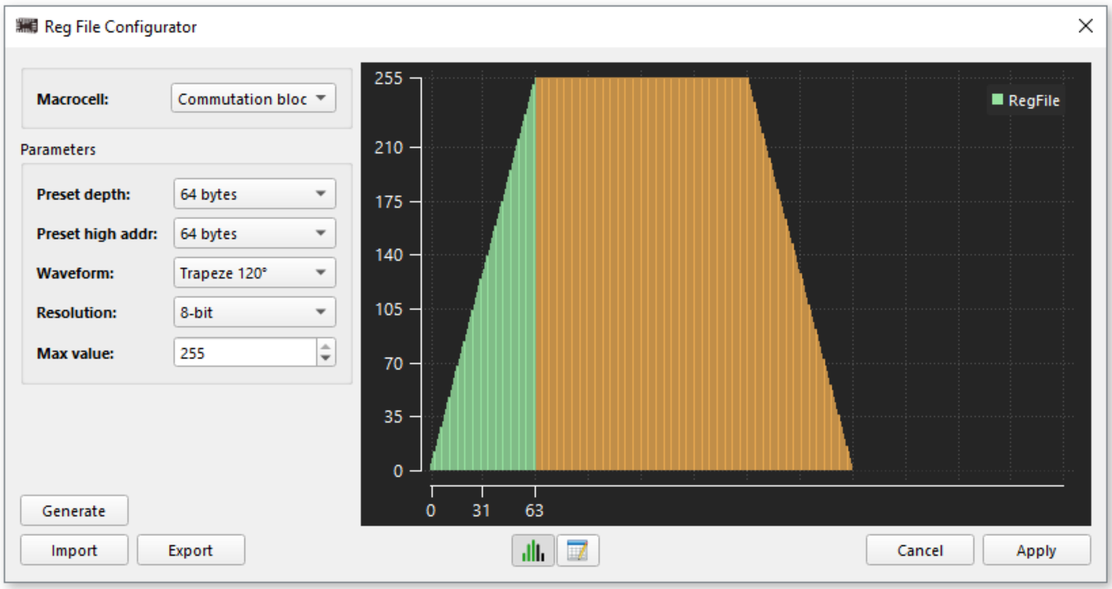
The set of parameters allows you to:

- 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:

- *Waveform* — 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



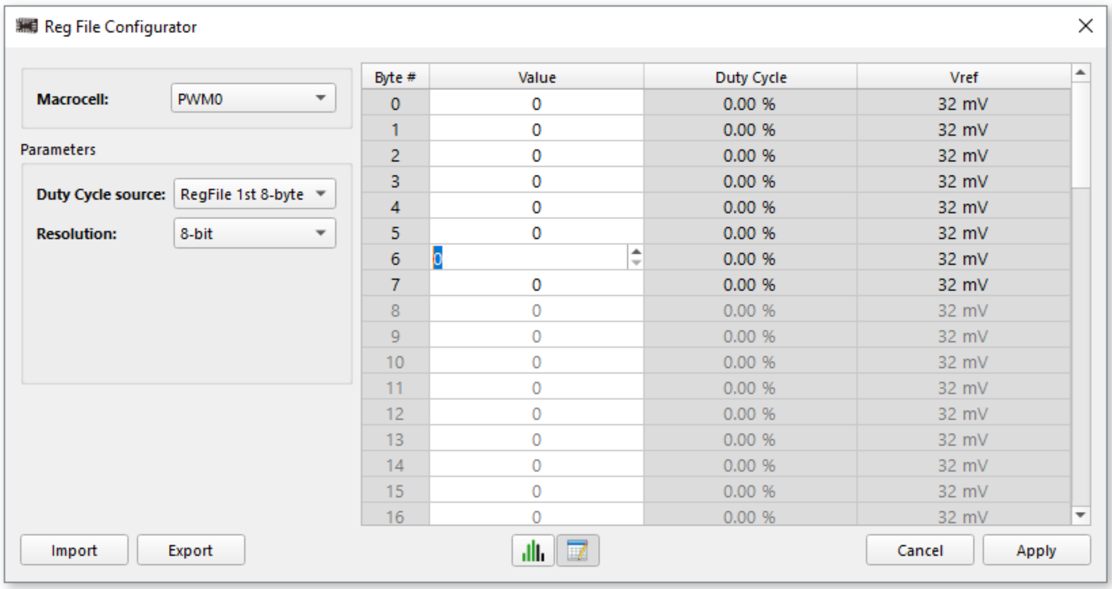
Macrocell: Commutation bloc

Parameters

Preset depth:	64 bytes
Preset high addr:	64 bytes
Waveform:	Trapeze 120°
Resolution:	8-bit
Max value:	255

Reg File Configurator window

➤ *Table* — in addition to generated values, you can also manually modify the *Value* column cells upon double-clicking



Byte #	Value	Duty Cycle	Vref
0	0	0.00 %	32 mV
1	0	0.00 %	32 mV
2	0	0.00 %	32 mV
3	0	0.00 %	32 mV
4	0	0.00 %	32 mV
5	0	0.00 %	32 mV
6	0	0.00 %	32 mV
7	0	0.00 %	32 mV
8	0	0.00 %	32 mV
9	0	0.00 %	32 mV
10	0	0.00 %	32 mV
11	0	0.00 %	32 mV
12	0	0.00 %	32 mV
13	0	0.00 %	32 mV
14	0	0.00 %	32 mV
15	0	0.00 %	32 mV
16	0	0.00 %	32 mV

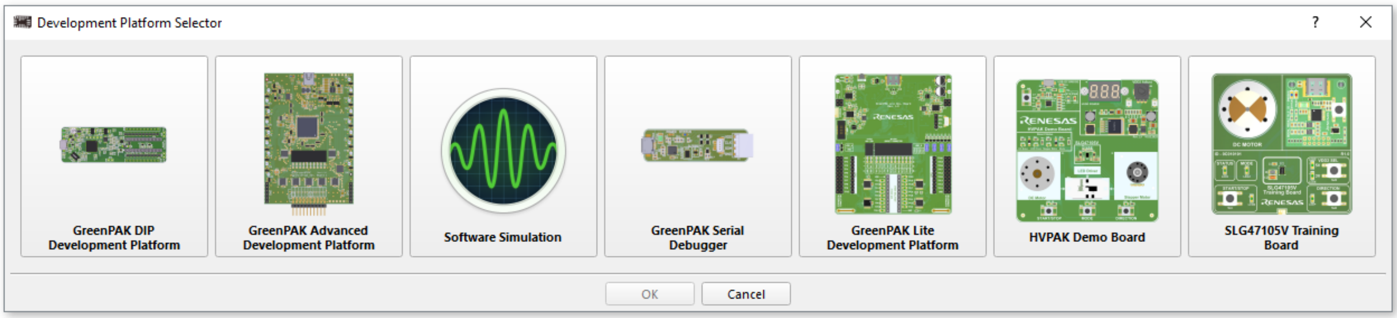
Manual data editing

Import/Export the calculation results in .csv or .txt format using the respective bottom buttons.

Note: If you wish to write the table data to NVM click the *Apply* button.

2.2 Debug tools

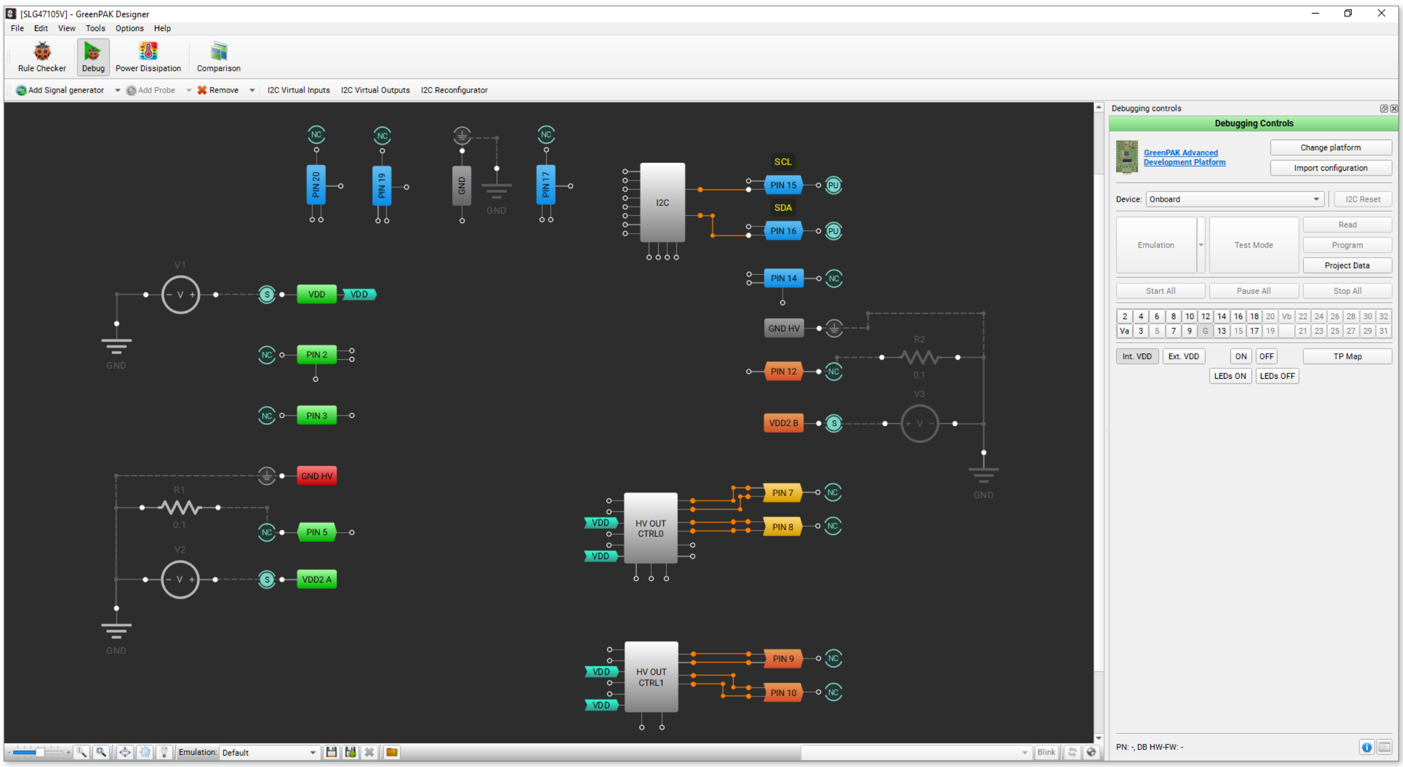
Debug tools are a set of instruments, that allow you to test and debug your design. To access *Debug tools*, click the *Debug* button on the toolbar. Since there are different hardware platforms available for a specific Part Number, select the platform most suitable for your project.


Development Platform Selector
GreenPAK DIP Development Platform
GreenPAK Advanced Development Platform
Software Simulation
GreenPAK Serial Debugger
GreenPAK Lite Development Platform
HVPAK Demo Board
SLG47105V Training Board
OK
Cancel

Platform selector window for SLG47105V

Note: You can only use the hardware in one application instance. If you wish to transfer control from one instance to another, disable *Debug tools* on the controlling instance.

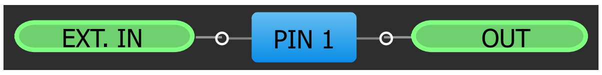
After you select a platform, the software activates a toolbar and a panel with controls for main procedures, including emulation and chip programming.



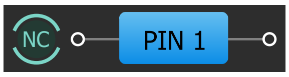
Software UI after Debug is enabled

2.2.1 Hardware sources

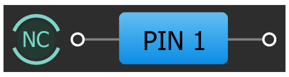
Go Configure Software Hub provides software tools to configure varied hardware sources that manage or generate input and test signals for a chip. Each signal source is connected to an external pin on a chip.



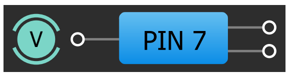
Debug ON →



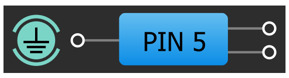
Below, you can find the complete list of all the hardware sources.



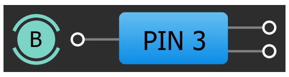
No source (Not Connected)



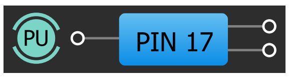
VDD



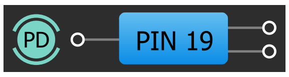
GND



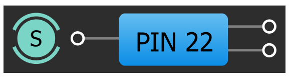
Button



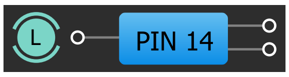
Pull up



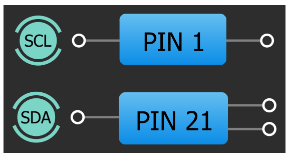
Pull down



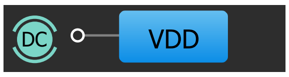
Signal Generator



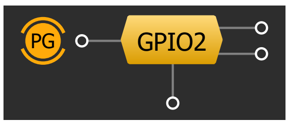
Logic Generator



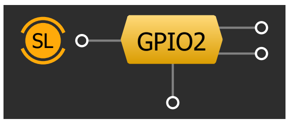
I2C Generator



Constant Voltage



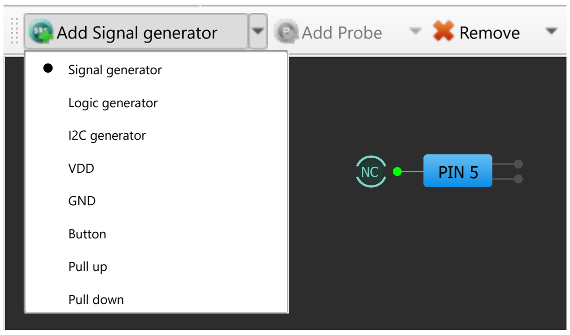
Parametric Generator



Synchronous Logic Generator

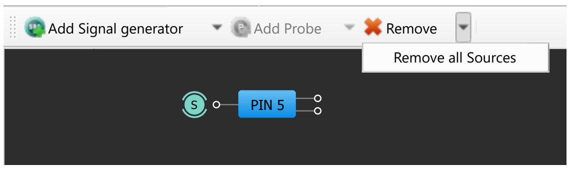
Note: The choice of hardware sources depends on the development platform features and chip restrictions.

You can switch the controls on/off via the context menu or from the toolbar via the *Add Signal Generator* or *Add VDD* button (the button view depends on the available sources).



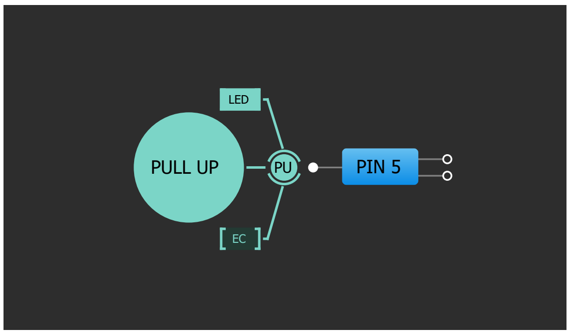
Debug toolbar

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

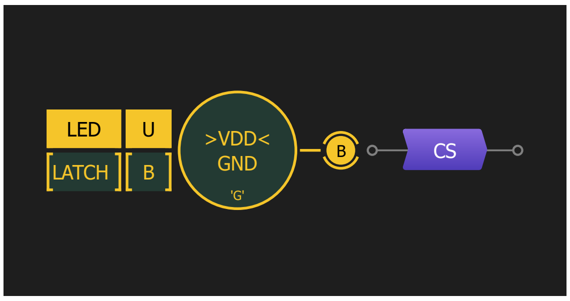


Remove button on Debug toolbar

Some of the chip pins may have additional controls, such as *LED indicators* or *Expansion Connectors (ECs)*, which are available even if a development platform is not connected yet. You can enable/disable the controls by hovering over the source and clicking the control you need.

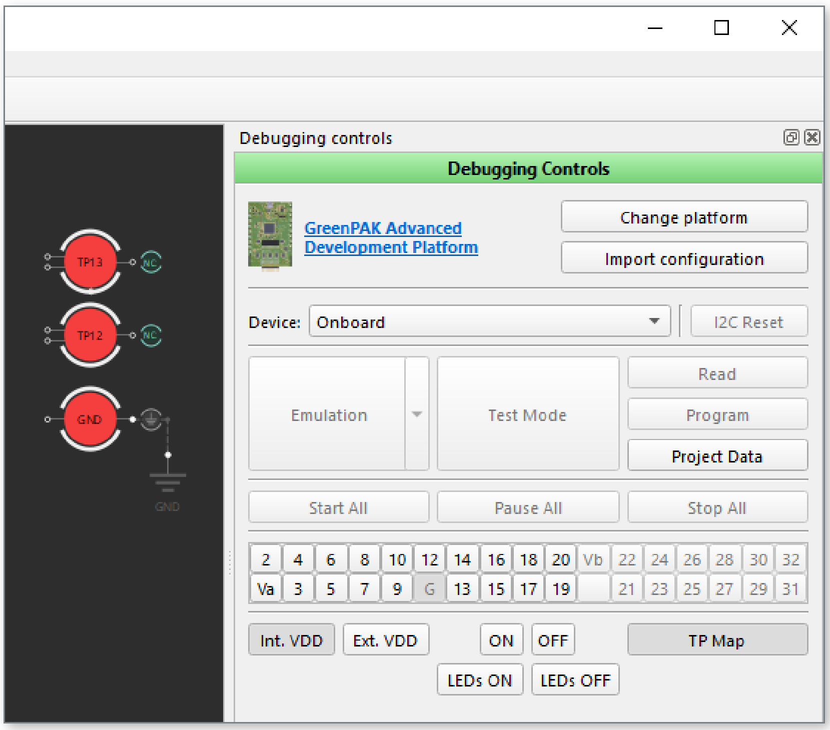


Buffered LED



Power GreenPAK buffered LED

If you need to identify the chip pin connected to a specific test point on a board, use the *Test Point (TP) Map* tool. Note that the pins mapping to test points varies based on the chip type.



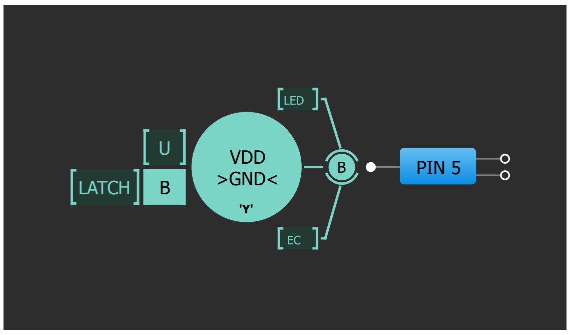
TP map

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 the section 2.2.2 Generators. Later in this chapter you can read about the *basic Hardware Sources*.

Basic Hardware Sources

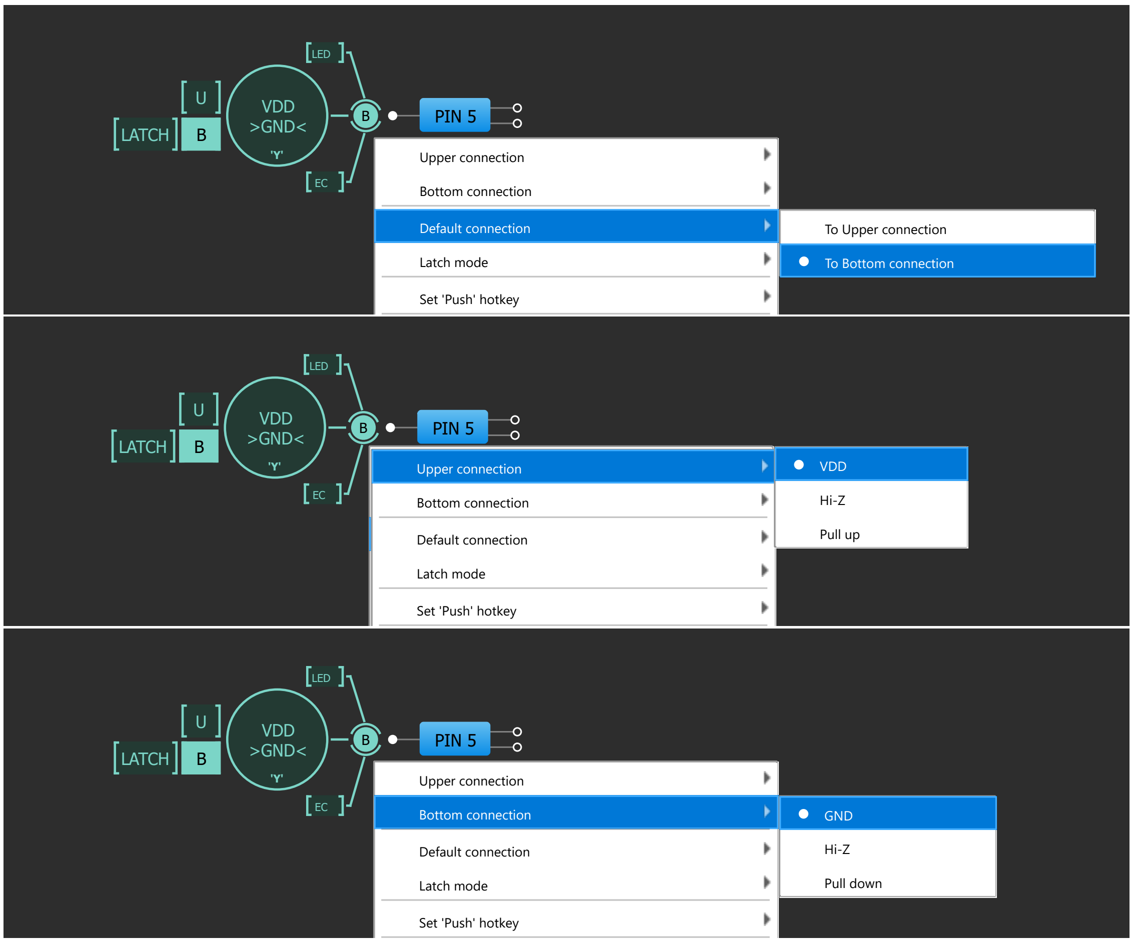
The available *basic Hardware Sources* are: *VDD*, *GND*, *Pull up*, *Pull down*, and *Button*. Unlike others, *Button* has more configuration possibilities (for the rest of the basic sources, all necessary information is already described above).

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*, or *Pull Up/Down*. Hover the mouse cursor over the *Button* control to see its configuration.



Configurable Button

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

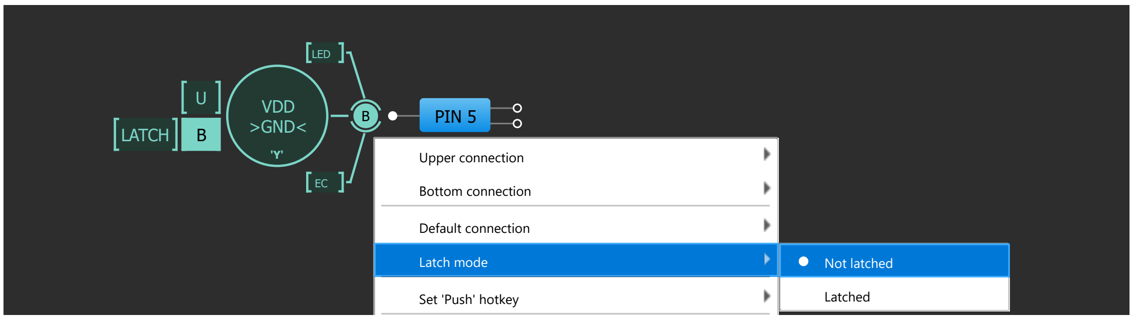


Default Key Connection

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

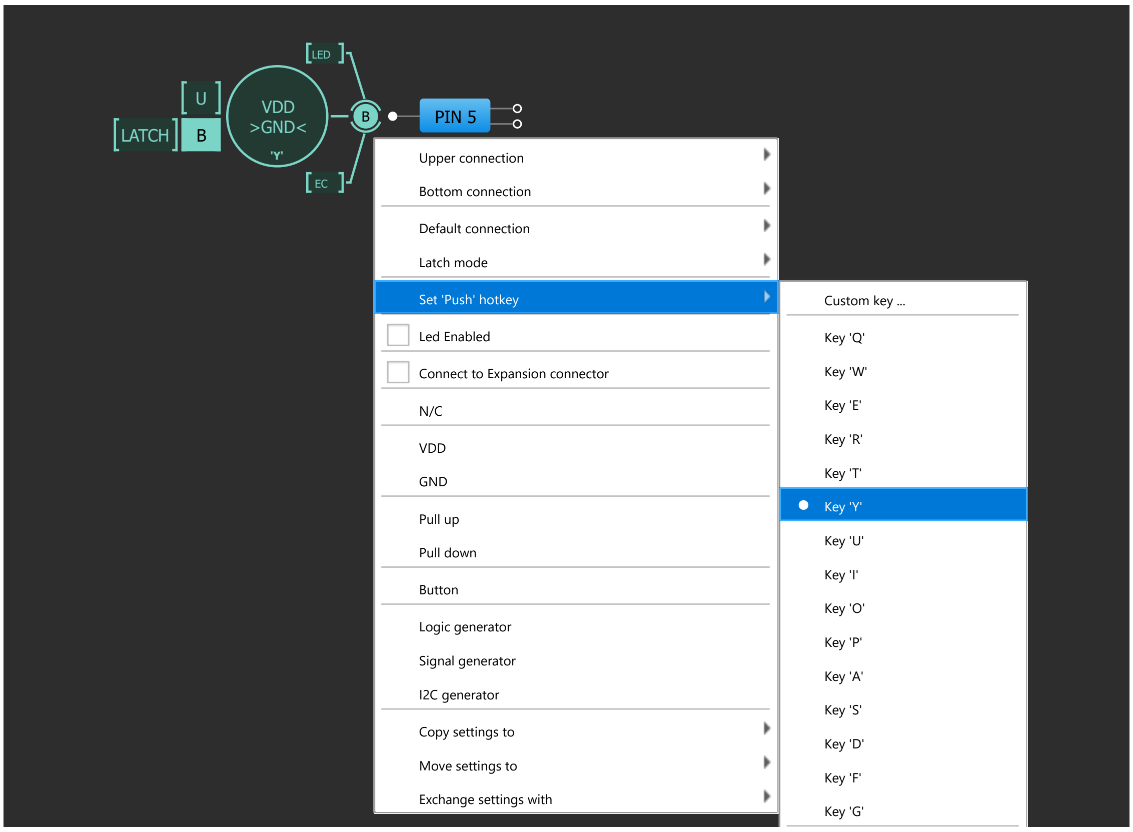
In *Latched* mode, the *Button* toggles its state on click and remains in the new state until the next click.

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



Key Mode

You can assign a hotkey for the *Push* action. Pressing the hotkey is equal to a mouse click.



Choosing hotkey

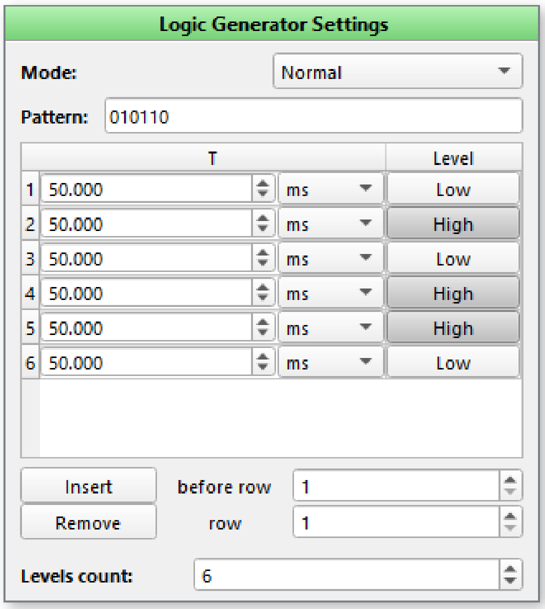
You can assign the same hotkey to multiple *Buttons*, which allows changing the states of all the *Buttons* with the same hotkey at once.

2.2.2 Generators

A *Generator* is a hardware source type, that produces analog or digital electronic signals onto test points on a board. *Go Configure Software Hub* allows setting up the generators in an easy and convenient way with visual control of their settings and states. You can control the *Generator* using the sticker, which appears upon hovering over the respective icon. For more advanced configurations use the *Signal Wizard* tool. To find out more refer to section 2.2.3 Signal Wizard.

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*).



Logic Generator Settings

Mode: Normal

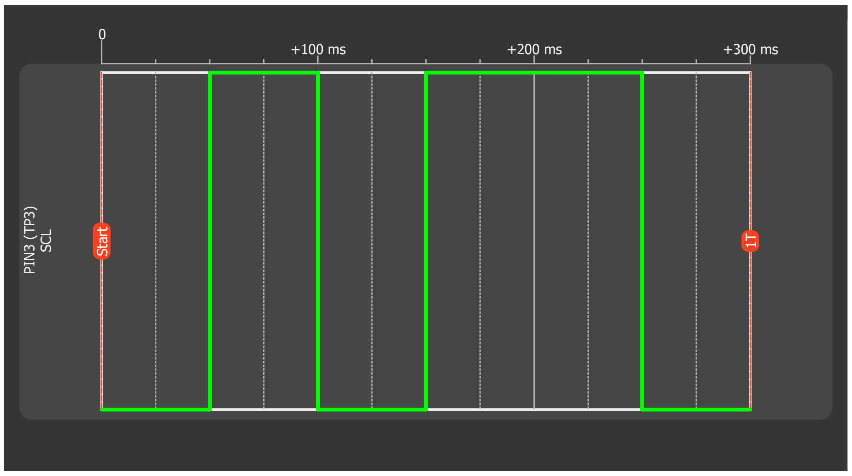
Pattern: 010110

	T		Level
1	50.000	ms	Low
2	50.000	ms	High
3	50.000	ms	Low
4	50.000	ms	High
5	50.000	ms	High
6	50.000	ms	Low

Insert before row 1

Remove row 1

Levels count: 6

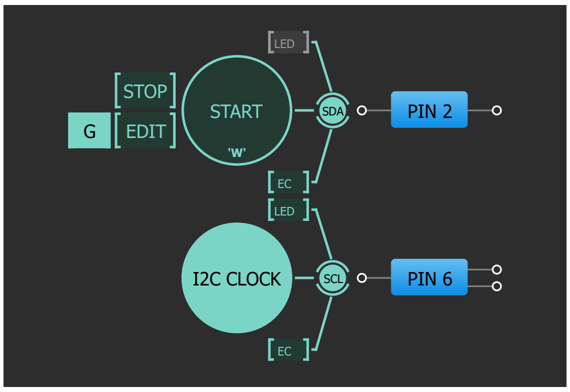


Logic Generator in Signal Wizard

You can export/import all *Logic Generator* settings. To do this, copy the *Pattern* field content and paste it to a text editor. The content will automatically convert to *XML* format text. You can use this feature to save your custom generators or load them from an external file.

I2C Generator

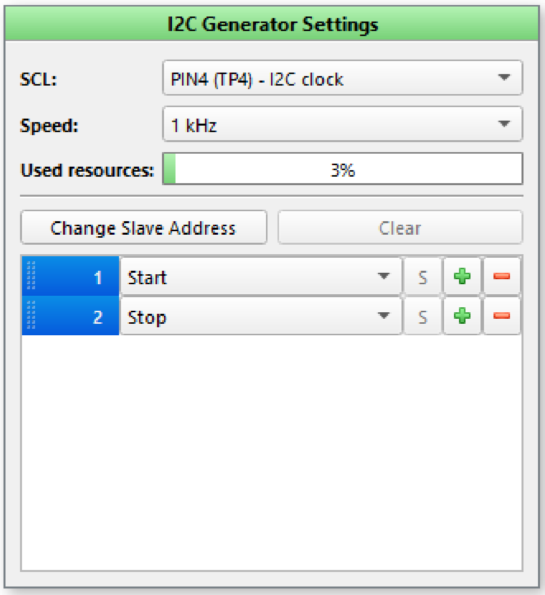
The *I2C Generator*'s purpose is to create *I2C* (Inter-Integrated Circuit) communication patterns based on *Logic Generators*. There are two *Logic Generators* combined as *SDA* (Serial Data) and *SCL* (Serial Clock) lines. You can combine predefined *I2C* primitives to generate a required waveform appropriately and choose the *SCL* frequency.



I2C Generator

SDA signal is a special case of *Logic Generator* used for sending data via *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.



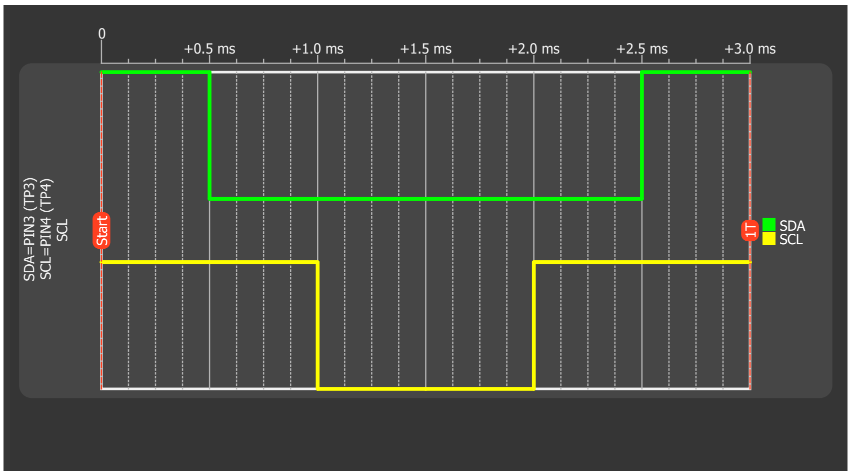
SCL: PIN4 (TP4) - I2C clock

Speed: 1 kHz

Used resources: 3%

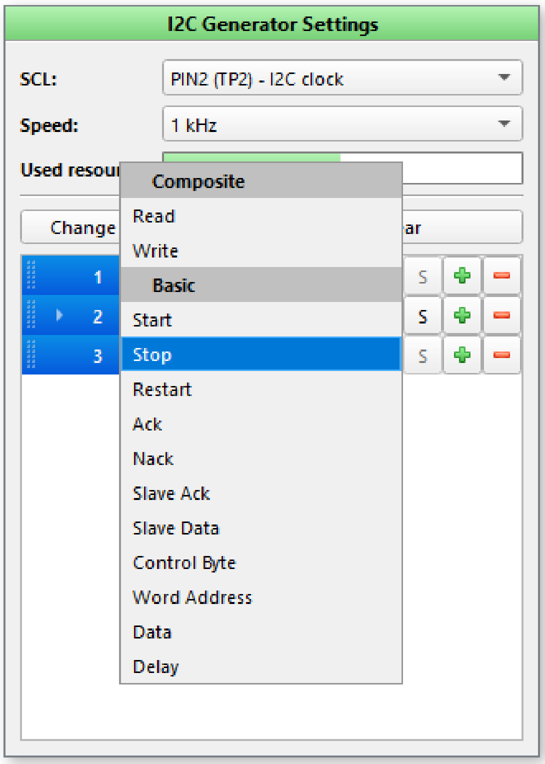
1	Start	S
2	Stop	S

I2C Generator command editor



I2C Generator Signal Wizard

If you decide to change a type of a command, click an arrow and the drop-down menu will show the available commands.

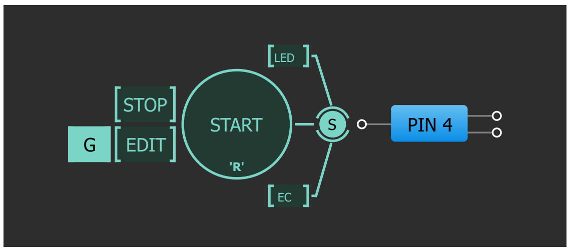


I2C command list

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

Signal (Analog) Generator

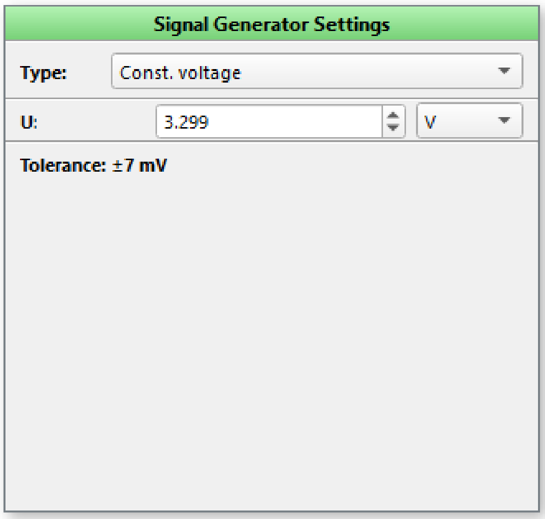
The *Signal Generator* produces the analog signal, and can be configured using one of the following pre-defined types: constant voltage, sine, trapeze, logic pattern, or custom signal.



Signal Generator

Below, you can see the *Signal Wizard* view for different signal types.

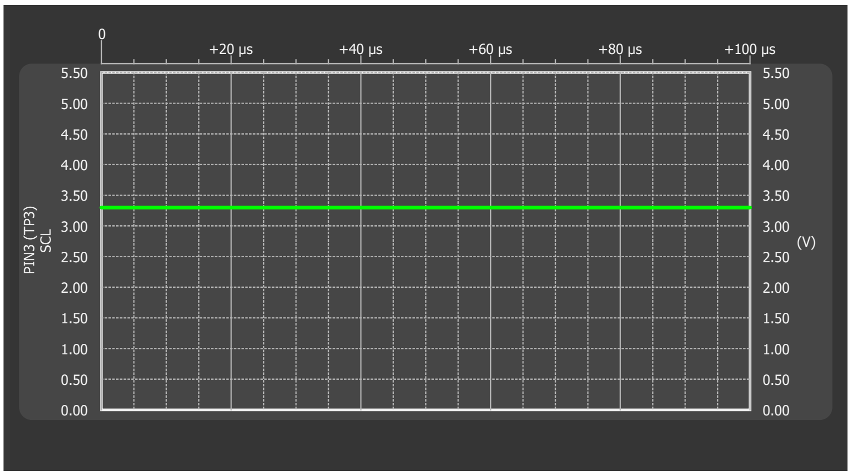
- *Constant voltage waveform type*


Signal Generator Settings

Type: Const. voltage

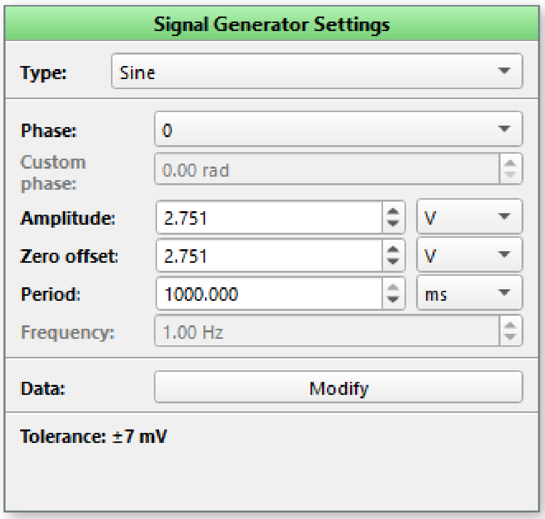
U: 3.299 V

Tolerance: ±7 mV



Configuration options of Signal Generator, type Constant Voltage

➤ *Sine waveform type*


Signal Generator Settings

Type: Sine

Phase: 0

Custom phase: 0.00 rad

Amplitude: 2.751 V

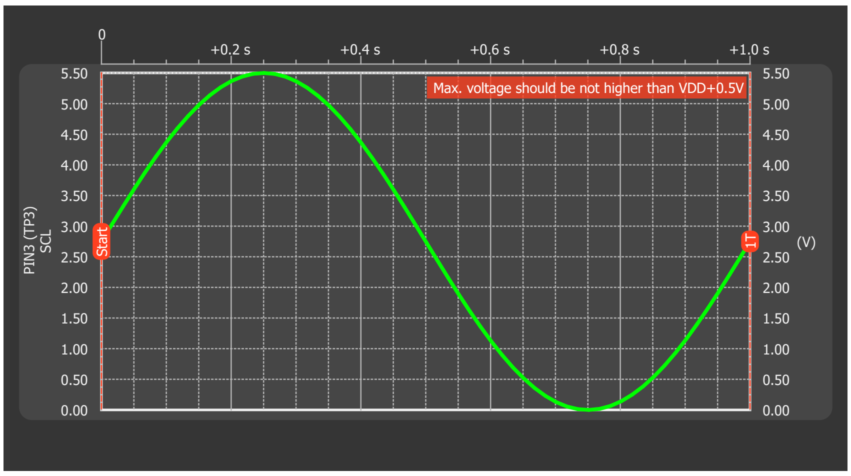
Zero offset: 2.751 V

Period: 1000.000 ms

Frequency: 1.00 Hz

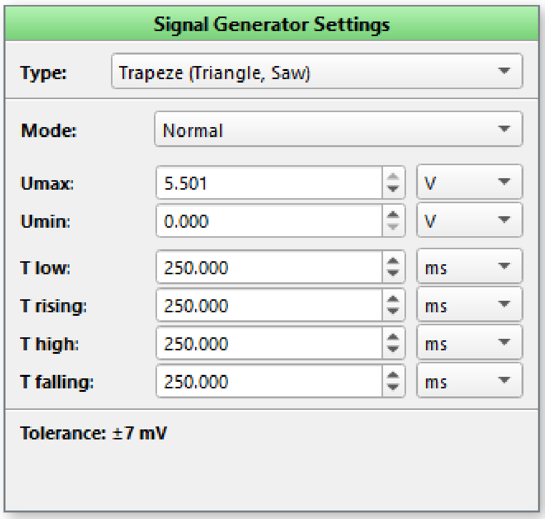
Data: Modify

Tolerance: ±7 mV



Configuration options of Signal Generator, type Sine

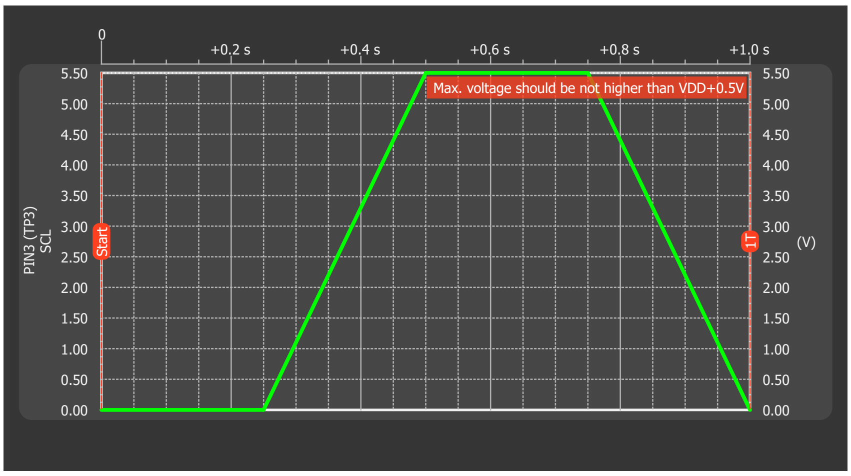
➤ *Trapeze waveform type*



Signal Generator Settings

Type:	Trapeze (Triangle, Saw)	
Mode:	Normal	
Umax:	5.501	V
Umin:	0.000	V
T low:	250.000	ms
T rising:	250.000	ms
T high:	250.000	ms
T falling:	250.000	ms

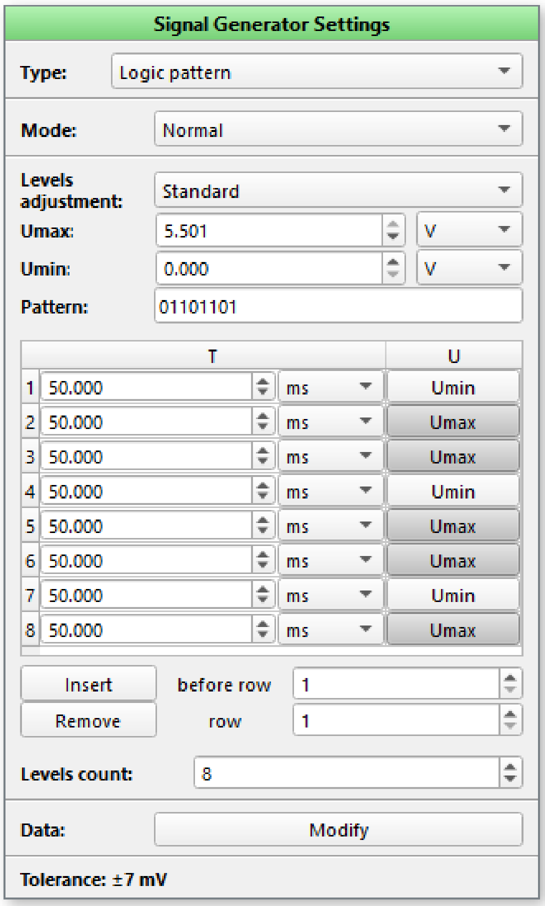
Tolerance: ±7 mV



Max. voltage should be not higher than VDD+0.5V

Configuration options of Signal Generator, type Trapezoid (Triangle, Sawtooth)

➤ *Logic pattern waveform type*



Signal Generator Settings

Type:	Logic pattern
Mode:	Normal
Levels adjustment:	Standard
Umax:	5.501 V
Umin:	0.000 V
Pattern:	01101101

	T		U
1	50.000	ms	Umin
2	50.000	ms	Umax
3	50.000	ms	Umax
4	50.000	ms	Umin
5	50.000	ms	Umax
6	50.000	ms	Umax
7	50.000	ms	Umin
8	50.000	ms	Umax

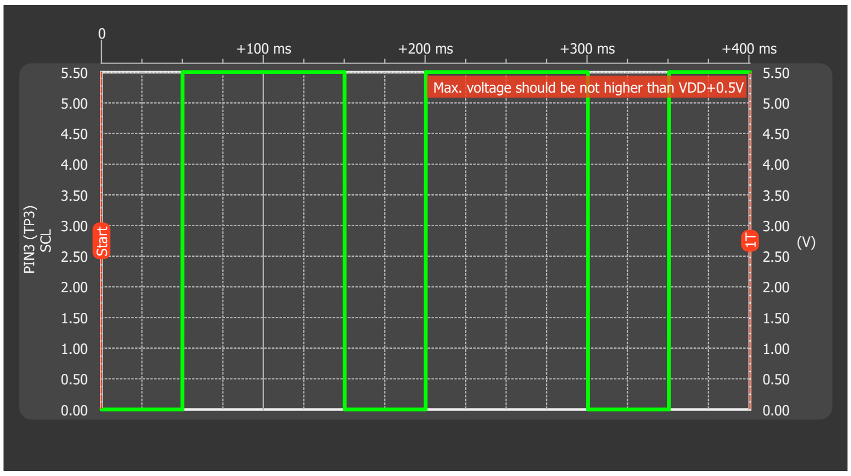
Insert before row 1

Remove row 1

Levels count: 8

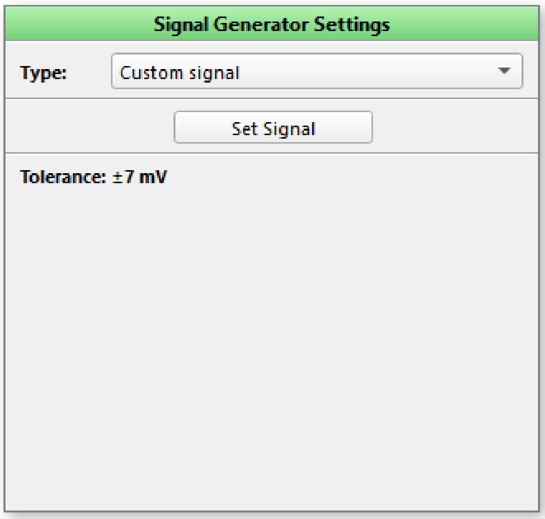
Data: Modify

Tolerance: ±7 mV



Max. voltage should be not higher than VDD+0.5V

Configuration options for Signal Generator, type Logic pattern

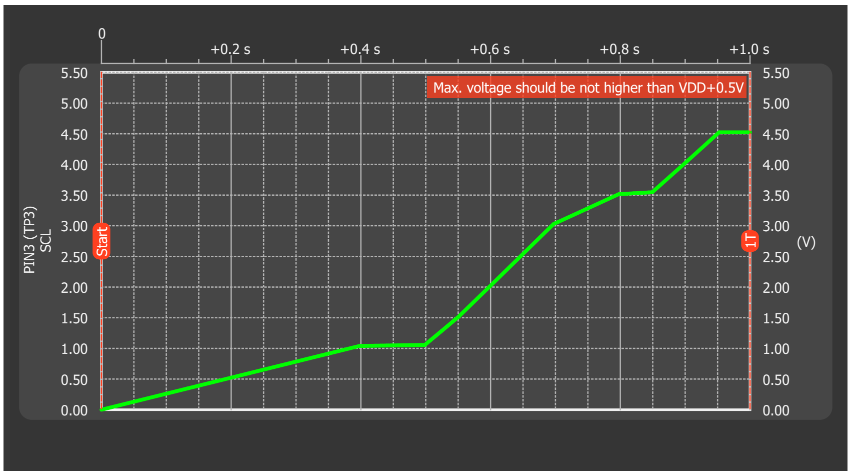


Signal Generator Settings

Type: Custom signal

Set Signal

Tolerance: ±7 mV



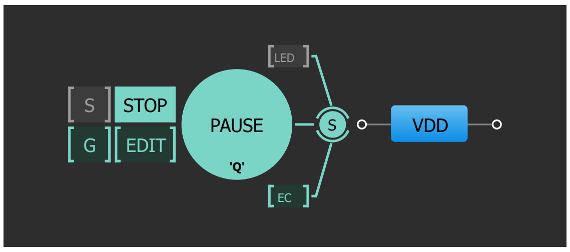
Max. voltage should be not higher than VDD+0.5V

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.

VDD/VDD2 Power Signal Generator

Some Part Numbers have an additional power supply (*VDD2*); if present, it allows to interface two independent voltage domains within the same design. You can configure the pins dedicated to each power supply as inputs, outputs, or both (controlled dynamically by the internal logic) to *VDD* and *VDD2* voltage domains. Using the available macrocell, you can implement mixed-signal functions bridging both domains or pass through level translation in *HIGH* to *LOW* and *LOW* to *HIGH* directions.

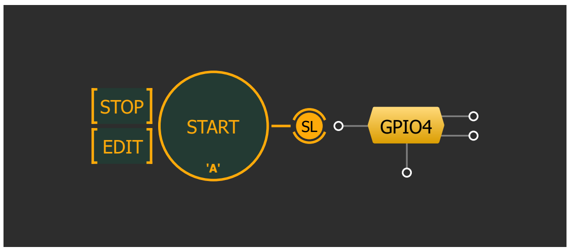


VDD/VDD2 Power Signal Generator

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.

Synchronous Logic Generator

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



Synchronous Logic Generator

You can enhance the efficiency of this generator 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.

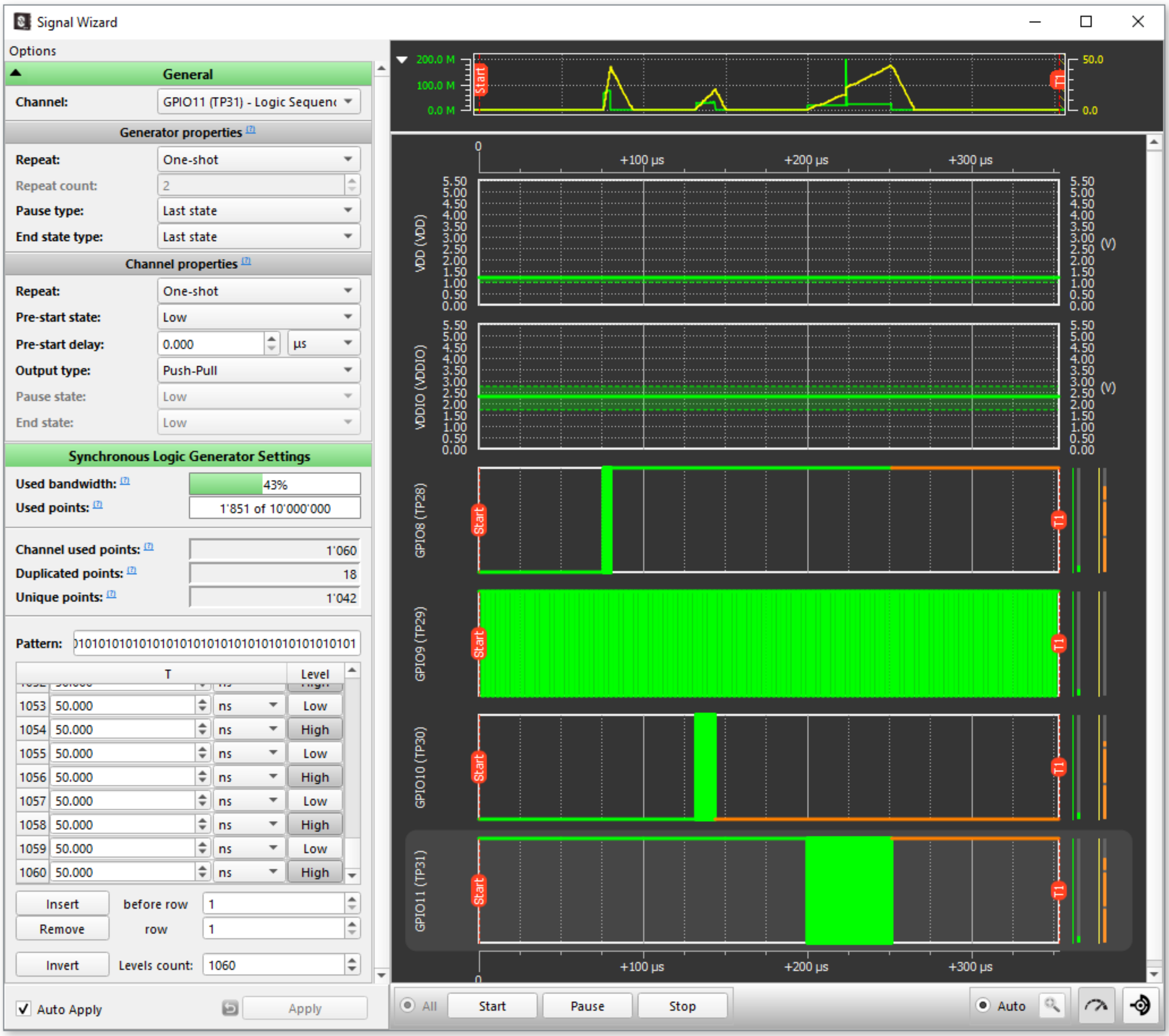
You can adjust your screen view and toggle the graph visibility using the button in the bottom right corner of the window.

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 will change colors depending on their values or percentages.

The legends are available for each indicator to check the value range.



General

Channel:	GPIO11 (TP31) - Logic Sequenc

Generator properties

Repeat:	One-shot
Repeat count:	2
Pause type:	Last state
End state type:	Last state

Channel properties

Repeat:	One-shot
Pre-start state:	Low
Pre-start delay:	0.000 µs
Output type:	Push-Pull
Pause state:	Low
End state:	Low

Synchronous Logic Generator Settings

Used bandwidth:	43%
Used points:	1'851 of 10'000'000
Channel used points:	1'060
Duplicated points:	18
Unique points:	1'042

Pattern: 01010101010101010101010101010101010101010101

	T		Level
1053	50.000	ns	Low
1054	50.000	ns	High
1055	50.000	ns	Low
1056	50.000	ns	High
1057	50.000	ns	Low
1058	50.000	ns	High
1059	50.000	ns	Low
1060	50.000	ns	High

Insert before row 1
Remove row 1
Invert Levels count: 1060

✓ Auto Apply Apply

Synchronous Logic Generator window

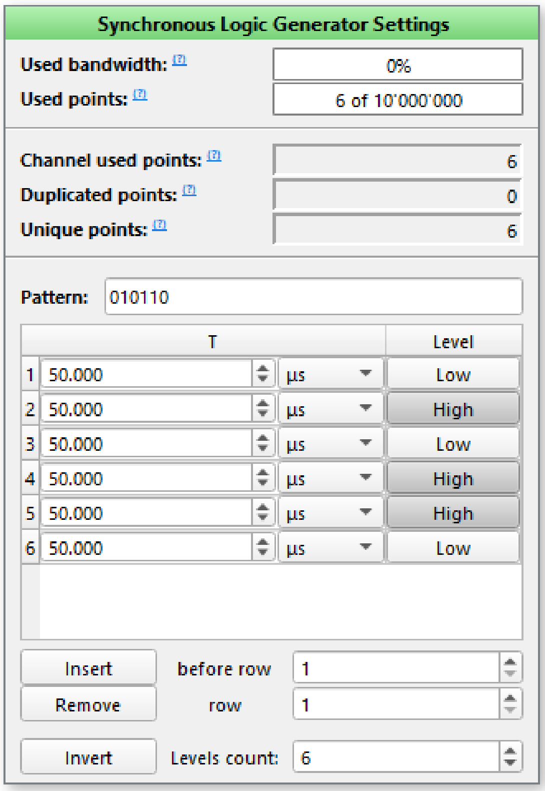
The *Used points* metric displays the cumulative number of points used by patterns across all channels. A point shows any change in the level across any of the 64 channels, including pre-start delay and endstate. However, if two or more channels change levels simultaneously, it will be counted as one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

The Wizard also helps to understand 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


Synchronous Logic Generator Settings

Used bandwidth:	0%
Used points:	6 of 10'000'000
Channel used points:	6
Duplicated points:	0
Unique points:	6

Pattern: 010110

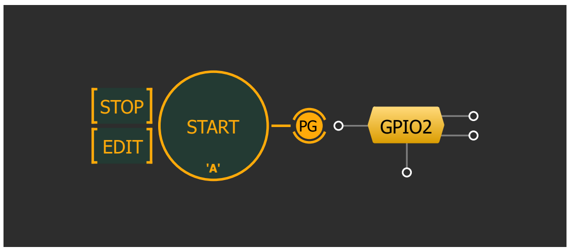
	T		Level
1	50.000	µs	Low
2	50.000	µs	High
3	50.000	µs	Low
4	50.000	µs	High
5	50.000	µs	High
6	50.000	µs	Low

Insert before row 1
Remove row 1
Levels count: 6

Synchronous Logic Generator properties

Parametric Generator

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

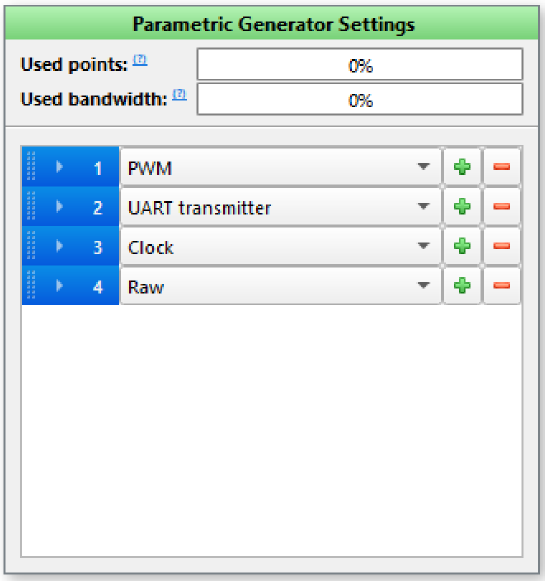


Parametric Generator

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



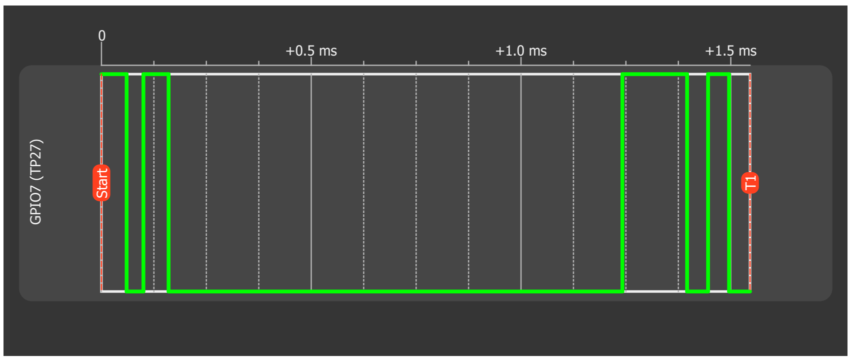
Parametric Generator Settings

Used points: 0%

Used bandwidth: 0%

1	PWM
2	UART transmitter
3	Clock
4	Raw

Parametric Generator command editor

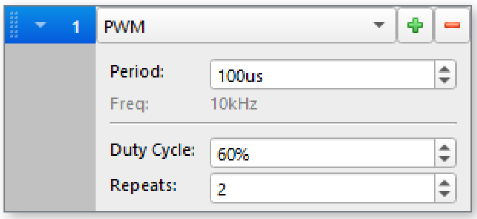


Signal Wizard for the Parametric Generator

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



1 PWM

Period: 100us

Freq: 10kHz

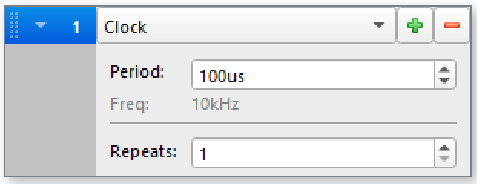
Duty Cycle: 60%

Repeats: 2

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

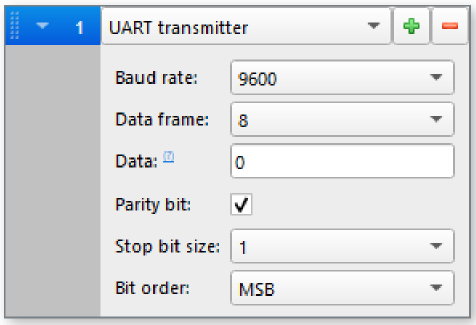


Period: 100us
Freq: 10kHz
Repeats: 1

Clock command editor

➤ *UART Transmitter* — generates signal according to the *UART* standard:

- *Baud rate* — signal's frequency
- *Data frame* — number of bits allocated for user input
- *Data* — user input in hex format. In case the data frame is lower than the bits required to represent data, more significant bits are ignored
- *Parity bit* — create parity bit for error detection
- *Stop bit size* — duration of the stop bit
- *Bit order* — serial data transfer format



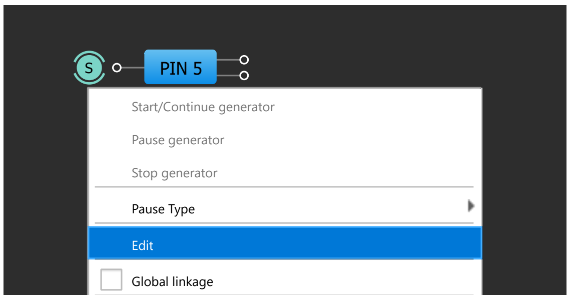
Baud rate: 9600
Data frame: 8
Data: 0
Parity bit: ✓
Stop bit size: 1
Bit order: MSB

UART transmitter command editor

➤ *Raw* — this pattern works as a typical *Logic Generator*

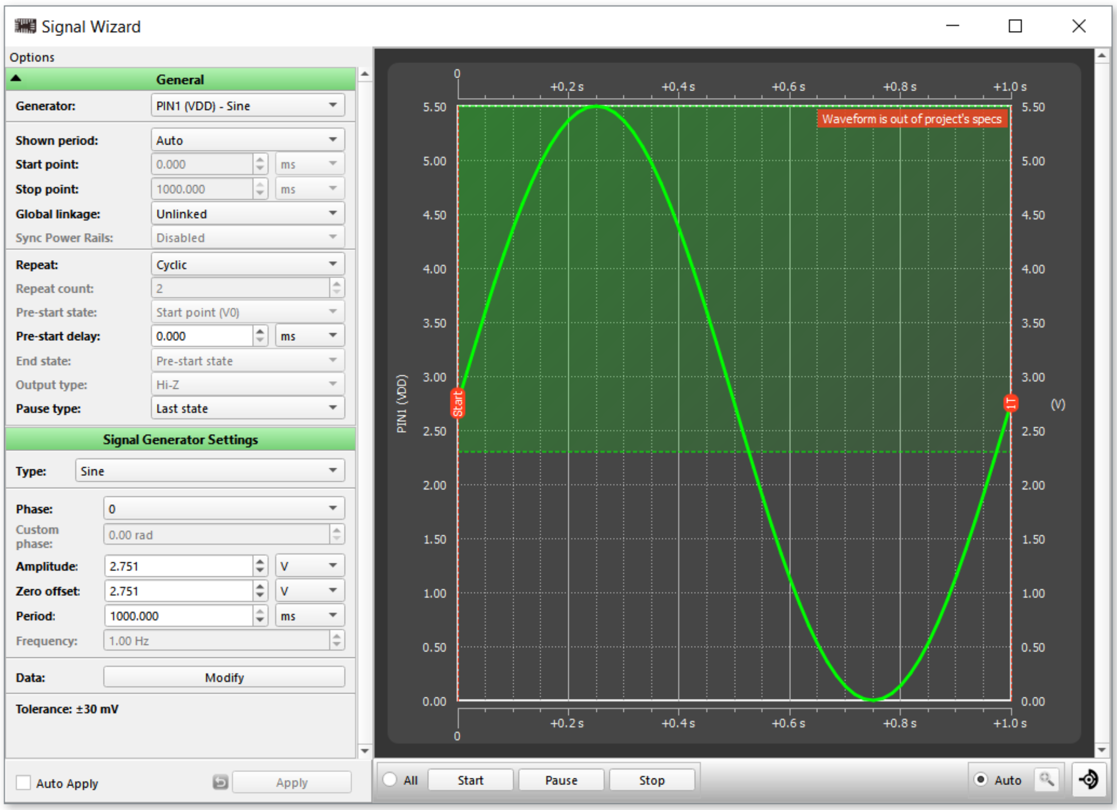
2.2.3 Signal Wizard

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



Opening Signal Wizard

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



General

Generator:	PIN1 (VDD) - Sine
Shown period:	Auto
Start point:	0.000 ms
Stop point:	1000.000 ms
Global linkage:	Unlinked
Sync Power Rails:	Disabled
Repeat:	Cyclic
Repeat count:	2
Pre-start state:	Start point (V0)
Pre-start delay:	0.000 ms
End state:	Pre-start state
Output type:	Hi-Z
Pause type:	Last state

Signal Generator Settings

Type:	Sine
Phase:	0
Custom phase:	0.00 rad
Amplitude:	2.751 V
Zero offset:	2.751 V
Period:	1000.000 ms
Frequency:	1.00 Hz
Data:	Modify

Tolerance: ±30 mV

Signal Wizard window

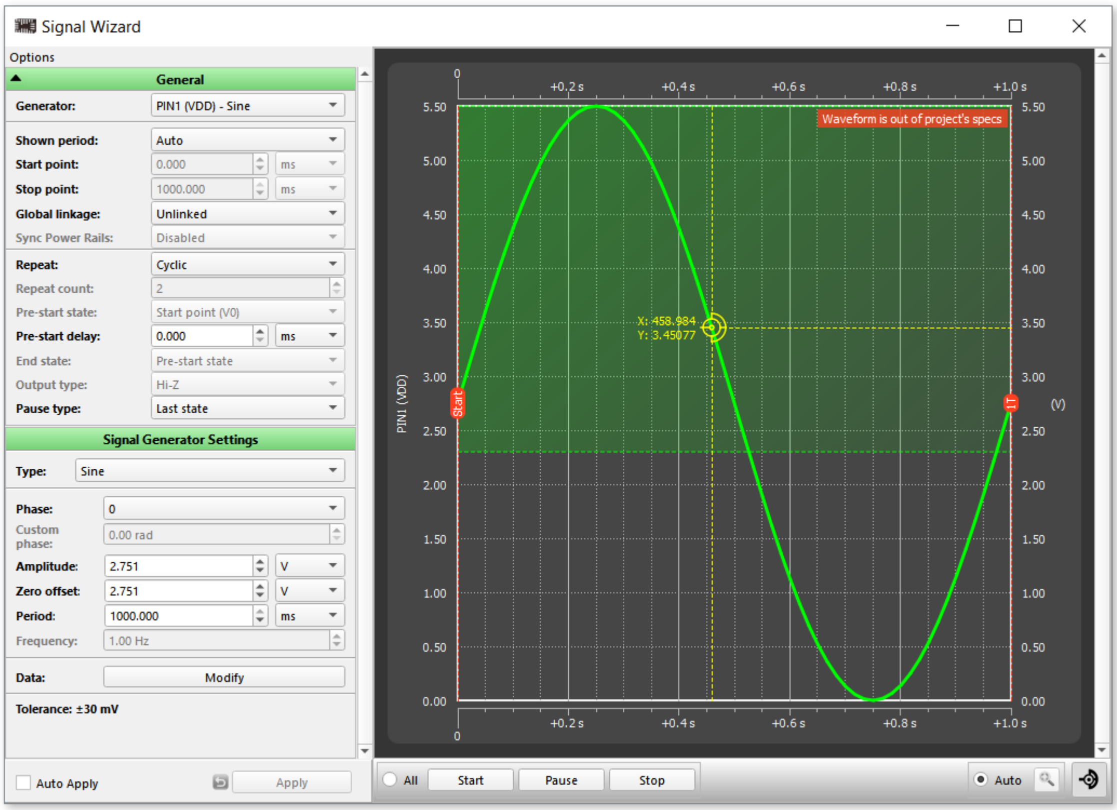
General settings category mostly provides signal configurations related to time, period, and state. Below you can see the settings which may require more detailed description.

- *Global Linkage* — when enabled, provides possibility to control *Generator* using *Start/Stop/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 part is generator-specific. Read more in section 2.2.2 Generators.

Scaling controls

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


Signal Wizard

Options

General

Generator:	PIN1 (VDD) - Sine
Shown period:	Auto
Start point:	0.000 ms
Stop point:	1000.000 ms
Global linkage:	Unlinked
Sync Power Rails:	Disabled
Repeat:	Cyclic
Repeat count:	2
Pre-start state:	Start point (V0)
Pre-start delay:	0.000 ms
End state:	Pre-start state
Output type:	Hi-Z
Pause type:	Last state

Signal Generator Settings

Type:	Sine
Phase:	0
Custom phase:	0.00 rad
Amplitude:	2.751 V
Zero offset:	2.751 V
Period:	1000.000 ms
Frequency:	1.00 Hz
Data:	Modify

Tolerance: ±30 mV

Waveform is out of project's specs

X: 458.984
Y: 3.45077

Scaling in Signal Wizard

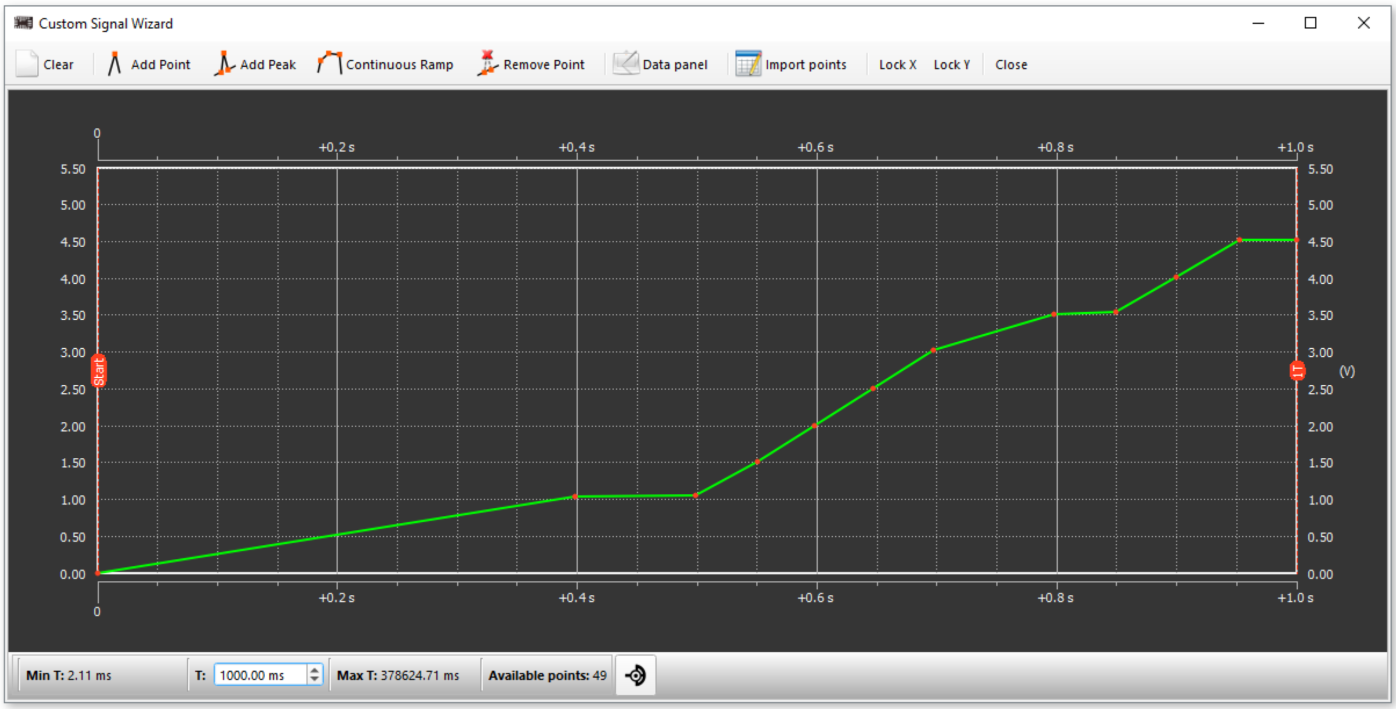
You can use the *Cursor* button to turn the mouse coordinates on/off in the timing diagrams.

➤ *Auto* — scale all generators to fit the biggest period among them. Only generators with *Shown period* set to *Auto* are affected.

➤ *Custom* — waveforms can be re-scaled manually. To do that, ensure that the *Auto* mode is OFF and then scale by `Ctrl` + mouse wheel.

2.2.4 Custom Signal Wizard

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



Min T: 2.11 ms

T: 1000.00 ms

Max T: 378624.71 ms

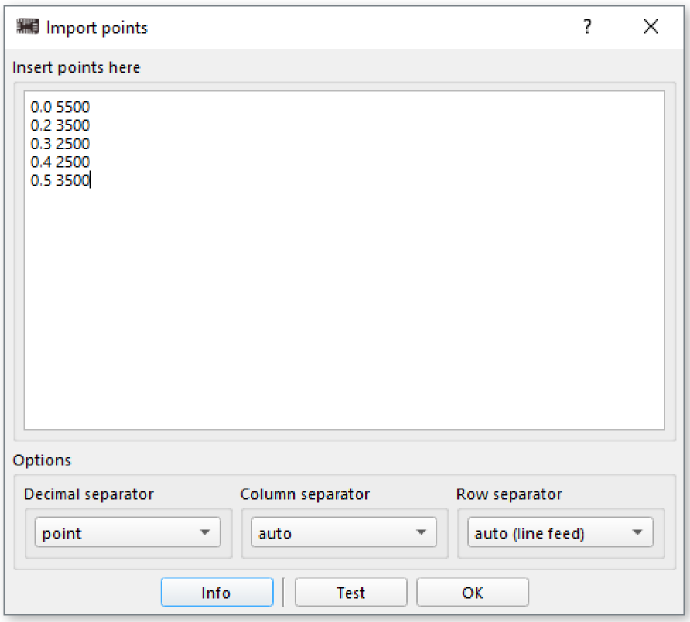
Available points: 49

Drawing Signal (Arbitrary waveform)

Use the controls on the toolbar to manipulate the waveform.

- *Add Point/Peak/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. You can remove/change values for a selected point or add a point in between

➤ *Import Points* — click the widget to open the *Import* window and insert the points' coordinates



```
0.0 5500
0.2 3500
0.3 2500
0.4 2500
0.5 3500
```

Decimal separator: point
Column separator: auto
Row separator: auto (line feed)

Import Points

See the coordinates formatting options:

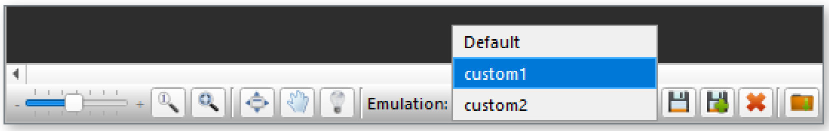
– *Decimal separator*: point/comma

– *Column separator*: auto/tab/other

– *Row separator*: auto(line feed)/tab/other

2.2.5 Hardware configurations

You can create configuration state snapshots to restore a previous configuration when needed. The snapshots include your hardware sources' configuration and are saved and loaded with your project file. This option is available for both *Generators* and *basic Hardware Sources*.



Saved configurations

💾	Save the current configuration state
💾	Save a new configuration state
✖	Delete the selected configuration
📁	Import new configuration
Default / custom1 / custom2	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.

In the software you may encounter different *Debugging Controls* panel types.

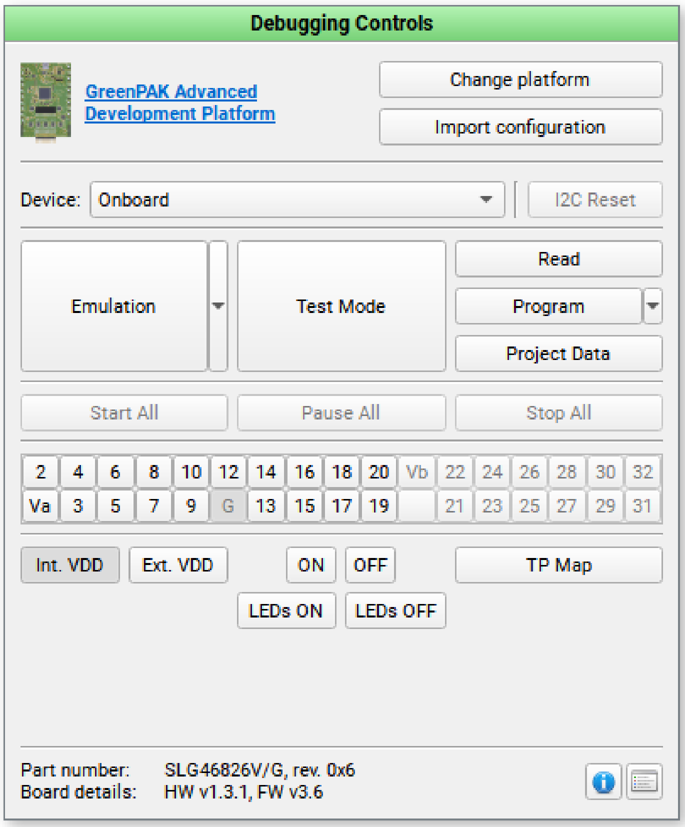
Within the *Standard* debugging controls, we provide the built-in validation checks. The list includes, but is not limited to, the following:

➤ IC detection check — it is verified whether the correct part number/revision is placed in the socket

➤ 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 if the chip is empty/programmed, or whether a chip protection configuration permits the operation

➤ VDDs range validation — the software ensures that the chip operates within the required VDDs voltage range

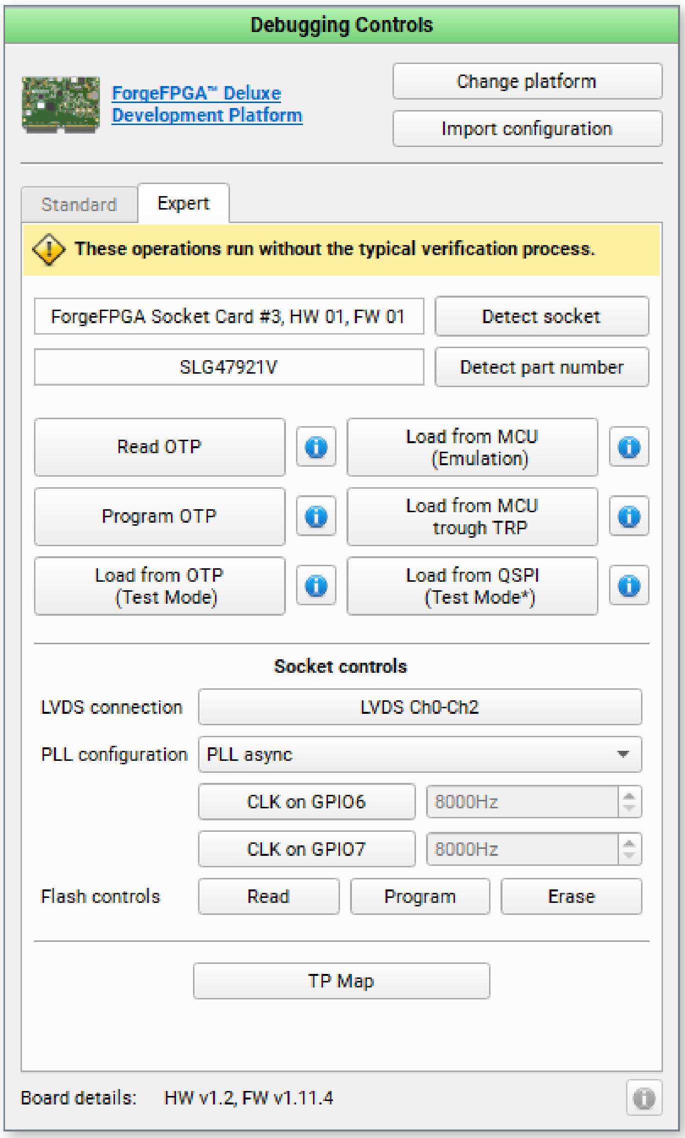


Device: Onboard

Part number: SLG46826V/G, rev. 0x6
Board details: HW v1.3.1, FW v3.6

Standard Debugging Controls panel example

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



These operations run without the typical verification process.

ForgeFPGA Socket Card #3, HW 01, FW 01

SLG47921V

LVDS connection: LVDS Ch0-Ch2

PLL configuration: PLL async

CLK on GPIO6: 8000Hz

CLK on GPIO7: 8000Hz

Board details: HW v1.2, FW v1.11.4

Expert Debugging Controls panel example

Later in this section, you can read about all *Debugging Controls* panel UI elements you may encounter while working with different platforms and ICs. In chapter 3 Devices, you can see the supported development platforms description and the *Debugging Controls* panel for each board.

The table listing *Debugging Controls* elements availability for all boards is given in the appendix, in section 6.3 Debugging Controls feature availability.

Click the reference below to quickly find the *Debugging Controls* panel interface for the certain platform.

GreenPAK Advanced Dev. Platform	GreenPAK Lite Dev. Platform
GreenPAK DIP Dev. Platform	GreenPAK Serial Debugger

ForgeFPGA™ Deluxe Dev. Board

ForgeFPGA™ Evaluation Board

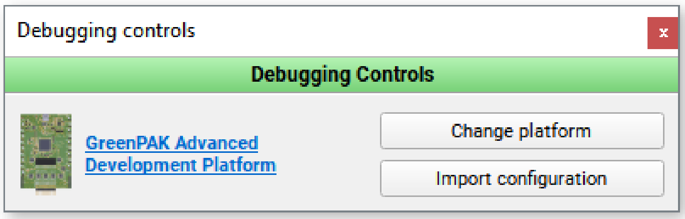
Power GreenPAK Dev. Motherboard

Power GreenPAK Demo Board

GreenPAK Serial Debugger (with SLG5100x)

Debug configuration

- *Recommended platform configuration* — find information about the supported adapter, development board and the devices' ordering information. Click on the platform name link to open the *Recommended platform configuration*

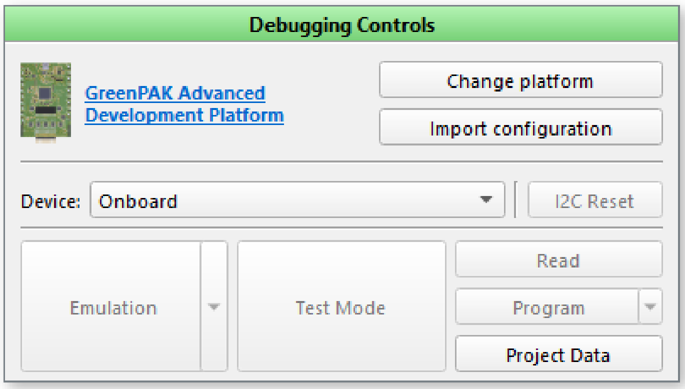


Recommended platform configuration

- *Change platform* — open the list of the development platforms that are supported by a Part Number
- *Import configuration* — upload configurations from a different development platform supported by a Part Number

Device selector

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

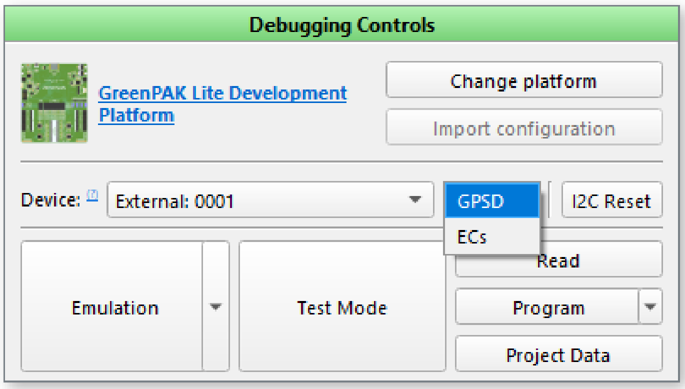


Device selector

Note: Make sure you do not connect the socket to the platform and external chip at the same time.

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/SDA* — EC according to the chip TP map



Lite board device selector

Chip procedures

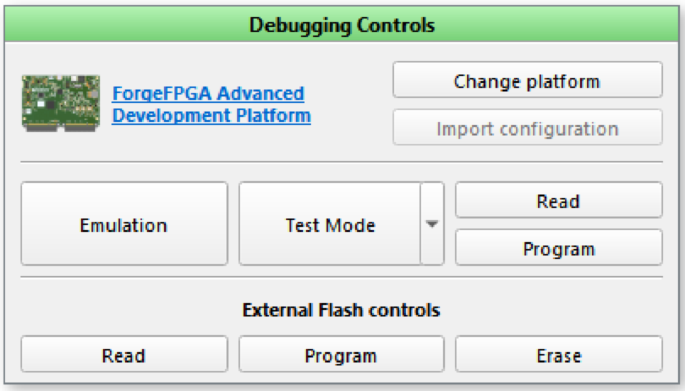
- *Test Mode* — debug the programmed project. Enable power and load the configured hardware sources
- *Load from OTP* — same as *Test Mode* (*Expert* procedure performed without additional validation checks)
- *Emulation* — debug the current project. Enable power and load the design with configured hardware 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
- *Sync* — load the current project to the chip once
- *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, e.g. 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 the reload of all registers data from NVM

Flash procedures

Flash memory is located on the FPGA sockets. See the available controls to work with it.

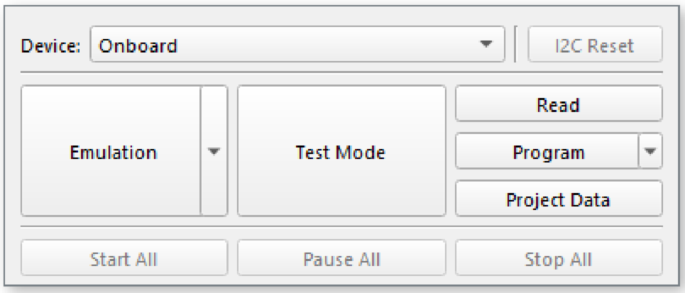
- *Test Mode(*)* — load data from flash to the chip
- *Load from QSPI* — same as *Test Mode(*)* (*Expert* procedure performed without additional validation checks)
- *Read* — read the chip project from the flash memory
- *Program* — program a flash memory with the current project
- *Erase* — clean flash memory (erased flash memory address values will be read as 0xFF)



Flash procedures

NVM Programmer window

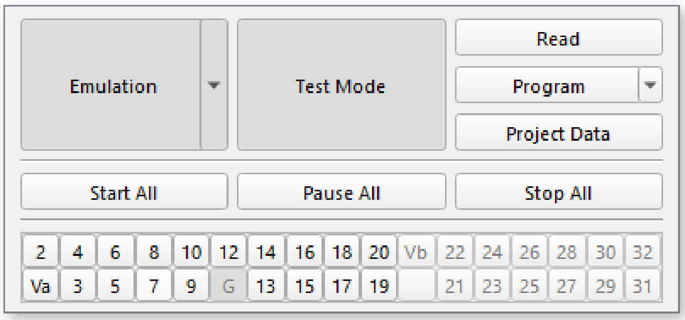
- *Project data* — a table with *NVM/EEPROM* bit sequences. You can change the bit values, import/export the sequences and use data to program the chip (for more info see section 2.2.7 NVM Programmer window)



Project data control

Generator controls

- ➤ *Start/Pause/Stop All* — control the generator state while *Global linkage* generator setting is enabled

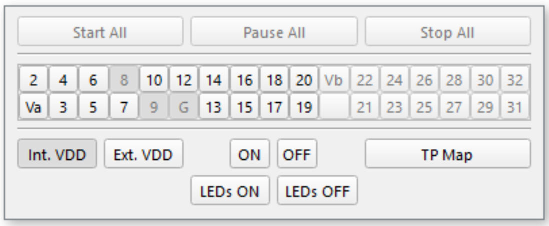

Emulation
Test Mode
Read
Program
Project Data
Start All
Pause All
Stop All
2 4 6 8 10 12 14 16 18 20 Vb 22 24 26 28 30 32
Va 3 5 7 9 G 13 15 17 19 21 23 25 27 29 31

Generator controls

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/disconnect the expansion connectors (also, control the ECs on the work area via a hardware source):

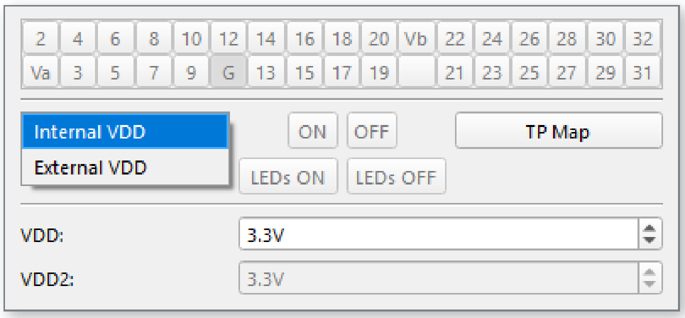
  - Connect/Disconnect all ECs by clicking *ON/OFF* buttons
  - Connect/Disconnect a specific EC by clicking the ECs *sequence number* button
  - Connect/Disconnect *Va EC* by clicking the *Ext. VDD* button


Start All
Pause All
Stop All
2 4 6 8 10 12 14 16 18 20 Vb 22 24 26 28 30 32
Va 3 5 7 9 G 13 15 17 19 21 23 25 27 29 31
Int. VDD
Ext. VDD
ON
OFF
TP Map
LEDs ON
LEDs OFF

Power source selector

- ➤ *Internal VDD* — the power is provided by the GreenPAK board

- ➤ *External VDD* — the board measures voltage on the active power connector, and provides the same voltage level



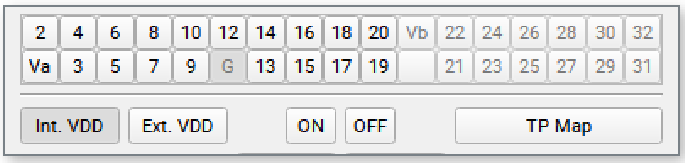
VDD: 3.3V

VDD2: 3.3V

Power source selector example

TP map

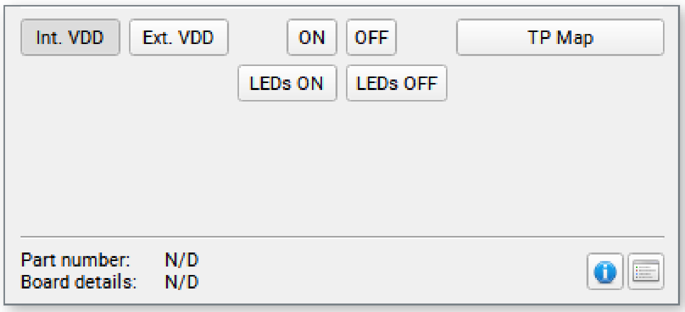
➤ *TP map* — show the test point map on the work area to reflect the physical test points on the development platform



Test point control

LEDs ON/LEDs OFF

➤ *LEDs ON/LEDs OFF* — enable/disable all LEDs on the current platform (applicable only for development platforms with LED support). You can control a particular LED on the work area via a hardware source

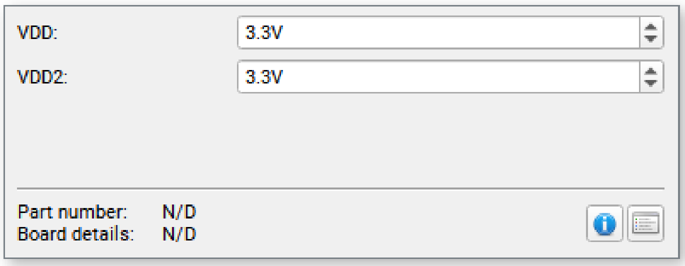


Part number: N/D
Board details: N/D

LEDs ON/OFF controls

Voltage level controls

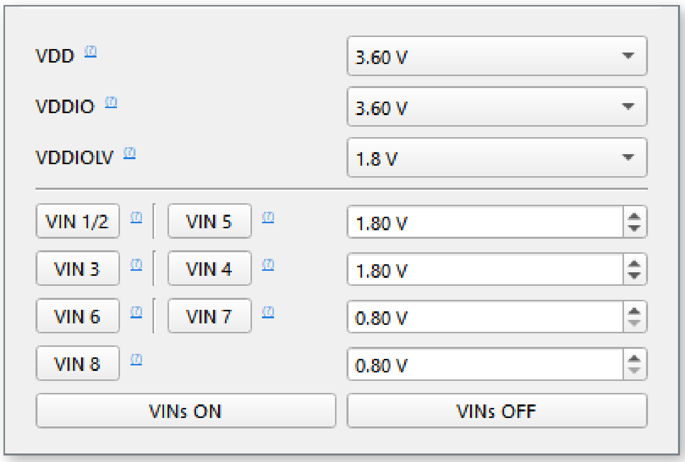
➤ *VDD* — set the voltage level on the corresponding test points. If a selected value exceeds any of the board limitations, the dependent controls are blocked, and a warning is shown.


VDD: 3.3V
VDD2: 3.3V
Part number: N/D
Board details: N/D

Voltage level controls

Power GreenPAK voltage controls

- *VDD/VDDIO/VDDLV* — power supply voltage for overall chip and GPIOs
- *VIN* — power supply voltage on the corresponding LDO component's VINs
- *VINs ON/OFF* — enable/disable all VINs

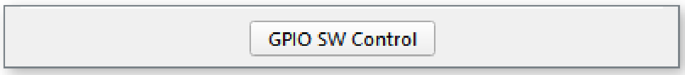

VDD 3.60 V
VDDIO 3.60 V
VDDIOLV 1.8 V
VIN 1/2 | VIN 5 | 1.80 V
VIN 3 | VIN 4 | 1.80 V
VIN 6 | VIN 7 | 0.80 V
VIN 8 | 0.80 V

Power GreenPAK voltage controls

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*, you can activate the feature using the corresponding button in the *Debugging Controls*.

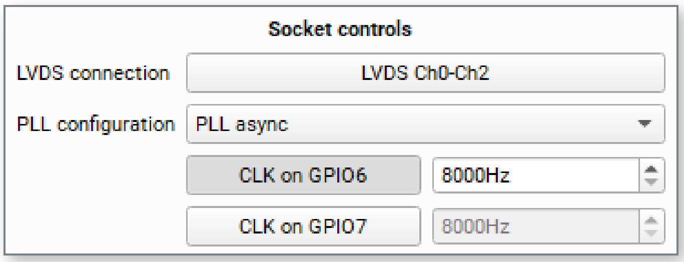


GPIO software control

Socket controls

- ➤ *LVDS connection* — the setting LVDS 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 individually enabled or disabled, and operates independently

Note: The described socket settings are applied dynamically during the procedure and can be modified on the fly.

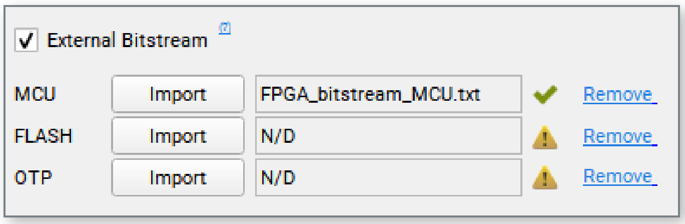


LVDS connection: LVDS Ch0-Ch2
PLL configuration: PLL async
CLK on GPIO6: 8000Hz
CLK on GPIO7: 8000Hz

Socket controls

External Bitstream

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



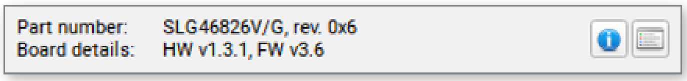
MCU	Import	FPGA_bitstream_MCU.txt	✓	Remove
FLASH	Import	N/D	⚠	Remove
OTP	Import	N/D	⚠	Remove

External Bitstream option

Info details

Brief information about the last detected chip (where e.g. V/G is a package name and e.g. 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.



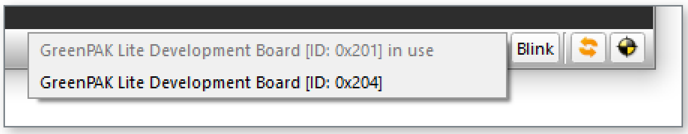
Part number: SLG46826V/G, rev. 0x6
Board details: HW v1.3.1, FW v3.6

Info details

●	Information about the connected Part Number, development platform, software version, and operating system
▤	Show operation log

Board selector

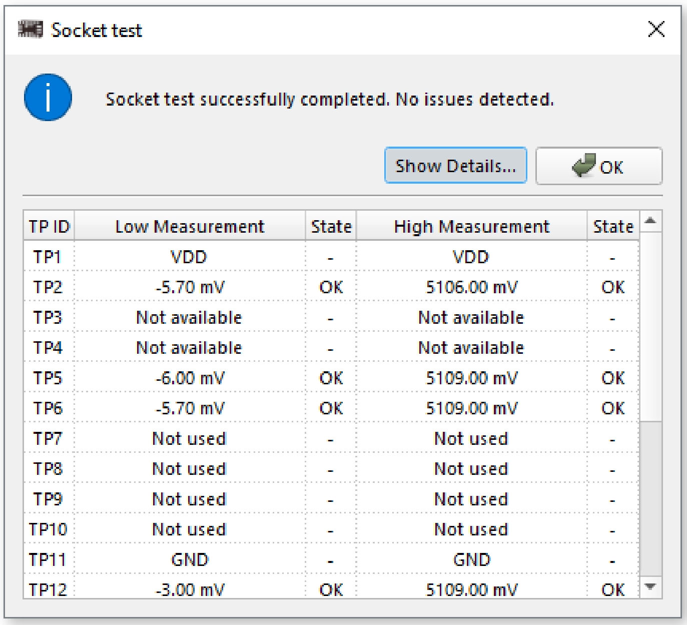
Once the board is connected, the platform info appears on the bottom toolbar. To switch to a different platform, use the Board selector.


GreenPAK Lite Development Board [ID: 0x201] in use
GreenPAK Lite Development Board [ID: 0x204]

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 if the chip is successfully connected to the board. Click *Show Details* to get the *Socket Test* report.


Socket test

Socket test successfully completed. No issues detected.

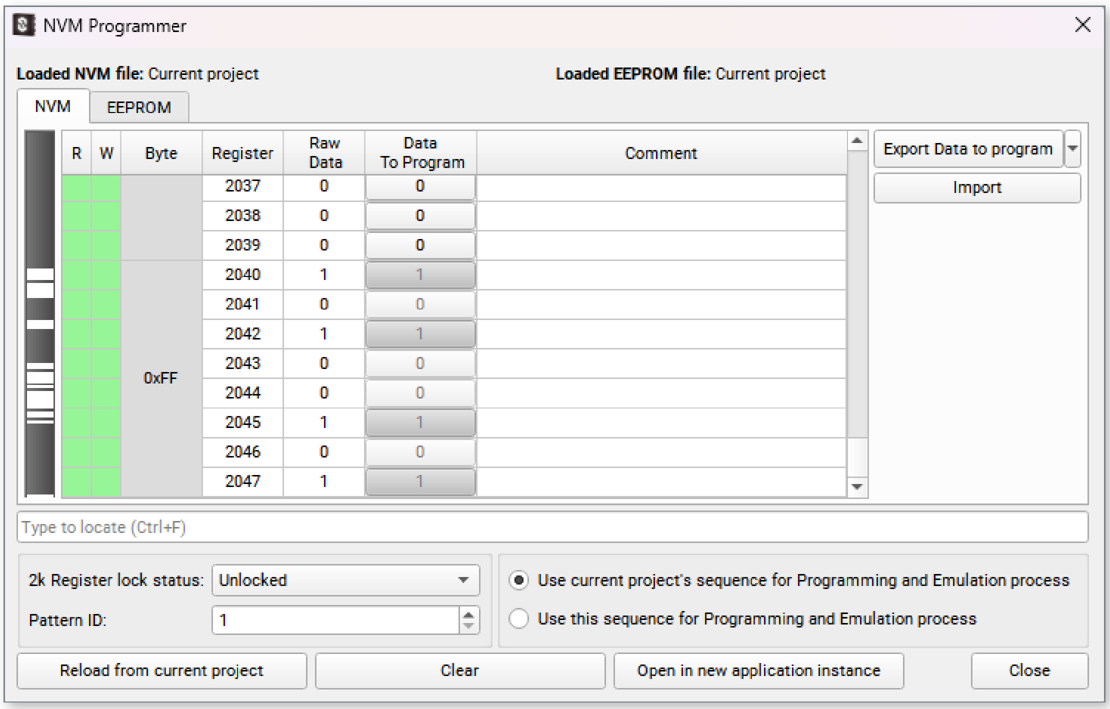
TP ID	Low Measurement	State	High Measurement	State
TP1	VDD	-	VDD	-
TP2	-5.70 mV	OK	5106.00 mV	OK
TP3	Not available	-	Not available	-
TP4	Not available	-	Not available	-
TP5	-6.00 mV	OK	5109.00 mV	OK
TP6	-5.70 mV	OK	5109.00 mV	OK
TP7	Not used	-	Not used	-
TP8	Not used	-	Not used	-
TP9	Not used	-	Not used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

Socket Test report

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

2.2.7 NVM Programmer window

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



NVM Programmer

Loaded NVM file: Current project   **Loaded EEPROM file:** Current project

NVM | EEPROM

R	W	Byte	Register	Raw Data	Data To Program	Comment
🟩	🟩		2037	0	0	
🟩	🟩		2038	0	0	
🟩	🟩		2039	0	0	
🟩	🟩	0xFF	2040	1	1	
🟩	🟩		2041	0	0	
🟩	🟩		2042	1	1	
🟩	🟩		2043	0	0	
🟩	🟩		2044	0	0	
🟩	🟩		2045	1	1	
🟩	🟩		2046	0	0	
🟩	🟩		2047	1	1	

Export Data to program | Import

Type to locate (Ctrl+F)

2k Register lock status: Unlocked

Pattern ID: 1

[✓ Use current project's sequence for Programming and Emulation process]

Use this sequence for Programming and Emulation process

Reload from current project | Clear | Open in new application instance | Close

NVM Programmer window with I2C Read/Write operations

NVM Programmer window table:

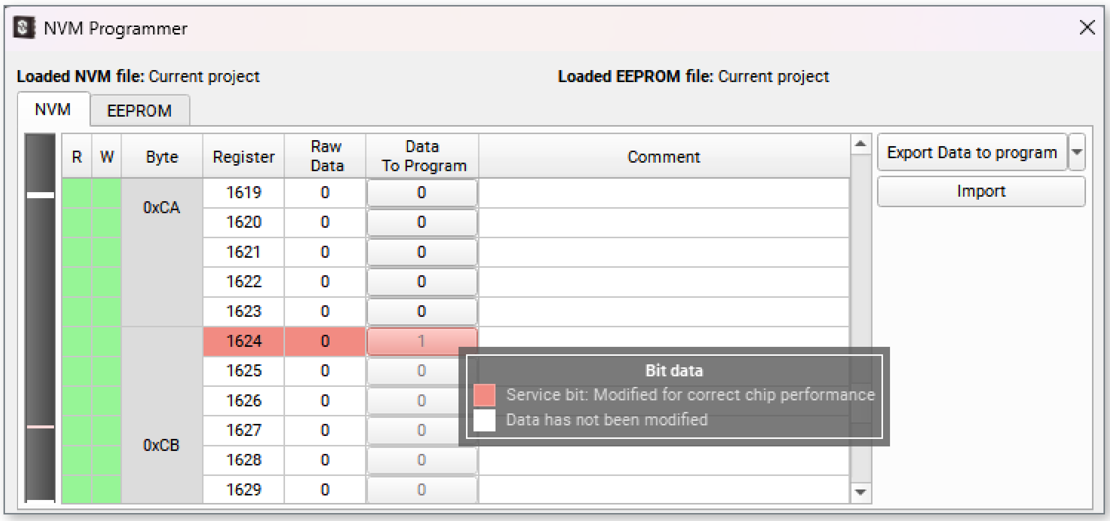
- *R* and *W* — shows if a register is readable and writable

🟩	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 user. It will be used for chip procedures, such as *Emulation* or programming
- *Comment* — add the notes

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

  Note 2: We may forcibly modify the service bit value upon NVM import to ensure proper chip performance.



NVM Programmer

Loaded NVM file: Current project

Loaded EEPROM file: Current project

NVM | EEPROM

R	W	Byte	Register	Raw Data	Data To Program	Comment
🟩	🟩	0xCA	1619	0	0	
🟩	🟩		1620	0	0	
🟩	🟩		1621	0	0	
🟩	🟩		1622	0	0	
🟩	🟩		1623	0	0	
🟩	🟩	0xCB	🟥 1624	🟥 0	🟥 1	
🟩	🟩		1625	0	0	
🟩	🟩		1626	0	0	
🟩	🟩		1627	0	0	
🟩	🟩		1628	0	0	
🟩	🟩		1629	0	0	

Export Data to program

Import

Bit data

🟥 Service bit: Modified for correct chip performance

⬜ Data has not been modified

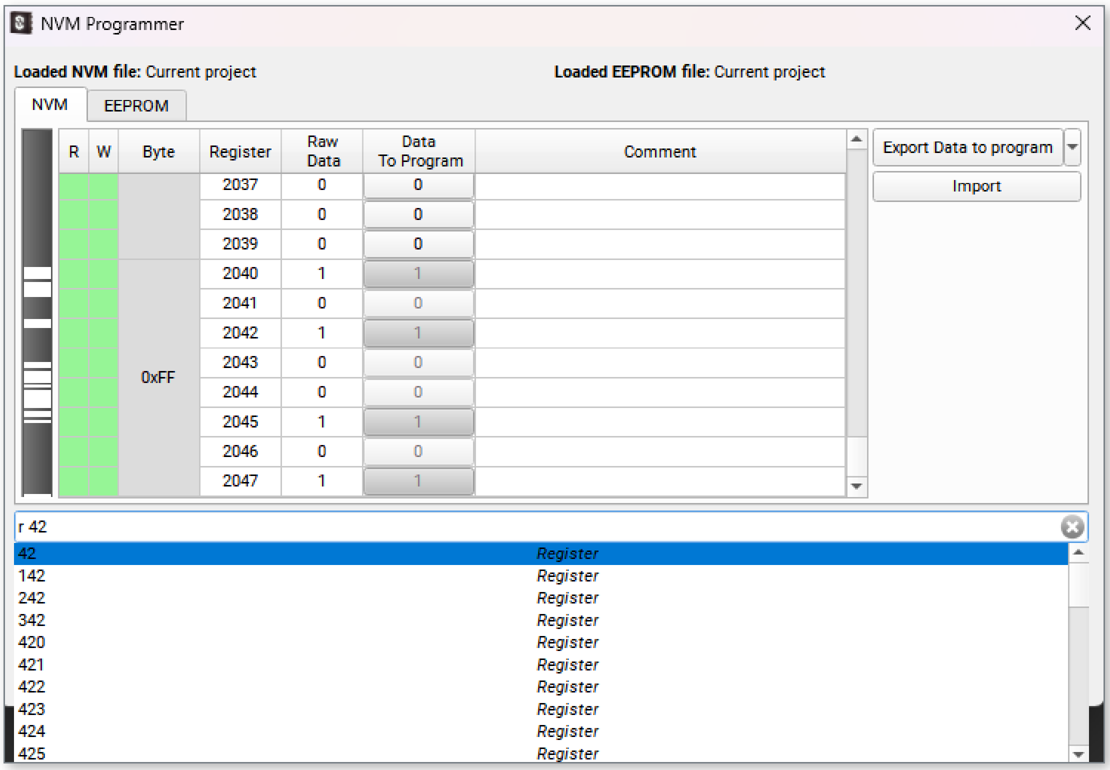
Forced service bit modification highlight

NVM Programmer window controls:

- *Lock status* — lock *NVM Reading/Writing*. Use this control to determine the possibility of *Read /Write* operations
- *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 bit sequence from the *NVM Viewer* to the *Project Data* table
- *Clear* — set the whole *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

You can find the required byte or register by typing its number in the search field. To filter out bytes or registers from the search, use the following markers:

- b — searching for bytes only
- r — searching for registers only



NVM Programmer

Loaded NVM file: Current project   **Loaded EEPROM file:** Current project

NVM | EEPROM

R	W	Byte	Register	Raw Data	Data To Program	Comment
🟩	🟩		2037	0	0	
🟩	🟩		2038	0	0	
🟩	🟩		2039	0	0	
🟩	🟩	0xFF	2040	1	1	
🟩	🟩		2041	0	0	
🟩	🟩		2042	1	1	
🟩	🟩		2043	0	0	
🟩	🟩		2044	0	0	
🟩	🟩		2045	1	1	
🟩	🟩		2046	0	0	
🟩	🟩		2047	1	1	

Export Data to program | Import

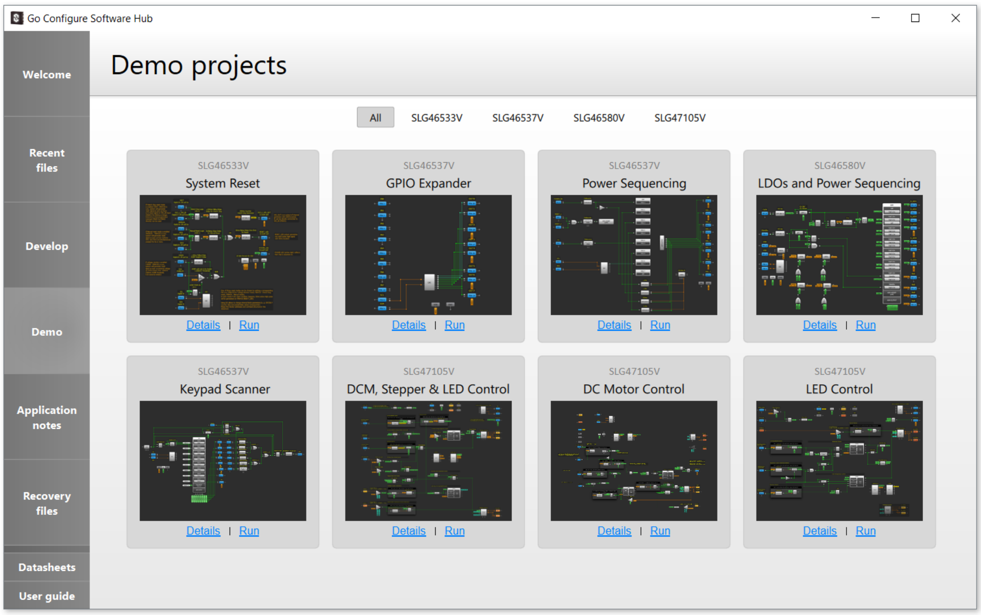
r 42

42	*Register*
142	*Register*
242	*Register*
342	*Register*
420	*Register*
421	*Register*
422	*Register*
423	*Register*
424	*Register*
425	*Register*

Searching for registers

2.2.8 Demo Board and Demo mode

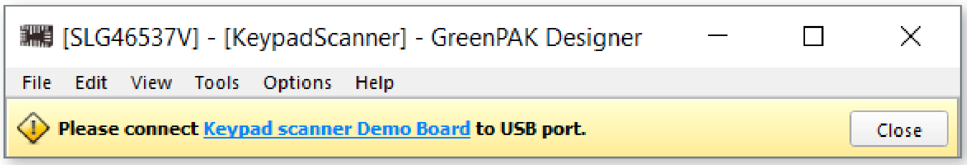
The *Demo* mode allows exploring possible applications of a specific Part Number. The *Demo* tab on the *Hub* window contains the list of the preconfigured projects. Click *Details* to find out more about the design. To open the project click *Run*.


Go Configure Software Hub
Welcome
Recent files
Develop
Demo
Application notes
Recovery files
Datasheets
User guide
Demo projects
All
SLG46533V
SLG46537V
SLG46580V
SLG47105V
SLG46533V System Reset Details | Run
SLG46537V GPIO Expander Details | Run
SLG46537V Power Sequencing Details | Run
SLG46580V LDOs and Power Sequencing Details | Run
SLG46537V Keypad Scanner Details | Run
SLG47105V DCM, Stepper & LED Control Details | Run
SLG47105V DC Motor Control Details | Run
SLG47105V LED Control Details | Run

Select a Demo project

To interact with the design, you can connect the specific *Demo Board*. Such board contains a soldered IC with a preprogrammed project.

Once you open the project, the workspace UI depends on whether the *Demo Board* is connected.


[SLG46537V] - [KeypadScanner] - GreenPAK Designer
File Edit View Tools Options Help
Close

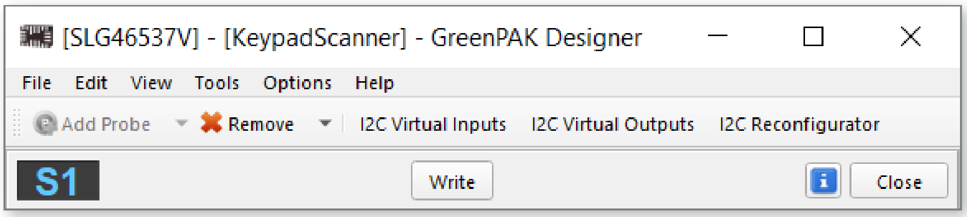
Please connect Keypad scanner Demo Board to USB port.

Waiting for the corresponding Demo Board connection

After the board detection is successful, 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 2.2.9 I2C Tools.
- — information about a Part Number, development platform, and operating system
- *Close* — exit the *Demo* mode


[SLG46537V] - [KeypadScanner] - GreenPAK Designer
File Edit View Tools Options Help
Add Probe
Remove
I2C Virtual Inputs
I2C Virtual Outputs
I2C Reconfigurator
S1
Write
Close

Demo Board detected

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

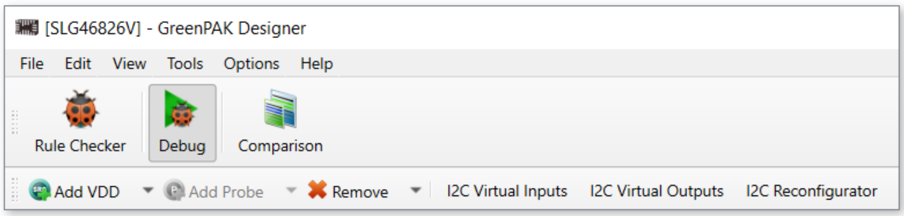
2.2.9 I2C Tools

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

A chip's I2C communication macrocell allows an I2C bus master to read and write information at any moment via a serial channel directly to the registers via I2C protocol. The tool allows configuring the macrocell data on the fly.

Once you enable *Debug* utility and select the platform, the following I2C Tools appear on the toolbar: *I2C Virtual Inputs*, *I2C Virtual Outputs*, and *I2C Reconfigurator*.

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

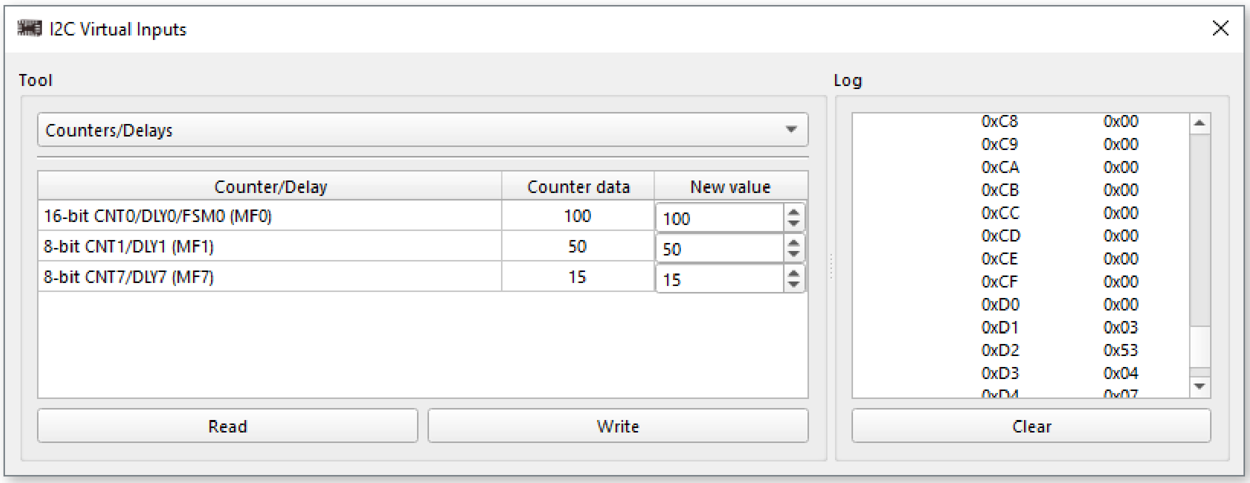


I2C Tools on the toolbar

I2C Virtual Inputs

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

➤ *Counters/Delays* — allows to read/write the counter data of a specific macrocell configured in CNT/DLY mode

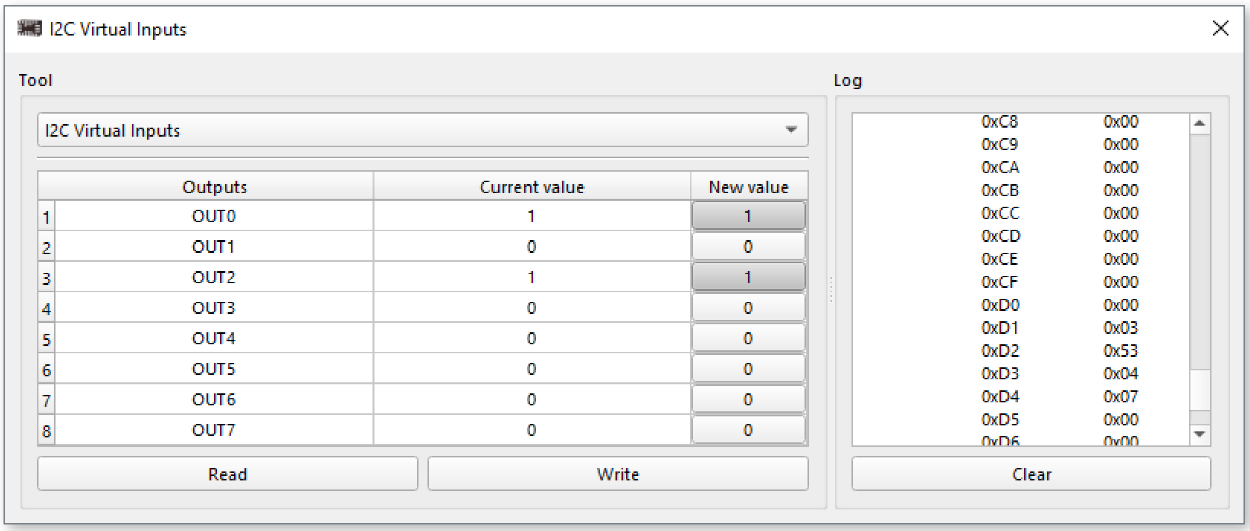


Counter/Delay	Counter data	New value
16-bit CNT0/DLY0/FSM0 (MF0)	100	100
8-bit CNT1/DLY1 (MF1)	50	50
8-bit CNT7/DLY7 (MF7)	15	15

```
0xC8    0x00
0xC9    0x00
0xCA    0x00
0xCB    0x00
0xCC    0x00
0xCD    0x00
0xCE    0x00
0xCF    0x00
0xD0    0x00
0xD1    0x03
0xD2    0x53
0xD3    0x04
0xD4    0x07
```

Managing Counters/Delays

➤ *I2C Virtual Inputs* — allows you to read/write I2C virtual OUTs values of the I2C macrocell



	Outputs	Current value	New value
1	OUT0	1	1
2	OUT1	0	0
3	OUT2	1	1
4	OUT3	0	0
5	OUT4	0	0
6	OUT5	0	0
7	OUT6	0	0
8	OUT7	0	0

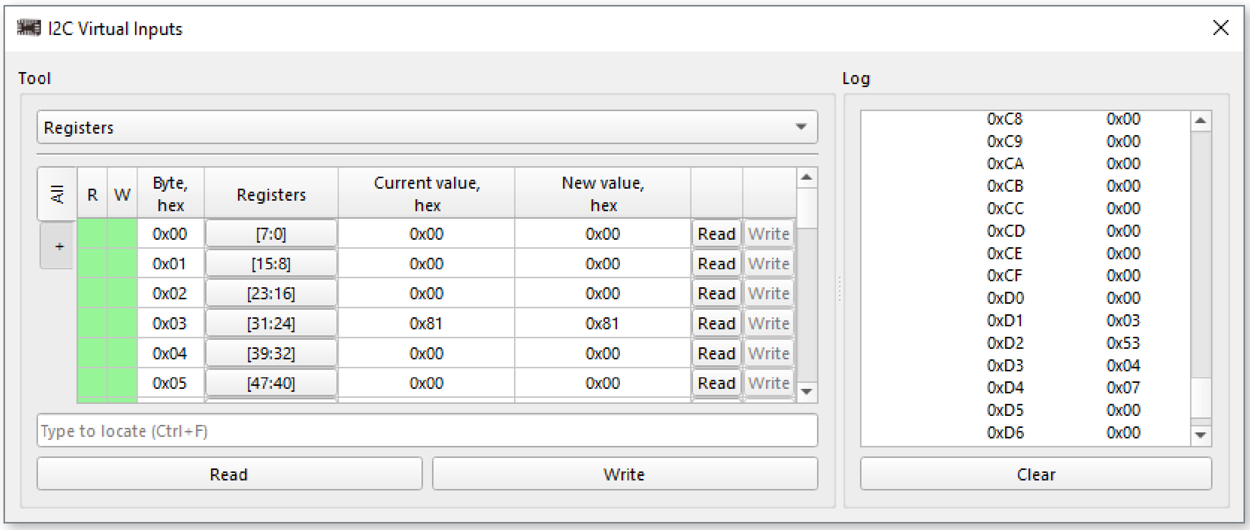
```
0xC8    0x00
0xC9    0x00
0xCA    0x00
0xCB    0x00
0xCC    0x00
0xCD    0x00
0xCE    0x00
0xCF    0x00
0xD0    0x00
0xD1    0x03
0xD2    0x53
0xD3    0x04
0xD4    0x07
0xD5    0x00
0xD6    0x00
```

I2C Virtual Inputs

➤ *Registers* — allows you to read/write all of the chip registers data. It is possible to read/write:

– the certain register in each row

– all registers using the bottom buttons



R	W	Byte, hex	Registers	Current value, hex	New value, hex		
🟩	🟩	0x00	[7:0]	0x00	0x00	Read	Write
🟩	🟩	0x01	[15:8]	0x00	0x00	Read	Write
🟩	🟩	0x02	[23:16]	0x00	0x00	Read	Write
🟩	🟩	0x03	[31:24]	0x81	0x81	Read	Write
🟩	🟩	0x04	[39:32]	0x00	0x00	Read	Write
🟩	🟩	0x05	[47:40]	0x00	0x00	Read	Write

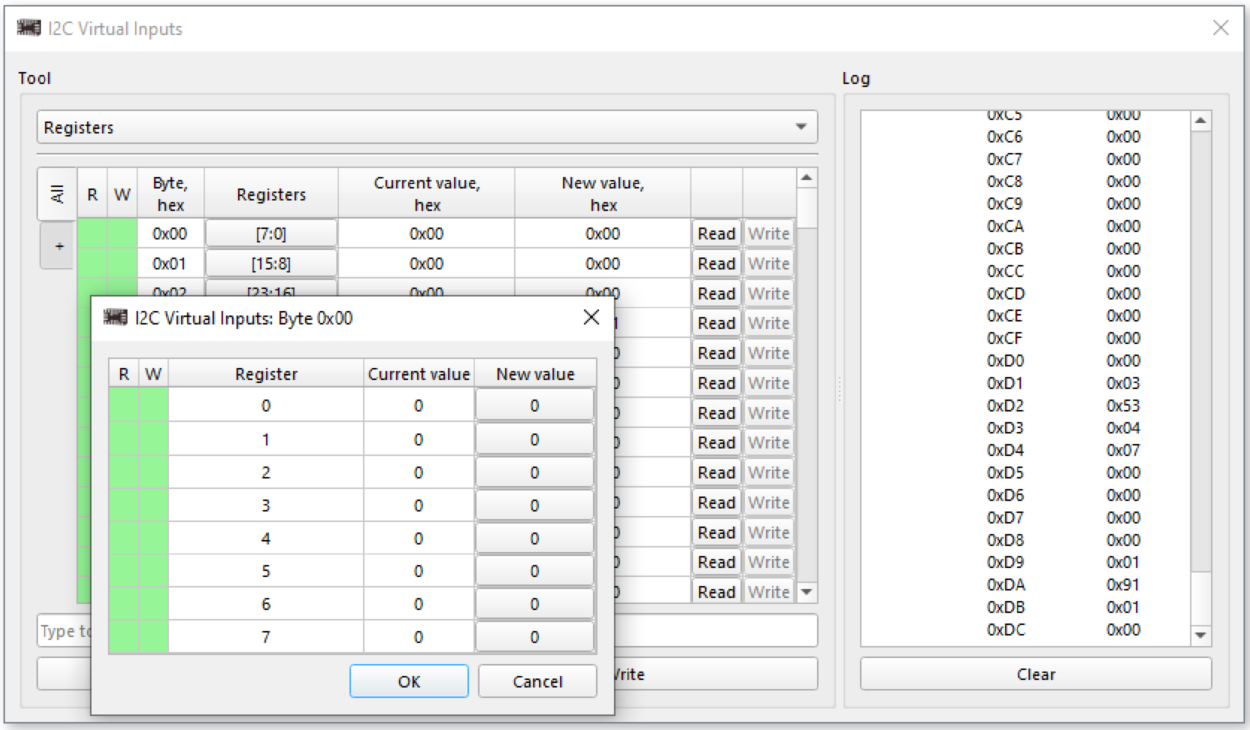
```
0xC8    0x00
0xC9    0x00
0xCA    0x00
0xCB    0x00
0xCC    0x00
0xCD    0x00
0xCE    0x00
0xCF    0x00
0xD0    0x00
0xD1    0x03
0xD2    0x53
0xD3    0x04
0xD4    0x07
0xD5    0x00
0xD6    0x00
```

Read/Write the registers data

You can edit a register value in the following ways:

– enter the value in the *New value* column

– change the register bits by clicking a cell in the *Registers* column


I2C Virtual Inputs

Tool

Registers

R	W	Byte, hex	Registers	Current value, hex	New value, hex		
🟩	🟩	0x00	[7:0]	0x00	0x00	Read	Write
🟩	🟩	0x01	[15:8]	0x00	0x00	Read	Write
🟩	🟩	0x02	[23:16]	0x00	0x00	Read	Write

I2C Virtual Inputs: Byte 0x00

R	W	Register	Current value	New value
🟩	🟩	0	0	0
🟩	🟩	1	0	0
🟩	🟩	2	0	0
🟩	🟩	3	0	0
🟩	🟩	4	0	0
🟩	🟩	5	0	0
🟩	🟩	6	0	0
🟩	🟩	7	0	0

OK Cancel

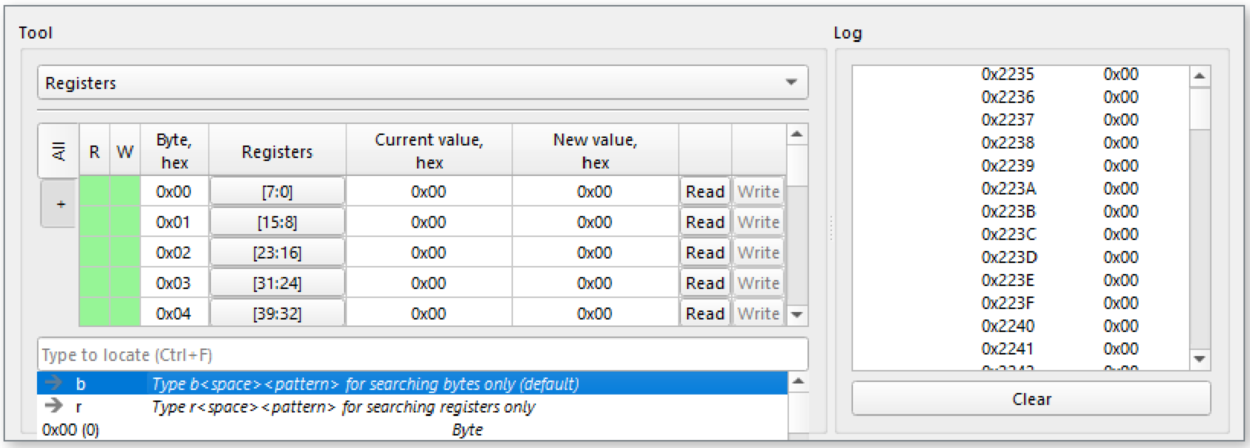
Log

```
0xC5    0x00
0xC6    0x00
0xC7    0x00
0xC8    0x00
0xC9    0x00
0xCA    0x00
0xCB    0x00
0xCC    0x00
0xCD    0x00
0xCE    0x00
0xCF    0x00
0xD0    0x00
0xD1    0x03
0xD2    0x53
0xD3    0x04
0xD4    0x07
0xD5    0x00
0xD6    0x00
0xD7    0x00
0xD8    0x00
0xD9    0x01
0xDA    0x91
0xDB    0x01
0xDC    0x00
```

Clear

Registers modification options

You can find a required byte or register(bit) by using the pattern below:


Tool

Registers

R	W	Byte, hex	Registers	Current value, hex	New value, hex		
🟩	🟩	0x00	[7:0]	0x00	0x00	Read	Write
🟩	🟩	0x01	[15:8]	0x00	0x00	Read	Write
🟩	🟩	0x02	[23:16]	0x00	0x00	Read	Write
🟩	🟩	0x03	[31:24]	0x00	0x00	Read	Write
🟩	🟩	0x04	[39:32]	0x00	0x00	Read	Write

Type to locate (Ctrl+F)

```
→ b        Type b<space><pattern> for searching bytes only (default)
→ r        Type r<space><pattern> for searching registers only
0x00 (0)                             Byte
```

Log

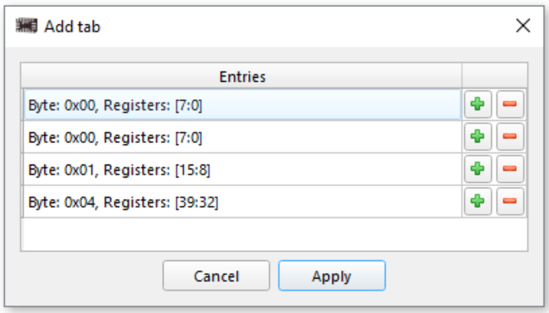
```
0x2235    0x00
0x2236    0x00
0x2237    0x00
0x2238    0x00
0x2239    0x00
0x223A    0x00
0x223B    0x00
0x223C    0x00
0x223D    0x00
0x223E    0x00
0x223F    0x00
0x2240    0x00
0x2241    0x00
```

Clear

Filter options

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

Note: Only the registers present under an active tab are read upon clicking the bottom *Read* button.

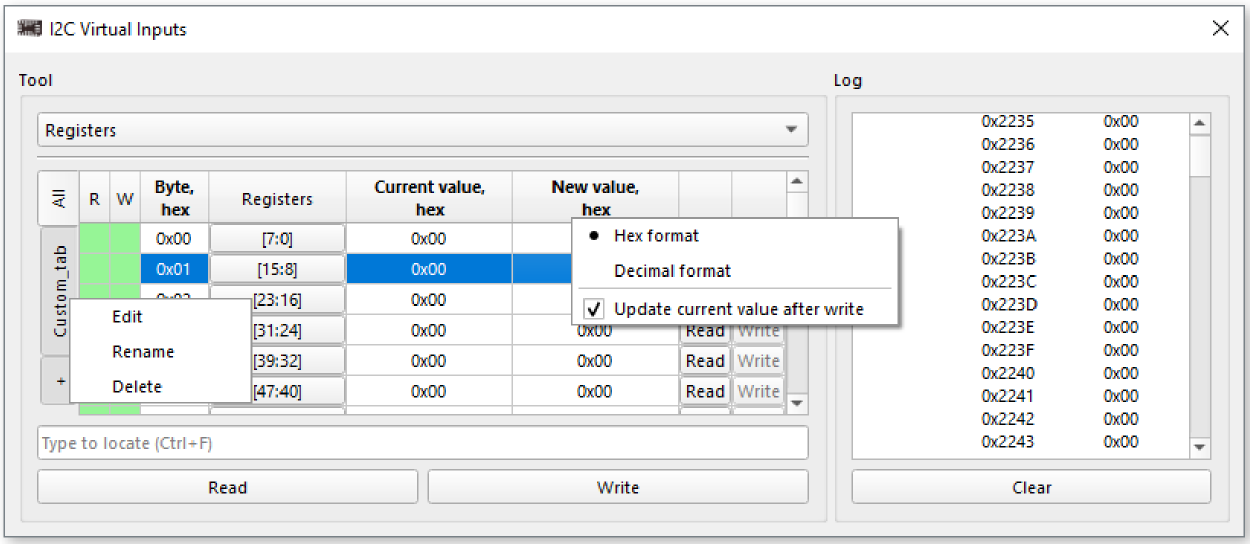


Entries
Byte: 0x00, Registers: [7:0]
Byte: 0x00, Registers: [7:0]
Byte: 0x01, Registers: [15:8]
Byte: 0x04, Registers: [39:32]

Add tab

You can also open the context menu for some additional options: edit, rename, or remove the tabs; switch between the hex or decimal value formats.

Remove a tick from *Update current value after write* checkbox if you do not wish to read the overwritten values.

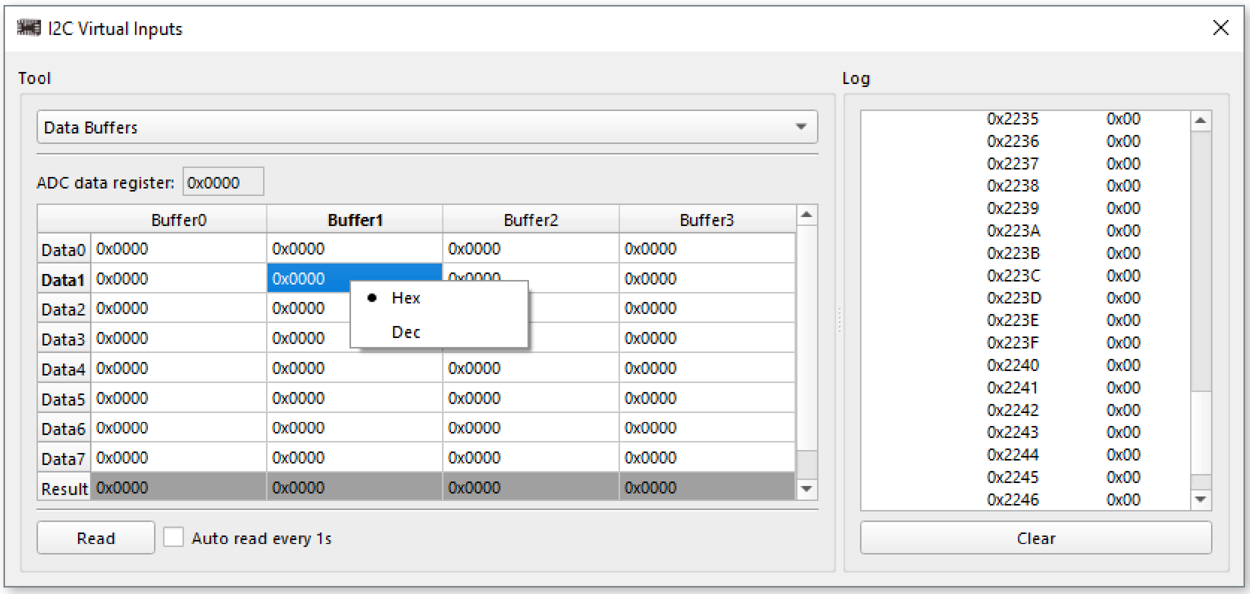


Context menus

The *R* and *W* columns show if a register is readable/writable:

🟩	I2C operations are supported
🟥	I2C operations are not supported
🟨	I2C operations are unsupported for some bits of the register

➤ *Data Buffers* — allows you to read 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*



Tool: Data Buffers

ADC data register: 0x0000

	Buffer0	Buffer1	Buffer2	Buffer3
Data0	0x0000	0x0000	0x0000	0x0000
Data1	0x0000	0x0000	0x0000	0x0000
Data2	0x0000	0x0000		0x0000
Data3	0x0000	0x0000		0x0000
Data4	0x0000	0x0000	0x0000	0x0000
Data5	0x0000	0x0000	0x0000	0x0000
Data6	0x0000	0x0000	0x0000	0x0000
Data7	0x0000	0x0000	0x0000	0x0000
Result	0x0000	0x0000	0x0000	0x0000

```
0x2235    0x00
0x2236    0x00
0x2237    0x00
0x2238    0x00
0x2239    0x00
0x223A    0x00
0x223B    0x00
0x223C    0x00
0x223D    0x00
0x223E    0x00
0x223F    0x00
0x2240    0x00
0x2241    0x00
0x2242    0x00
0x2243    0x00
0x2244    0x00
0x2245    0x00
0x2246    0x00
```

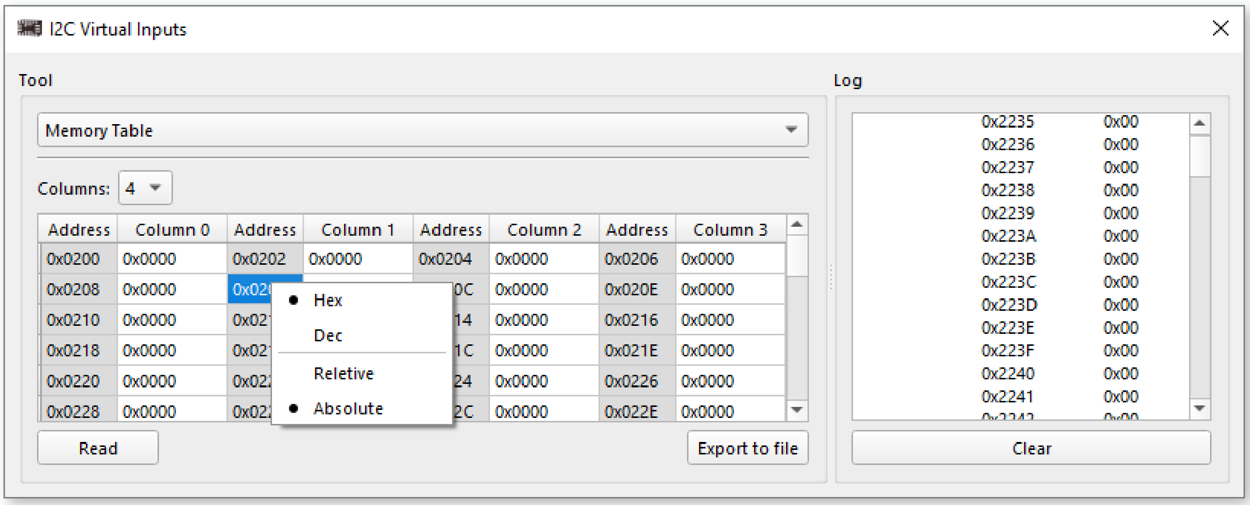
Managing Data Buffers

➤ *Memory Table* — read data from the Memory Table macrocell in the *Column X* section. To change the amount of displayed columns use the *Columns* selector. In addition to the decimal and hexadecimal formats, applicable for both column types, the *Address* columns data view can be also changed using the following options:

– *Absolute* — show the actual address of the register in the macrocell

– *Relative* — show the register address starting with zero

You can save the read data in .csv format by clicking the *Export to file* button.



Tool: Memory Table

Columns: 4

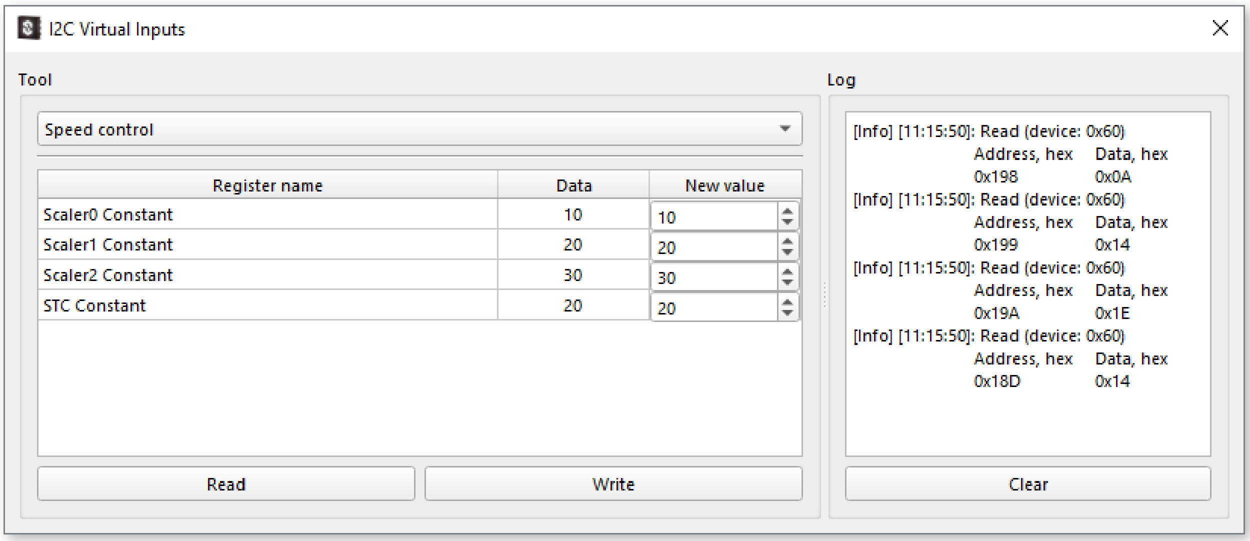
Address	Column 0	Address	Column 1	Address	Column 2	Address	Column 3
0x0200	0x0000	0x0202	0x0000	0x0204	0x0000	0x0206	0x0000
0x0208	0x0000	0x020		0C	0x0000	0x020E	0x0000
0x0210	0x0000	0x021		14	0x0000	0x0216	0x0000
0x0218	0x0000	0x021		1C	0x0000	0x021E	0x0000
0x0220	0x0000	0x022		24	0x0000	0x0226	0x0000
0x0228	0x0000	0x022		2C	0x0000	0x022E	0x0000

```
0x2235    0x00
0x2236    0x00
0x2237    0x00
0x2238    0x00
0x2239    0x00
0x223A    0x00
0x223B    0x00
0x223C    0x00
0x223D    0x00
0x223E    0x00
0x223F    0x00
0x2240    0x00
0x2241    0x00
```

Managing Memory Table

➤ *Speed control* — displays and allows to read/write the constant of the Scalers and

Speed/Torque Control macrocells

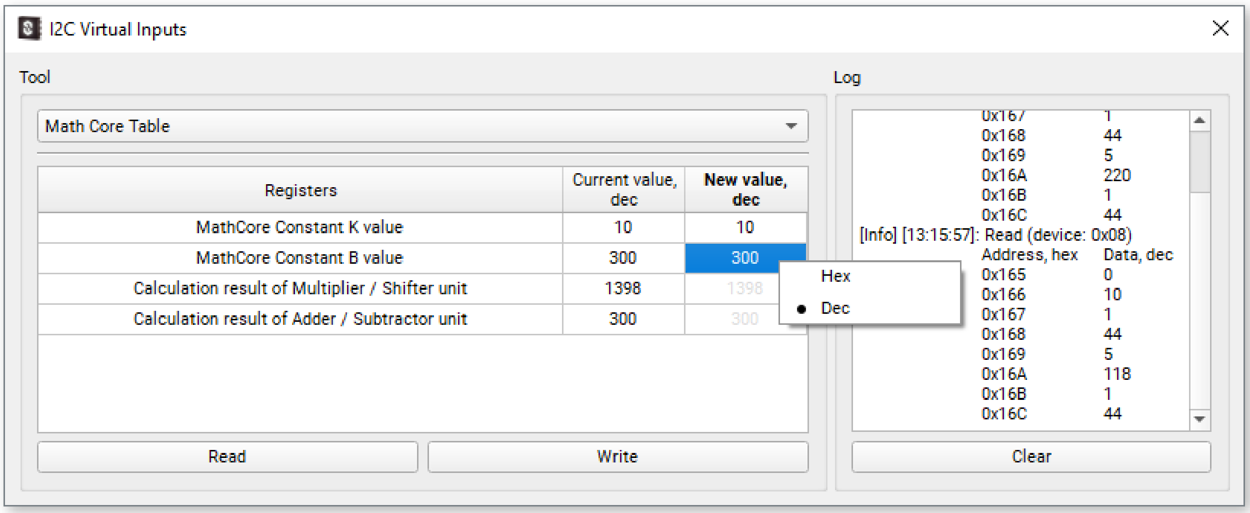


Register name	Data	New value
Scaler0 Constant	10	10
Scaler1 Constant	20	20
Scaler2 Constant	30	30
STC Constant	20	20

```
[Info] [11:15:50]: Read (device: 0x60)
                Address, hex    Data, hex
                0x198           0x0A
[Info] [11:15:50]: Read (device: 0x60)
                Address, hex    Data, hex
                0x199           0x14
[Info] [11:15:50]: Read (device: 0x60)
                Address, hex    Data, hex
                0x19A           0x1E
[Info] [11:15:50]: Read (device: 0x60)
                Address, hex    Data, hex
                0x18D           0x14
```

Speed control tool

➤ *Math Core Table* — is designed to manage the Math Core component data from its *Properties* panel. You can set and read/write the K and B constants, as well as the results of mathematical operations (addition/subtraction and multiplication/division). Change the data format in the context menu

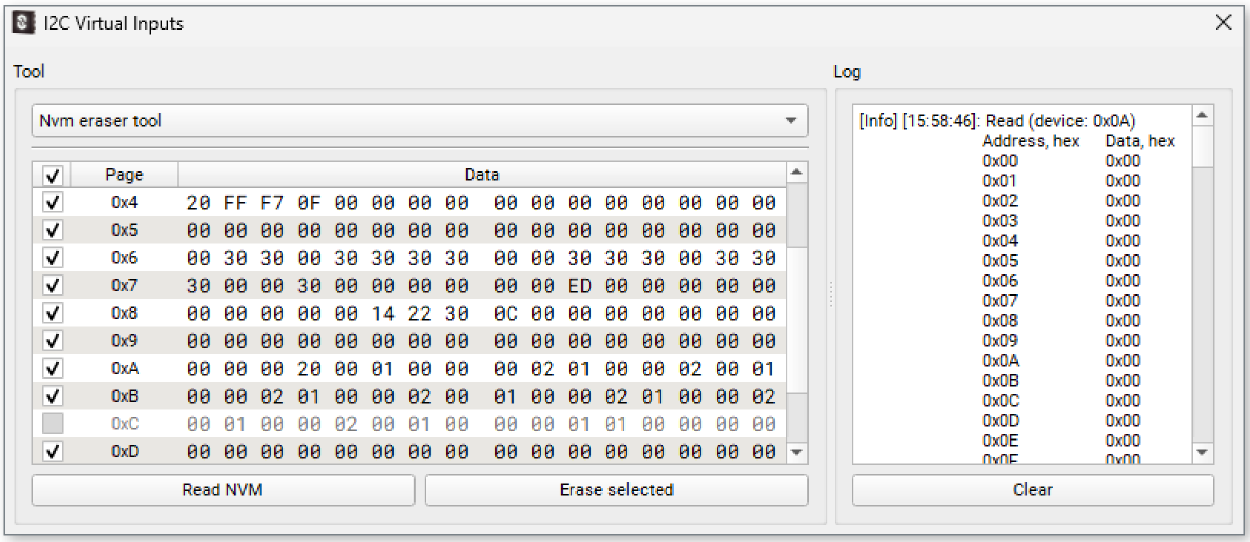


Registers	Current value, dec	New value, dec
MathCore Constant K value	10	10
MathCore Constant B value	300	300
Calculation result of Multiplier / Shifter unit	1398	1398
Calculation result of Adder / Subtractor unit	300	300

```
                0x167           1
                0x168           44
                0x169           5
                0x16A           220
                0x16B           1
                0x16C           44
[Info] [13:15:57]: Read (device: 0x08)
                Address, hex    Data, dec
                0x165           0
                0x166           10
                0x167           1
                0x168           44
                0x169           5
                0x16A           118
                0x16B           1
                0x16C           44
```

Math Core Table

➤ *NVM/EEPROM eraser tool* — allows clearing the selected NVM/EEPROM data programmed onto your chip (supports only multiple-time programmable (MTP) ICs). Once you click the *Read NVM/EEPROM* buttons, the programmed data will be displayed in the respective column. Use *Erase selected* to clear the data. While NVM/EEPROM is cleared, the *Data* column is not updated until the memory is read again. Certain pages cannot be erased as they are essential for proper chip operation

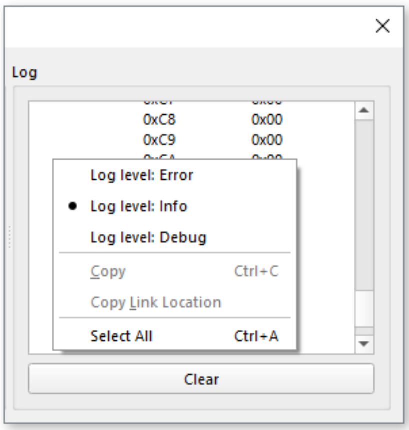


✓	Page	Data
✓	0x4	20 FF F7 0F 00 00 00 00 00 00 00 00 00 00 00 00
✓	0x5	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
✓	0x6	00 30 30 00 30 30 30 30 00 00 30 30 30 00 30 30
✓	0x7	30 00 00 30 00 00 00 00 00 00 ED 00 00 00 00 00
✓	0x8	00 00 00 00 00 14 22 30 0C 00 00 00 00 00 00 00
✓	0x9	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
✓	0xA	00 00 00 20 00 01 00 00 00 02 01 00 00 02 00 01
✓	0xB	00 00 02 01 00 00 02 00 01 00 00 02 01 00 00 02
	0xC	00 01 00 00 02 00 01 00 00 00 01 01 00 00 00 00
✓	0xD	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```
[Info] [15:58:46]: Read (device: 0x0A)
                     Address, hex    Data, hex
                     0x00            0x00
                     0x01            0x00
                     0x02            0x00
                     0x03            0x00
                     0x04            0x00
                     0x05            0x00
                     0x06            0x00
                     0x07            0x00
                     0x08            0x00
                     0x09            0x00
                     0x0A            0x00
                     0x0B            0x00
                     0x0C            0x00
                     0x0D            0x00
                     0x0E            0x00
                     0x0F            0x00
```

NVM eraser tool

➤ *Log* — the results of recent read and write operations. The *Log level* has three severity settings, such as *Debug*, *Error* and *Info* displaying the most/least detailed information



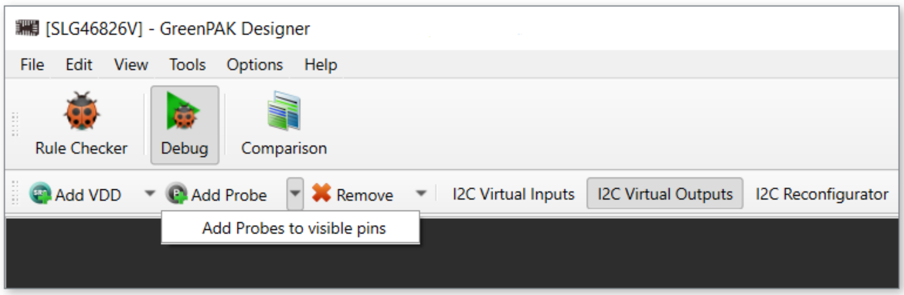
Log context menu

I2C Virtual Outputs

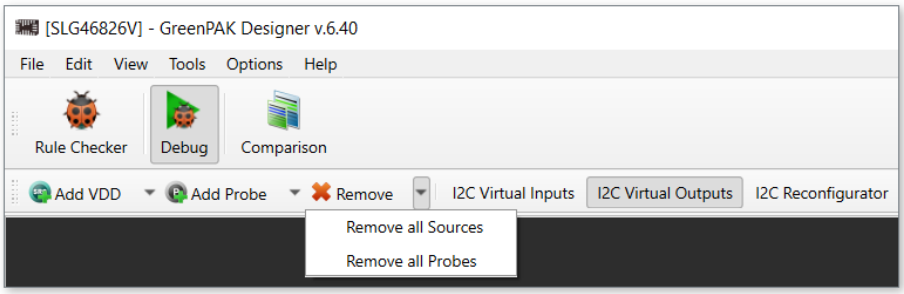
I2C Virtual Outputs allows you to read the state of certain macrocells. The tool includes the hardware probes, ASM state, and the current counted data of the specific counters.

➤ *Probes (Matrix inputs)* — reflect the current logic level on the macrocell output pins. You can add a probe by clicking *Add Probe* and then the required pin. Internal pins that support adding a probe are highlighted in green. Also, you can add probes to all visible pins at once by clicking

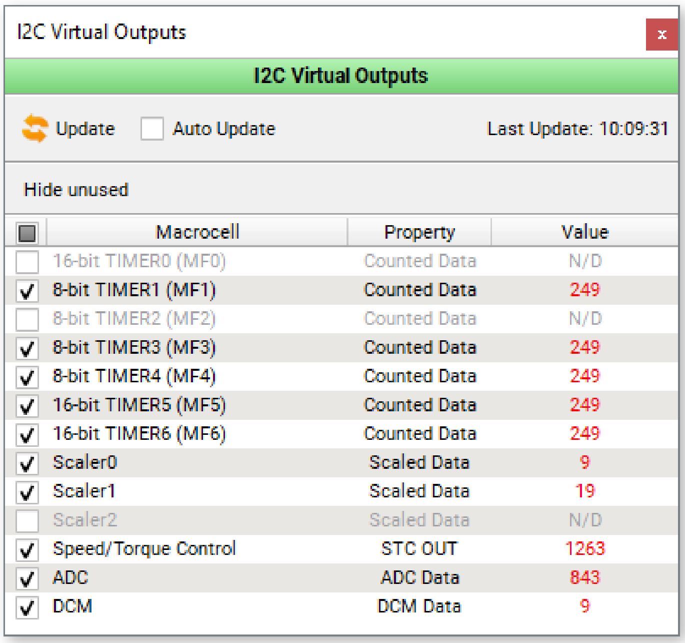
Add Probes to visible pins.



You can remove probes one by one or remove all probes at once.



➤ *I2C Virtual Outputs* tool — displays the current value of the selected components, such as e.g. CNT/DLY, ASM, Scaler, DCM, etc., read from the chip registers (see the Datasheet for the selected Part Number). To update the data or hide the unused blocks click the respective controls on the tool panel



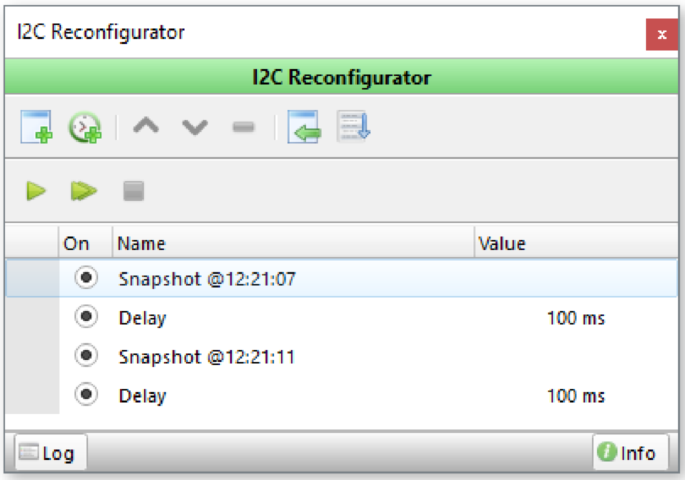
Last Update: 10:09:31

	Macrocell	Property	Value
☐	16-bit TIMER0 (MF0)	Counted Data	N/D
✓	8-bit TIMER1 (MF1)	Counted Data	249
☐	8-bit TIMER2 (MF2)	Counted Data	N/D
✓	8-bit TIMER3 (MF3)	Counted Data	249
✓	8-bit TIMER4 (MF4)	Counted Data	249
✓	16-bit TIMER5 (MF5)	Counted Data	249
✓	16-bit TIMER6 (MF6)	Counted Data	249
✓	Scaler0	Scaled Data	9
✓	Scaler1	Scaled Data	19
☐	Scaler2	Scaled Data	N/D
✓	Speed/Torque Control	STC OUT	1263
✓	ADC	ADC Data	843
✓	DCM	DCM Data	9

I2C Virtual Outputs tool

I2C Reconfigurator

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



On	Name	Value
●	Snapshot @12:21:07	
●	Delay	100 ms
●	Snapshot @12:21:11	
●	Delay	100 ms

Snapshot configuration

You can work with the *Reconfiguration scenarios* 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 see the list of changes which are applied, refer to *Reconfiguration scenario* controls:

- open a dialog with the list of changes applied; delays are included. You can also export the whole list to a file
- Import presets from a different project

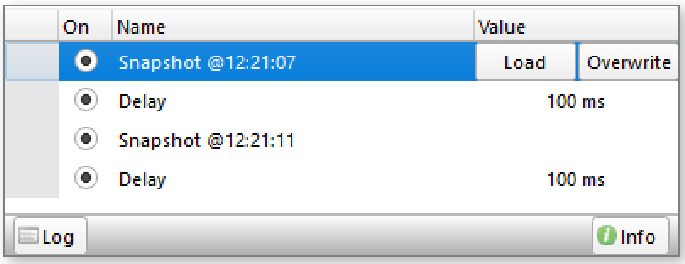
Here is the list of snapshot sending controls:

- ▶ 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

You can also manipulate the selected snapshot using the following buttons:

➤ *Load* — send the configurations of the selected snapshot to the chip

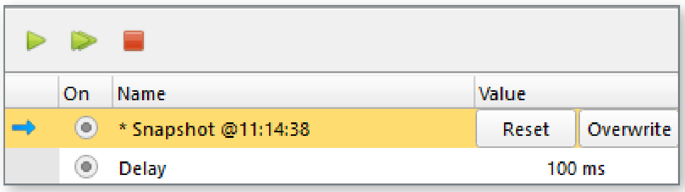
- *Overwrite* — make the desired changes and substitute the existing snapshot data



On	Name	Value
●	Snapshot @12:21:07	Load Overwrite
●	Delay	100 ms
●	Snapshot @12:21:11	
●	Delay	100 ms

Load/Overwrite snapshot configurations

- *Reset* — the button becomes available once you load a snapshot clicking the *Send one* button and make any configuration modifications afterwards. Click the button to return the snapshot to the primarily saved state



On	Name	Value
●	* Snapshot @11:14:38	Reset Overwrite
●	Delay	100 ms

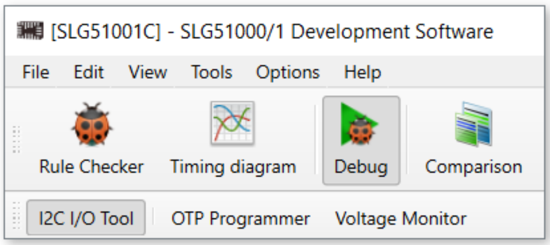
Reset snapshot configurations

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

2.2.10 I2C I/O Tool

Just like I2C Tools described above, I2C I/O Tool allows debugging the project configuration by reading or writing the register data on the chip (the tools type depends on the selected Part Number).

Once you enable *Debug* and select the required platform, the *I2C I/O Tool* button appears on the toolbar.



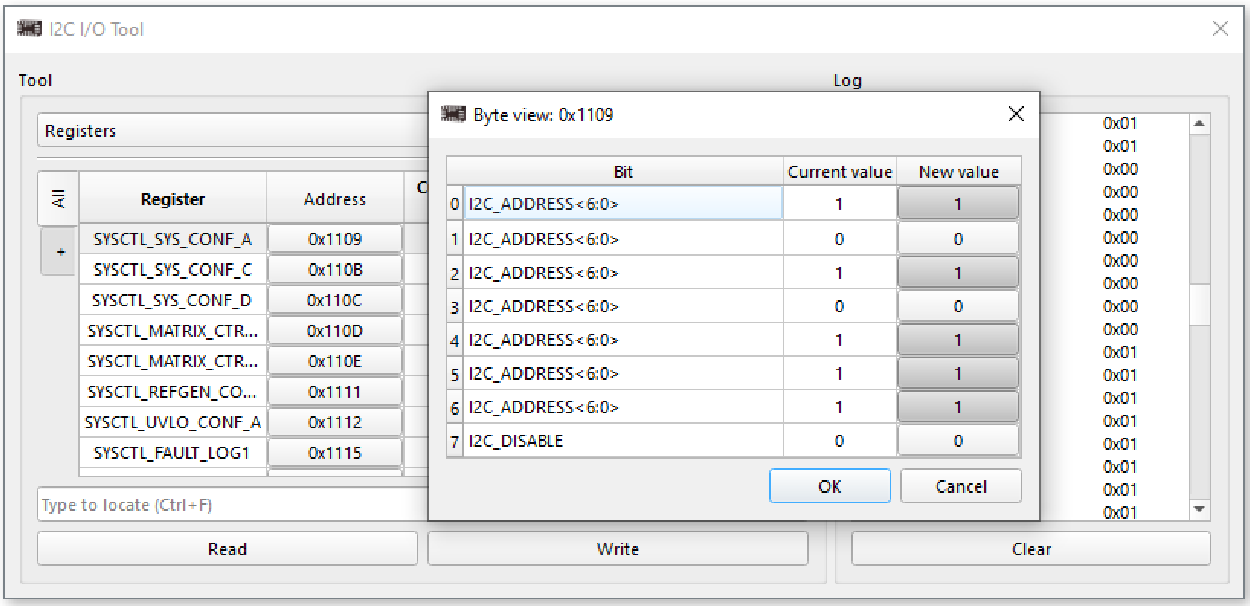
I2C I/O Tool on the toolbar

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

Note: To read/write data via 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:



I2C I/O Tool

Tool

Registers

Register	Address
SYSCTL_SYS_CONF_A	0x1109
SYSCTL_SYS_CONF_C	0x110B
SYSCTL_SYS_CONF_D	0x110C
SYSCTL_MATRIX_CTR...	0x110D
SYSCTL_MATRIX_CTR...	0x110E
SYSCTL_REFGEN_CO...	0x1111
SYSCTL_UVLO_CONF_A	0x1112
SYSCTL_FAULT_LOG1	0x1115

Type to locate (Ctrl+F)

Byte view: 0x1109

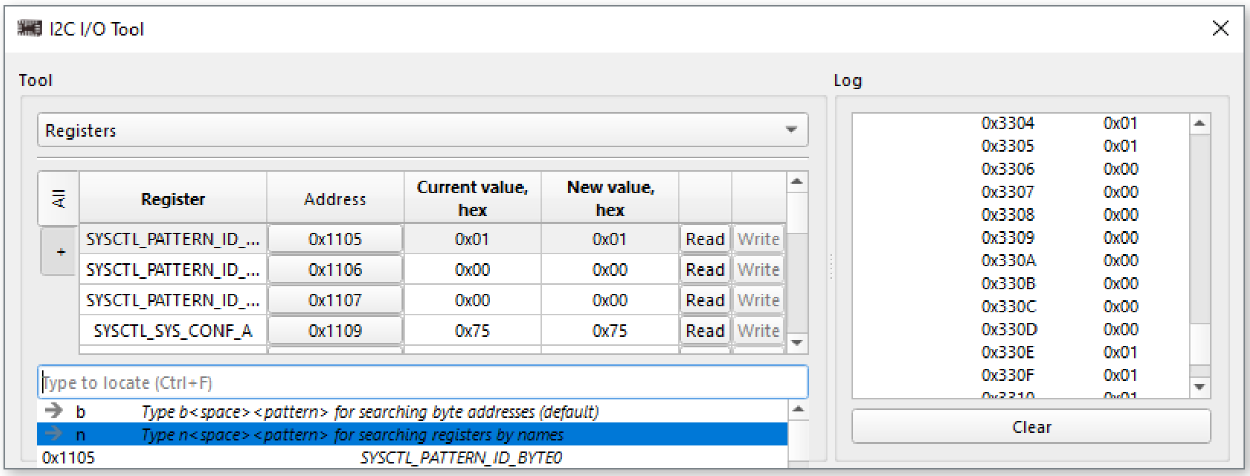
	Bit	Current value	New value
0	I2C_ADDRESS<6:0>	1	1
1	I2C_ADDRESS<6:0>	0	0
2	I2C_ADDRESS<6:0>	1	1
3	I2C_ADDRESS<6:0>	0	0
4	I2C_ADDRESS<6:0>	1	1
5	I2C_ADDRESS<6:0>	1	1
6	I2C_ADDRESS<6:0>	1	1
7	I2C_DISABLE	0	0

Log

```
0x01
0x01
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x01
0x01
0x01
0x01
0x01
0x01
0x01
0x01
```

Change specific bit

Search by byte or register name is available:



I2C I/O Tool

Tool

Registers

Register	Address	Current value, hex	New value, hex		
SYSCTL_PATTERN_ID_...	0x1105	0x01	0x01	Read	Write
SYSCTL_PATTERN_ID_...	0x1106	0x00	0x00	Read	Write
SYSCTL_PATTERN_ID_...	0x1107	0x00	0x00	Read	Write
SYSCTL_SYS_CONF_A	0x1109	0x75	0x75	Read	Write

Type to locate (Ctrl+F)

```
b   Type b<space><pattern> for searching byte addresses (default)
n   Type n<space><pattern> for searching registers by names
0x1105                SYSCTL_PATTERN_ID_BYTE0
```

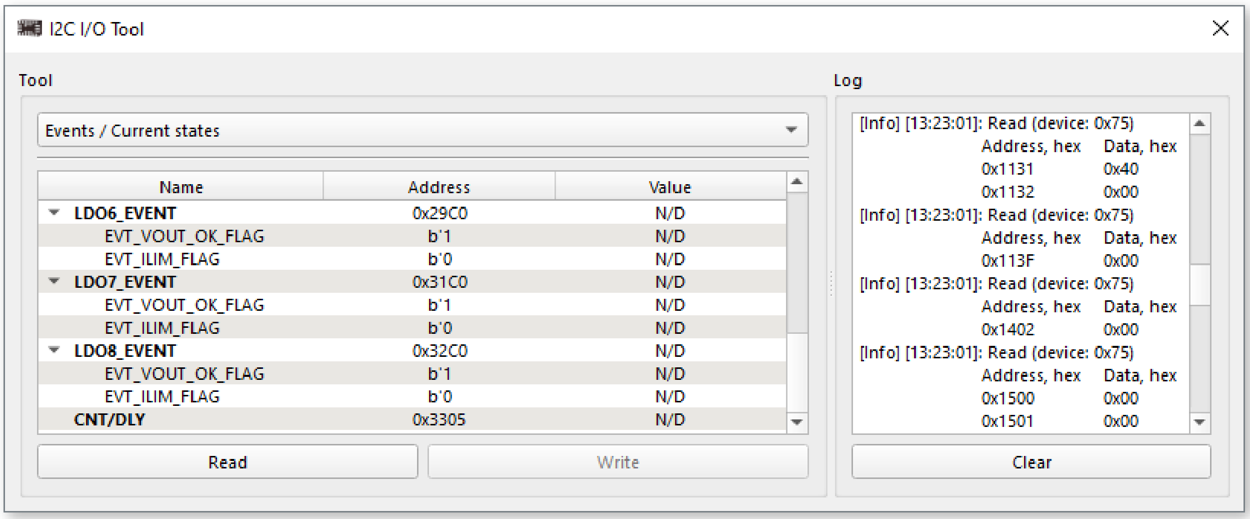
Log

```
0x3304   0x01
0x3305   0x01
0x3306   0x00
0x3307   0x00
0x3308   0x00
0x3309   0x00
0x330A   0x00
0x330B   0x00
0x330C   0x00
0x330D   0x00
0x330E   0x01
0x330F   0x01
0x3310   0x01
```

Search by byte or register

➤ *Counters/Delays* — same description as in I2C Tools section is applicable

➤ *Events/Current states* — shows the read-only list of the specific components' (e.g. LDO, CNT/DLY) events


I2C I/O Tool

Tool: Events / Current states

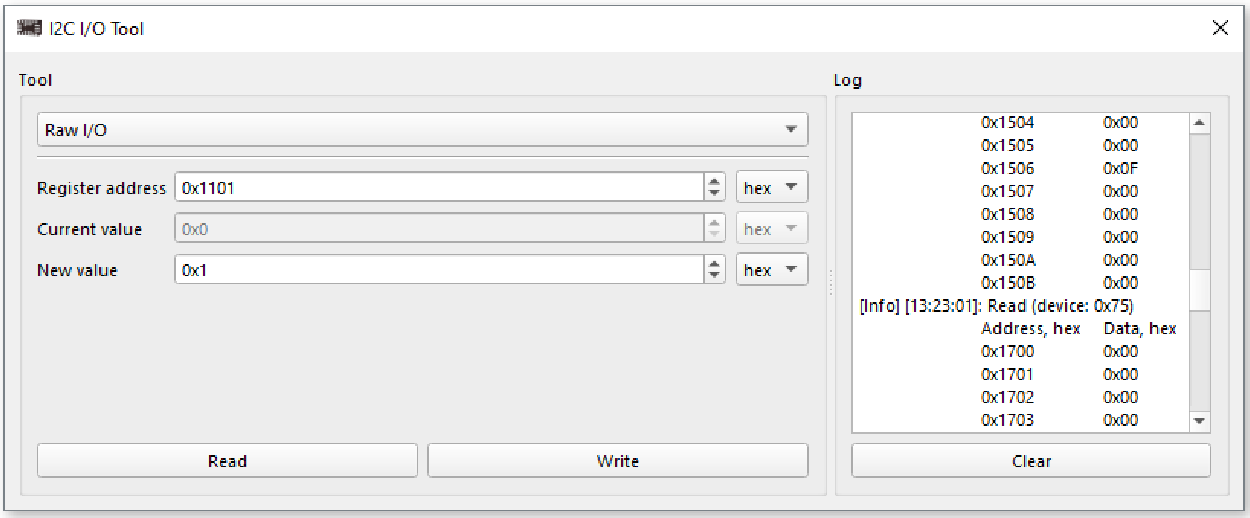
Name	Address	Value
LDO6_EVENT	0x29C0	N/D
EVT_VOUT_OK_FLAG	b'1	N/D
EVT_ILIM_FLAG	b'0	N/D
LDO7_EVENT	0x31C0	N/D
EVT_VOUT_OK_FLAG	b'1	N/D
EVT_ILIM_FLAG	b'0	N/D
LDO8_EVENT	0x32C0	N/D
EVT_VOUT_OK_FLAG	b'1	N/D
EVT_ILIM_FLAG	b'0	N/D
CNT/DLY	0x3305	N/D

Log:

```
[Info] [13:23:01]: Read (device: 0x75)
            Address, hex    Data, hex
            0x1131          0x40
            0x1132          0x00
[Info] [13:23:01]: Read (device: 0x75)
            Address, hex    Data, hex
            0x113F          0x00
[Info] [13:23:01]: Read (device: 0x75)
            Address, hex    Data, hex
            0x1402          0x00
[Info] [13:23:01]: Read (device: 0x75)
            Address, hex    Data, hex
            0x1500          0x00
            0x1501          0x00
```

I2C I/O Tool window with events

➤ *Raw I/O* — provides extended access via I2C to the chip registers. It allows reaching the registers that may be inaccessible from the *Registers* instrument


I2C I/O Tool

Tool: Raw I/O

Register address: 0x1101 hex
Current value: 0x0 hex
New value: 0x1 hex

Log:

```
            0x1504          0x00
            0x1505          0x00
            0x1506          0x0F
            0x1507          0x00
            0x1508          0x00
            0x1509          0x00
            0x150A          0x00
            0x150B          0x00
[Info] [13:23:01]: Read (device: 0x75)
            Address, hex    Data, hex
            0x1700          0x00
            0x1701          0x00
            0x1702          0x00
            0x1703          0x00
```

I2C I/O Tool window with Raw I/O

➤ *Log* — refer to I2C Tools section

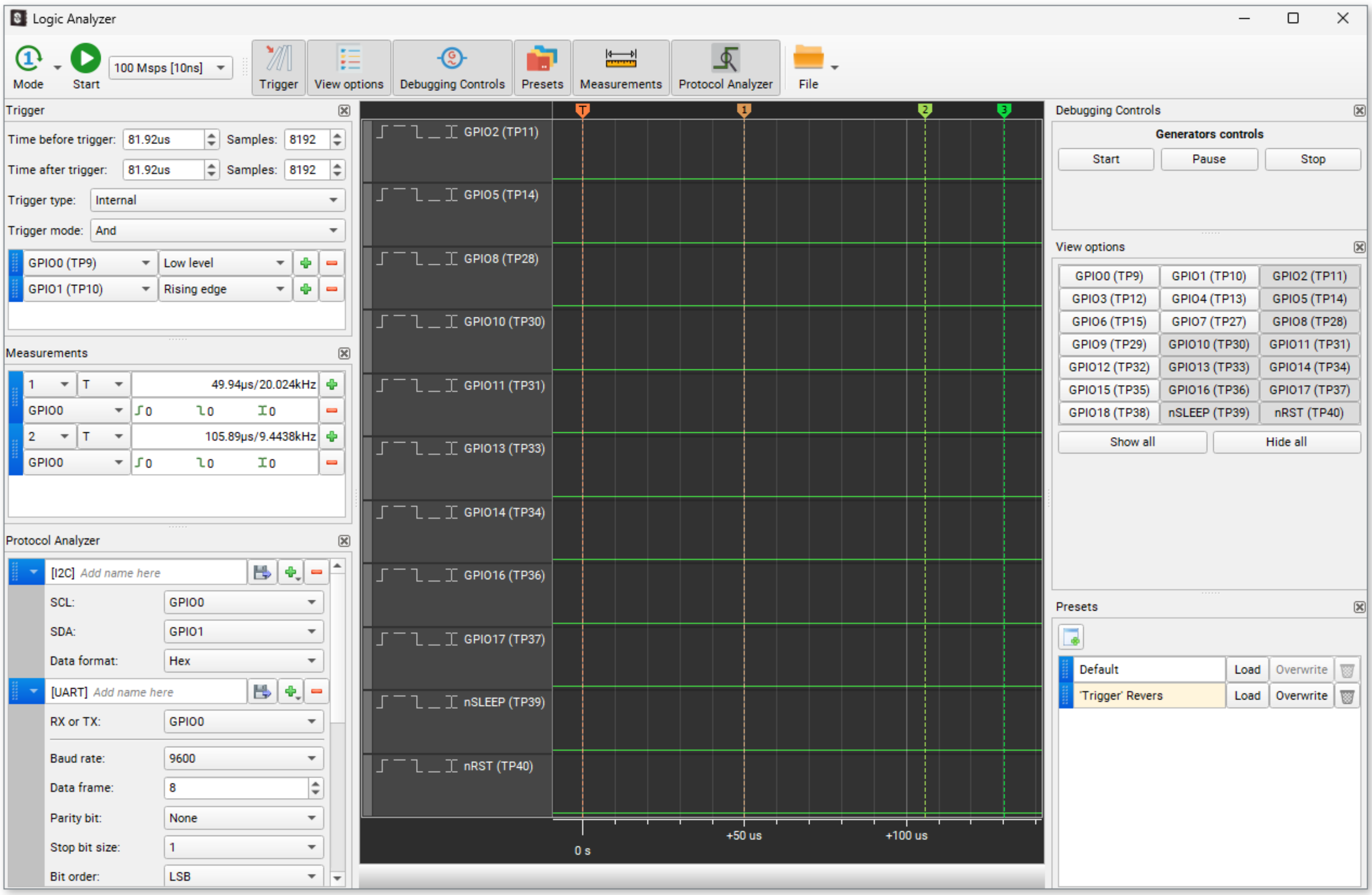
2.2.11 Logic Analyzer

Logic Analyzer tool allows you to capture multiple signals from a digital circuit. It includes triggering capabilities and a protocol decoder that helps to see 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



Trigger

Time before trigger: 81.92us Samples: 8192

Time after trigger: 81.92us Samples: 8192

Trigger type: Internal

Trigger mode: And

GPIO0 (TP9) Low level

GPIO1 (TP10) Rising edge

Measurements

1 T 49.94µs/20.024kHz

GPIO0

2 T 105.89µs/9.4438kHz

GPIO0

Protocol Analyzer

[I2C] Add name here

SCL: GPIO0

SDA: GPIO1

Data format: Hex

[UART] Add name here

RX or TX: GPIO0

Baud rate: 9600

Data frame: 8

Parity bit: None

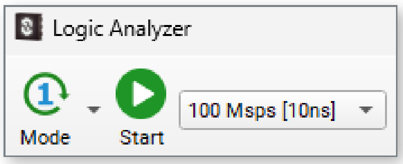
Stop bit size: 1

Bit order: LSB

Logic Analyzer window

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 once a trigger event is detected
  - – *Normal mode* — refreshes the waveform pattern each time when a 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



Main controls

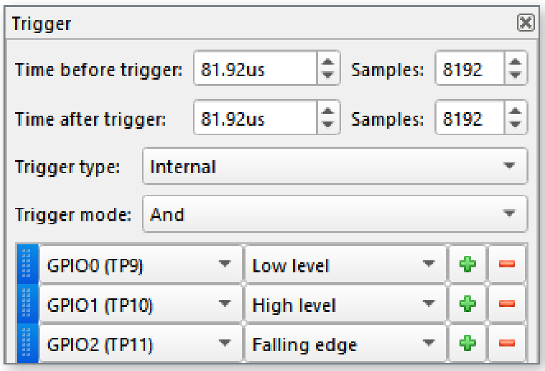
Triggers

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*), any or all of the trigger conditions added to the trigger list will be met
  - *Trigger list* — add/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 (e.g. *Rising edge*, *Falling edge* together with *And* trigger mode may result in improper trigger work).



Trigger

Time before trigger: 81.92us Samples: 8192

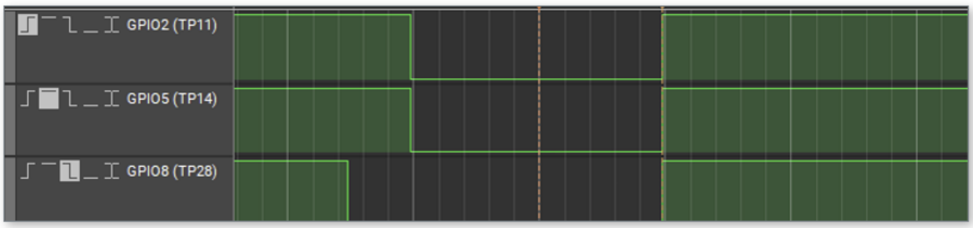
Time after trigger: 81.92us Samples: 8192

Trigger type: Internal

Trigger mode: And

GPIO0 (TP9)	Low level
GPIO1 (TP10)	High level
GPIO2 (TP11)	Falling edge

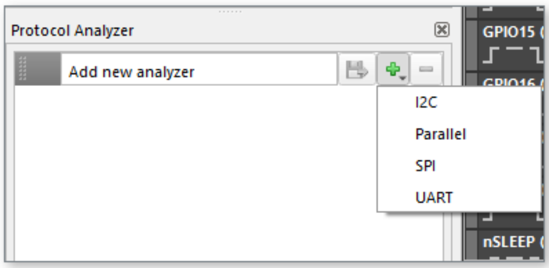
Trigger parameters



Trigger configuration

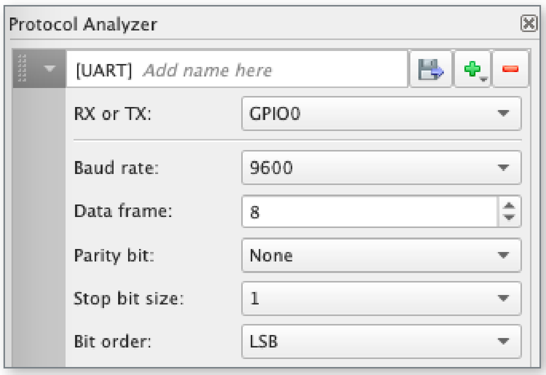
Protocol Analyzer

The *Protocol Analyzer* allows you to decode data according to a protocol.



Protocol Analyzer

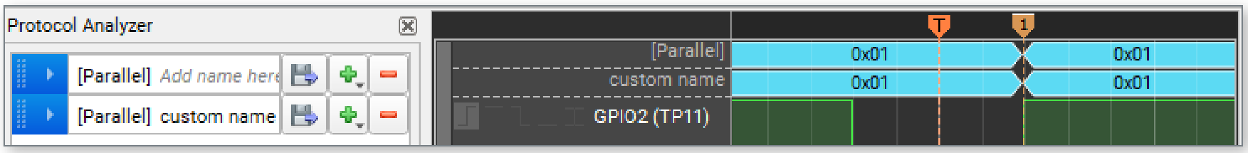
To analyze captured data, click the ✚ button and select one of the protocols.



RX or TX: GPIO0
Baud rate: 9600
Data frame: 8
Parity bit: None
Stop bit size: 1
Bit order: LSB

Protocol settings

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

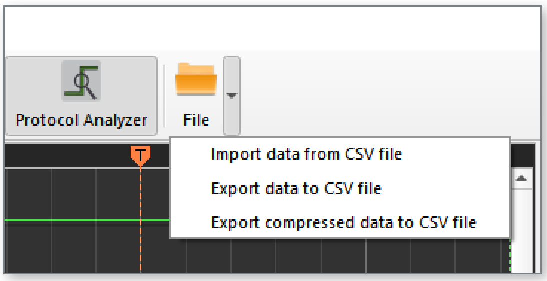


Decoded data

You can easily export data from a *Protocol Analyzer* by clicking the *Save* button next to the *Protocol Analyzer* name field. If the parameters are selected correctly, an automatic download of a CSV file will initiate.

Import / Export actions

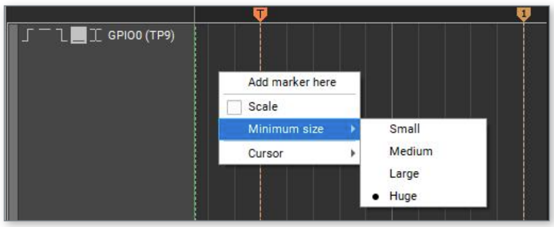
You can save/import the waveform data in the CSV format. These options are grouped under the *File* button at the top toolbar.



Export and import operations

Plot widget

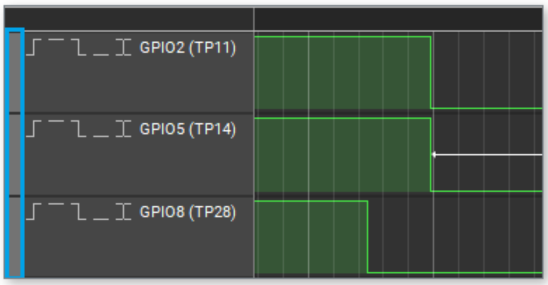
The plot widget displays the waveforms in the *Logic Analyzer* window. You can change the way a plot is shown from the plot context menu. Right-click on a plot area to add a marker, show or hide the time scale, change the plot height, and select the cursor width.



Logic Analyzer Plot menu

You may do the following actions with the plot area:

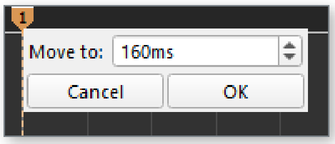
- Move left/right by click + drag left/right
- Zoom in/out by `Ctrl` + `mouse wheel` up/down
- Quick zoom in/out with the `MMB` click + drag up/down
- Reorder waveforms by drag and drop



Markers

You can do the following actions with 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



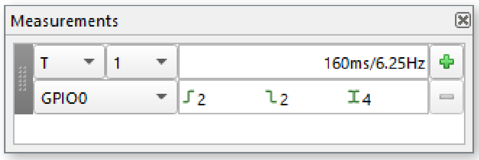
Marker Move To

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

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.

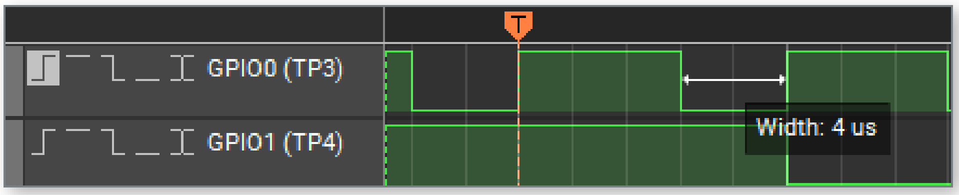


Markers measurements

Cursors

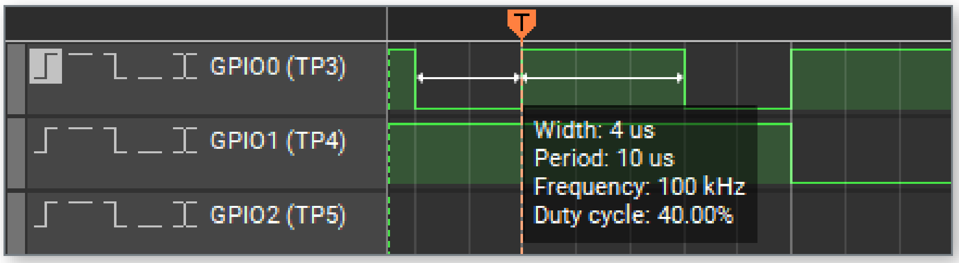
A cursor is a measurement tool for calculating waveform data between the edges of one or more plots. To see the cursor, hover a measured section of a waveform.

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



Half Period cursor and sample measurements

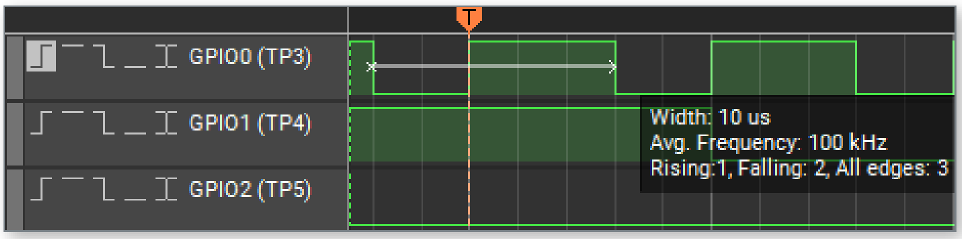
A *Period* cursor measures the width of the hovered half period, the duration of a full period, calculates the frequency, and which fraction of the full period the hovered part of the period is, which is the *Duty cycle*.



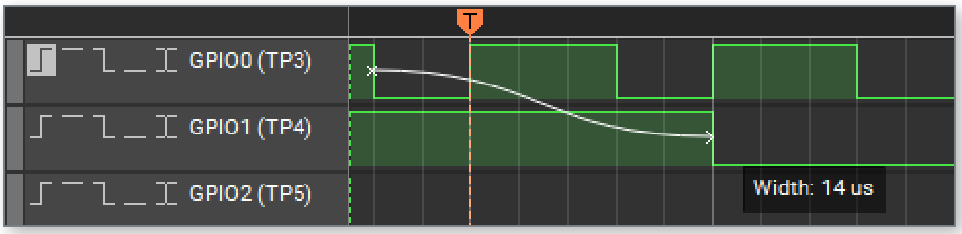
Period cursor and sample measurements

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

To measure data at a distance that exceeds one period, click the edge you want to measure from and hover over the edge you want to measure to. This will give you the total duration of the measured section, the calculated average frequency, and the quantity of rising and falling edges as well as their sum.

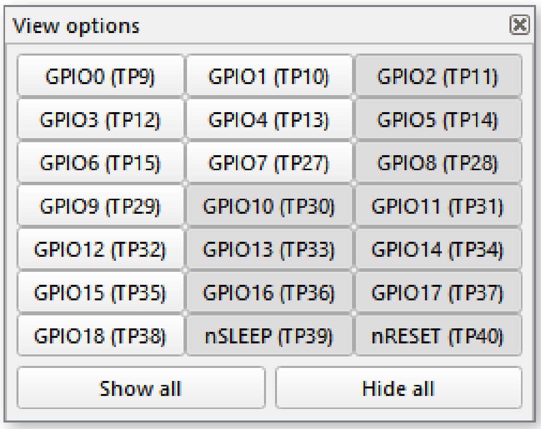


You can also measure the width between the edges on the distinct waveforms.



View options

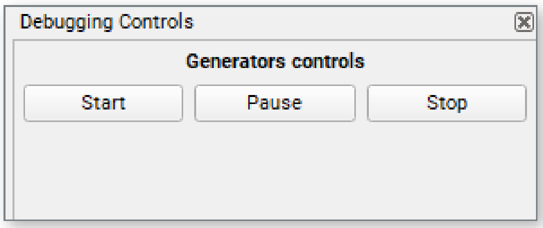
Click the channel in the *View options* panel to show/hide the corresponding plot on the plot widget.



View options panel

Debugging Controls

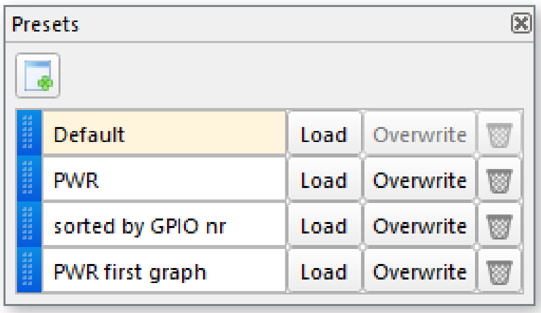
The *Debugging Controls* panel is responsible for *Emulation* and *Test mode* chip procedures, along with *Generator controls* functionality.



Debugging controls

Presets

You can store your *Logic Analyzer* configurations in presets and restore a previous configuration when needed.



Presets			
Default	Load	Overwrite	
PWR	Load	Overwrite	
sorted by GPIO nr	Load	Overwrite	
PWR first graph	Load	Overwrite	

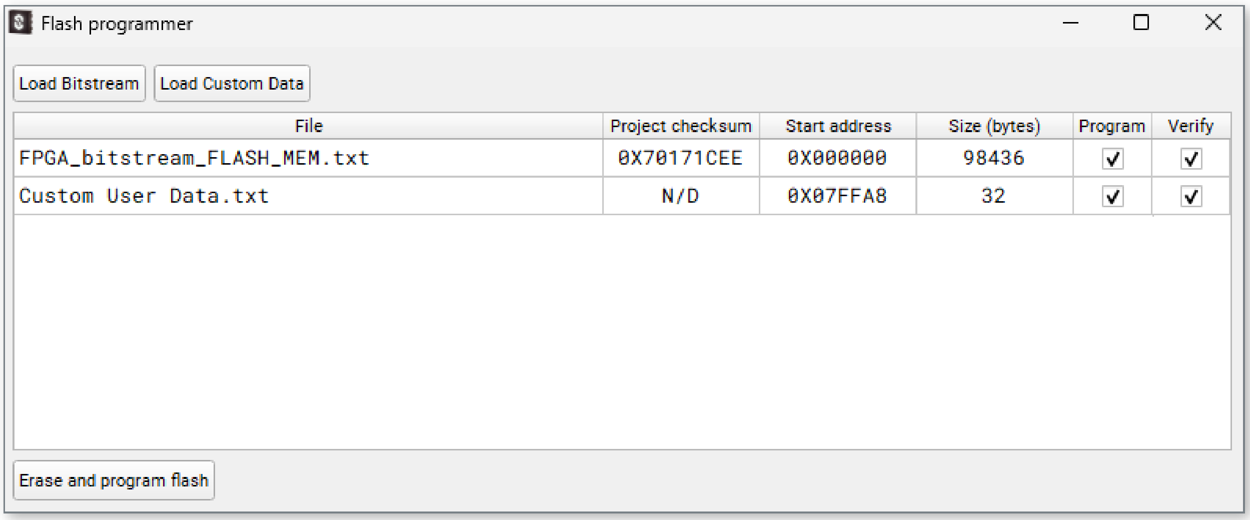
Presets

- — create a new preset with the current *Logic Analyzer* configuration
- *Load* — load a selected preset configuration
- *Overwrite* — overwrite 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

2.2.12 Flash Programmer

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

To start working with the tool, choose data via the *Load Bitstream* or *Load Custom Data* buttons in the upper section:



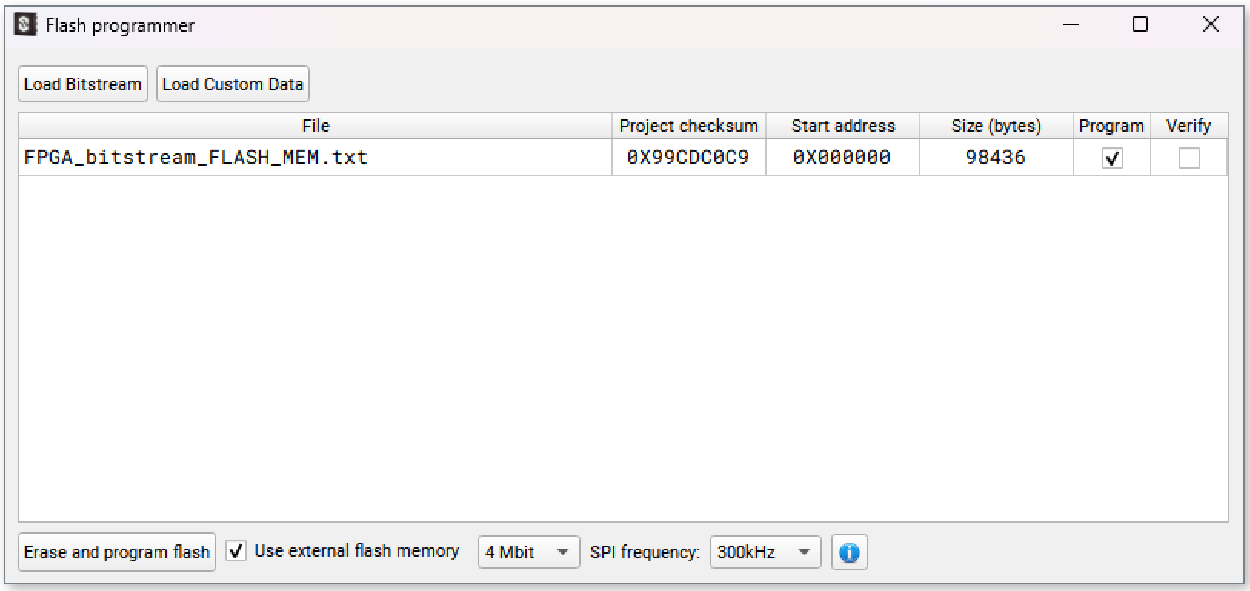
File	Project checksum	Start address	Size (bytes)	Program	Verify
FPGA_bitstream_FLASH_MEM.txt	0X70171CEE	0X000000	98436	✓	✓
Custom User Data.txt	N/D	0X07FFA8	32	✓	✓

Flash programmer tool

- *Load Bitstream* — upon adding a file, the tool validates file 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 footer. Click the info icon to check the instruction for the flash parameter configuration. Specify the external flash size and set the SPI frequency to match your device specifications.

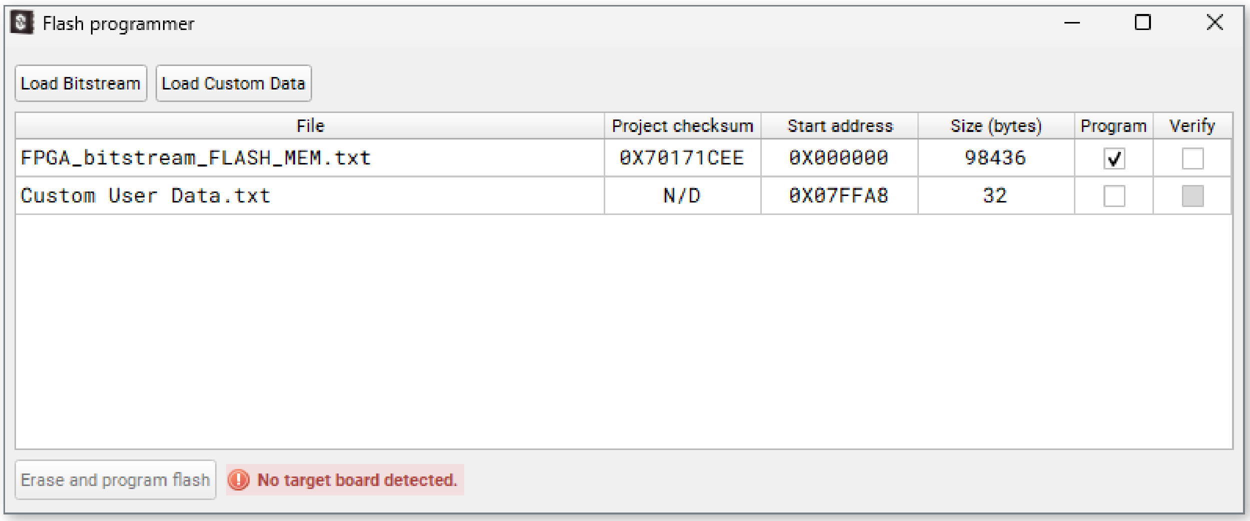


File	Project checksum	Start address	Size (bytes)	Program	Verify
FPGA_bitstream_FLASH_MEM.txt	0X99CDC0C9	0X000000	98436	✓	☐

✓ Use external flash memory: 4 Mbit; SPI frequency: 300kHz

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 the validation.



File	Project checksum	Start address	Size (bytes)	Program	Verify
FPGA_bitstream_FLASH_MEM.txt	0X70171CEE	0X000000	98436	✓	☐
Custom User Data.txt	N/D	0X07FFA8	32	☐	☐

No target board detected.

Flash programmer tool info hint example

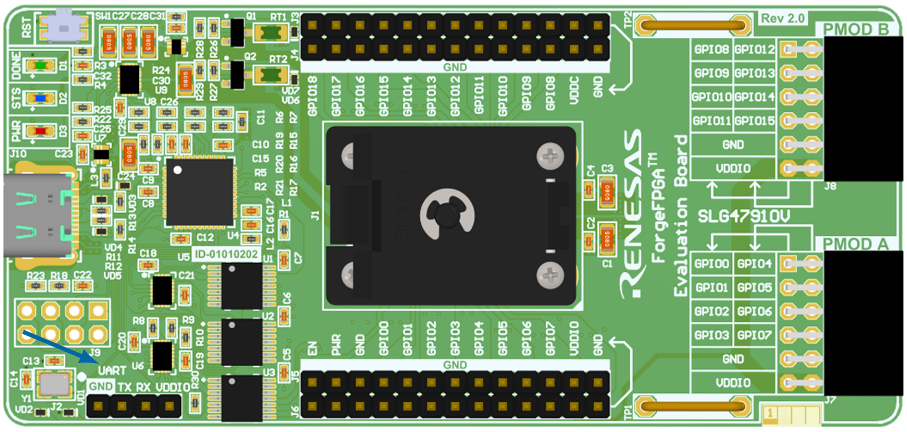
Optionally, enable the *Verify* column. When activated, *Flash programmer* reads the programmed range defined by the *Start address* and *Size* columns and compares it with the loaded file data.

Right-click the file name to access options to remove or replace the file via the context menu. Hover over the file to see its location on your computer.

Note: The entire flash memory is erased before programming.

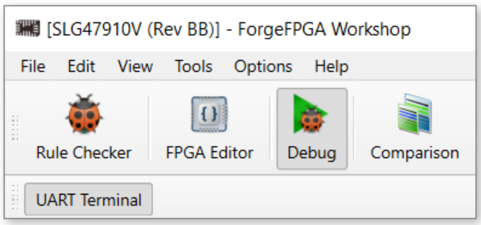
2.2.13 UART Terminal

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

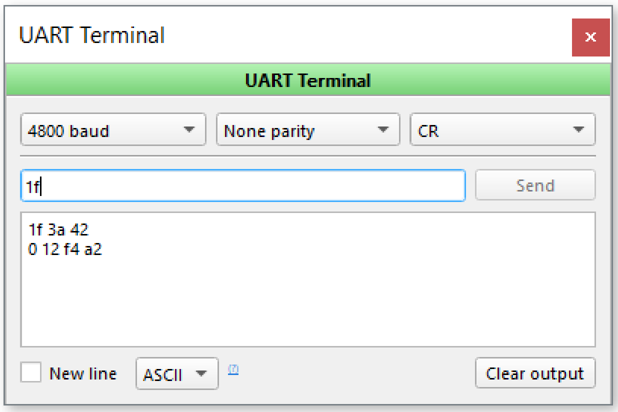


The UART port on the ForgeFPGA Evaluation Board

To start working with the terminal, make sure you select the platform supporting this feature. Open the *UART Terminal* tool at the top toolbar.



UART Terminal on the toolbar



```
1f 3a 42
0 12 f4 a2
```

UART Terminal window

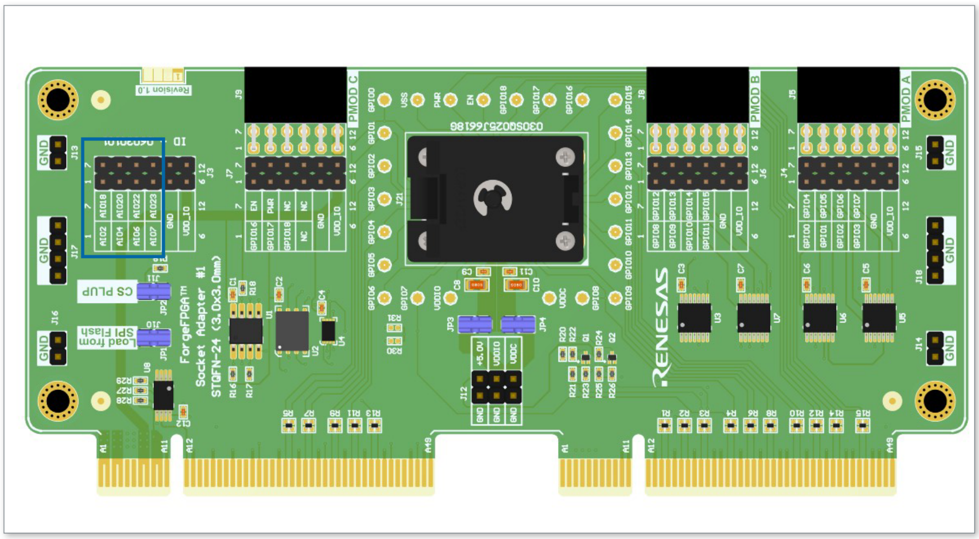
The *UART Terminal* key features are as follows:

- *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 the hex format, the input case is independent, numbers are separated by a space

The input field allows copy/paste operations. The output field allows copying the received data.

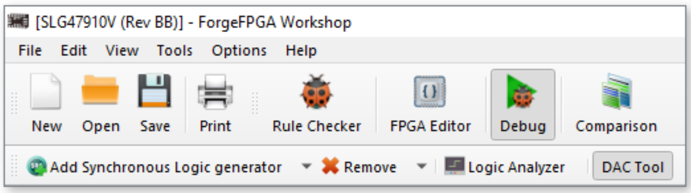
2.2.14 DAC Tool

The *DAC Tool* allows you to configure the voltage level on special analog output socket pins.



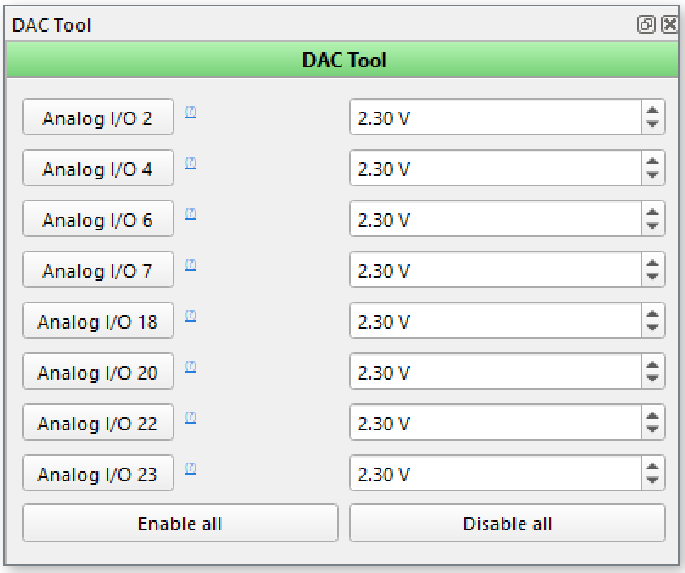
Analog I/O pins on the ForgeFPGA socket adapter

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



DAC Tool on the toolbar

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

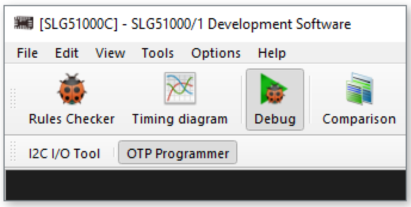


DAC Tool window

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

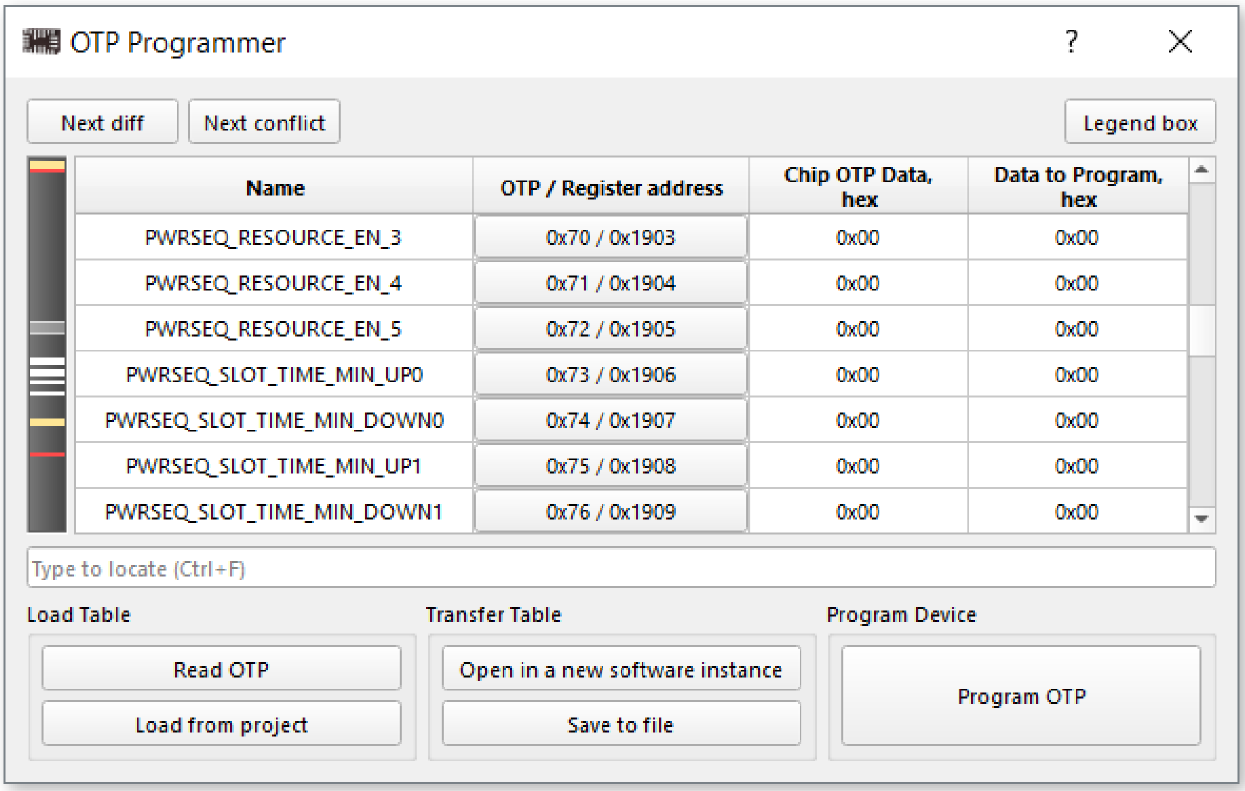
2.2.15 OTP Programmer

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



OTP Programmer on the toolbar

To reach the *OTP Programmer* tool, click on *Debug* button and select suitable development platform.



Name	OTP / Register address	Chip OTP Data, hex	Data to Program, hex
PWRSEQ_RESOURCE_EN_3	0x70 / 0x1903	0x00	0x00
PWRSEQ_RESOURCE_EN_4	0x71 / 0x1904	0x00	0x00
PWRSEQ_RESOURCE_EN_5	0x72 / 0x1905	0x00	0x00
PWRSEQ_SLOT_TIME_MIN_UP0	0x73 / 0x1906	0x00	0x00
PWRSEQ_SLOT_TIME_MIN_DOWN0	0x74 / 0x1907	0x00	0x00
PWRSEQ_SLOT_TIME_MIN_UP1	0x75 / 0x1908	0x00	0x00
PWRSEQ_SLOT_TIME_MIN_DOWN1	0x76 / 0x1909	0x00	0x00

OTP Programmer tool

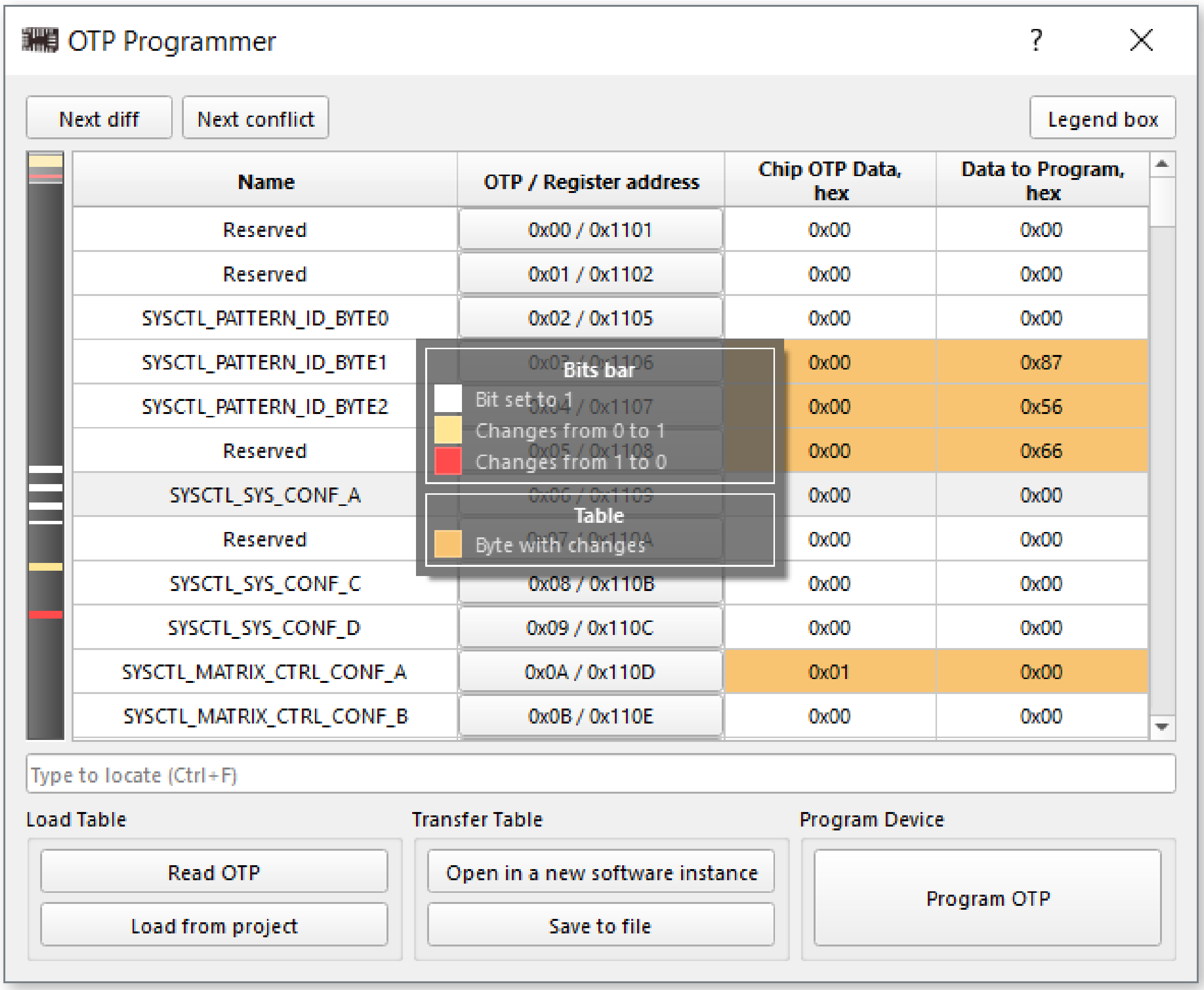
Before you start working with *OTP Programmer*, check whether the chip is detected (platform and chip info should appear in the *Info details* on the *Debugging Controls* panel).

The tool contains the following controls:

- *Read OTP* — read the data from the chip and load/set 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 the new software instance

- *Save to file* — save the *Data to Program* into 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 using *Read OTP* or *Load from project* buttons.


OTP Programmer

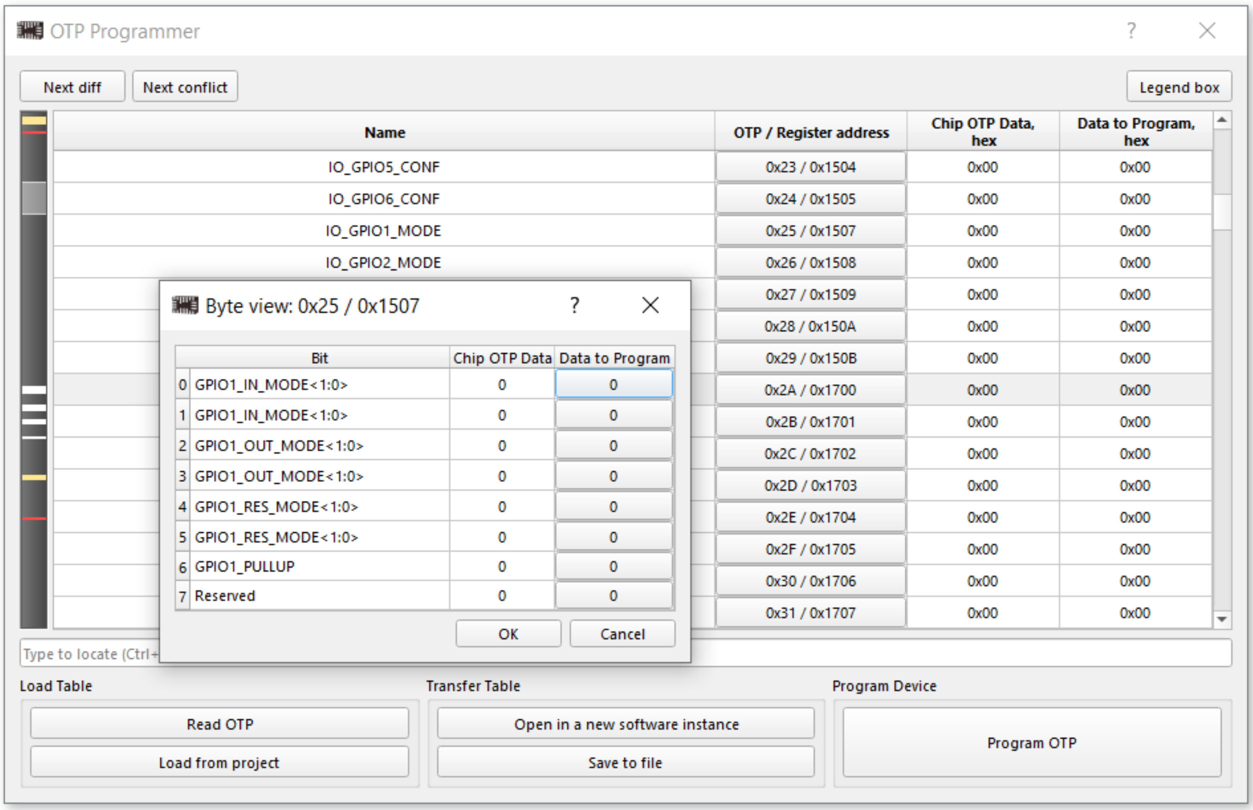
Name	OTP / Register address	Chip OTP Data, hex	Data to Program, hex
Reserved	0x00 / 0x1101	0x00	0x00
Reserved	0x01 / 0x1102	0x00	0x00
SYSCTL_PATTERN_ID_BYTE0	0x02 / 0x1105	0x00	0x00
SYSCTL_PATTERN_ID_BYTE1	0x03 / 0x1106	0x00	0x87
SYSCTL_PATTERN_ID_BYTE2	0x04 / 0x1107	0x00	0x56
Reserved	0x05 / 0x1108	0x00	0x66
SYSCTL_SYS_CONF_A	0x06 / 0x1109	0x00	0x00
Reserved	0x07 / 0x110A	0x00	0x00
SYSCTL_SYS_CONF_C	0x08 / 0x110B	0x00	0x00
SYSCTL_SYS_CONF_D	0x09 / 0x110C	0x00	0x00
SYSCTL_MATRIX_CTRL_CONF_A	0x0A / 0x110D	0x01	0x00
SYSCTL_MATRIX_CTRL_CONF_B	0x0B / 0x110E	0x00	0x00

Bits bar
- ⬜ Bit set to 1
- 🟨 Changes from 0 to 1
- 🟥 Changes from 1 to 0

Table
- 🟧 Byte with changes

Legend Box

- *Legend Box* — shows the color scheme of Bits changes in *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



OTP Programmer

Next diff | Next conflict | Legend box

Name	OTP / Register address	Chip OTP Data, hex	Data to Program, hex
IO_GPIO5_CONF	0x23 / 0x1504	0x00	0x00
IO_GPIO6_CONF	0x24 / 0x1505	0x00	0x00
IO_GPIO1_MODE	0x25 / 0x1507	0x00	0x00
IO_GPIO2_MODE	0x26 / 0x1508	0x00	0x00
	0x27 / 0x1509	0x00	0x00
	0x28 / 0x150A	0x00	0x00
	0x29 / 0x150B	0x00	0x00
	0x2A / 0x1700	0x00	0x00
	0x2B / 0x1701	0x00	0x00
	0x2C / 0x1702	0x00	0x00
	0x2D / 0x1703	0x00	0x00
	0x2E / 0x1704	0x00	0x00
	0x2F / 0x1705	0x00	0x00
	0x30 / 0x1706	0x00	0x00
	0x31 / 0x1707	0x00	0x00

Byte view: 0x25 / 0x1507

	Bit	Chip OTP Data	Data to Program
0	GPIO1_IN_MODE<1:0>	0	0
1	GPIO1_IN_MODE<1:0>	0	0
2	GPIO1_OUT_MODE<1:0>	0	0
3	GPIO1_OUT_MODE<1:0>	0	0
4	GPIO1_RES_MODE<1:0>	0	0
5	GPIO1_RES_MODE<1:0>	0	0
6	GPIO1_PULLUP	0	0
7	Reserved	0	0

OK | Cancel

Type to locate (Ctrl+

Load Table: Read OTP | Load from project

Transfer Table: Open in a new software instance | Save to file

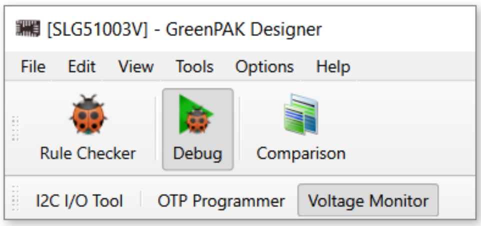
Program Device: Program OTP

Byte view

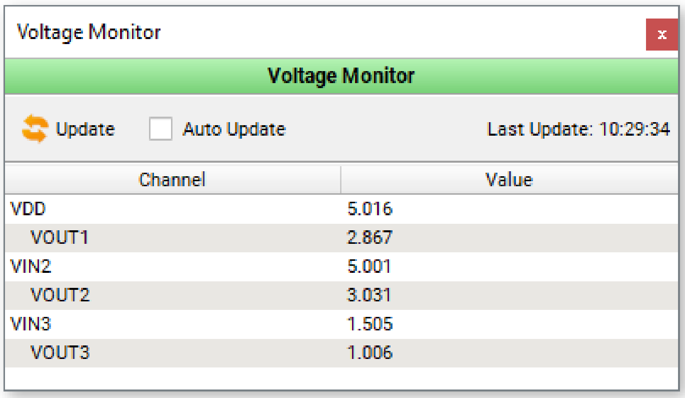
You can edit *Data to Program* by double-clicking a cell in the *Data to Program* column or open the *Byte view window* by clicking the required cell in the *OTP/Register address* column.

2.2.16 Voltage Monitor

The *Voltage Monitor* tool helps you to measure the voltage on the ADC channels. The tool appears on the toolbar after you click *Debug* and select the appropriate development platform.



Voltage Monitor on the toolbar


Voltage Monitor

Voltage Monitor

Update | Auto Update | Last Update: 10:29:34

Channel	Value
VDD	5.016
VOUT1	2.867
VIN2	5.001
VOUT2	3.031
VIN3	1.505
VOUT3	1.006

Voltage Monitor tool

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

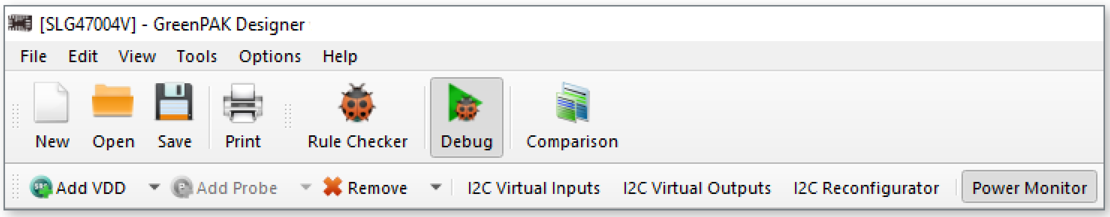
The tool also contains the *Channel/Value* table.

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

2.2.17 Power Monitor

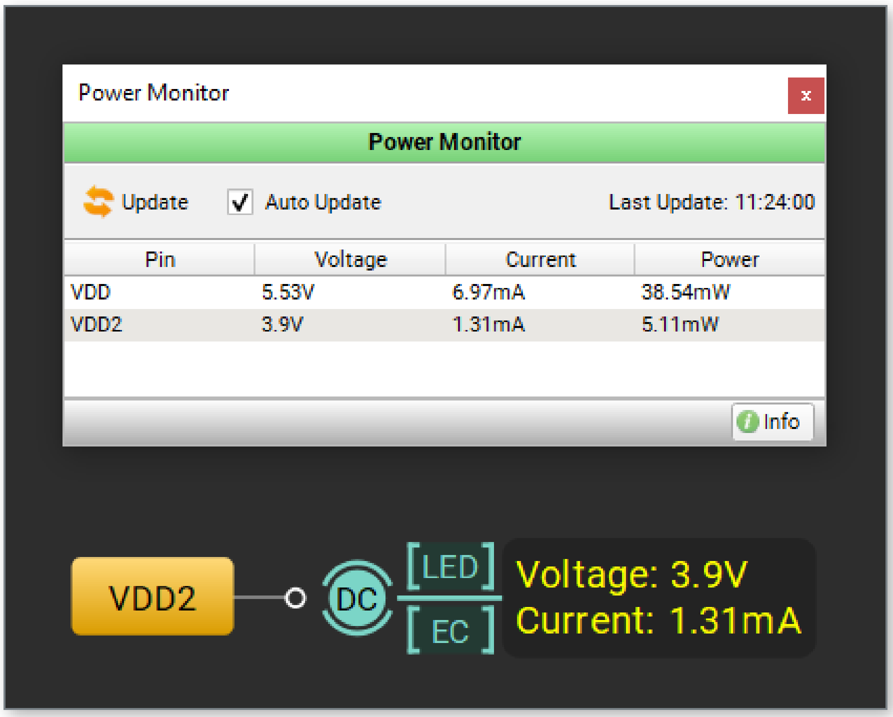
The *Power Monitor* tool is designed to measure the chip voltage and current for power consumption calculation. Find the tool on the toolbar after you click *Debug* and select the appropriate development platform.

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



Power Monitor on the toolbar

To update or auto-update the data use the corresponding controls. For more tool details click the information button.



Last Update: 11:24:00

Pin	Voltage	Current	Power
VDD	5.53V	6.97mA	38.54mW
VDD2	3.9V	1.31mA	5.11mW

Voltage: 3.9V
Current: 1.31mA

Power Monitor with work area label

Work area labels are a helpful reference for displaying voltage and current values. Once *Emulation* or *Test mode* is activated and the data is refreshed, these labels light up, duplicating values from the *Power Monitor* window for a convenient 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 calculations for power consumption, since current cannot be measured by the tool under this condition

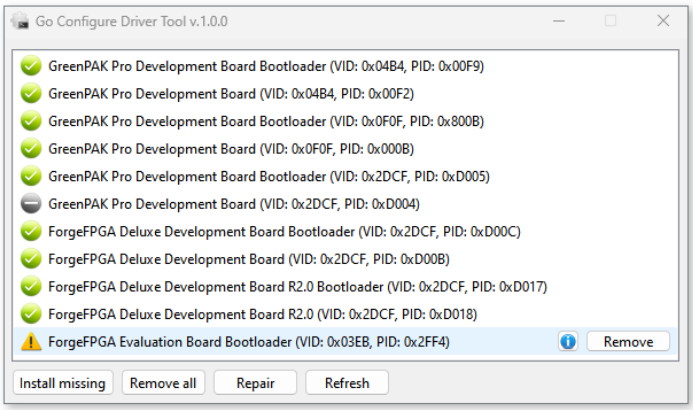
2.2.18 Go Configure Driver Tool

If the driver required for the development platform detection is missing, or a conflicting driver is present in the system, the board will not be connected successfully. *Go Configure Driver Tool* is designed to resolve the described 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 conflict (remove conflicting drivers and install the missing ones if required)

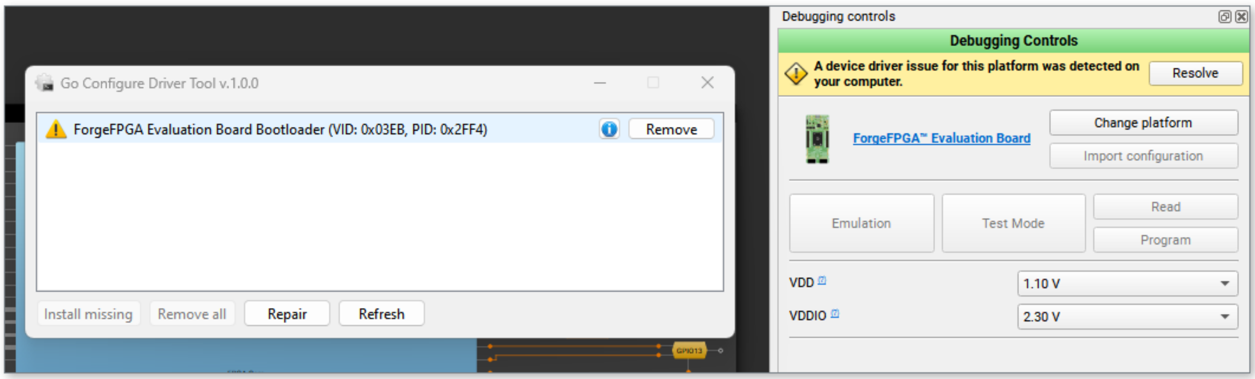
You can launch the tool manually via the main menu, *Tools → Go Configure Driver Tool*. It displays the list of the development platforms and the driver statuses (whether the driver is present, missing, or conflicting).


Go Configure Driver Tool v.1.0.0

✓ GreenPAK Pro Development Board Bootloader (VID: 0x04B4, PID: 0x00F9)
✓ GreenPAK Pro Development Board (VID: 0x04B4, PID: 0x00F2)
✓ GreenPAK Pro Development Board Bootloader (VID: 0x0F0F, PID: 0x800B)
✓ GreenPAK Pro Development Board (VID: 0x0F0F, PID: 0x000B)
✓ GreenPAK Pro Development Board Bootloader (VID: 0x2DCF, PID: 0xD005)
⊖ GreenPAK Pro Development Board (VID: 0x2DCF, PID: 0xD004)
✓ ForgeFPGA Deluxe Development Board Bootloader (VID: 0x2DCF, PID: 0xD00C)
✓ ForgeFPGA Deluxe Development Board (VID: 0x2DCF, PID: 0xD00B)
✓ ForgeFPGA Deluxe Development Board R2.0 Bootloader (VID: 0x2DCF, PID: 0xD017)
✓ ForgeFPGA Deluxe Development Board R2.0 (VID: 0x2DCF, PID: 0xD018)
⚠ ForgeFPGA Evaluation Board Bootloader (VID: 0x03EB, PID: 0x2FF4)

Go Configure Driver Tool

Once the platform is selected the banner on the *Debugging Controls* panel will notify about 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.


Go Configure Driver Tool v.1.0.0

⚠ ForgeFPGA Evaluation Board Bootloader (VID: 0x03EB, PID: 0x2FF4)

Debugging controls

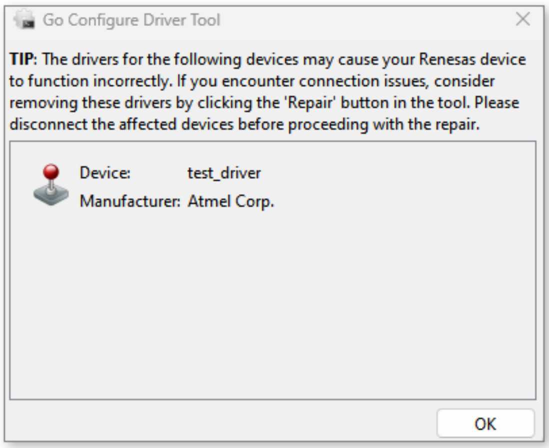
A device driver issue for this platform was detected on your computer.

VDD 1.10 V

VDDIO 2.30 V

Driver issue detection

Click the info icon to find out more about the conflicting driver that prevents successful board detection.


Go Configure Driver Tool

TIP: The drivers for the following devices may cause your Renesas device to function incorrectly. If you encounter connection issues, consider removing these drivers by clicking the 'Repair' button in the tool. Please disconnect the affected devices before proceeding with the repair.

Device:	test_driver
Manufacturer:	Atmel Corp.

Conflicting driver info pop-up

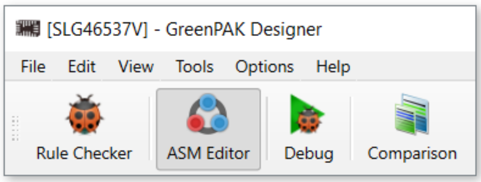
Once the driver issues are resolved, the pop-up will prompt you to reconnect the platform.

2.3 Configuration tools

2.3.1 Asynchronous State Machine Editor

GreenPAK family devices featuring the *Asynchronous State Machine (ASM)* macrocell allows you to develop personalized state machine designs. You can establish state definitions, define permissible state transitions, and specify the signals responsible for initiating each state transition. Moreover, you can link this macrocell to various I/O Pins and other internal *GreenPAK* components to activate inputs for state transitions. Outputs from the macrocell can be conveniently 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.

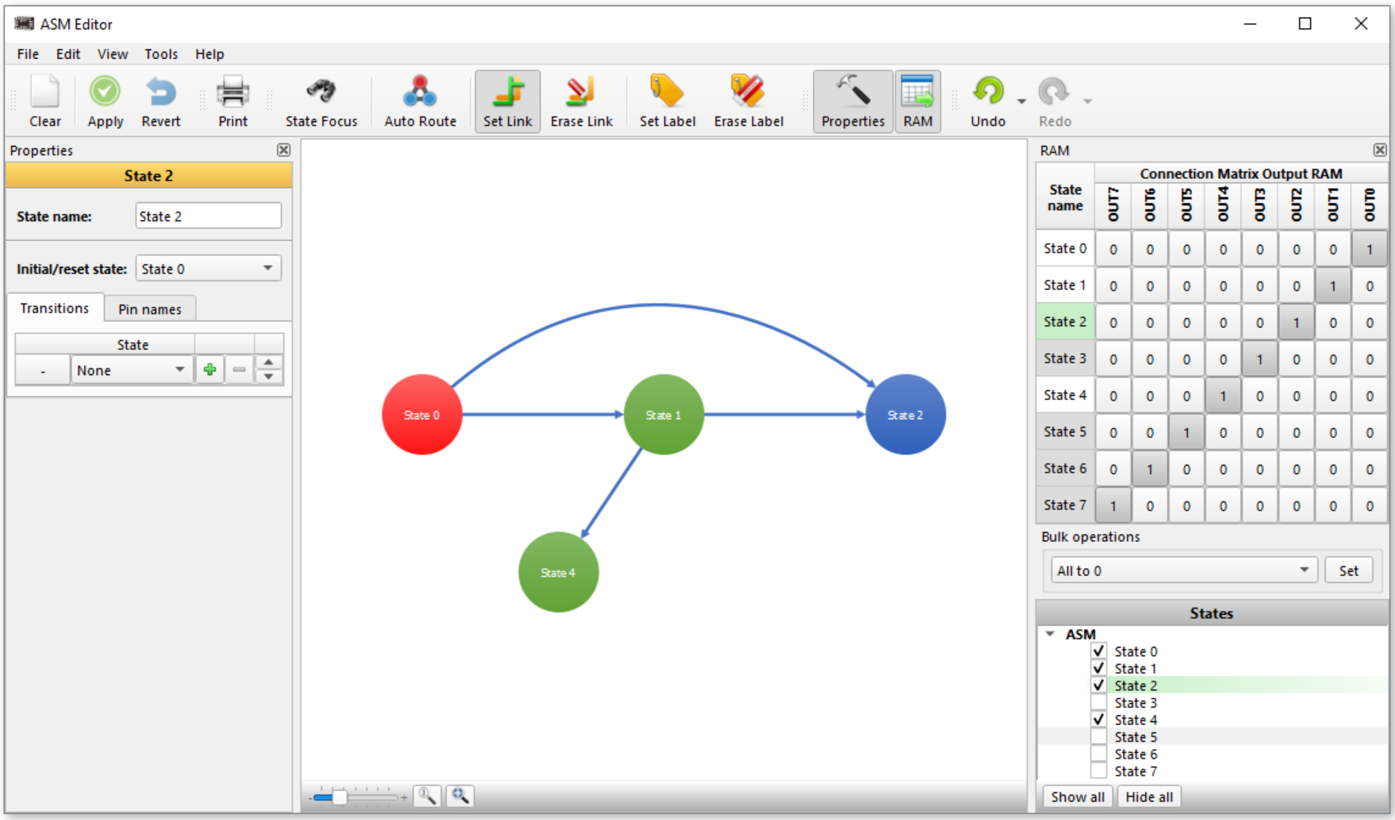


ASM Editor on the toolbar

ASM Editor's interface

You can view and set a state's properties 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.



State 2

State name: State 2

Initial/reset state: State 0

Transitions | Pin names

	State
-	None

Connection Matrix Output RAM

State name	OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
State 0	0	0	0	0	0	0	0	1
State 1	0	0	0	0	0	0	1	0
State 2	0	0	0	0	0	1	0	0
State 3	0	0	0	0	1	0	0	0
State 4	0	0	0	1	0	0	0	0
State 5	0	0	1	0	0	0	0	0
State 6	0	1	0	0	0	0	0	0
State 7	1	0	0	0	0	0	0	0

Bulk operations

All to 0 | Set

States

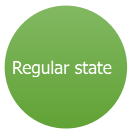
- ASM
  - ✓ State 0
  - ✓ State 1
  - ✓ State 2
  - State 3
  - ✓ State 4
  - State 5
  - State 6
  - State 7

Setting a state transition

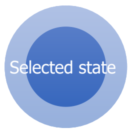
Bulk operations allow applying a selected operation (*All to 0*, *All to 1*, *Invert*, and resetting to *Default*) to the whole table.

You can show or hide states by toggling the corresponding checkboxes within the States section located in the lower right area of the *RAM* properties panel.

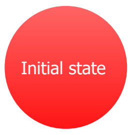
The *main area* of the ASM editor depicts states. The color of a state changes according to the state's interaction options.



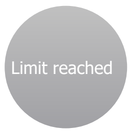
regular state



selected state



initial (reset) state



state transitions limit reached

Configuring States

If states haven't been established, the *ASM Editor* will display 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 will not produce any changes. To establish a connection click the outer ring of one state and then click on the ring of another state.

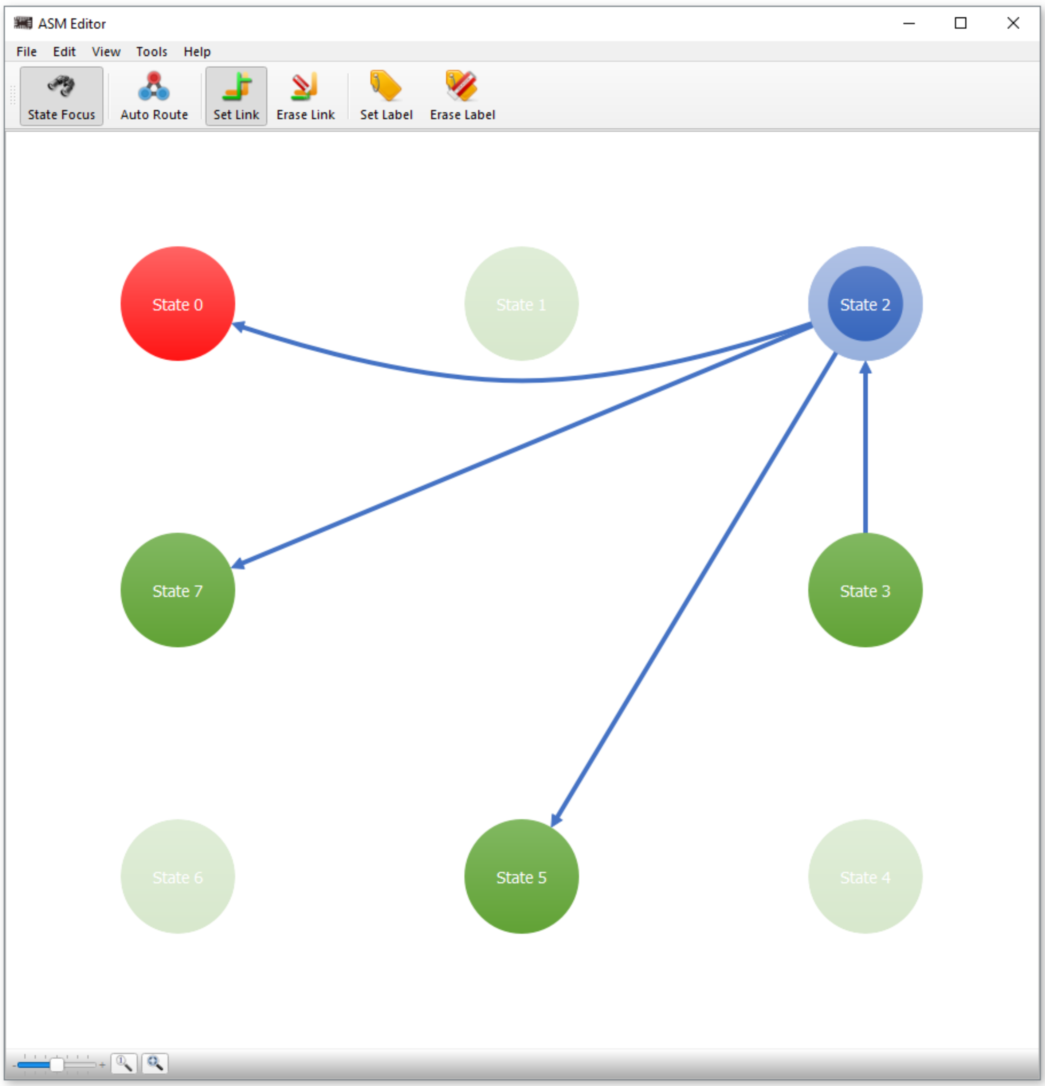
For some chips, a state may be directed to itself by selecting the state's outside ring on both the first and second click. The number of states a particular state can transition to is unrestricted; it can transition to all the utilized states or none at all.

To rename a state, double-click on 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 your scheme. Drag the link to change its shape. Select the *Edit path* mode from a connection's context menu if you need more tweaking. **Note:** A link's shape is updated automatically and the manual adjustments get discarded every time a state is repositioned.

You can let the software reorder the states and transitions by clicking *Auto Route* on the toolbar. You can also add a label to a transition. To do that, click *Set label* at the top menu, or click the mouse right button on an existing transition and select *Set label*.

You can also focus on the states by clicking the *State focus* button on the toolbar.



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

You can configure a state's properties in the *Properties* panel at the left. The available tabs are:

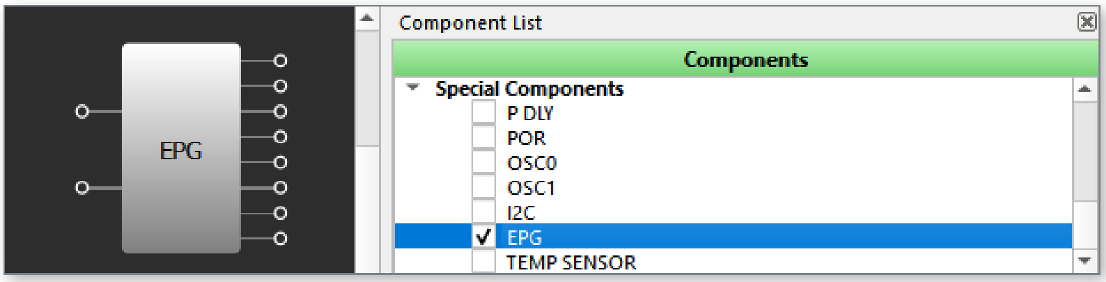
- *Transitions* — lists of the states to which the current state transitions. Allows adding a new transition by selecting another state from the list.
- *Pin names* — this tab's data is common for all states. It lists pin names set the current state as

the initial

- *RE (Rising Edge)* — lists the states the current state transitions to and apply *Rising Edge* to the transition, otherwise ASM will be level sensitive.

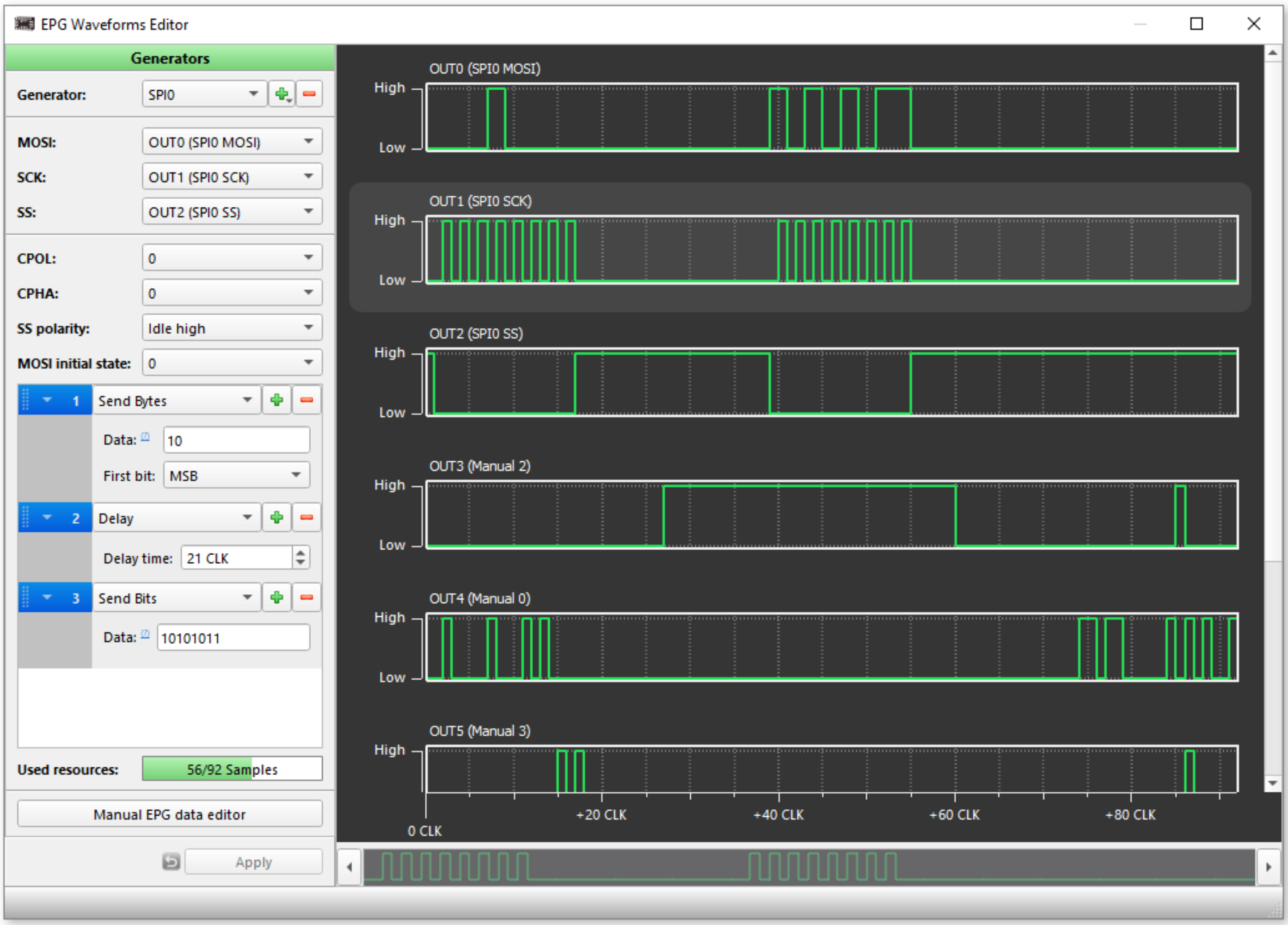
2.3.2 EPG Waveform Editor

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.



Extended Pattern Generator

You can configure the *Extended Pattern Generator* in the *EPG Waveforms Editor*. This tool allows you to choose from the available predefined generators, including *SPI*, *I2C*, *PWM*, or *Manual*, and assign the output accordingly. Each of these generators offers a range of adjustable settings, making it more convenient for you to create desired patterns.



Generator: SPI0

MOSI: OUT0 (SPI0 MOSI)
SCK: OUT1 (SPI0 SCK)
SS: OUT2 (SPI0 SS)

CPOL: 0
CPHA: 0
SS polarity: Idle high
MOSI initial state: 0

1 Send Bytes
Data: 10
First bit: MSB

2 Delay
Delay time: 21 CLK

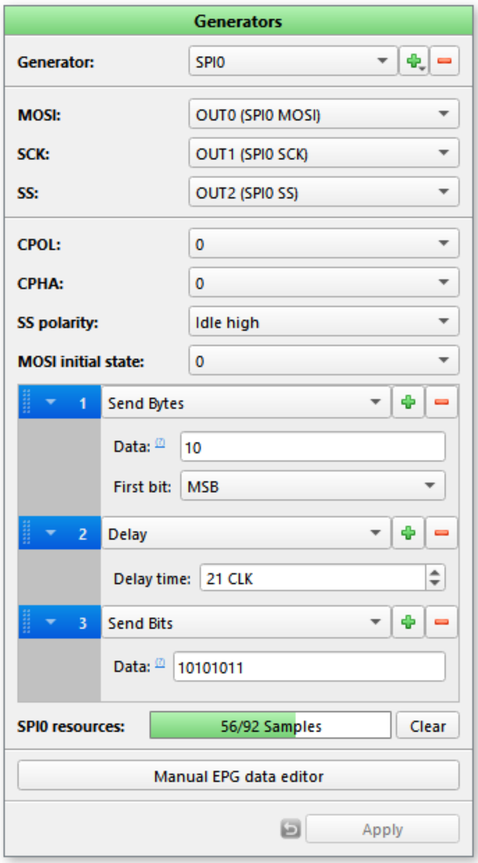
3 Send Bits
Data: 10101011

Used resources: 56/92 Samples

EPG Waveforms Editor

Within the *EPG Waveforms Editor*, you can also find a *resource meter* that visually represents the number of bits used in the pattern. This feature provides transparent view of the memory and resources allocated to the present pattern.

The *SPI* generator allows you to select and configure the output pin using the *command editor*.


Generators

Generator: SPI0

MOSI: OUT0 (SPI0 MOSI)

SCK: OUT1 (SPI0 SCK)

SS: OUT2 (SPI0 SS)

CPOL: 0

CPHA: 0

SS polarity: Idle high

MOSI initial state: 0

1 Send Bytes

Data: 10

First bit: MSB

2 Delay

Delay time: 21 CLK

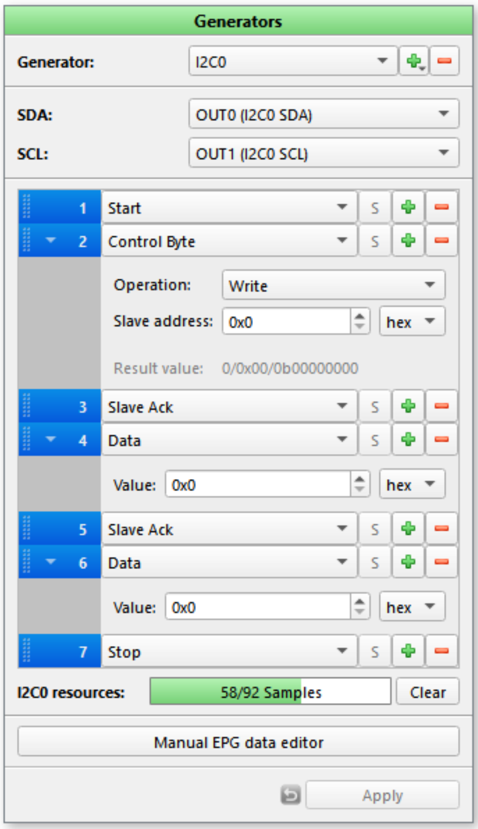
3 Send Bits

Data: 10101011

SPI0 resources: 56/92 Samples

SPI generator command editor and resource meter

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


Generators

Generator: I2C0

SDA: OUT0 (I2C0 SDA)

SCL: OUT1 (I2C0 SCL)

1 Start

2 Control Byte

Operation: Write

Slave address: 0x0 hex

Result value: 0/0x00/0b00000000

3 Slave Ack

4 Data

Value: 0x0 hex

5 Slave Ack

6 Data

Value: 0x0 hex

7 Stop

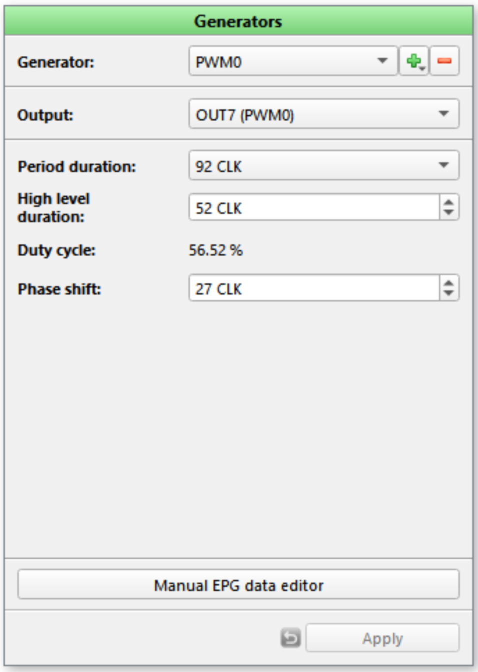
I2C0 resources: 58/92 Samples Clear

Manual EPG data editor

Apply

I2C generator command editor and resources meter

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


Generators

Generator: PWM0

Output: OUT7 (PWM0)

Period duration: 92 CLK

High level duration: 52 CLK

Duty cycle: 56.52 %

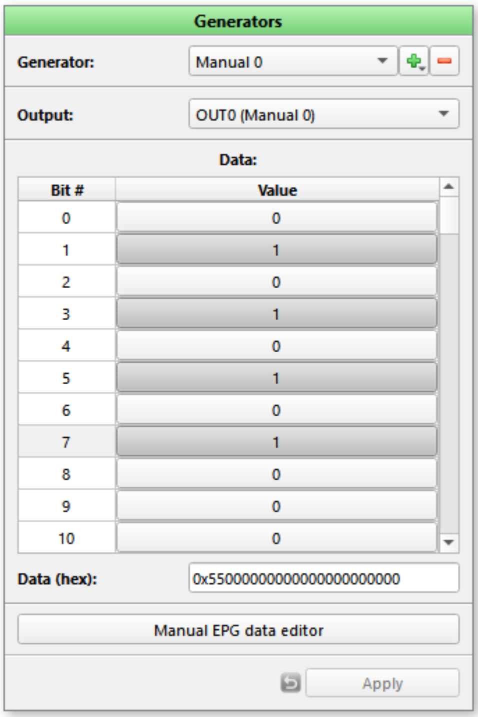
Phase shift: 27 CLK

Manual EPG data editor

Apply

PWM generator options

The *Manual* generator allows configuring the bit values.



Bit #	Value
0	0
1	1
2	0
3	1
4	0
5	1
6	0
7	1
8	0
9	0
10	0

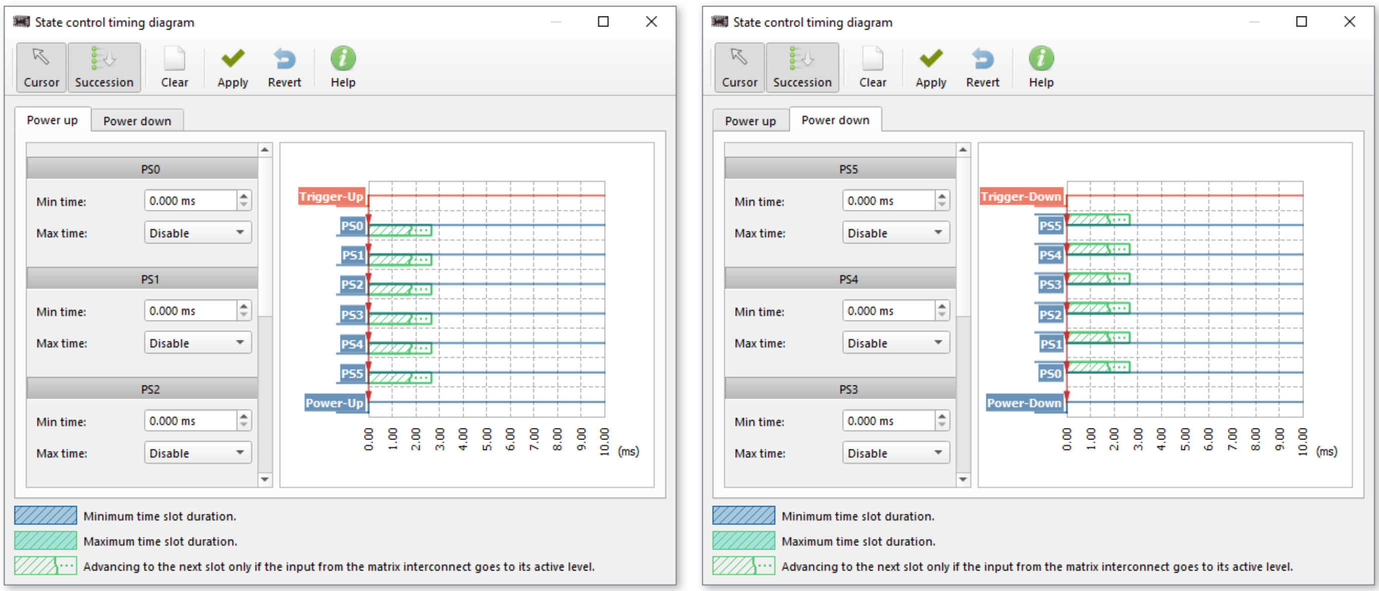
Data (hex): 0x55000000000000000000000

Manual generator options

When the system is powered up, the behavior of the *EPG* varies depending on the signal that 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 enables you to customize 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, which allows managing 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. You can set the minimum and maximum duration per each slot. The sequence is initiated with the *trigger-up* and *trigger-down* control signals from the matrix interconnect. The *Power* *sequencer* and, therefore, *Timing diagram* tool are available e.g. SLG51000/1 and SLG51002.



PS0	
Min time:	0.000 ms
Max time:	Disable
PS1	
Min time:	0.000 ms
Max time:	Disable
PS2	
Min time:	0.000 ms
Max time:	Disable

PS5	
Min time:	0.000 ms
Max time:	Disable
PS4	
Min time:	0.000 ms
Max time:	Disable
PS3	
Min time:	0.000 ms
Max time:	Disable

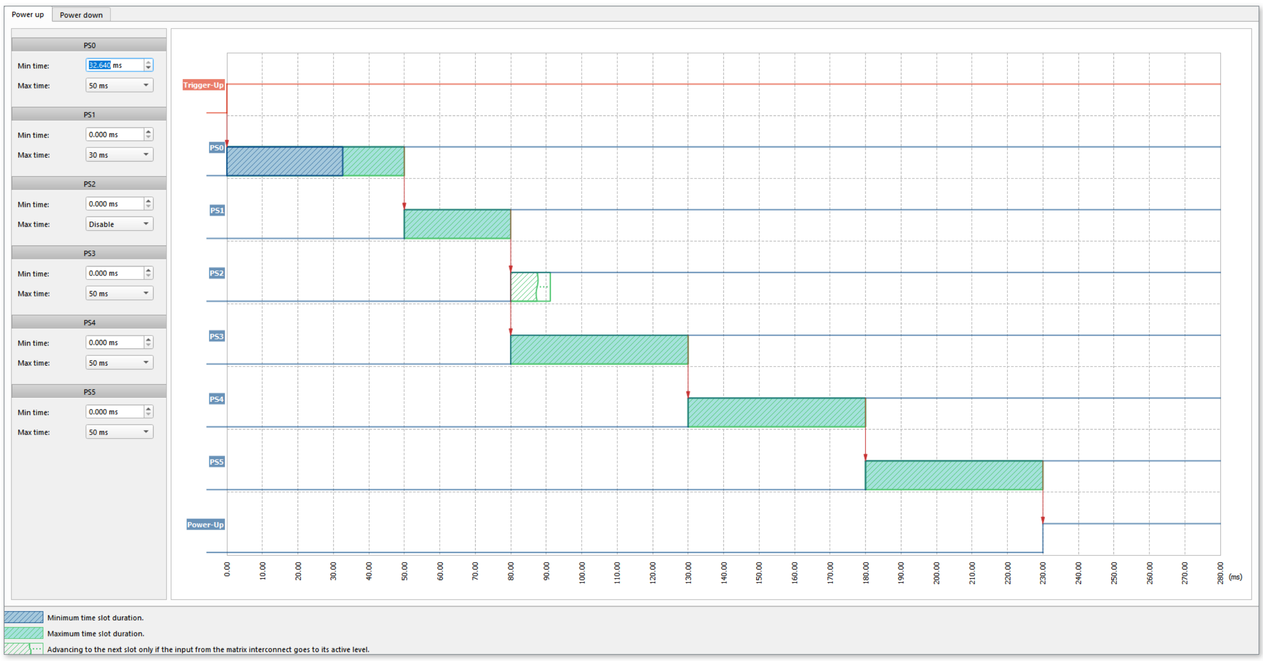
▨ Minimum time slot duration.

▨ Maximum time slot duration.

▨ Advancing to the next slot only if the input from the matrix interconnect goes to its active level.

Timing diagram

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



PS0	
Min time:	32.640 ms
Max time:	50 ms
PS1	
Min time:	0.000 ms
Max time:	30 ms
PS2	
Min time:	0.000 ms
Max time:	Disable
PS3	
Min time:	0.000 ms
Max time:	50 ms
PS4	
Min time:	0.000 ms
Max time:	50 ms
PS5	
Min time:	0.000 ms
Max time:	50 ms

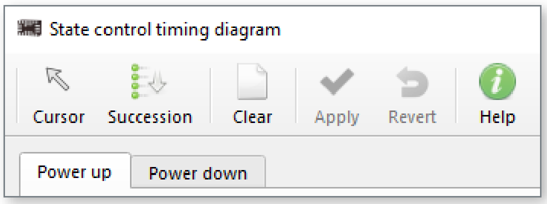
▨ Minimum time slot duration.

▨ Maximum time slot duration.

▨ Advancing to the next slot only if the input from the matrix interconnect goes to its active level.

Slot view examples

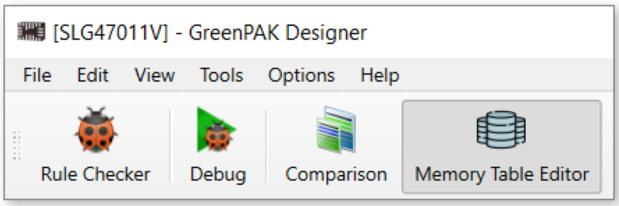
Use the toolbar controls to work with the tool.



Timing diagram options

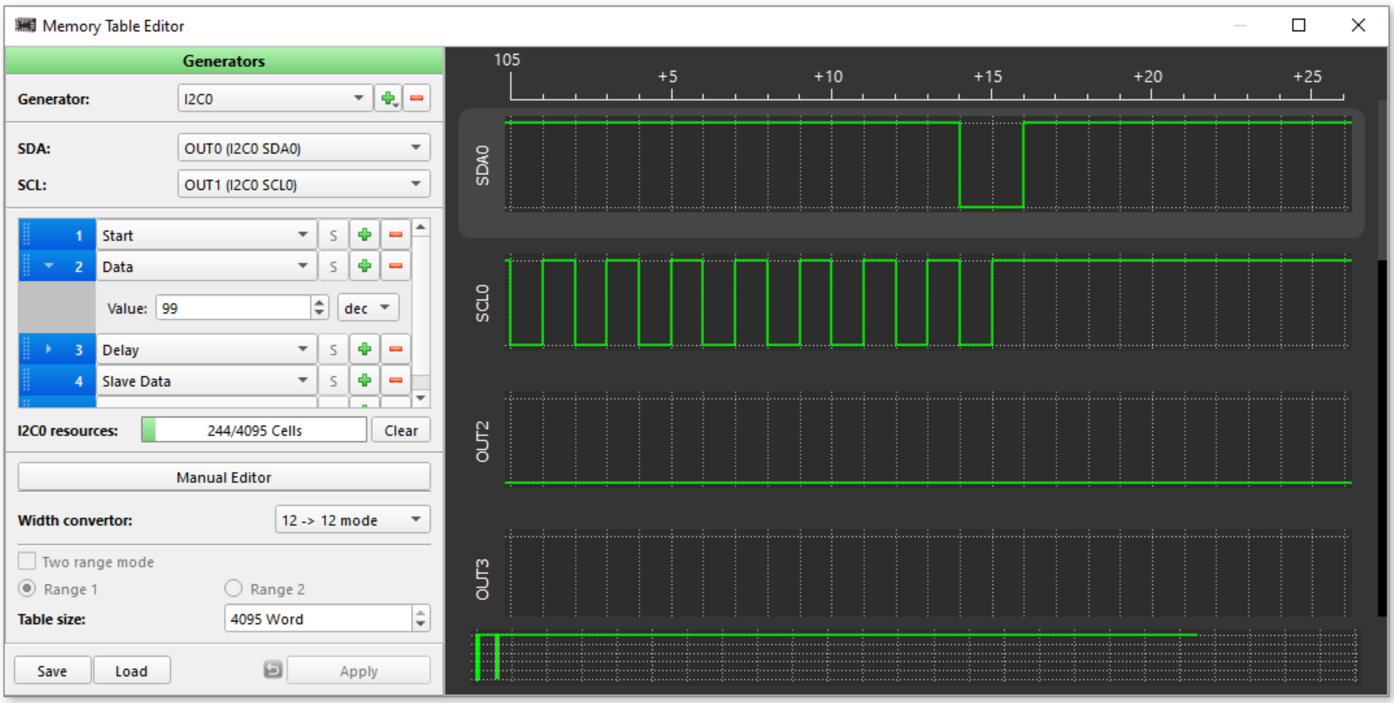
2.3.4 Memory Table Editor

Memory Table Editor allows you to configure the Memory table macrocell (make sure you select the ROM mode 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 on the work area.



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).



Memory Table Editor

Generators

Generator: I2C0

SDA: OUT0 (I2C0 SDA0)

SCL: OUT1 (I2C0 SCL0)

1	Start	S
2	Data	S
	Value: 99	dec
3	Delay	S
4	Slave Data	S

I2C0 resources: 244/4095 Cells Clear

Manual Editor

Width convertor: 12 -> 12 mode

Two range mode

Range 1 Range 2

Table size: 4095 Word

Save Load Apply

Memory Table Editor window

The tool provides you with the following possibilities:

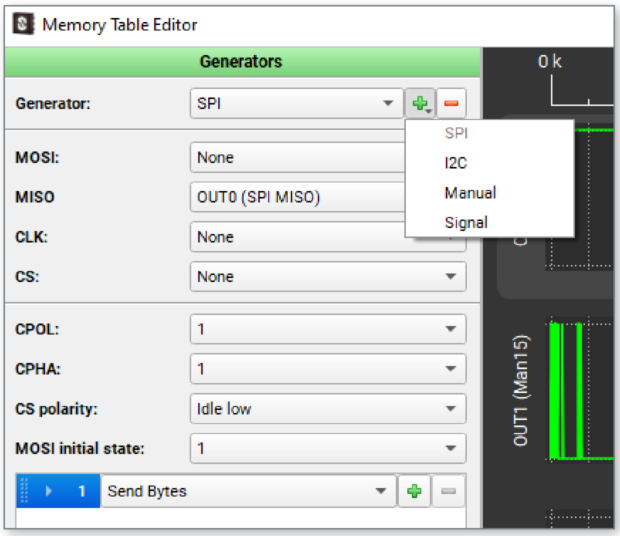
➤ Add the generator type. For *I2C* and *SPI* use the available generator-specific settings, e.g. specify which Width Converter output corresponds to which generator signal.

Upon selecting *Manual* generator the word (digit) is represented in binary format. The manual mode allows you to fill in the data through the provided table, which you can bind to the specific output.

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

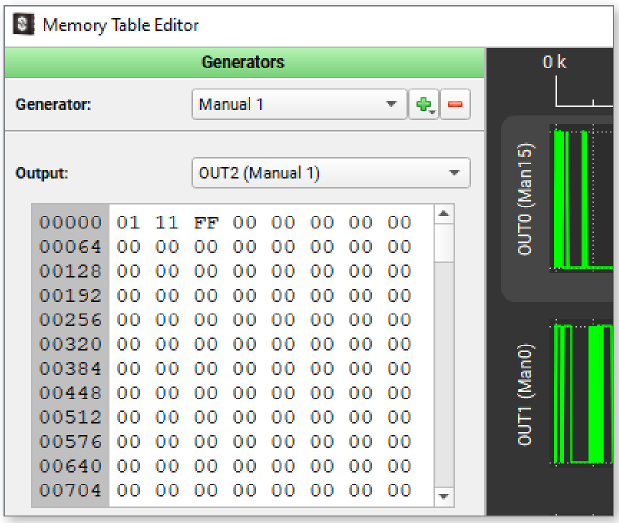
The x-axis on the graph is the word sequence number, a memory cell (*Word index*). The y-axis is a word saved in the memory cell (*Word decimal*).

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



Generator: SPI
MOSI: None
MISO OUT0 (SPI MISO)
CLK: None
CS: None
CPOL: 1
CPHA: 1
CS polarity: Idle low
MOSI initial state: 1
1 Send Bytes

SPI generator configuration



Generator: Manual 1
Output: OUT2 (Manual 1)

```
00000 01 11 FF 00 00 00 00 00
00064 00 00 00 00 00 00 00 00
00128 00 00 00 00 00 00 00 00
00192 00 00 00 00 00 00 00 00
00256 00 00 00 00 00 00 00 00
00320 00 00 00 00 00 00 00 00
00384 00 00 00 00 00 00 00 00
00448 00 00 00 00 00 00 00 00
00512 00 00 00 00 00 00 00 00
00576 00 00 00 00 00 00 00 00
00640 00 00 00 00 00 00 00 00
00704 00 00 00 00 00 00 00 00
```

Manual generator configuration

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

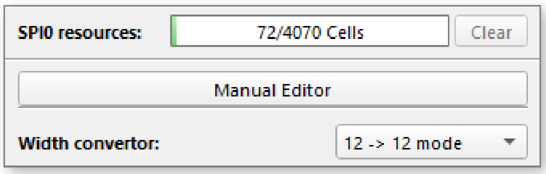
➤ Check the *Resources* bar, indicating the number of available cells. The number of cells, as well as the displayed output signals, depends on the selected *Width convertor* mode, which are:

– 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)



SPI0 resources: 72/4070 Cells

Width convertor: 12 -> 12 mode

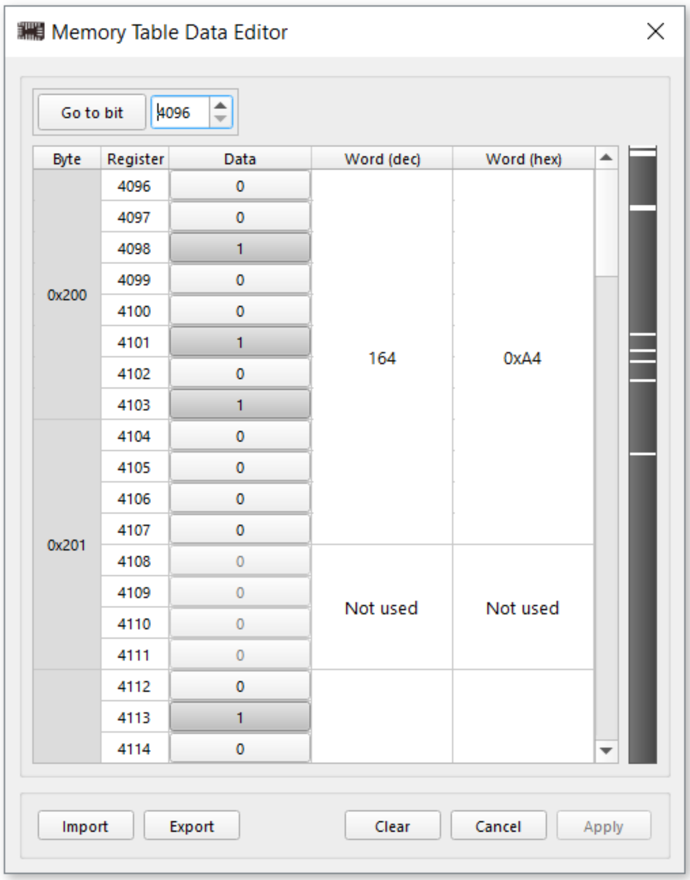
Resources/Width converter mode

➤ *Manual Editor* allows you to modify the macrocell's bit values of the specific registers, and consists of three main parts:

– Navigation controls — enter the bit number and click *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



Byte	Register	Data	Word (dec)	Word (hex)
0x200	4096	0	164	0xA4
	4097	0		
	4098	1		
	4099	0		
	4100	0		
	4101	1		
	4102	0		
	4103	1		
0x201	4104	0		
	4105	0		
	4106	0		
	4107	0		
	4108	0	Not used	Not used
	4109	0		
	4110	0		
	4111	0		
	4112	0		
	4113	1		
	4114	0		

Memory Table Data Editor

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

➤ The additional settings provide flexibility in table usage defining the 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

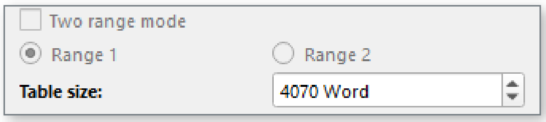
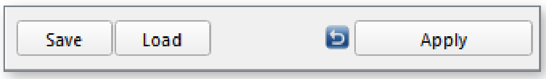


Table size: 4070 Word

Additional settings

➤ Save/Load the file with configurations using the corresponding bottom controls



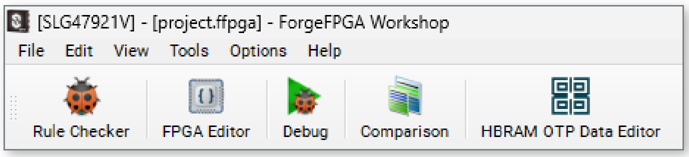
Save/Load controls

See some additional plot controls:

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

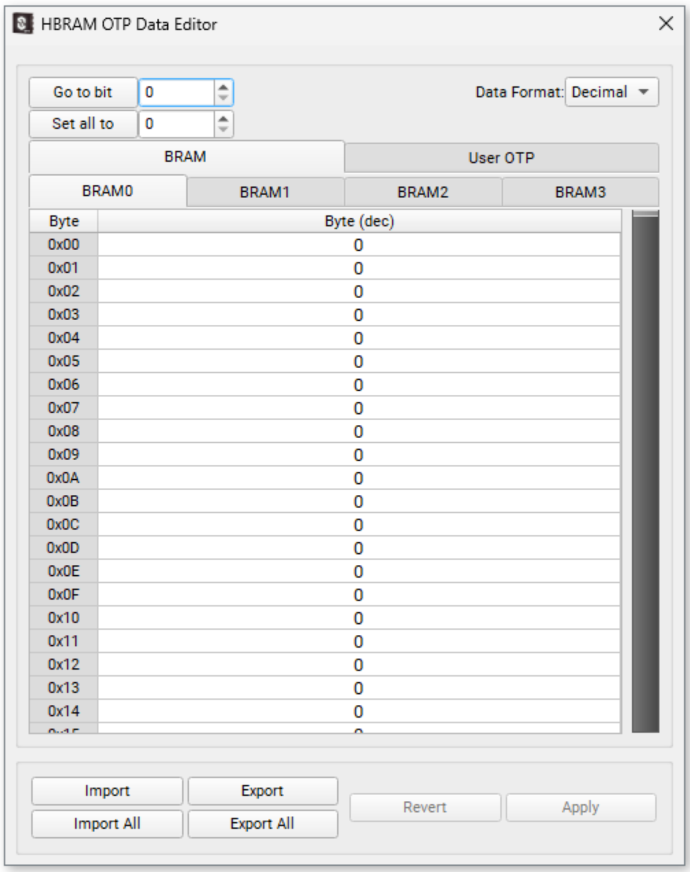
2.3.5 HBRAM OTP Data Editor

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


[SLG47921V] - [project.ffpga] - ForgeFPGA Workshop
File Edit View Tools Options Help
Rule Checker
FPGA Editor
Debug
Comparison
HBRAM OTP Data Editor

HBRAM OTP Data Editor on the toolbar

The tool provides separate tables for configuring BRAM and User OTP macrocell bits. Choose whether to represent the data in decimal or hexadecimal format.



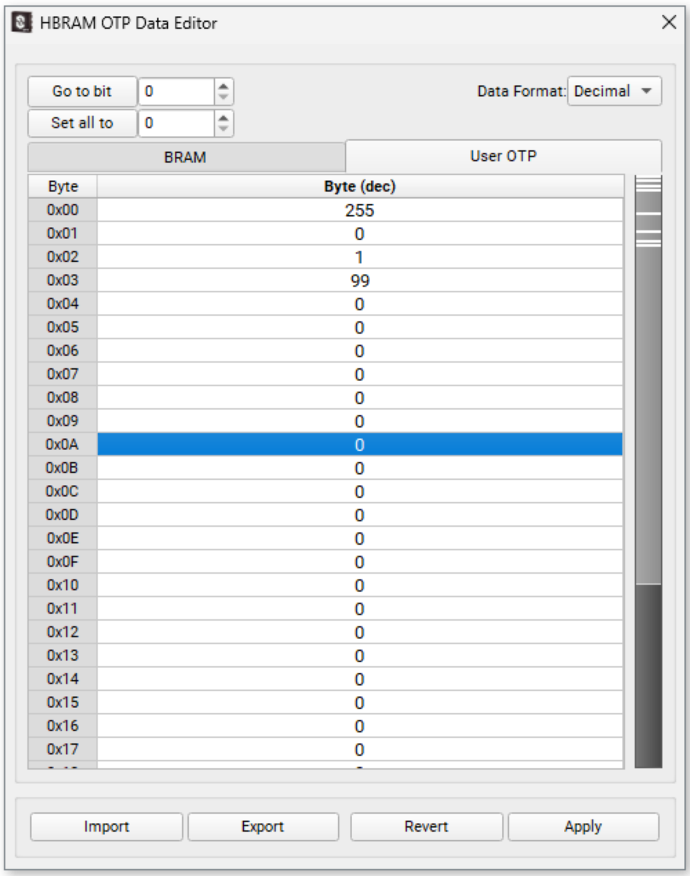
Go to bit: 0

Set all to: 0

Data Format: Decimal

Byte	Byte (dec)
0x00	0
0x01	0
0x02	0
0x03	0
0x04	0
0x05	0
0x06	0
0x07	0
0x08	0
0x09	0
0x0A	0
0x0B	0
0x0C	0
0x0D	0
0x0E	0
0x0F	0
0x10	0
0x11	0
0x12	0
0x13	0
0x14	0

HBRAM OTP Data Editor (BRAM tab)



HBRAM OTP Data Editor

Go to bit: 0

Set all to: 0

Data Format: Decimal

Byte	Byte (dec)
0x00	255
0x01	0
0x02	1
0x03	99
0x04	0
0x05	0
0x06	0
0x07	0
0x08	0
0x09	0
0x0A	0
0x0B	0
0x0C	0
0x0D	0
0x0E	0
0x0F	0
0x10	0
0x11	0
0x12	0
0x13	0
0x14	0
0x15	0
0x16	0
0x17	0

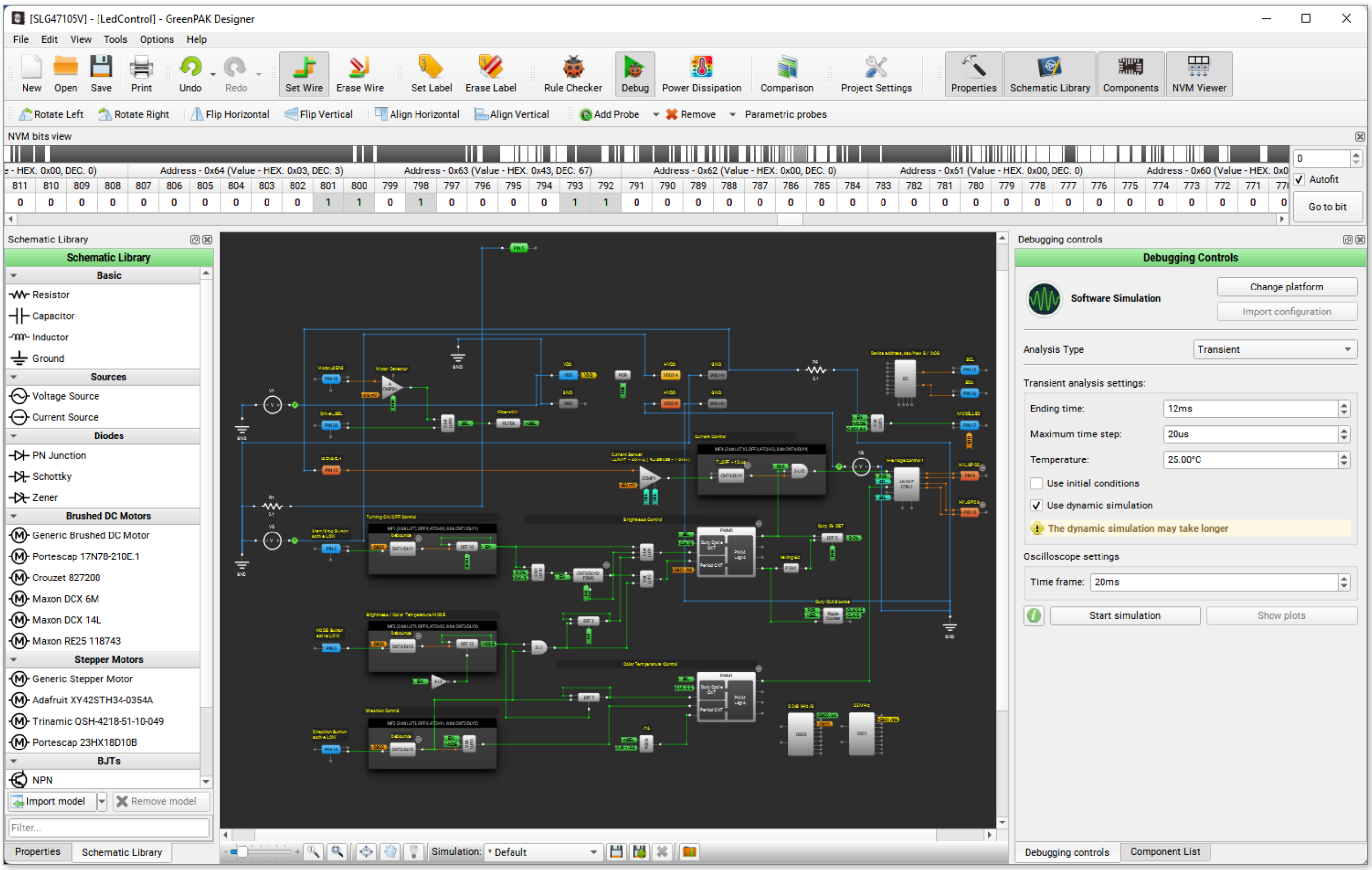
HBRAM OTP Data Editor (User OTP tab)

Use the upper controls to navigate through the table with the *Go to bit* button. You can also set a specified value for all bits in the active tab using *Set all to*.

The bottom controls provide an option to import or export the data either from the active tab using the *Import/Export* buttons or from all tabs with *Import all/Export all* buttons (the latter options are applicable for BRAM only).

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 in the *Development Platform Selector*. Refer to the *Hub* window → *Development* tab → *Details* to check the simulation support availability for a certain Part Number.



Analysis Type: Transient

Transient analysis settings:

Ending time: 12ms

Maximum time step: 20us

Temperature: 25.00°C

Use initial conditions

✓ Use dynamic simulation

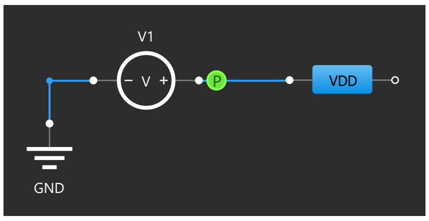
The dynamic simulation may take longer

Oscilloscope settings

Time frame: 20ms

Software Simulation tools

Before starting an analysis, you can use the external components from *Schematic Library* and add probes to configure the simulation environment parameters.

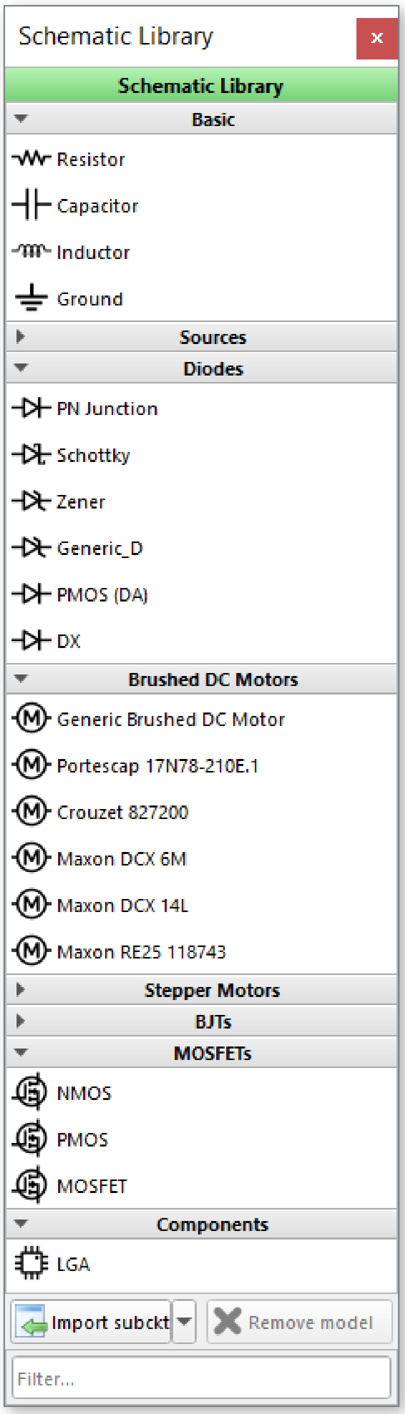


Basic Simulation Tools

2.4.1 Schematic Library

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

You can also add a custom component to the *Schematic Library* list by using the *Import model/subcircuit* feature. See the corresponding buttons on the *Schematic Library* panel (read more in section 2.4.3 Working with models and subcircuits). Find imported models and subcircuits in the *Schematic Library* in the designated location.

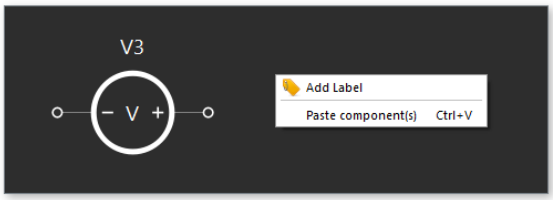


Schematic Library

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

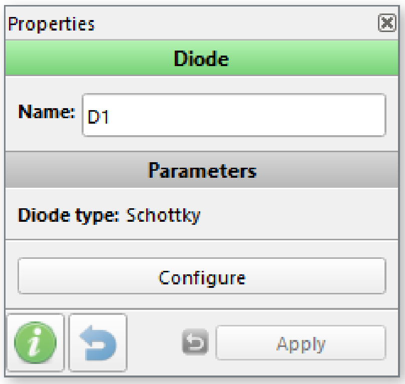
2.4.2 External components

See the list of external components on the Schematic Library panel. Click the component and then the work area to add it (discard adding with a right-click). Once added, find the *Copy component(s)*/*Paste component(s)* options in the desired component's/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.



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.


Properties

Diode

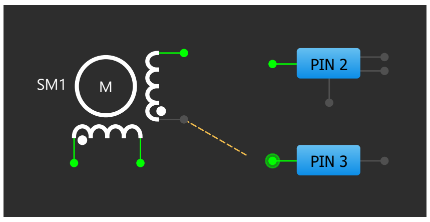
Name: D1

Parameters

Diode type: Schottky

Configuring basic parameters

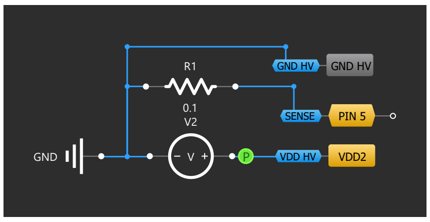
Connect the external components to the external I/O macrocell pins or to the other external component's pins using the *Set Wire* tool. Upon setting a connection, the available pins become highlighted. By default, voltage source(s) and GND are added to specific pins.



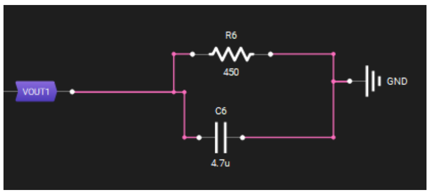
Adding wire to an external component

Here, you can see the external connection type (all connection types are described in the section

2.1.5 Connections).



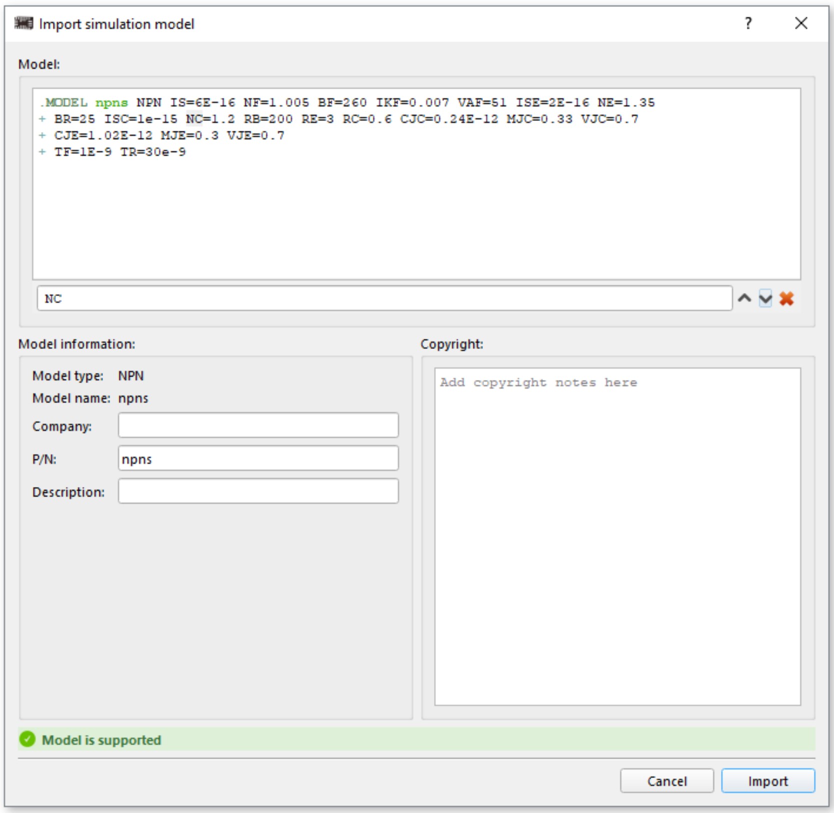
External connections



External connections in SLG51001

2.4.3 Working with models and subcircuits

You can import third-party *SPICE* models and subcircuits to the project. *Import model/subcircuit* opens the *Import* window, where you can insert a *SPICE* simulation model/subcircuit, fill in additional information, and add it to the library. The dialog validates a *SPICE* syntax of a model/subcircuit. Only one model can be imported at a time.



Model:

```
.MODEL npns NPN IS=6E-16 NF=1.005 BF=260 IKF=0.007 VAF=51 ISE=2E-16 NE=1.35
+ BR=25 ISC=1e-15 NC=1.2 RB=200 RE=3 RC=0.6 CJC=0.24E-12 MJC=0.33 VJC=0.7
+ CJE=1.02E-12 MJE=0.3 VJE=0.7
+ TF=1E-9 TR=30e-9
```

NC

Model information:

Model type:	NPN
Model name:	npns
Company:	
P/N:	npns
Description:	

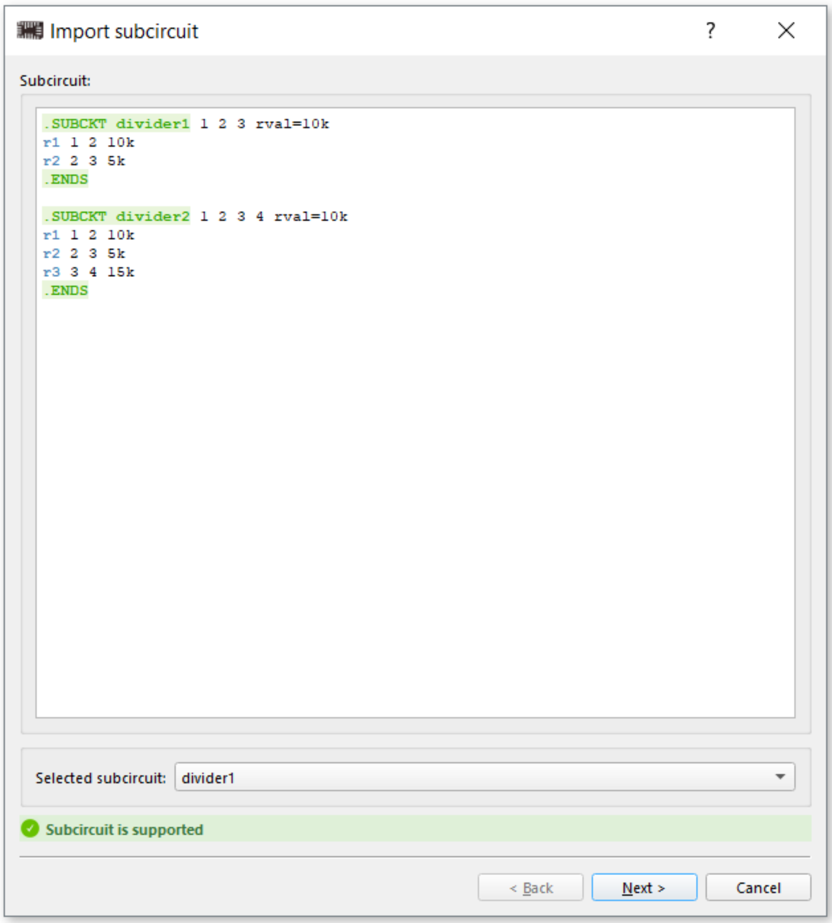
Copyright:

Add copyright notes here

Model is supported

Import simulation model window

Unlike models, you can import multiple subcircuits and select one of them as main:



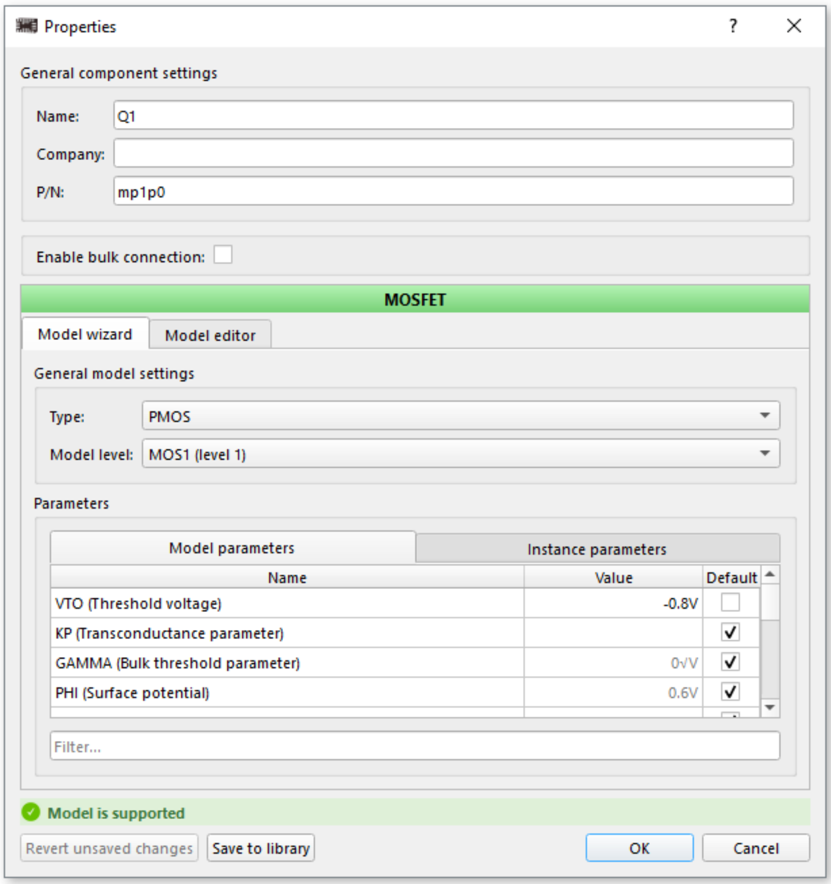
```
.SUBCKT divider1 1 2 3 rval=10k
r1 1 2 10k
r2 2 3 5k
.ENDS

.SUBCKT divider2 1 2 3 4 rval=10k
r1 1 2 10k
r2 2 3 5k
r3 3 4 15k
.ENDS
```

Selected subcircuit: divider1

Subcircuit is supported

Importing a model containing multiple subcircuits

Model/Subcircuit parameters can be edited via the *Properties* dialog window. Double-click a component in the work area or click *Configure* in the *Properties* panel. The *Properties* window allows configuring the external component's parameters and editing the model/subcircuit.



Properties

General component settings

Name: Q1

Company:

P/N: mp1p0

Enable bulk connection: ☐

MOSFET

Model wizard | Model editor

General model settings

Type: PMOS

Model level: MOS1 (level 1)

Parameters

Model parameters | Instance parameters

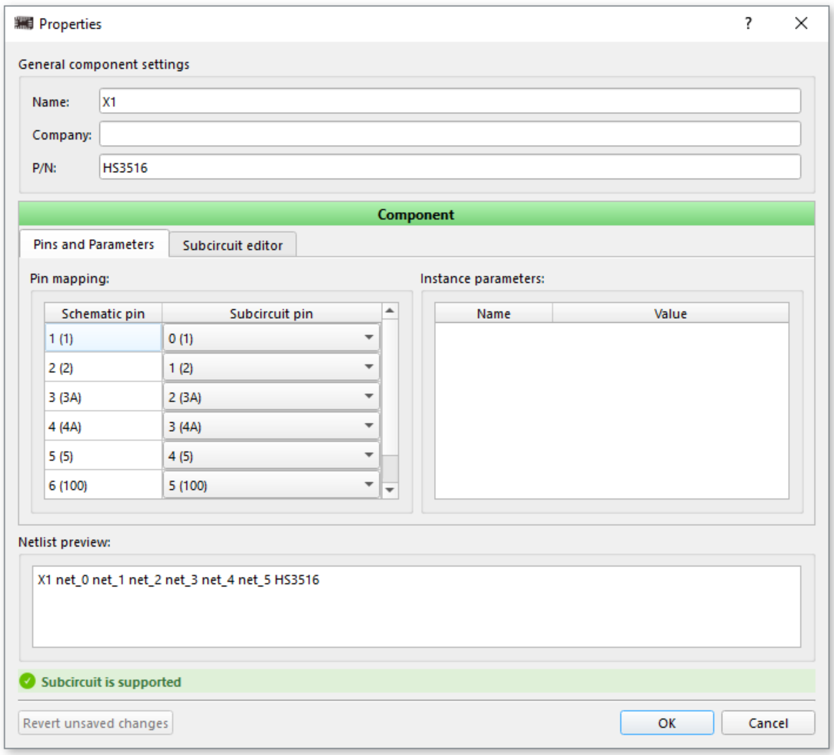
Name	Value	Default
VTO (Threshold voltage)	-0.8V	☐
KP (Transconductance parameter)		✓
GAMMA (Bulk threshold parameter)	0√V	✓
PHI (Surface potential)	0.6V	✓

Filter...

Model is supported

Revert unsaved changes | Save to library | OK | Cancel

Model properties window



Properties

General component settings

Name: X1

Company:

P/N: HS3516

Component

Pins and Parameters | Subcircuit editor

Pin mapping:

Schematic pin	Subcircuit pin
1 (1)	0 (1)
2 (2)	1 (2)
3 (3A)	2 (3A)
4 (4A)	3 (4A)
5 (5)	4 (5)
6 (100)	5 (100)

Instance parameters:

Name	Value

Netlist preview:

```
X1 net_0 net_1 net_2 net_3 net_4 net_5 HS3516
```

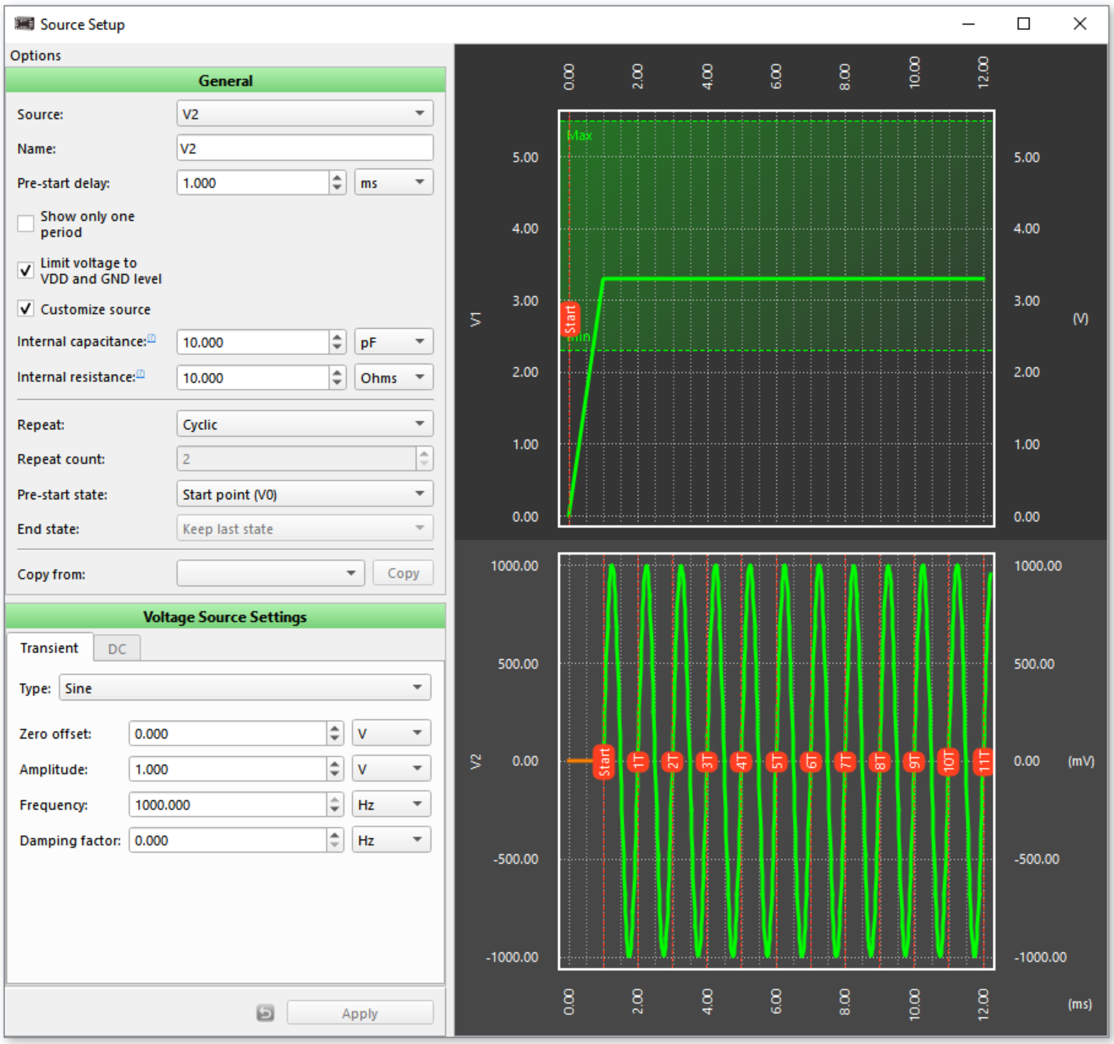
Subcircuit is supported

Revert unsaved changes | OK | Cancel

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.


Source Setup

Options

General

Source:	V2
Name:	V2
Pre-start delay:	1.000 ms
☐ Show only one period	
✓ Limit voltage to VDD and GND level	
✓ Customize source	
Internal capacitance:	10.000 pF
Internal resistance:	10.000 Ohms
Repeat:	Cyclic
Repeat count:	2
Pre-start state:	Start point (V0)
End state:	Keep last state
Copy from:	

Voltage Source Settings

Transient | DC

Type:	Sine
Zero offset:	0.000 V
Amplitude:	1.000 V
Frequency:	1000.000 Hz
Damping factor:	0.000 Hz

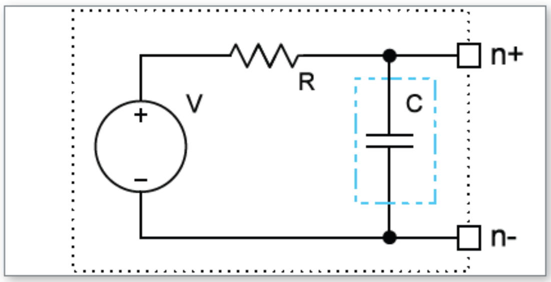
Source Setup Window

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

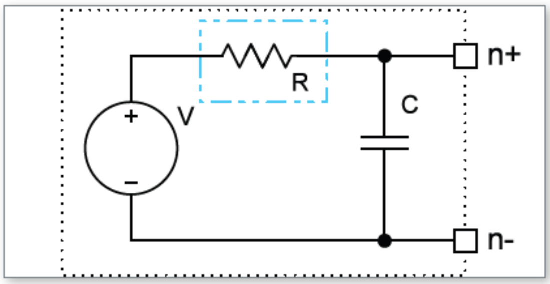
You can use the *Customize source* property to activate two additional parameters:

➤ *Internal capacitance* — the capacitance between the voltage source terminal and the ground.

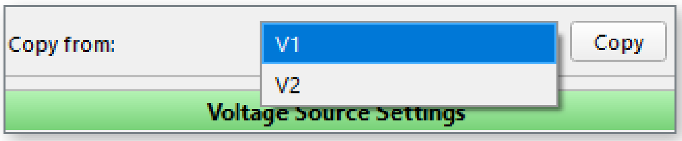
This value can be used to model a VDD bypass capacitor or IC pin parasitic capacitance



➤ *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 you need to duplicate the configured settings from one source to another, use the *Copy from* property.

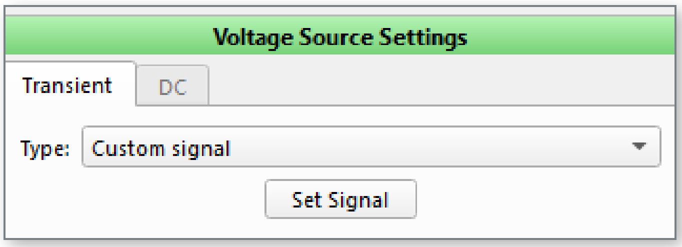


Copy from property

Custom Signal Wizard

In addition to configuring the specific waveform types (e.g., DC, sine, trapeze, etc.), it is also possible to set a custom waveform shape. *Custom Signal Wizard* allows creating, importing, and editing the signal for the simulation voltage or current sources. The signal may be constructed by manually adding points or importing a custom list of points from an external source.

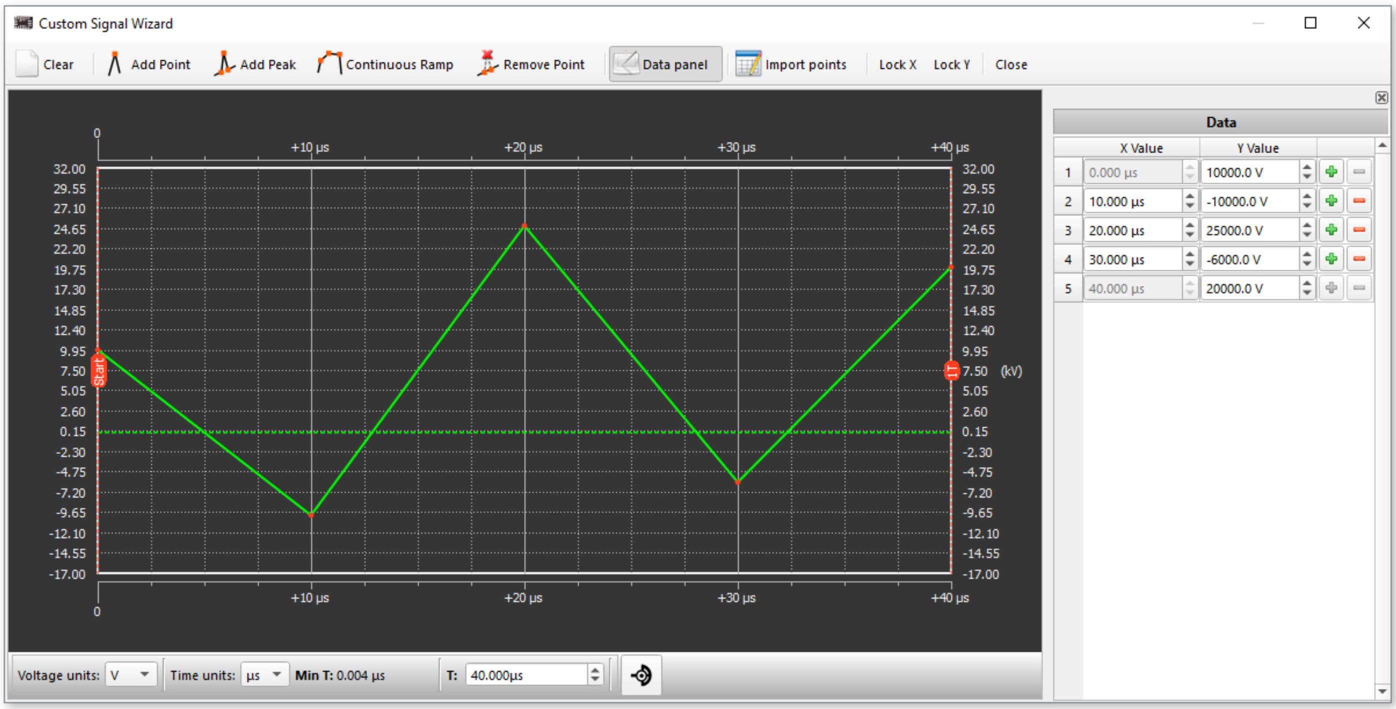
To open the *Custom Signal Wizard*, select *Custom signal* in the *Type* dropdown and click *Set Signal*.



Type: Custom signal

Custom waveform option

The examples of possible signals are shown in the pictures below.

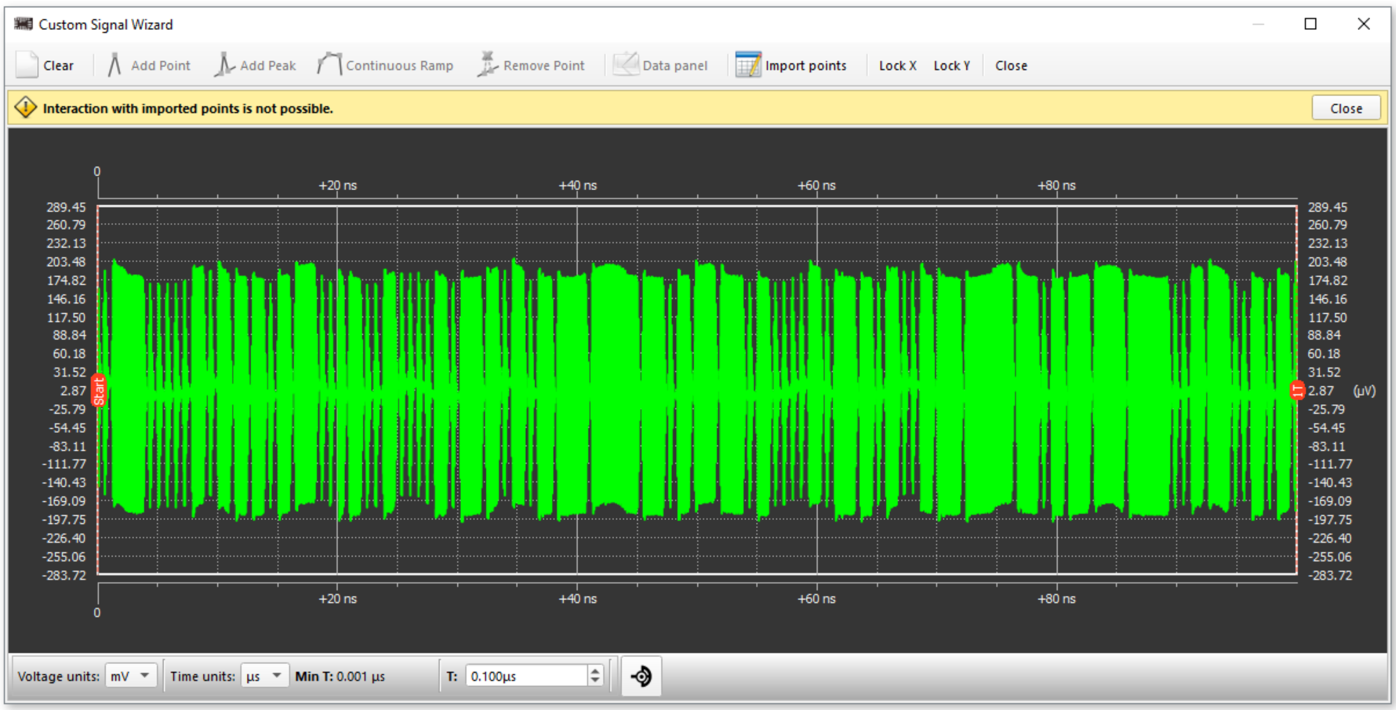


Voltage units: V Time units: µs Min T: 0.004 µs T: 40.000µs

	X Value	Y Value
1	0.000 µs	10000.0 V
2	10.000 µs	-10000.0 V
3	20.000 µs	25000.0 V
4	30.000 µs	-6000.0 V
5	40.000 µs	20000.0 V

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 the simplified signal is shown. See the example below.



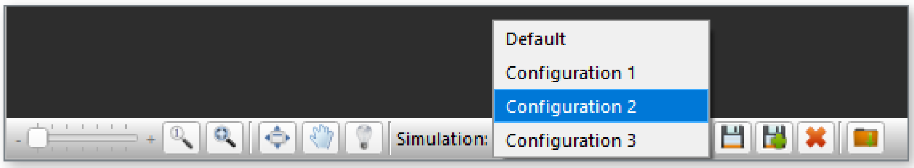
Interaction with imported points is not possible.

Min T: 0.001 µs

An imported waveform with a large number of points (above 1000)

2.4.5 Simulation configurations

You can save the snapshots with the simulation configuration state and switch between them. The snapshot stores only the elements that belong to simulation. An asterisk next to its name indicates the modified state of the snapshot. You can also import the saved configuration state from another project for the same Part Number.



Simulation configuration

💾	Save the current configuration state
💾	Save a new configuration state
✖	Delete the selected configuration
📂	Import new configuration
Default Configuration 1 Configuration 2 Configuration 3	The list of saved configurations

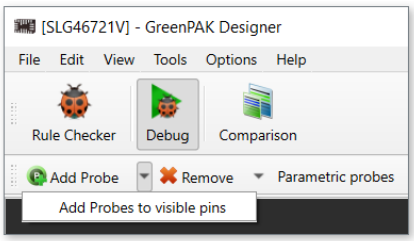
2.4.6 Probes

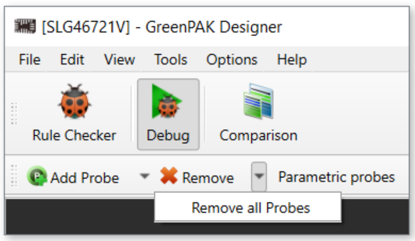
The probes provide you with the data that components yield during the simulation. You can attach a probe to a pin to reflect the component parameters. The active probes remain on the hidden components, and their data will be displayed in the simulation results.

Two types of probes are available, namely voltage and parametric probes.

Voltage probes

The voltage probe is used to capture an output signal from a pin. You can add a probe to the external component terminals and macrocell output pins. Click the respective buttons on the toolbar and then the pin to add or remove a probe. It is possible to add the probes to all available pins or remove all of them. In one click (see the arrow next to the add/remove button). You can also delete the probe from the context menu of a probe icon.

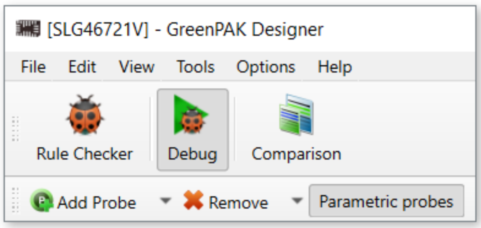




Add/Remove voltage probes

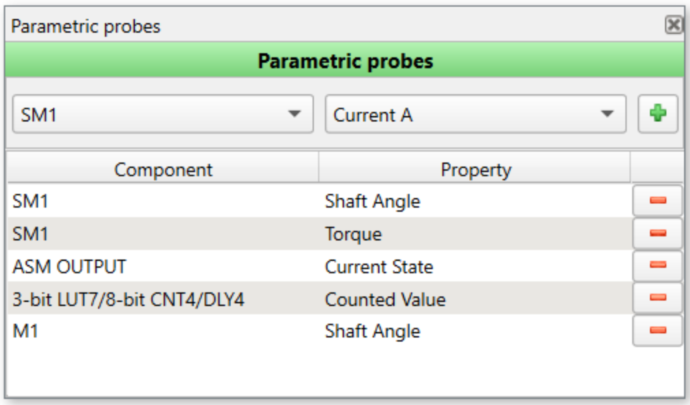
Parametric probes

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



Active parametric probes button

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


Parametric probes

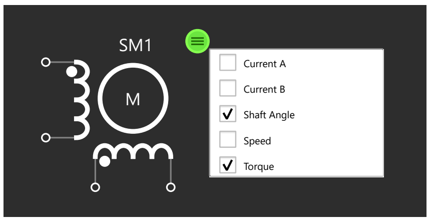
Parametric probes

SM1 | Current A

Component	Property
SM1	Shaft Angle
SM1	Torque
ASM OUTPUT	Current State
3-bit LUT7/8-bit CNT4/DLY4	Counted Value
M1	Shaft Angle

Parametric probes window

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



Parametric probe sticker

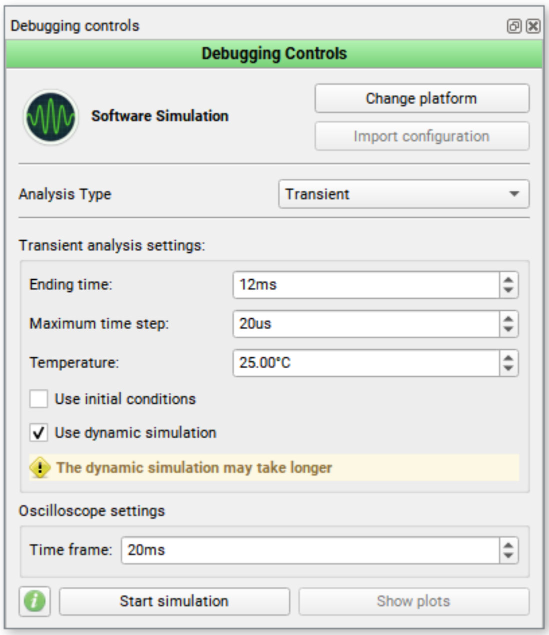
2.4.7 Debugging Controls

The *Debugging Controls* panel appears once *Software Simulation* is selected in *Development Platform Selector*.

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

Transient analysis

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


Debugging controls

Debugging Controls

Software Simulation

Analysis Type: Transient

Transient analysis settings:

Ending time: 12ms

Maximum time step: 20us

Temperature: 25.00°C

☐ Use initial conditions

☑ Use dynamic simulation

The dynamic simulation may take longer

Oscilloscope settings

Time frame: 20ms

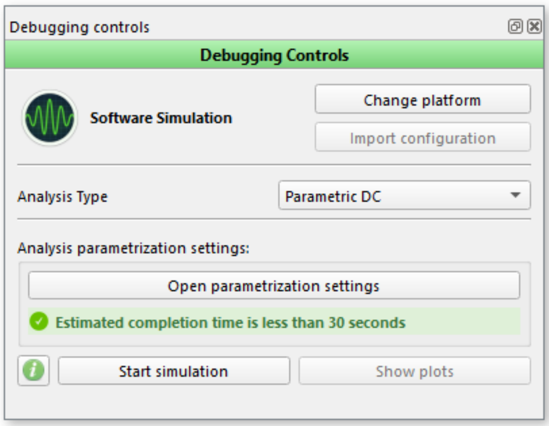
Transient Analysis Debugging controls

Use initial conditions is an optional checkbox that allows setting an initial charge different from the default value (0) for components like a capacitor or inductor. Also, this checkbox enables setting all node voltages in a circuit 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*).

Parametric DC Analysis

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



Analysis Type: Parametric DC

Analysis parametrization settings:

Estimated completion time is less than 30 seconds

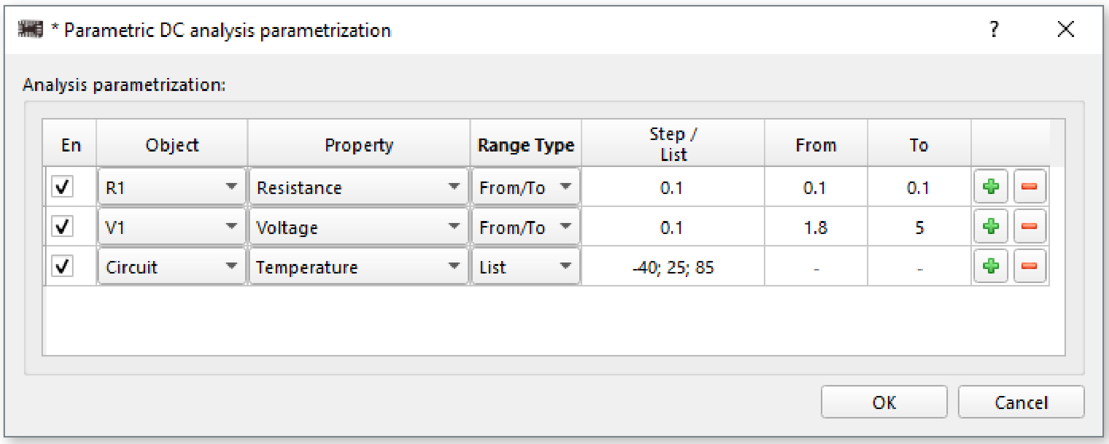
Parametric DC Analysis Debugging controls

Analysis parametrization can set active parameters for external components, circuit temperature, and macrocell properties (e.g., *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



Analysis parametrization:

En	Object	Property	Range Type	Step / List	From	To
✓	R1	Resistance	From/To	0.1	0.1	0.1
✓	V1	Voltage	From/To	0.1	1.8	5
✓	Circuit	Temperature	List	-40; 25; 85	-	-

Parametric DC Analysis Window

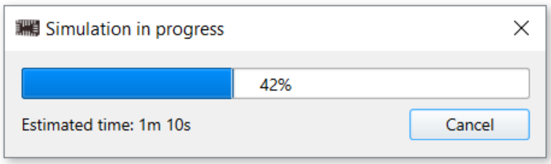
Estimated completion time

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

The software estimates the runtime based on sample points in three broad categories: *green*, *yellow*, and *red*. The estimated runtime may vary depending on different factors (e.g., the computer's performance).

2.4.8 Simulation progress

The *Simulation in progress* window appears while processing data. Simulation time remaining is estimated dynamically.

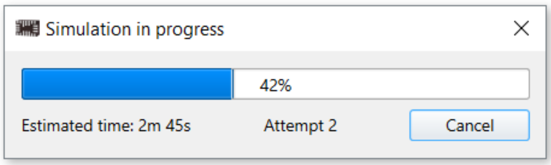


Simulation in progress

You can interrupt the process by clicking *Cancel*. The simulation results will still be displayed based on data that the tool managed to process before it was canceled.

Scheme issue resolution

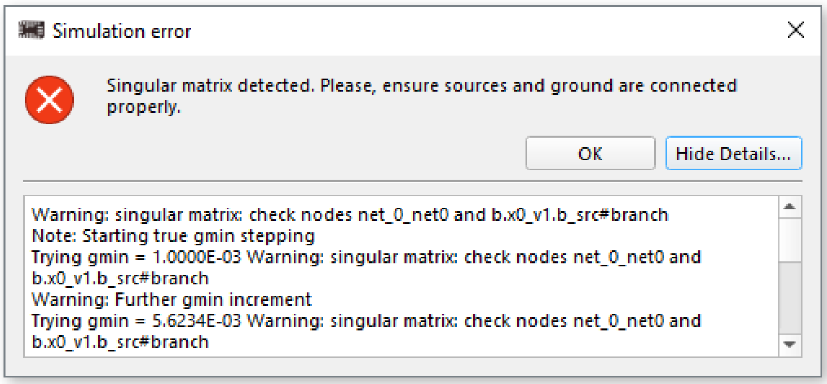
If a simulation attempt encounters a known simulation error of the scheme convergence, the simulation internal algorithm attempts to adjust the circuit by changing the component properties or scheme parameters and re-init the simulation. In this case, the *Simulation in progress* window indicates the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



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.

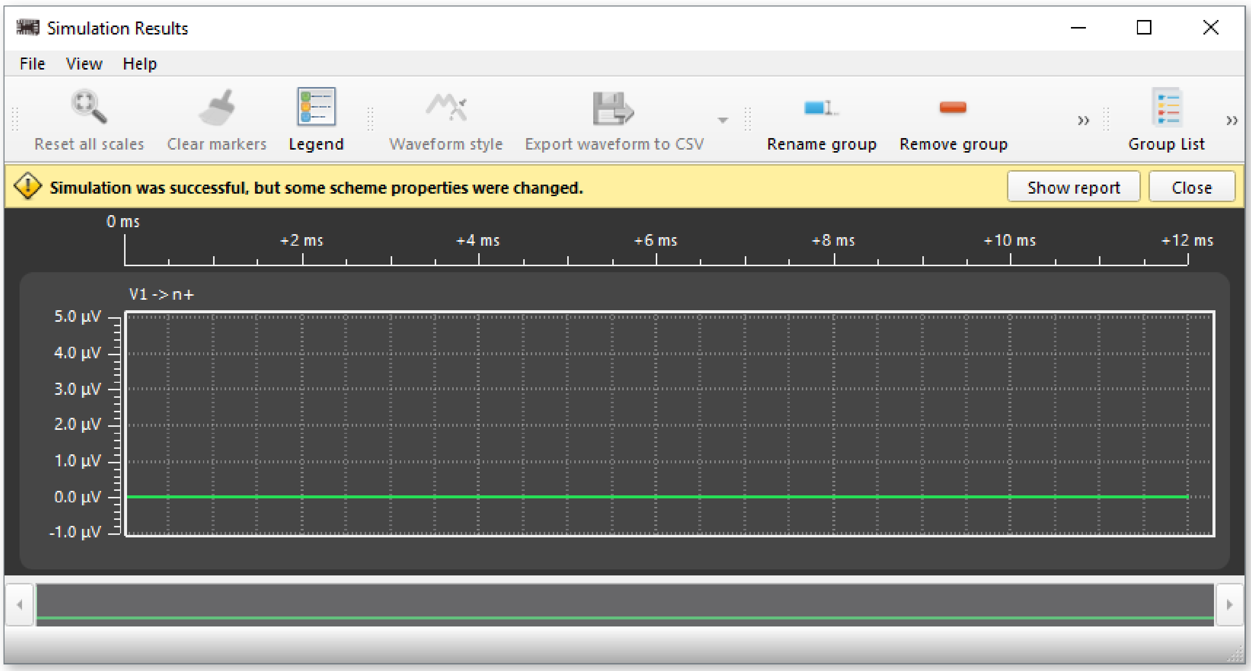


Singular matrix detected. Please, ensure sources and ground are connected properly.

```
Warning: singular matrix: check nodes net_0_net0 and b.x0_v1.b_src#branch
Note: Starting true gmin stepping
Trying gmin = 1.0000E-03 Warning: singular matrix: check nodes net_0_net0 and
b.x0_v1.b_src#branch
Warning: Further gmin increment
Trying gmin = 5.6234E-03 Warning: singular matrix: check nodes net_0_net0 and
b.x0_v1.b_src#branch
```

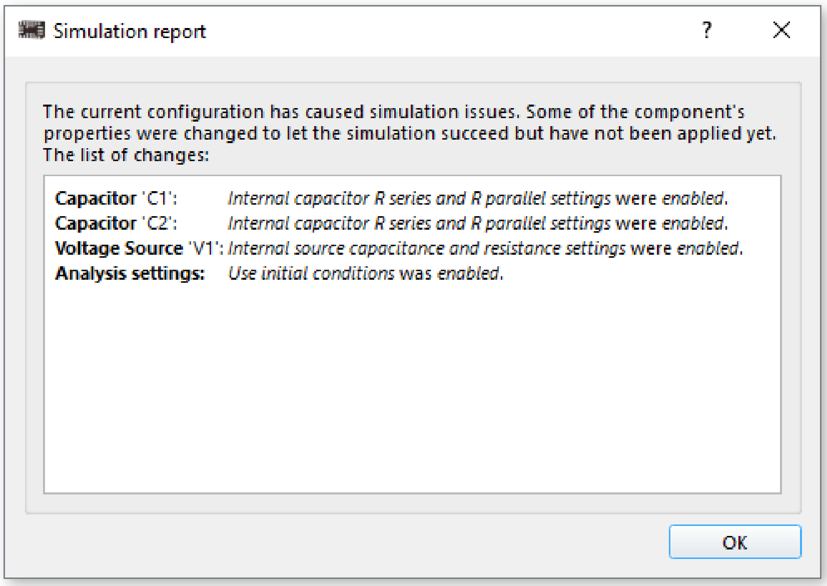
Simulation error window

After the issues are resolved and the simulation process is completed, you can observe the simulation report by clicking the *Show report* button on the *Simulation Results* window.



Simulation was successful, but some scheme properties were changed.

Show report button



The current configuration has caused simulation issues. Some of the component's properties were changed to let the simulation succeed but have not been applied yet. The list of changes:

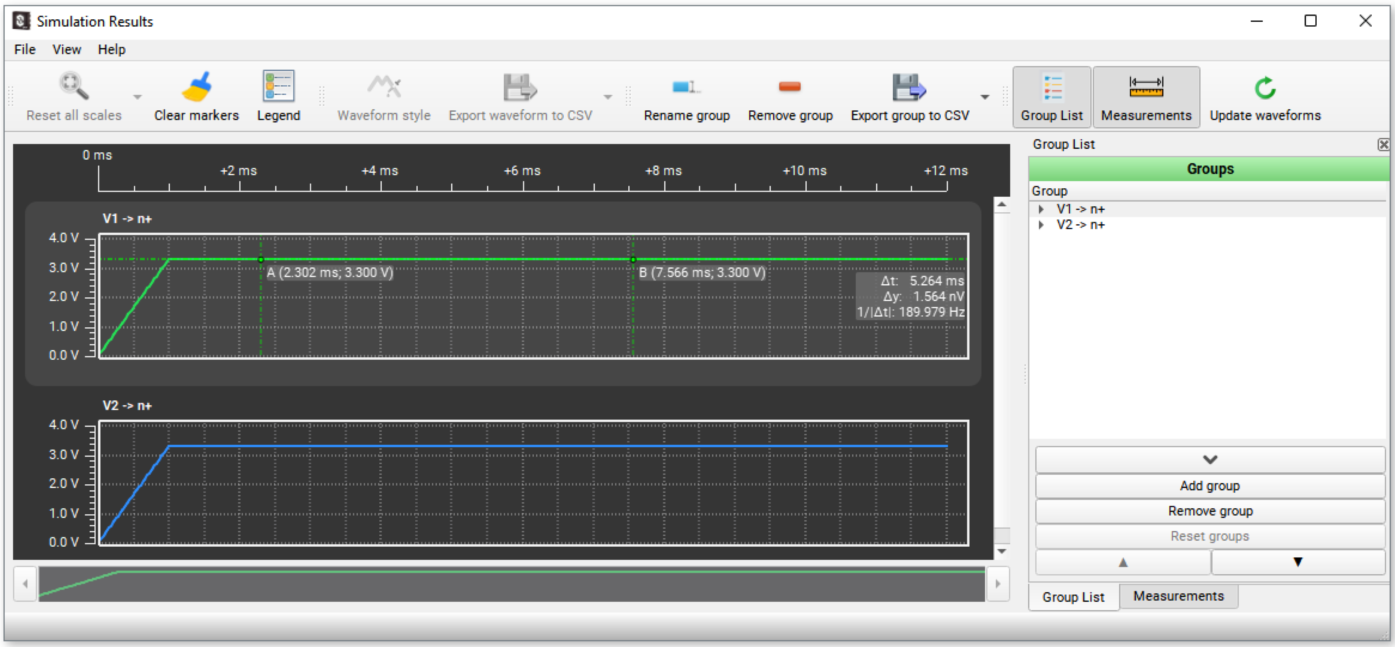
Capacitor 'C1':	*Internal capacitor R series and R parallel settings were enabled.*
Capacitor 'C2':	*Internal capacitor R series and R parallel settings were enabled.*
Voltage Source 'V1':	*Internal source capacitance and resistance settings were enabled.*
Analysis settings:	*Use initial conditions was enabled.*

Simulation report window

2.4.9 Simulation Results window

Depending on the selected simulation type (regular or dynamic) on the *Debugging Controls*, the *Simulation Results* window has different behavior.

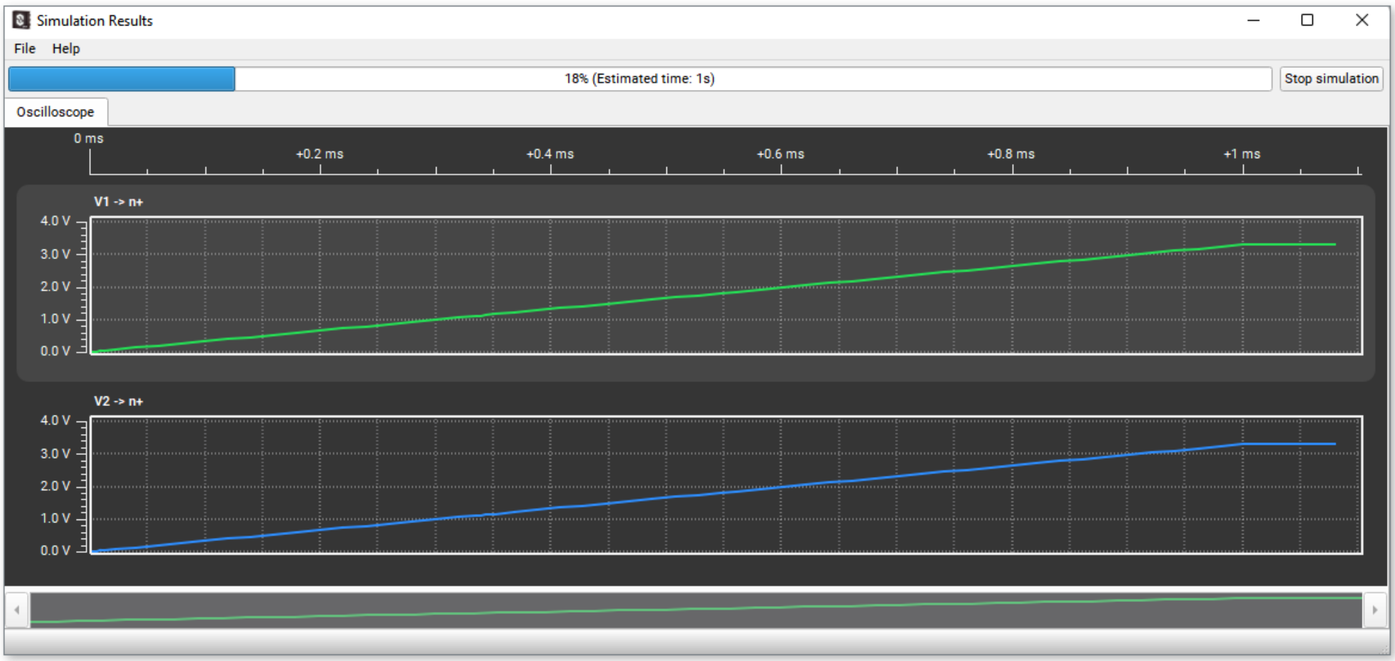
While regular simulation type is enabled, *Simulation Results* opens only after the completed/interrupted simulation process.



Simulation Results (regular simulation)

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

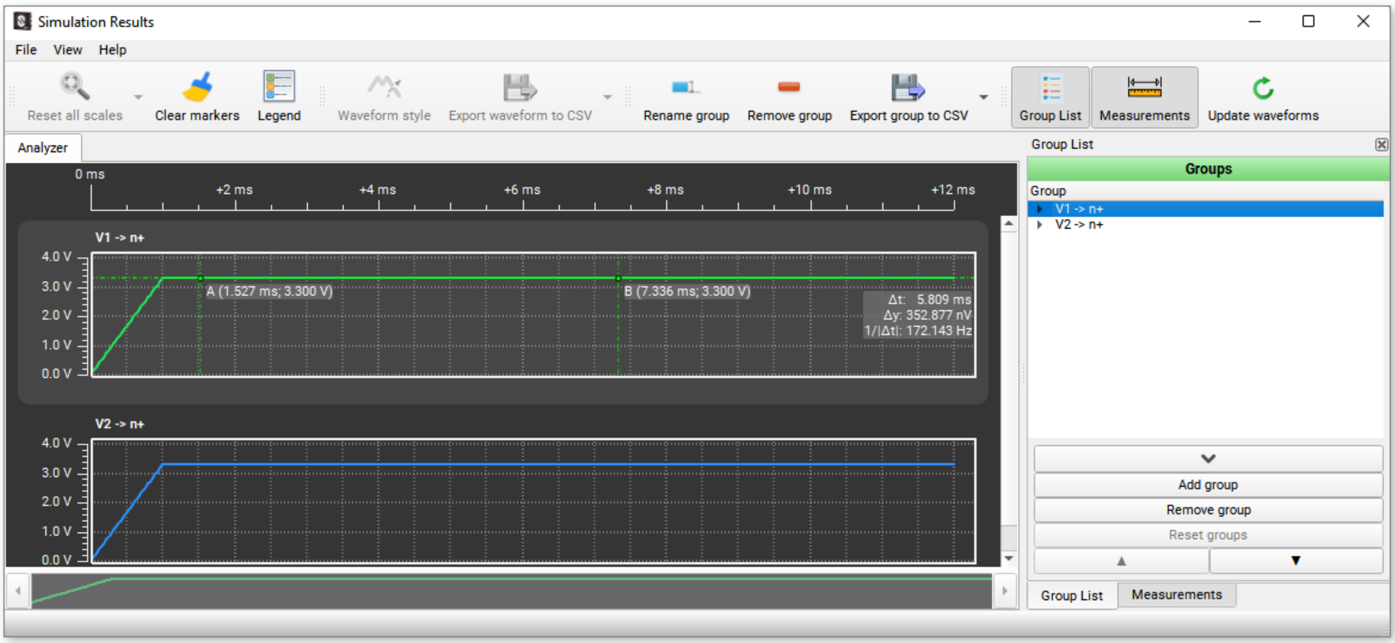
➤ *Oscilloscope* — displays the results at runtime. Interaction with waveforms on this tab is disabled. Here you may also see the progress bar with the estimated completion time along with the *Stop simulation* button for process termination. Once the simulation run is completed, the *Analyzer* tab appears instead



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 on the *Debugging Controls*. This allows to see more accurate changes in too precise signals.

➤ *Analyzer* — shows the completed/interrupted simulation results, which are now available for interaction



Simulation Results Analyzer tab (dynamic simulation)

If you close the window, you can reopen the latest simulation results by clicking *Show plots* on the *Debugging Controls* panel.

Regardless of the simulation type, you can work with simulation results using the tools and features mentioned below.

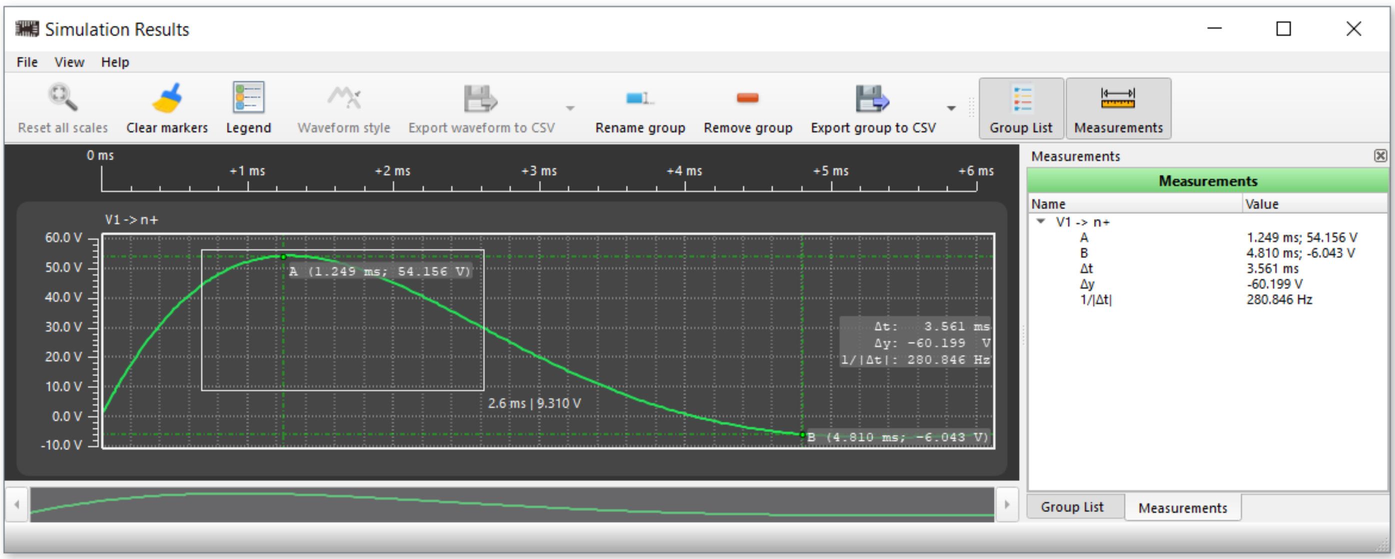
Toolbar

Toolbar provides operations with the plot area, waveform style, group management, and results data export. You can export a single waveform or a whole group to a .csv, .vcd, or .png file. You can also export all waveforms as a single *.png* file.

Plot widget

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

You can set the markers with a `Ctrl` + right and left click to see the signal information in the particular point. See the examples of markers and ∆t and ∆V parameters on the *Plot* widget:



Measurements

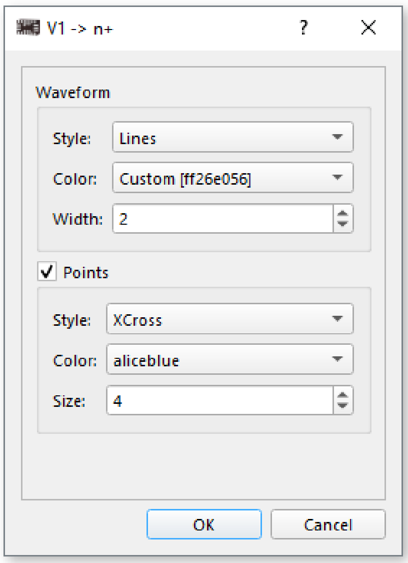
Name	Value
V1 -> n+	
A	1.249 ms; 54.156 V
B	4.810 ms; -6.043 V
Δt	3.561 ms
Δy	-60.199 V
1/|Δt|	280.846 Hz

Plot widget and Measurements

Group List

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

Click the waveform on the panel to modify its style, color, or width by choosing the *Waveform style* option from the toolbar or context menu. Also, use the context menu to rename the group.



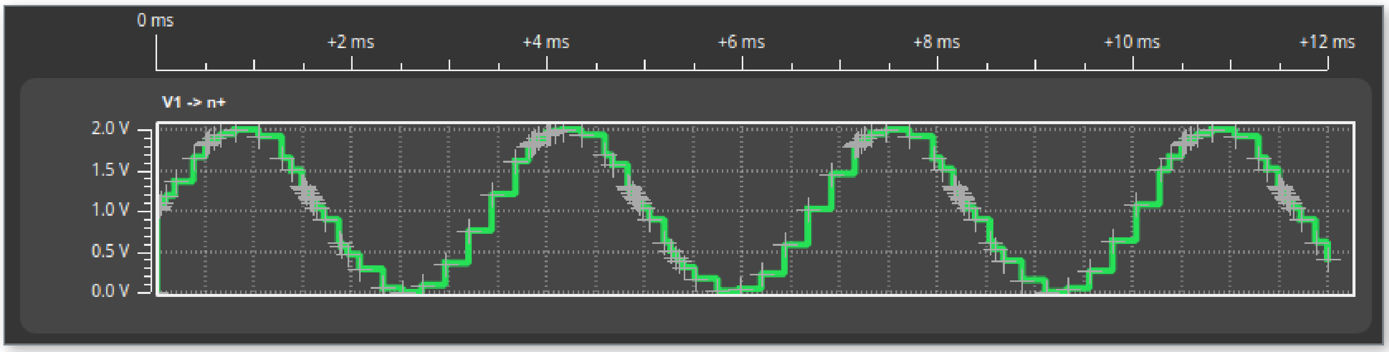
Waveform

Style: Lines
Color: Custom [ff26e056]
Width: 2

☑ Points

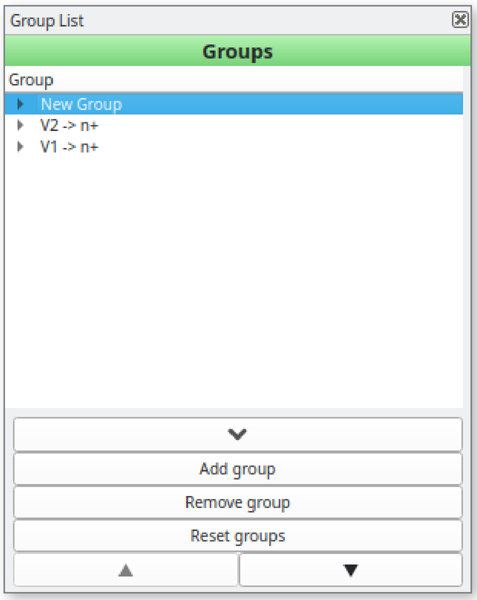
Style: XCross
Color: aliceblue
Size: 4

Waveform Plot Configuration window



A sine simulated with huge interval settings

The *Group List* bottom controls let you operate the waveforms.

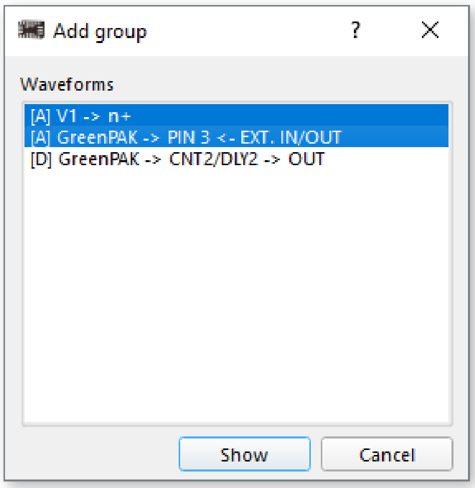


Group List

Group controls include:

- ⌄ / ⌃ — fold/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. You can create

multiple groups with the same waveform reused.

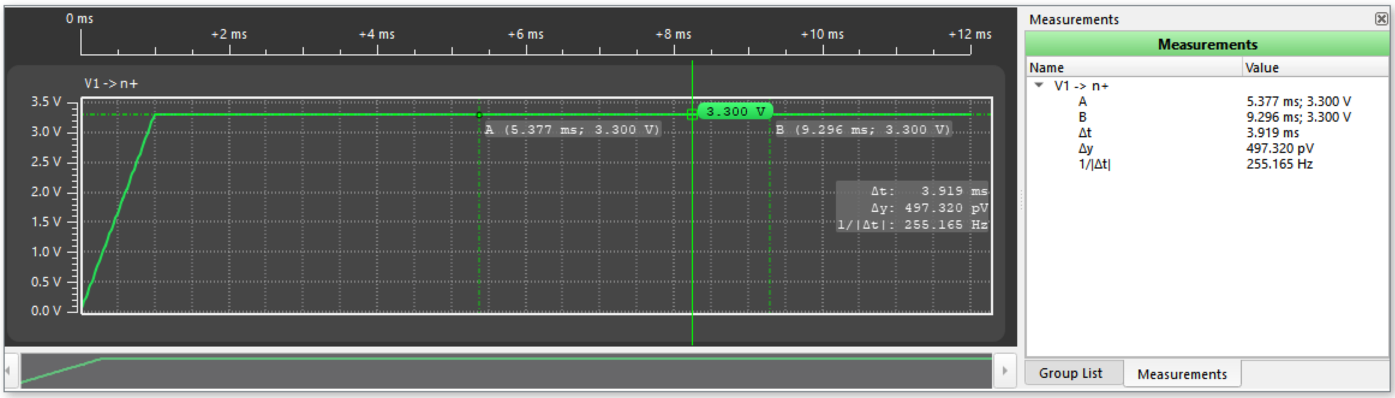

Add group
Waveforms
[A] V1 -> n+
[A] GreenPAK -> PIN 3 <- EXT. IN/OUT
[D] GreenPAK -> CNT2/DLY2 -> OUT
Show
Cancel

Add group menu

- *Remove group* — delete the selected group
- ▾ / ▴ — move a selected group up or down
- *Reset groups* — restore default and remove all added groups

Measurements

This panel displays measurement data for the added plot markers.



Measurements

Name	Value
V1 -> n+	
A	5.377 ms; 3.300 V
B	9.296 ms; 3.300 V
Δt	3.919 ms
Δy	497.320 pV
1/|Δt|	255.165 Hz

Measurements panel

2.5 FPGA Editor tool

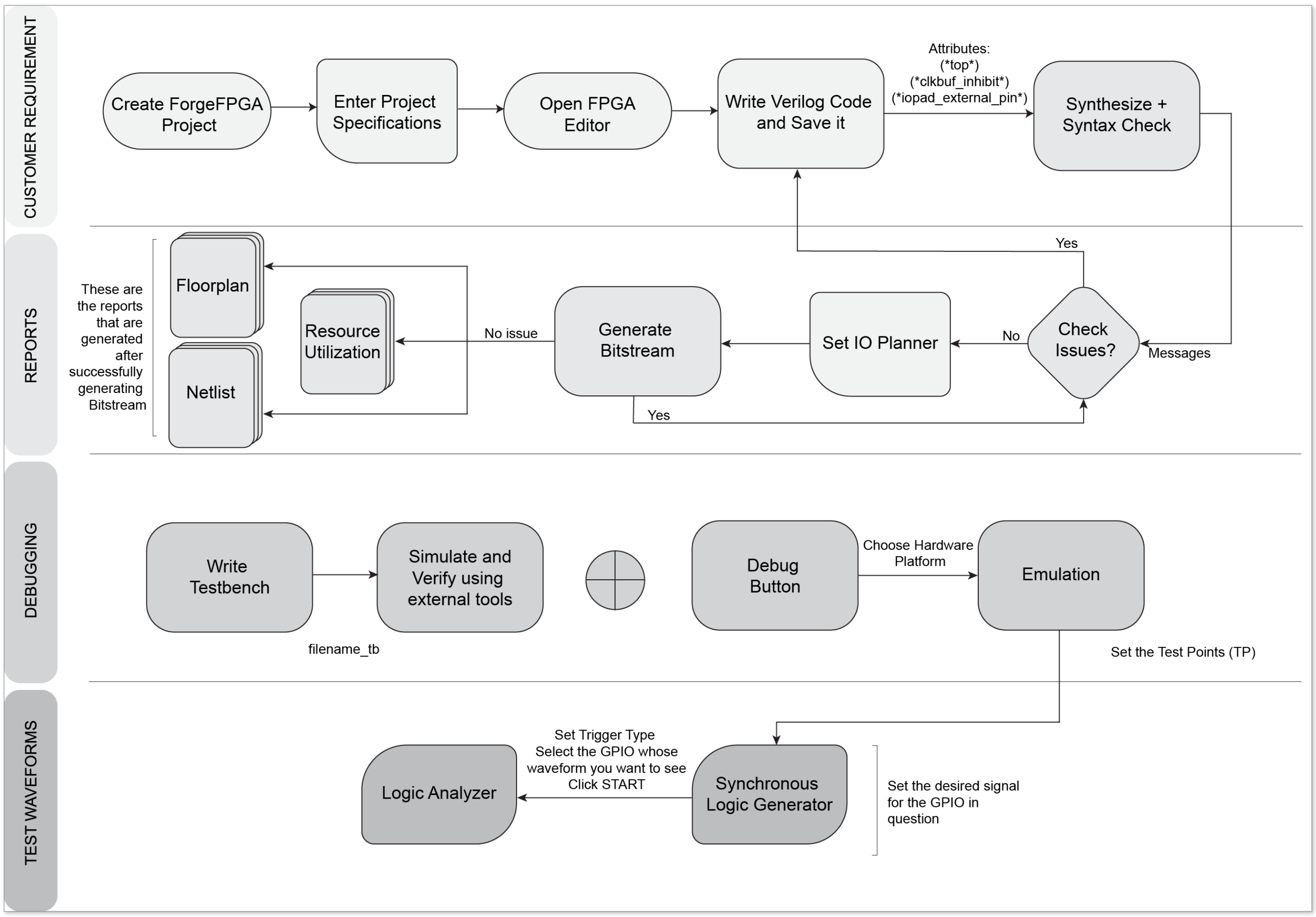
The *FPGA Editor* is a complex software tool designed to create and configure the FPGA logic. The editor allows you to create designs for Field-Programmable Gate Arrays (FPGAs) using Verilog, a Hardware Description Language (HDL). Before you program your design onto the FPGA, the editor provides an option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

The *FPGA Editor* includes a graphical representation of the Register Transfer Level (RTL) for the FPGA logic you designed. You can also use an additional macrocell constructor instead of writing a Verilog code, manually place the required blocks to the work area, connect them, and convert the complete design into the code representation.

In addition, *FPGA Editor* allows you to work with external designs by importing files with the created logic mapped to the components. Find the description of these and some additional features 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.

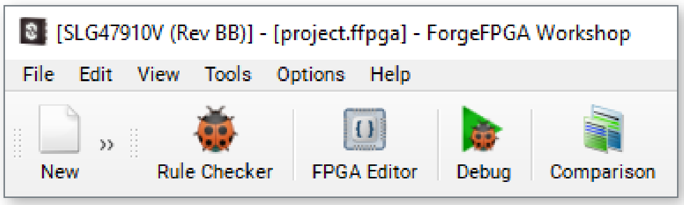


CUSTOMER REQUIREMENT
Create ForgeFPGA Project
Enter Project Specifications
Open FPGA Editor
Write Verilog Code and Save it
Attributes: (*top*) (*clkbuf_inhibit*) (*iopad_external_pin*)
Synthesize + Syntax Check
REPORTS
These are the reports that are generated after successfully generating Bitstream
Floorplan
Netlist
Resource Utilization
No issue
Generate Bitstream
Set IO Planner
No
Yes
Check Issues?
Messages
Yes
DEBUGGING
Write Testbench
filename_tb
Simulate and Verify using external tools
Debug Button
Choose Hardware Platform
Emulation
Set the Test Points (TP)
TEST WAVEFORMS
Logic Analyzer
Set Trigger Type Select the GPIO whose waveform you want to see Click START
Synchronous Logic Generator
Set the desired signal for the GPIO in question

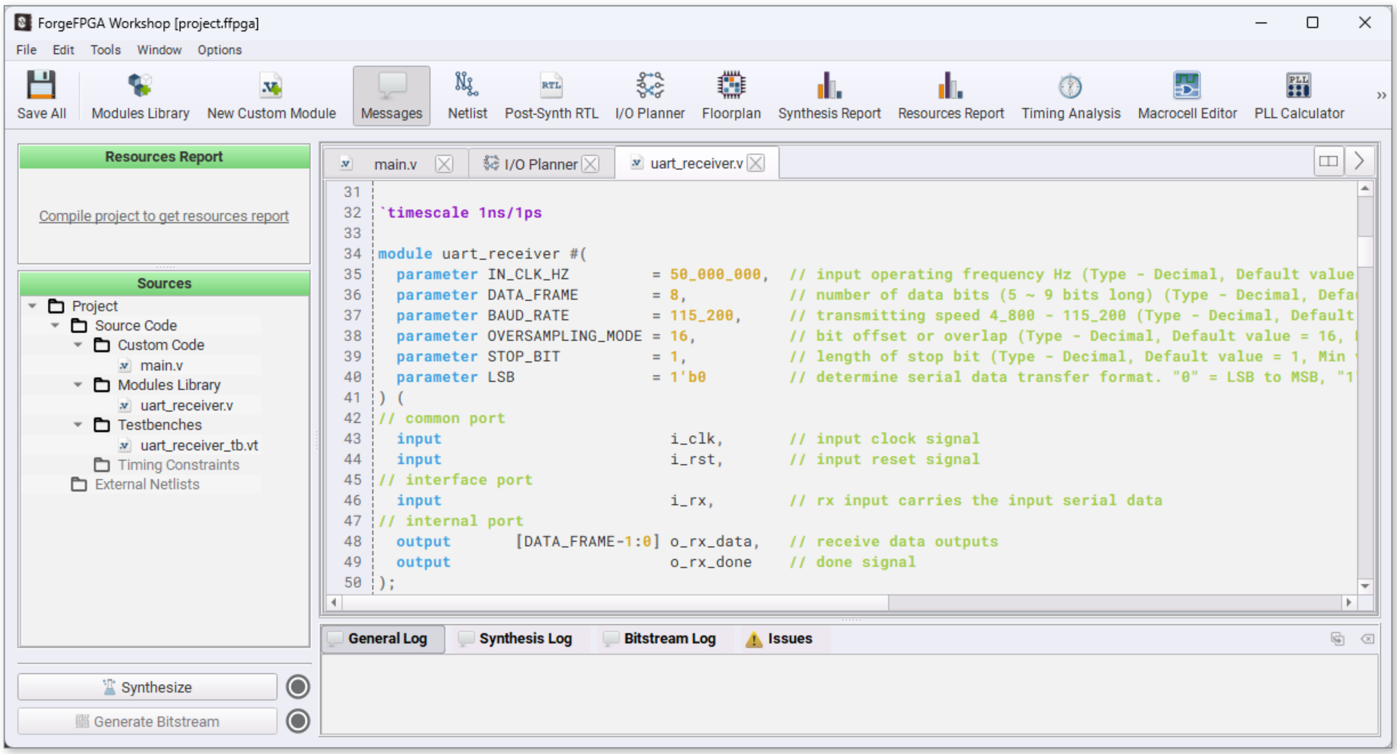
Software flow

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.



FPGA Editor button on the toolbar

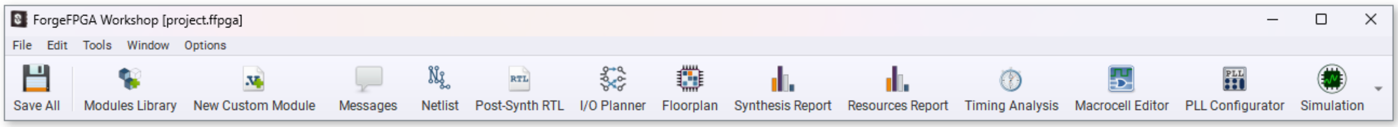


```
`timescale 1ns/1ps

module uart_receiver #(
  parameter IN_CLK_HZ          = 50_000_000,  // input operating frequency Hz (Type - Decimal, Default value
  parameter DATA_FRAME         = 8,           // number of data bits (5 ~ 9 bits long) (Type - Decimal, Defa
  parameter BAUD_RATE          = 115_200,     // transmitting speed 4_800 - 115_200 (Type - Decimal, Default
  parameter OVERSAMPLING_MODE = 16,           // bit offset or overlap (Type - Decimal, Default value = 16,
  parameter STOP_BIT           = 1,           // length of stop bit (Type - Decimal, Default value = 1, Min
  parameter LSB                = 1'b0         // determine serial data transfer format. "0" = LSB to MSB, "1
) (
// common port
  input                        i_clk,         // input clock signal
  input                        i_rst,         // input reset signal
// interface port
  input                        i_rx,          // rx input carries the input serial data
// internal port
  output       [DATA_FRAME-1:0] o_rx_data,    // receive data outputs
  output                       o_rx_done      // done signal
);
```

FPGA Editor

Toolbar

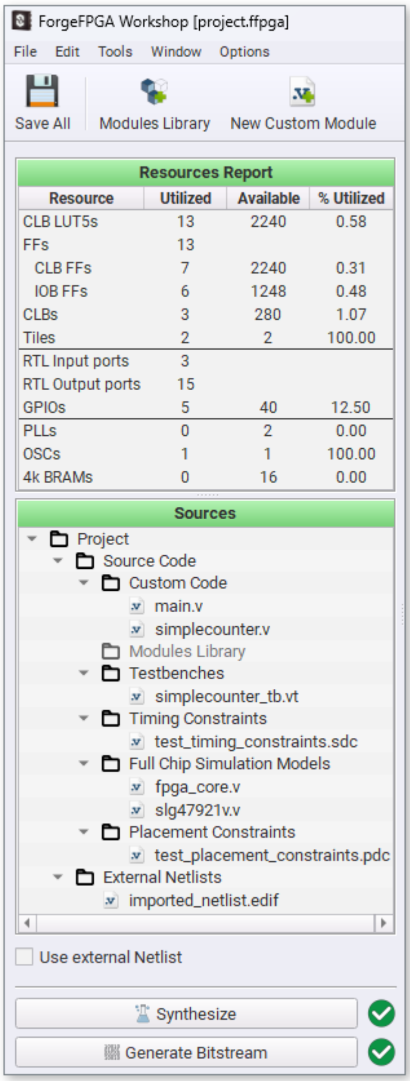
The top toolbar provides quick access to a set of tools and actions. See the description of all available tools in the next sections.



Toolbar

Control panel

From the left control panel, you can access:


ForgeFPGA Workshop [project.ffpga]
File Edit Tools Window Options
Save All
Modules Library
New Custom Module

Resources Report

Resource	Utilized	Available	% Utilized
CLB LUT5s	13	2240	0.58
FFs	13		
CLB FFs	7	2240	0.31
IOB FFs	6	1248	0.48
CLBs	3	280	1.07
Tiles	2	2	100.00
RTL Input ports	3		
RTL Output ports	15		
GPIOs	5	40	12.50
PLLs	0	2	0.00
OSCs	1	1	100.00
4k BRAMs	0	16	0.00

Sources

- Project
  - Source Code
    - Custom Code
      - main.v
      - simplecounter.v
    - Modules Library
    - Testbenches
      - simplecounter_tb.vt
    - Timing Constraints
      - test_timing_constraints.sdc
    - Full Chip Simulation Models
      - fpga_core.v
      - slg47921v.v
    - Placement Constraints
      - test_placement_constraints.pdc
  - External Netlists
    - imported_netlist.edif

Control panel

➤ *Resources Report* — an extract from the resource usage report. It shows the part 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 the Verilog sources and other supported files your project may contain. Add the module via the toolbar by clicking *New Custom Module* or select one from *Modules Library*. All supported sources can also be added via the main menu → *File*. In addition, you can import the external sources by reaching *File* → *Import* (to read more about project structure, click here)

Here you may find the following subcategories:

– *Custom Code*

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

  Customize the *Sources* list via context menu with the options to add, delete, or rename the sources.

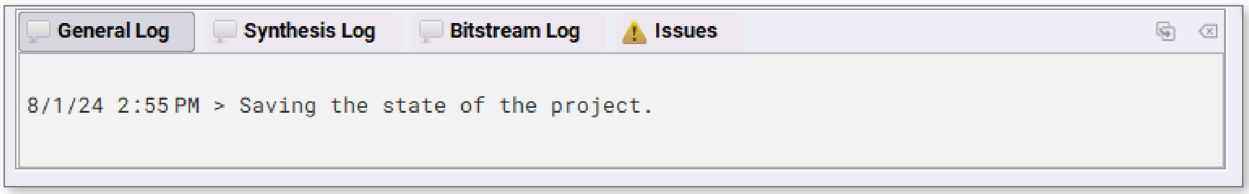
- *Synthesize and Generate Bitstream* — main controls to run toolchain on your design to produce chip configuration. Hover over the icon next to the procedure button to see the status details, such as whether the procedure was successful, failed, or if the Verilog file has been changed

Note: Make sure you save the changes for all modified modules before performing procedures to process the most updated data. Configure the saving options in Settings, *Processing* tab.

Messages panel

Here you can see the generated messages that appear after you use *Synthesize*, *Generate Bitstream*, or *Simulation* procedure.

- *General Log* — shows the information messages along with warnings and errors that were recorded while processing the design
- *Synthesis/Bitstream Log* — provides the output generated while the procedure is performed
- *Issues* — displays the warning and error messages that are automatically generated by the software when required. Once you receive a syntax error, you can right-click it and select *Show in Editor* to highlight a line in your Verilog code with a detected mistake



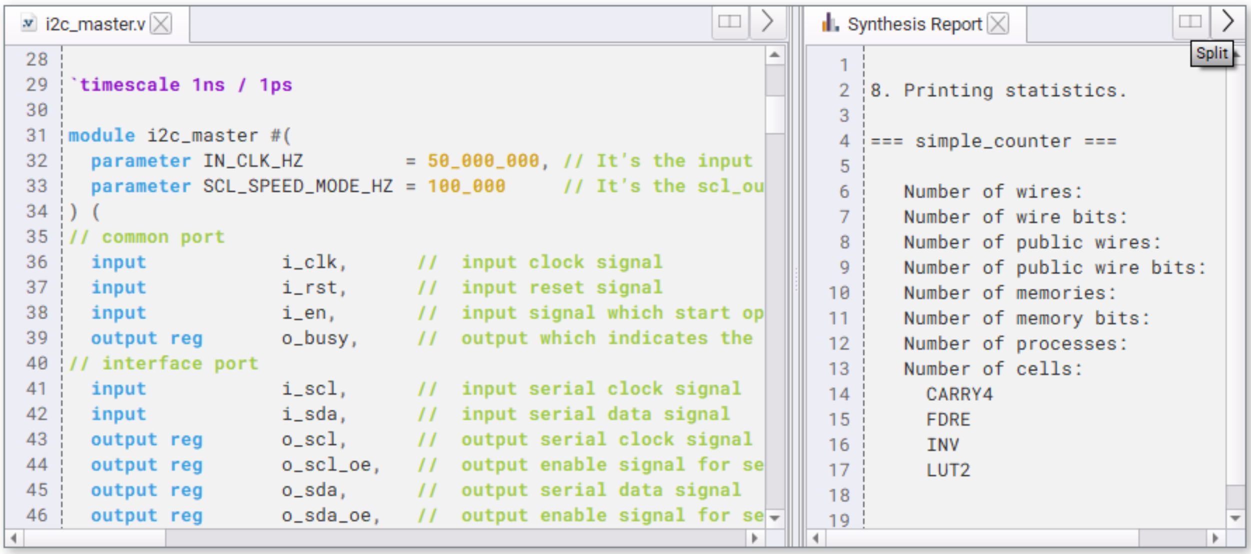
```
8/1/24 2:55 PM > Saving the state of the project.
```

Messages panel

Additional controls

See the list of *FPGA Editor* controls that can enhance your user experience (the controls availability may depend on the selected tool):

➤ *Split* — divide your work area side-by-side to work simultaneously with more than one tool. Focus on the split area and click the tool to add it to the desired side. You may chose to duplicate the same tool on both sides, which is available for most of the tools, e.g. *I/O Planner*, *Netlist*, *Post-Synth RTL*, etc.

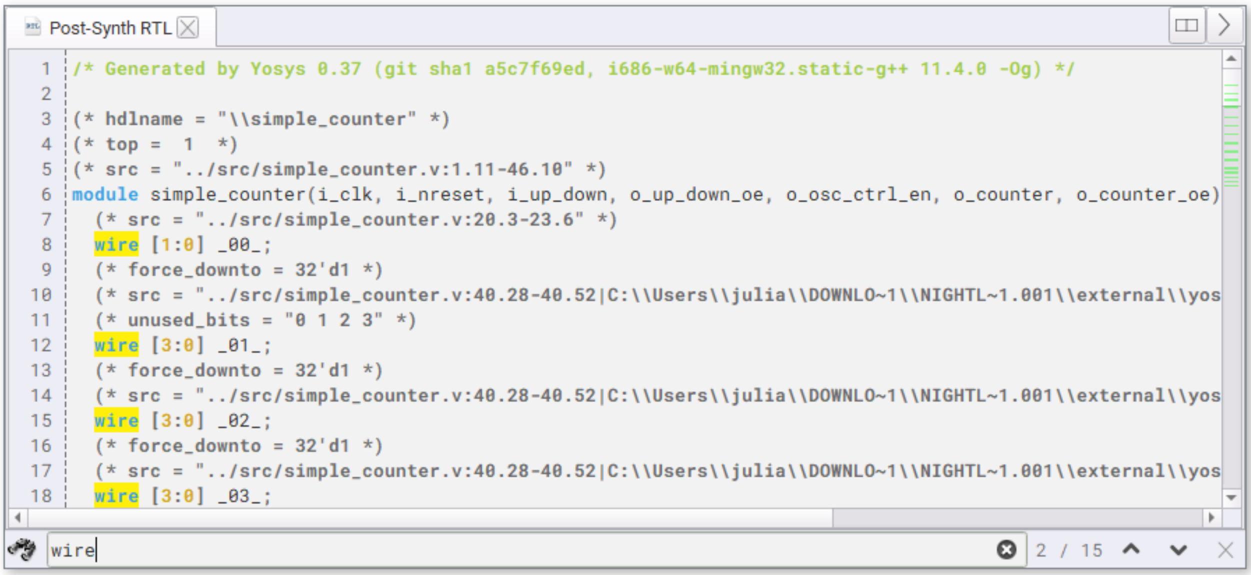


```
`timescale 1ns / 1ps

module i2c_master #(
  parameter IN_CLK_HZ          = 50_000_000, // It's the input
  parameter SCL_SPEED_MODE_HZ = 100_000      // It's the scl_ou
) (
// common port
  input            i_clk,      //  input clock signal
  input            i_rst,      //  input reset signal
  input            i_en,       //  input signal which start op
  output reg       o_busy,     //  output which indicates the
// interface port
  input            i_scl,      //  input serial clock signal
  input            i_sda,      //  input serial data signal
  output reg       o_scl,      //  output serial clock signal
  output reg       o_scl_oe,   //  output enable signal for se
  output reg       o_sda,      //  output serial data signal
  output reg       o_sda_oe,   //  output enable signal for se
```

```

8. Printing statistics.

=== simple_counter ===

   Number of wires:
   Number of wire bits:
   Number of public wires:
   Number of public wire bits:
   Number of memories:
   Number of memory bits:
   Number of processes:
   Number of cells:
     CARRY4
     FDRE
     INV
     LUT2

```

Split

➤ *Find* — activate the search field in the main menu, *Edit* → *Find* (`Ctrl` + `F`). You can enable the feature for most of the toolbar tools, including the *Messages* panel



```
/* Generated by Yosys 0.37 (git sha1 a5c7f69ed, i686-w64-mingw32.static-g++ 11.4.0 -Og) */

(* hdlname = "\\simple_counter" *)
(* top =  1  *)
(* src = "../src/simple_counter.v:1.11-46.10" *)
module simple_counter(i_clk, i_nreset, i_up_down, o_up_down_oe, o_osc_ctrl_en, o_counter, o_counter_oe)
  (* src = "../src/simple_counter.v:20.3-23.6" *)
  wire [1:0] _00_;
  (* force_downto = 32'd1 *)
  (* src = "../src/simple_counter.v:40.28-40.52|C:\\Users\\julia\\DOWNLO~1\\NIGHTL~1.001\\external\\yos
  (* unused_bits = "0 1 2 3" *)
  wire [3:0] _01_;
  (* force_downto = 32'd1 *)
  (* src = "../src/simple_counter.v:40.28-40.52|C:\\Users\\julia\\DOWNLO~1\\NIGHTL~1.001\\external\\yos
  wire [3:0] _02_;
  (* force_downto = 32'd1 *)
  (* src = "../src/simple_counter.v:40.28-40.52|C:\\Users\\julia\\DOWNLO~1\\NIGHTL~1.001\\external\\yos
  wire [3:0] _03_;
```

Find

➤ *Footer controls:*

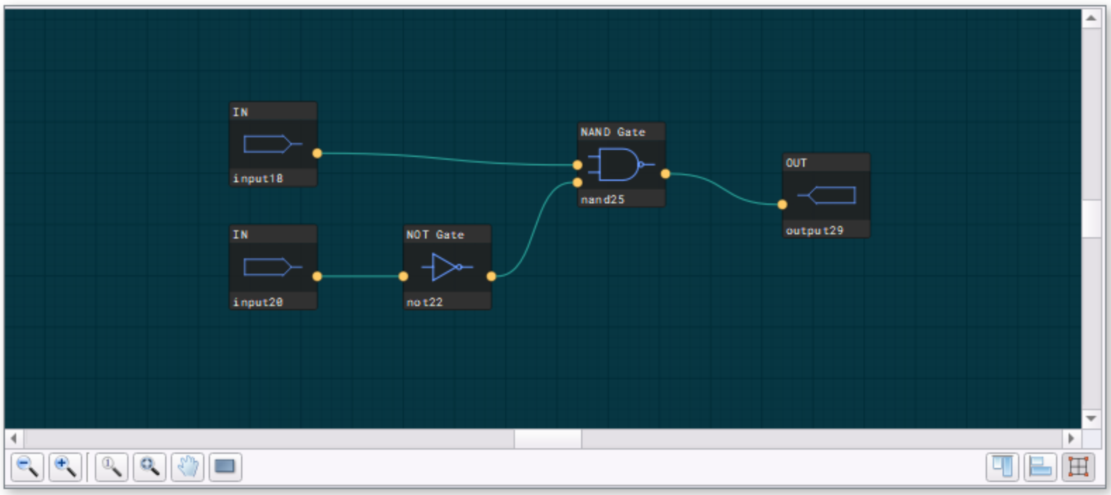
– *Zoom* — zoom in/out the work area

– *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 you want to zoom

– *Align Vertical/Horizontal* — align the macrocells relative to each other on the work area

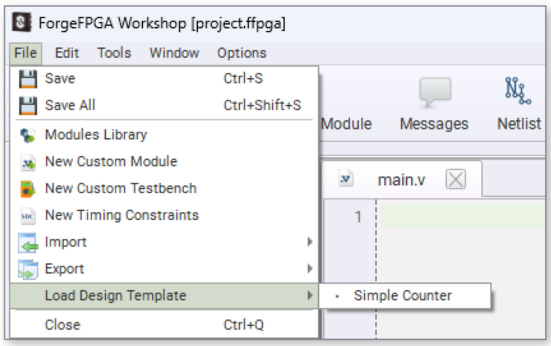
– *Snap to grid* — use for precise graphic alignment of the macrocells



Footer controls

2.5.3 Design Templates

Design Templates offer pre-built designs that you can seamlessly integrate into your project. You can find them in *FPGA Editor* → *File* → *Load Design Templates*.



FPGA Design Templates

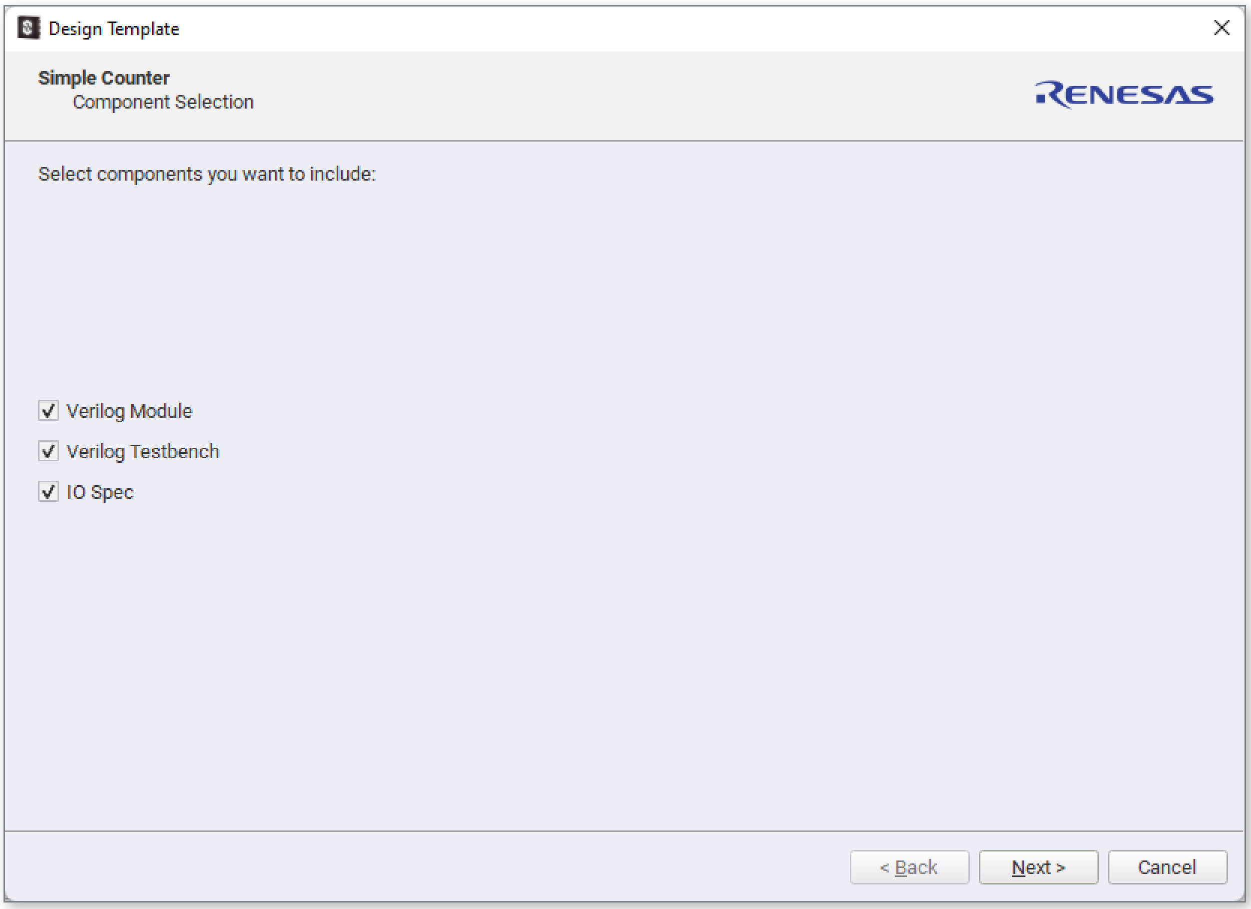
Upon selecting a design template, the *Design Template Wizard* will appear. It can guide you through component selection, IO spec diff review, and module name assignment.

➤ *Component Selection* offers you to choose the components you'd like to proceed with:

– *Verilog Module*

– *Verilog Testbench*

– *IO Spec*


Design Template

Simple Counter
Component Selection

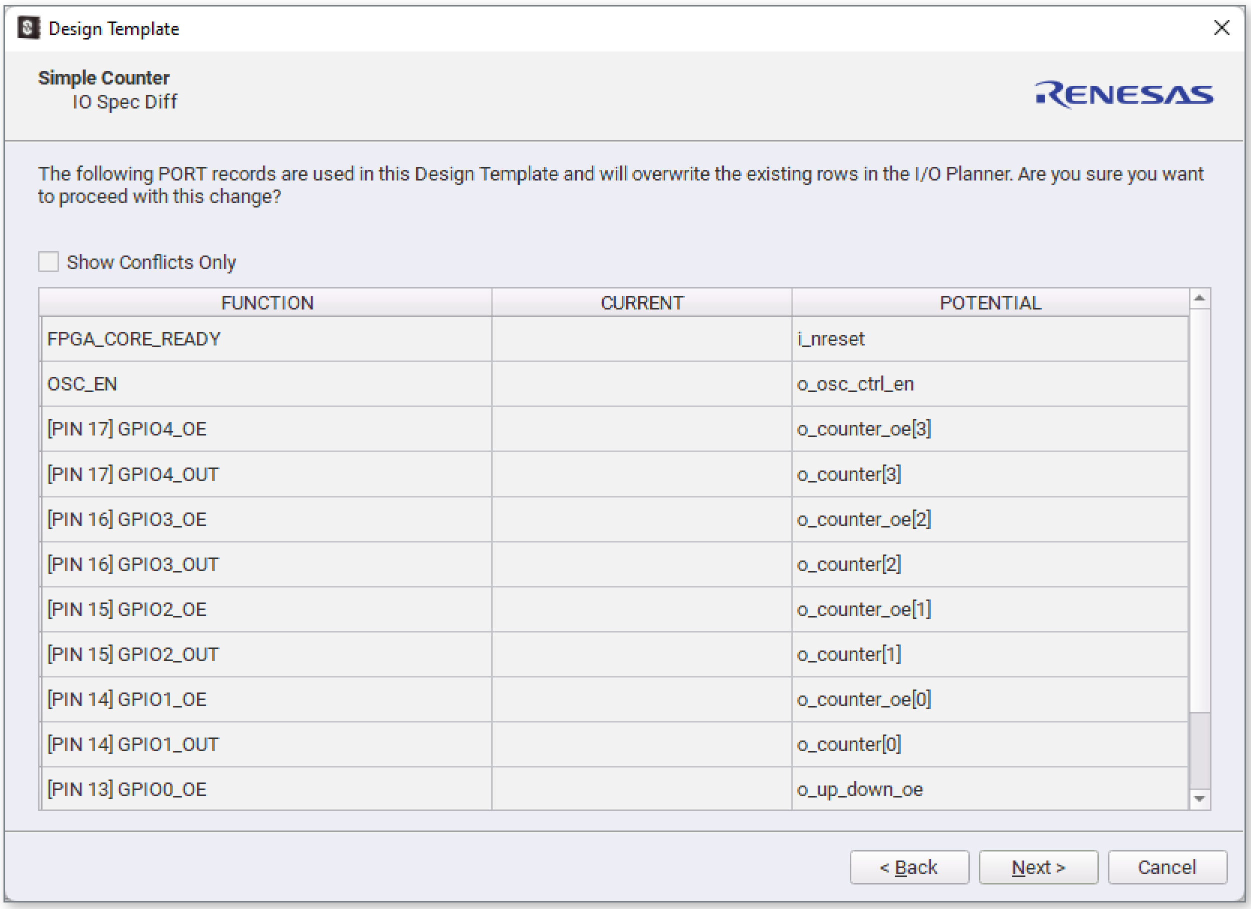
Select components you want to include:

- [x] Verilog Module
- [x] Verilog Testbench
- [x] IO Spec

Component Selection

Select all components simultaneously or individually, depending on the desired outcome for your project. The choices you make at this stage determine your next steps in the process.

➤ *IO Spec* generates an *IO Spec Diff*, presenting a comparison between your existing design port records and the ones associated with the selected template. This serves as a cautionary step as your current port configurations could potentially be overwritten by those of the chosen template. The software highlights conflicting entries in yellow. You can choose to focus solely on conflicts by selecting the *Show Conflicts Only* option. However, if you prefer not to review the *IO Specs*, the software streamlines the process by skipping the *IO Spec Diff*. In this scenario, the tool will prompt you straight to the *Module name*.


Design Template

Simple Counter
IO Spec Diff

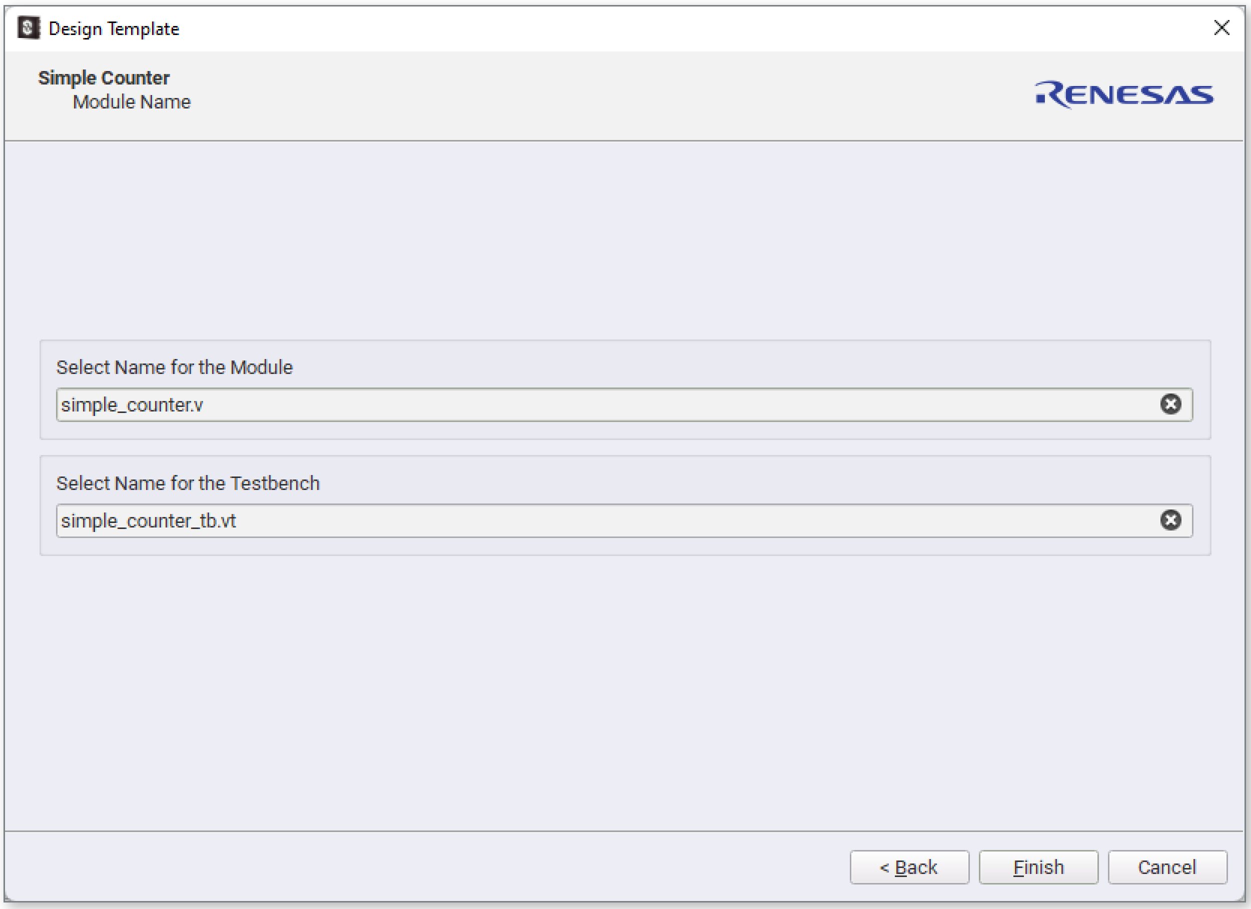
The following PORT records are used in this Design Template and will overwrite the existing rows in the I/O Planner. Are you sure you want to proceed with this change?

☐ Show Conflicts Only

FUNCTION	CURRENT	POTENTIAL
FPGA_CORE_READY		i_nreset
OSC_EN		o_osc_ctrl_en
[PIN 17] GPIO4_OE		o_counter_oe[3]
[PIN 17] GPIO4_OUT		o_counter[3]
[PIN 16] GPIO3_OE		o_counter_oe[2]
[PIN 16] GPIO3_OUT		o_counter[2]
[PIN 15] GPIO2_OE		o_counter_oe[1]
[PIN 15] GPIO2_OUT		o_counter[1]
[PIN 14] GPIO1_OE		o_counter_oe[0]
[PIN 14] GPIO1_OUT		o_counter[0]
[PIN 13] GPIO0_OE		o_up_down_oe

IO Spec Diff

➤ *Module name* is a component that assists you in assigning names to your *Verilog Module* and/or *Testbench*. If you choose to skip *Verilog Module* and *Verilog Testbench* in the *Component Selection* window, the default names are assigned.



Select Name for the Module

simple_counter.v

Select Name for the Testbench

simple_counter_tb.vt

Module Name

Once the template setup is complete, you will see new elements in your workspace. The *Design Template* Verilog code and *Testbench* names will appear in the *Sources* list, with corresponding tabs displaying the actual code and *Testbench* details. Additionally, your *I/O Planner* tab will reflect changes, showcasing the inclusion of port records from the selected design.

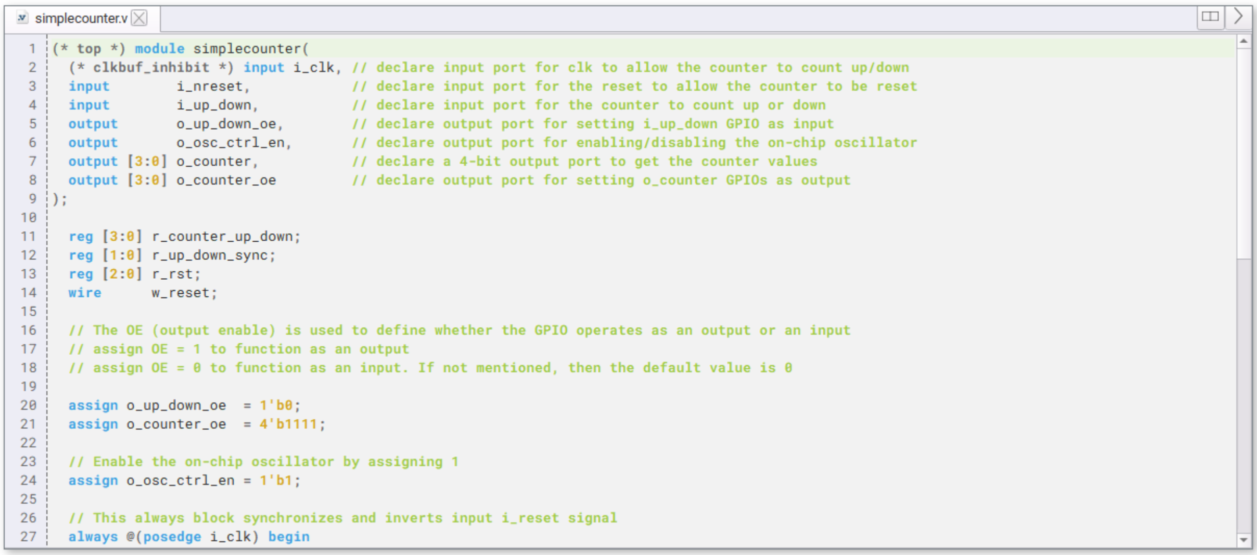
When the integration of the design template is done, you can assess its functionality by running a simulation.

2.5.4 Writing Verilog code

The toolchain of the *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:

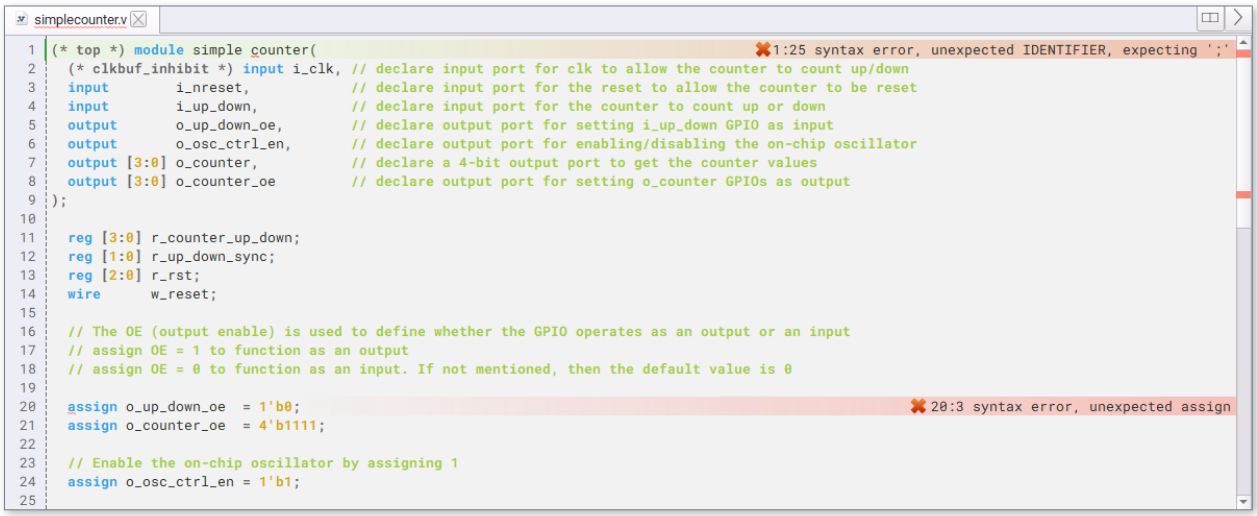
- `(* top *)` — the main module of your design should be marked with this attribute so the toolchain can successfully recognize which of the modules in the design is the top one
- `(* clkbuf_inhibit *)` — clocks signals in the input list of the main module should be marked with this attribute. This prevents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



```
(* top *) module simplecounter(
  (* clkbuf_inhibit *) input i_clk, // declare input port for clk to allow the counter to count up/down
  input         i_nreset,           // declare input port for the reset to allow the counter to be reset
  input         i_up_down,          // declare input port for the counter to count up or down
  output        o_up_down_oe,       // declare output port for setting i_up_down GPIO as input
  output        o_osc_ctrl_en,      // declare output port for enabling/disabling the on-chip oscillator
  output [3:0] o_counter,           // declare a 4-bit output port to get the counter values
  output [3:0] o_counter_oe         // declare output port for setting o_counter GPIOs as output
);

  reg [3:0] r_counter_up_down;
  reg [1:0] r_up_down_sync;
  reg [2:0] r_rst;
  wire      w_reset;

  // The OE (output enable) is used to define whether the GPIO operates as an output or an input
  // assign OE = 1 to function as an output
  // assign OE = 0 to function as an input. If not mentioned, then the default value is 0

  assign o_up_down_oe  = 1'b0;
  assign o_counter_oe  = 4'b1111;

  // Enable the on-chip oscillator by assigning 1
  assign o_osc_ctrl_en = 1'b1;

  // This always block synchronizes and inverts input i_reset signal
  always @(posedge i_clk) begin
```

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.



```
(* top *) module simple counter(                                    ✖1:25 syntax error, unexpected IDENTIFIER, expecting ';'
  (* clkbuf_inhibit *) input i_clk, // declare input port for clk to allow the counter to count up/down
  input         i_nreset,           // declare input port for the reset to allow the counter to be reset
  input         i_up_down,          // declare input port for the counter to count up or down
  output        o_up_down_oe,       // declare output port for setting i_up_down GPIO as input
  output        o_osc_ctrl_en,      // declare output port for enabling/disabling the on-chip oscillator
  output [3:0] o_counter,           // declare a 4-bit output port to get the counter values
  output [3:0] o_counter_oe         // declare output port for setting o_counter GPIOs as output
);

  reg [3:0] r_counter_up_down;
  reg [1:0] r_up_down_sync;
  reg [2:0] r_rst;
  wire      w_reset;

  // The OE (output enable) is used to define whether the GPIO operates as an output or an input
  // assign OE = 1 to function as an output
  // assign OE = 0 to function as an input. If not mentioned, then the default value is 0

  assign o_up_down_oe  = 1'b0;                                       ✖20:3 syntax error, unexpected assign
  assign o_counter_oe  = 4'b1111;

  // Enable the on-chip oscillator by assigning 1
  assign o_osc_ctrl_en = 1'b1;

```

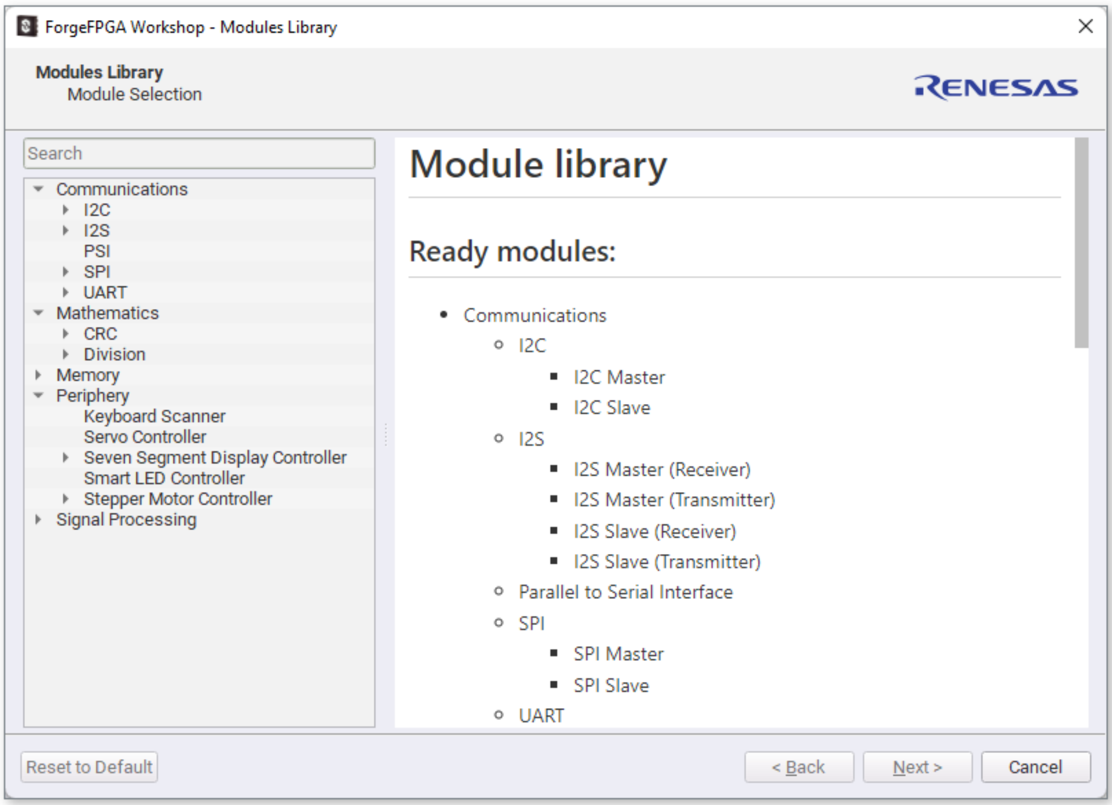
Linter error examples

2.5.5 Modules Library

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

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

Inside the *Modules Library* GUI, you can find the schematic, resource estimation along with the description of the block selected. The GUI gives a detailed explanation of all the input and output pins of the block and allows you to change the parameters as desired. This gives you the flexibility to create the Verilog code of the module needed and its associated testbench with just a few clicks.


ForgeFPGA Workshop - Modules Library

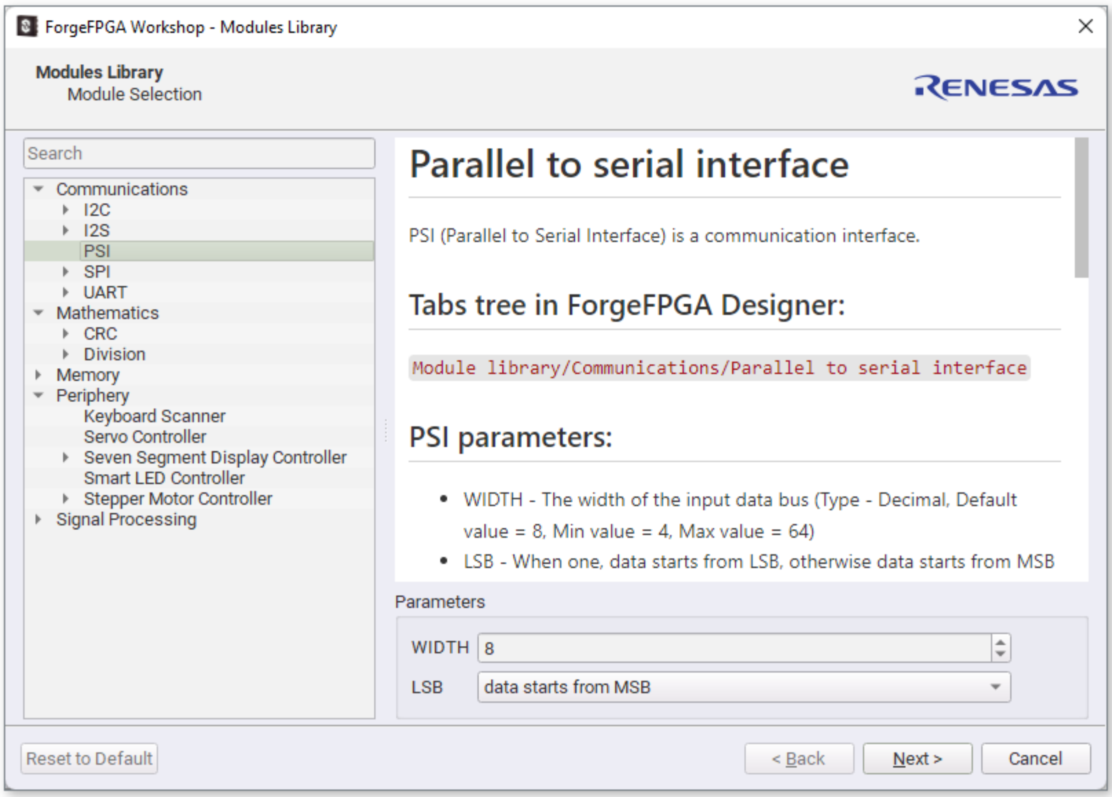
Modules Library
Module Selection

Module library

Ready modules:

- Communications
  - I2C
    - I2C Master
    - I2C Slave
  - I2S
    - I2S Master (Receiver)
    - I2S Master (Transmitter)
    - I2S Slave (Receiver)
    - I2S Slave (Transmitter)
  - Parallel to Serial Interface
  - SPI
    - SPI Master
    - SPI Slave
  - UART

Modules Library GUI


ForgeFPGA Workshop - Modules Library

Modules Library
Module Selection

Parallel to serial interface

PSI (Parallel to Serial Interface) is a communication interface.

Tabs tree in ForgeFPGA Designer:

`Module library/Communications/Parallel to serial interface`

PSI parameters:

- WIDTH - The width of the input data bus (Type - Decimal, Default value = 8, Min value = 4, Max value = 64)
- LSB - When one, data starts from LSB, otherwise data starts from MSB

Parameters

WIDTH: 8
LSB: data starts from MSB

Modules Library GUI with parameters

You can check the added block on the *Sources* control panel under the *Ip Blocks* group. Along with the block, the testbench module is added, and you can find it under the *Testbenches* group. You can also add multiple modules to your design and work with them simultaneously.

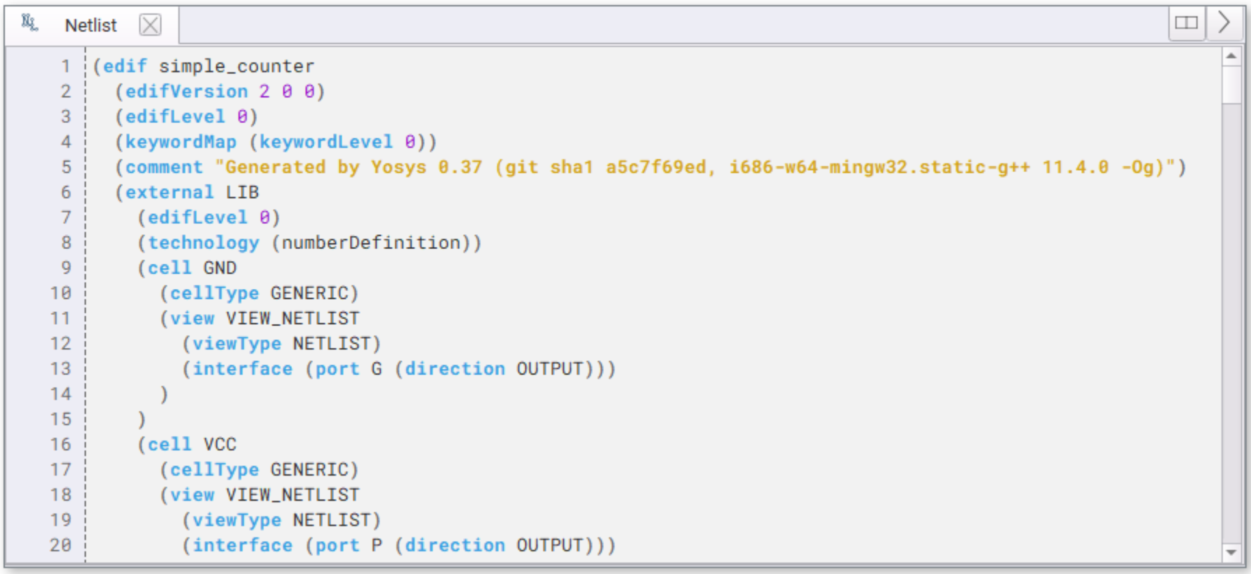
2.5.6 Synthesis

FPGA Editor has a built-in synthesis tool (Yosys, v0.37) 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 on your design, click the *Synthesize* button in the bottom left corner of *FPGA Editor* or from the main menu *Tools* → *Run Synthesis*.

During the synthesis the software also checks the Verilog code for syntax errors. You can check the log output on the *Messages* panel. The synthesis is successful only when the code is syntax error-free.

Netlist

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



```
(edif simple_counter
  (edifVersion 2 0 0)
  (edifLevel 0)
  (keywordMap (keywordLevel 0))
  (comment "Generated by Yosys 0.37 (git sha1 a5c7f69ed, i686-w64-mingw32.static-g++ 11.4.0 -Og)")
  (external LIB
    (edifLevel 0)
    (technology (numberDefinition))
    (cell GND
      (cellType GENERIC)
      (view VIEW_NETLIST
        (viewType NETLIST)
        (interface (port G (direction OUTPUT)))
      )
    )
    (cell VCC
      (cellType GENERIC)
      (view VIEW_NETLIST
        (viewType NETLIST)
        (interface (port P (direction OUTPUT)))
```

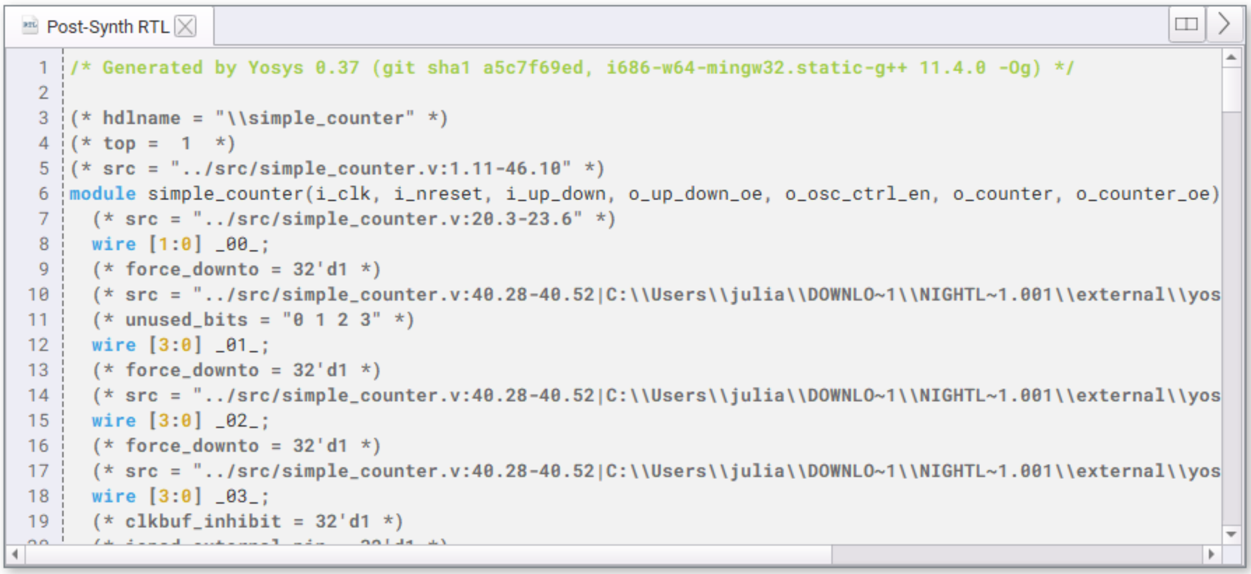
Netlist

You can also import an existing netlist by clicking *File* → *Import* → *Netlist*. The external netlist will appear in the *Sources* tree.

Post-Synth RTL

At the Register-Transfer Level, the design is represented by combinational data paths and registers. RTL synthesis is easy as 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. It helps in providing information about the clock and other clock-related logic in the design, which enables additional I/O planning.

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



```
/* Generated by Yosys 0.37 (git sha1 a5c7f69ed, i686-w64-mingw32.static-g++ 11.4.0 -Og) */

(* hdlname = "\\simple_counter" *)
(* top =  1  *)
(* src = "../src/simple_counter.v:1.11-46.10" *)
module simple_counter(i_clk, i_nreset, i_up_down, o_up_down_oe, o_osc_ctrl_en, o_counter, o_counter_oe)
  (* src = "../src/simple_counter.v:20.3-23.6" *)
  wire [1:0] _00_;
  (* force_downto = 32'd1 *)
  (* src = "../src/simple_counter.v:40.28-40.52|C:\\Users\\julia\\DOWNLO~1\\NIGHTL~1.001\\external\\yos
  (* unused_bits = "0 1 2 3" *)
  wire [3:0] _01_;
  (* force_downto = 32'd1 *)
  (* src = "../src/simple_counter.v:40.28-40.52|C:\\Users\\julia\\DOWNLO~1\\NIGHTL~1.001\\external\\yos
  wire [3:0] _02_;
  (* force_downto = 32'd1 *)
  (* src = "../src/simple_counter.v:40.28-40.52|C:\\Users\\julia\\DOWNLO~1\\NIGHTL~1.001\\external\\yos
  wire [3:0] _03_;
  (* clkbuf_inhibit = 32'd1 *)
```

Post-Synth RTL

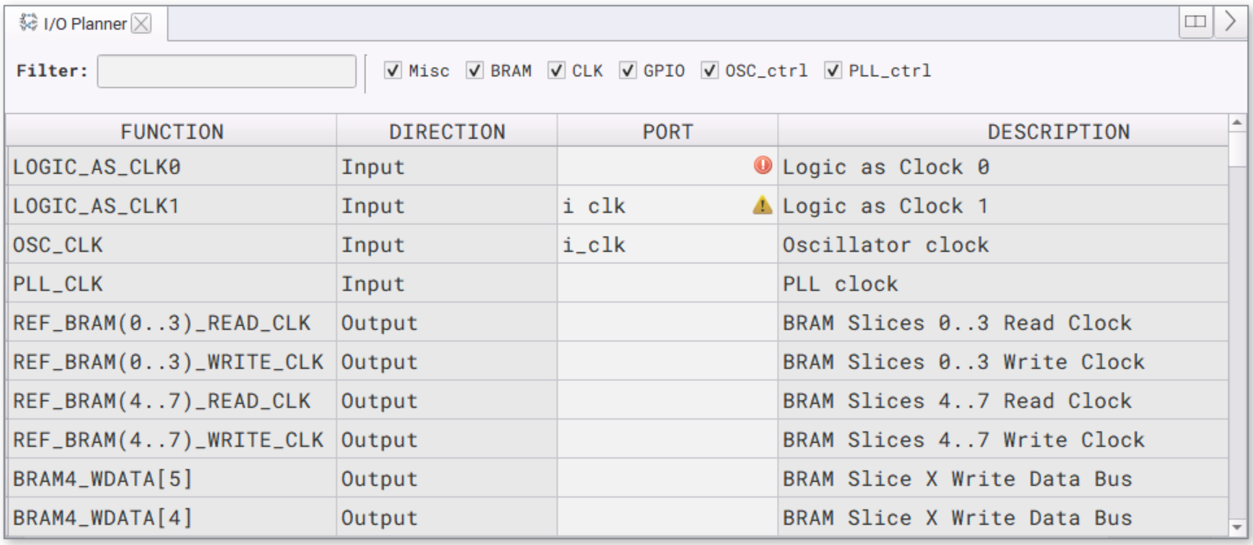
I/O Planner

Each I/O port available on FPGA has a dedicated function that can be mapped to your 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 pin mode
- *Port* — editable column, where you can input ports from your Verilog design to connect them to the desired functionality. Double-click a cell to see the list of all the ports defined in the Verilog code
- *Description* — data describing the port type and the function it fulfills

Note: The cell suggests when the port name is invalid by showing the warning icon. Hover over the cell to trigger the info hint with more details. Correct the name to perform the procedures successfully.



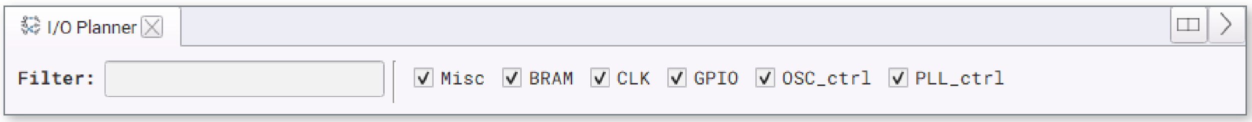
I/O Planner

Filter: ✓ Misc ✓ BRAM ✓ CLK ✓ GPIO ✓ OSC_ctrl ✓ PLL_ctrl

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

I/O Planner

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



I/O Planner

Filter: ✓ Misc ✓ BRAM ✓ CLK ✓ GPIO ✓ OSC_ctrl ✓ PLL_ctrl

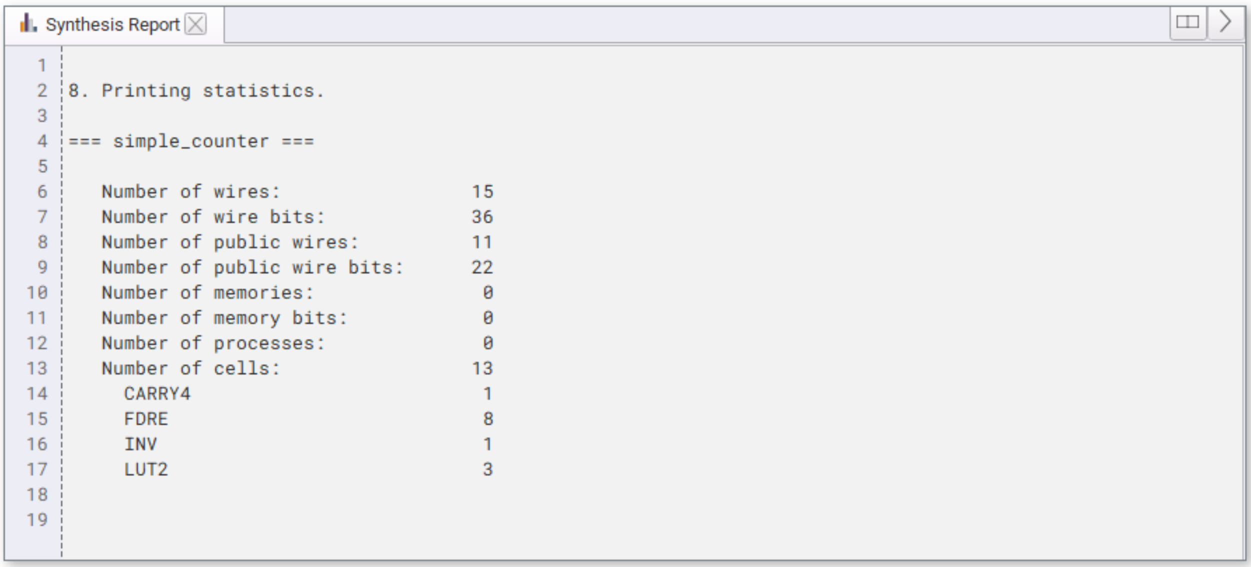
Filter controls

You can also clear or import/export the port data (.txt or .csv formats) via main menu, *Tools* → *I/O Planner*, or using the table's context menu.

Synthesis Report

After performing the synthesis of your current design, 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 reach the main menu *Window* → *Synthesis Report*.



```

8. Printing statistics.

=== simple_counter ===

   Number of wires:                 15
   Number of wire bits:             36
   Number of public wires:          11
   Number of public wire bits:      22
   Number of memories:               0
   Number of memory bits:            0
   Number of processes:              0
   Number of cells:                 13
     CARRY4                          1
     FDRE                            8
     INV                             1
     LUT2                            3

```

Synthesis Report

2.5.7 Generating Bitstream

To prepare your design to be sent to the device, you need to perform the place-and-route and the bitstream generation procedures. You can do this once the netlist and bitstream files are successfully generated. Place-and-route takes the elements of the synthesized netlist and maps its primitives to FPGA physical resources. To receive bitstream data, click the *Generate Bitstream* button in the bottom left corner of *FPGA Editor* or from the main menu, *Tools* → *Generate Bitstream*.

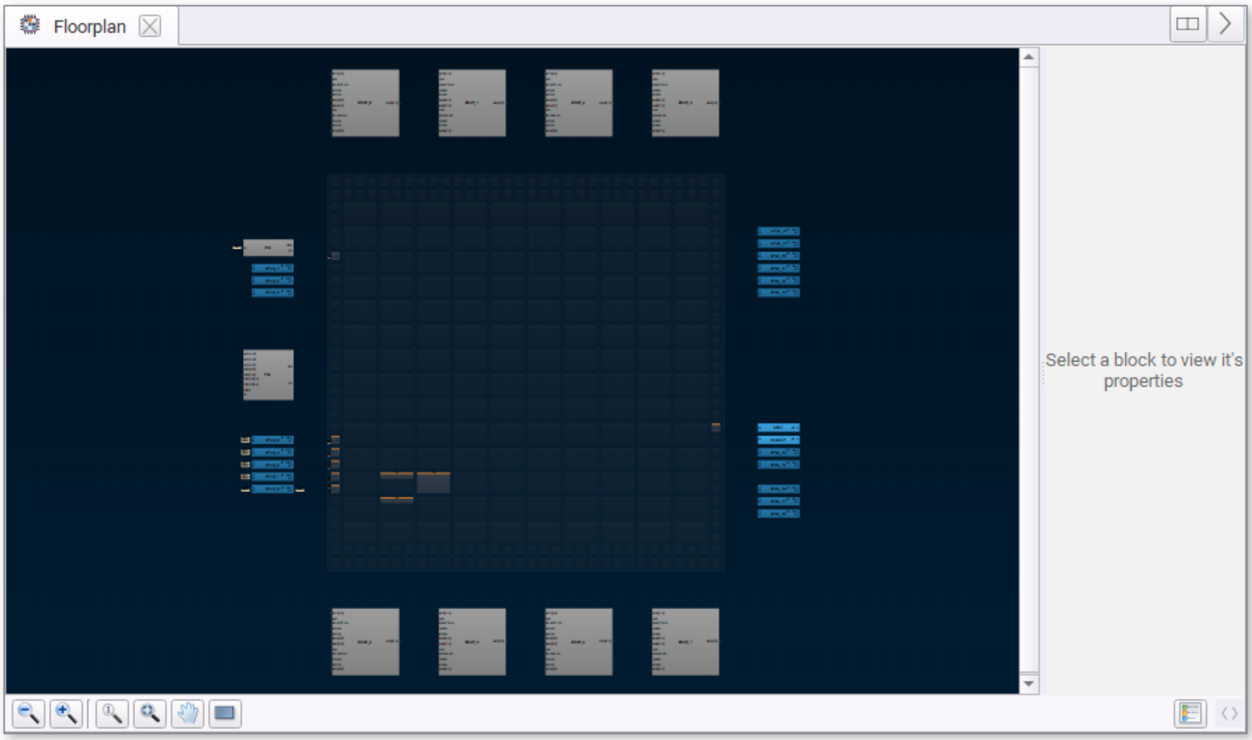
You can check the *Bitstream Log* tab on the *Messages* panel to inspect the background steps after you click *Generate Bitstream*. In the background, the software automatically performs technology mapping, clustering and floor planning, placement and optimization, routing and resource calculation. If the issue occurs in any of these steps, the bitstream generation will be incomplete, and you will see outcome on the *Messages* panel.

The generated bitstream is a hex file which, if generated correctly, you can further use for debugging your design. Read more about the chip/flash procedures and where to find them in section 2.2.6 Debugging Controls.

After bitstream generation is completed, you can see the generated info for the tools described later in this section.

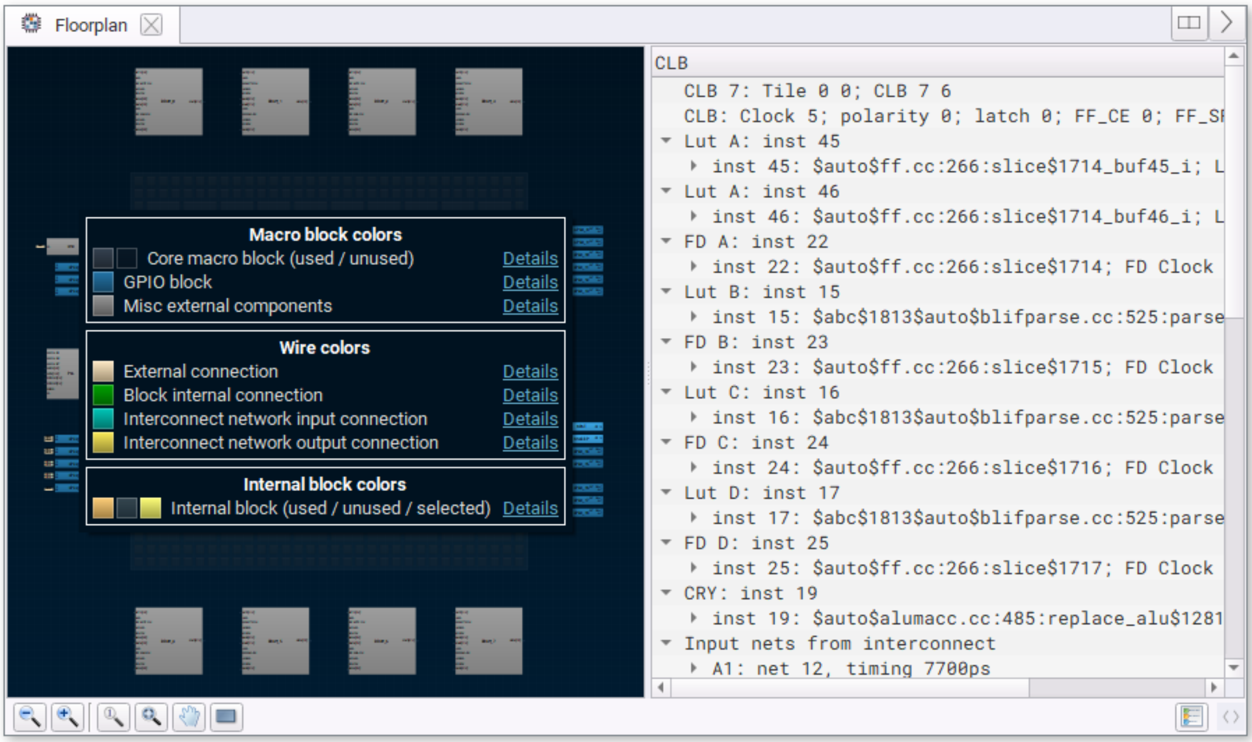
Floorplan

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 via the toolbar or *Window* section in the main menu. Use the bottom toolbar controls to take a closer look and navigate inside the tool.



Floorplan

Click the internal component to see its details on the block configuration info panel. Also, see the components and connections color scheme by clicking the *Legend box* icon at the right bottom.



Macro block colors
- Core macro block (used / unused) Details
- GPIO block Details
- Misc external components Details

Wire colors
- External connection Details
- Block internal connection Details
- Interconnect network input connection Details
- Interconnect network output connection Details

Internal block colors
- Internal block (used / unused / selected) Details

```
CLB
  CLB 7: Tile 0 0; CLB 7 6
  CLB: Clock 5; polarity 0; latch 0; FF_CE 0; FF_S
▾ Lut A: inst 45
    ▸ inst 45: $auto$ff.cc:266:slice$1714_buf45_i; L
▾ Lut A: inst 46
    ▸ inst 46: $auto$ff.cc:266:slice$1714_buf46_i; L
▾ FD A: inst 22
    ▸ inst 22: $auto$ff.cc:266:slice$1714; FD Clock
▾ Lut B: inst 15
    ▸ inst 15: $abc$1813$auto$blifparse.cc:525:parse
▾ FD B: inst 23
    ▸ inst 23: $auto$ff.cc:266:slice$1715; FD Clock
▾ Lut C: inst 16
    ▸ inst 16: $abc$1813$auto$blifparse.cc:525:parse
▾ FD C: inst 24
    ▸ inst 24: $auto$ff.cc:266:slice$1716; FD Clock
▾ Lut D: inst 17
    ▸ inst 17: $abc$1813$auto$blifparse.cc:525:parse
▾ FD D: inst 25
    ▸ inst 25: $auto$ff.cc:266:slice$1717; FD Clock
▾ CRY: inst 19
    ▸ inst 19: $auto$alumacc.cc:485:replace_alu$1281
▾ Input nets from interconnect
    ▸ A1: net 12, timing 7700ps
```

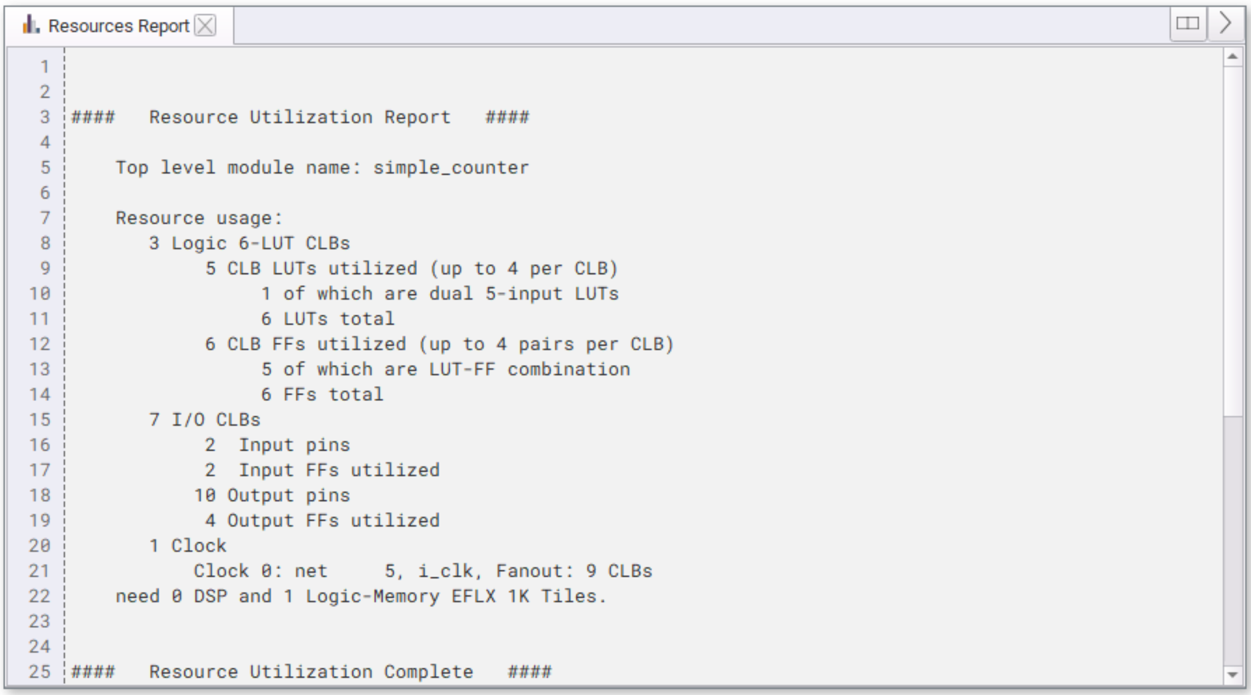
Informational elements

In addition, you can manually upload the configurations by accessing *Load custom PNR* in the main menu, *Tools* → *Floorplan*.

Resources Report

After you perform the bitstream generation procedure, a full report of the used resources is generated. This report shows the amount of CLBs, FFs, I/Os, and LUTs used for the design synthesis. You can also see the list of utilized resources on the control panel. To open the *Resources Report* window, click the corresponding button on the toolbar or reach the main menu, *Window* →

Resources Report.


Resources Report

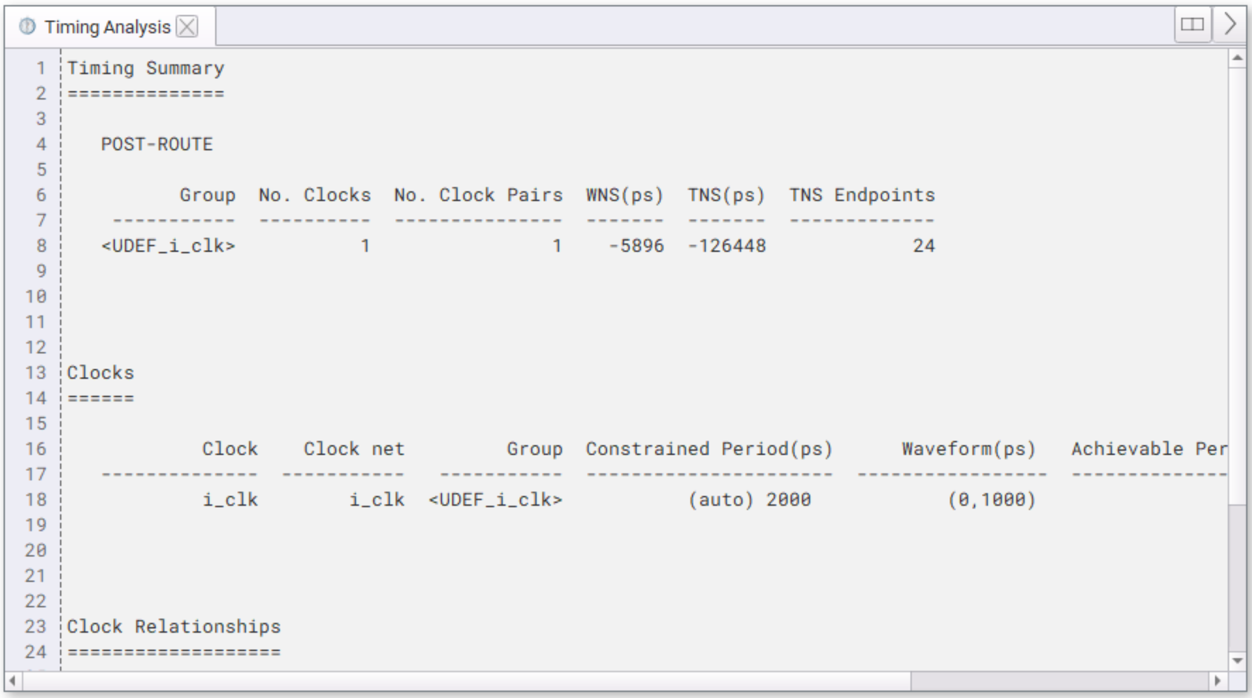
```

####   Resource Utilization Report   ####

    Top level module name: simple_counter

    Resource usage:
       3 Logic 6-LUT CLBs
            5 CLB LUTs utilized (up to 4 per CLB)
                 1 of which are dual 5-input LUTs
                 6 LUTs total
            6 CLB FFs utilized (up to 4 pairs per CLB)
                 5 of which are LUT-FF combination
                 6 FFs total
       7 I/O CLBs
            2  Input pins
            2  Input FFs utilized
           10 Output pins
            4 Output FFs utilized
       1 Clock
           Clock 0: net     5, i_clk, Fanout: 9 CLBs
    need 0 DSP and 1 Logic-Memory EFLX 1K Tiles.


####   Resource Utilization Complete   ####
```

Resources Report

Timing Analysis

The tool is designed to extract timing and check for any timing violations associated with any of the internal registers. The results show whether all set-up, hold, and pulse-width time is being met.

You can find the tool on the toolbar, or by clicking *Window* → *Timing Analysis* in the main menu.


Timing Analysis

```
Timing Summary
==============

   POST-ROUTE

          Group  No. Clocks  No. Clock Pairs  WNS(ps)  TNS(ps)  TNS Endpoints
   -----------  ----------  ---------------  -------  -------  -------------
   <UDEF_i_clk>          1                1    -5896  -126448             24




Clocks
======

            Clock    Clock net        Group  Constrained Period(ps)       Waveform(ps)   Achievable Per
   --------------  -----------  -----------  ---------------------  -----------------   --------------
            i_clk        i_clk  <UDEF_i_clk>           (auto) 2000            (0,1000)




Clock Relationships
===================
```

Timing Analysis

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

See the list of the supported SDC commands and arguments:

- `create_clock` — creates clock object and defines its characteristics
  - `-name clockName [-add] {objectList} | -name clockName [-add] [{objectList}] | [-name clockName [-add]] {objectList}`
  - `-period value`
  - `[-waveform {riseValue fallValue}]`
  - `[-disable]`
  - `[-comment commentString]`
- `set_clock_groups` — defines relationship between groups of clocks
  - `-asynchronous | -physically_exclusive | -logically_exclusive`
  - `[-name clockGroupname]`
  - `-group {clockList} [-group {clockList} <85> ]`
  - `-derive`
  - `[-disable]`
  - `[-comment commentString]`
- `set_false_path` — sets specific timing paths as being false and excluded from the timing analysis
  - `[-setup | -hold]`
  - `[-from {objectList}]`
  - `[-through {objectList} [-through {objectList} ...] ]`
  - `[-to {objectList}]`
  - `[-forward_propagate]`
  - `[-disable]`
  - `[-comment commentString]`
- `set_multicycle_path` — determines how many clock cycles a path can take

  - [-start | -end]
  - [-setup | -hold]
  - [-from {objectList}]
  - [-through {objectList} [-through {objectList} ...]]
  - [-to {objectList}]
  - pathMultiplier
  - [-disable]
  - [-comment commentString]

- set_hierarchy_separator — defines the character used to separate different levels of hierarchy in the design
  - {sepchar} | sepchar

The commands below have basic support:

- get_cells
- get_ports
- get_nets
- get_pins
- get_clocks

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

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. 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. Clicking the block within the FPGA core opens the info panel with the coordinates data in the first line.

- `create_cluster_only_placement_group name_of_the_group` — creates a specialized placement group that is restricted to clustered placement only. Upon 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 1: 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.

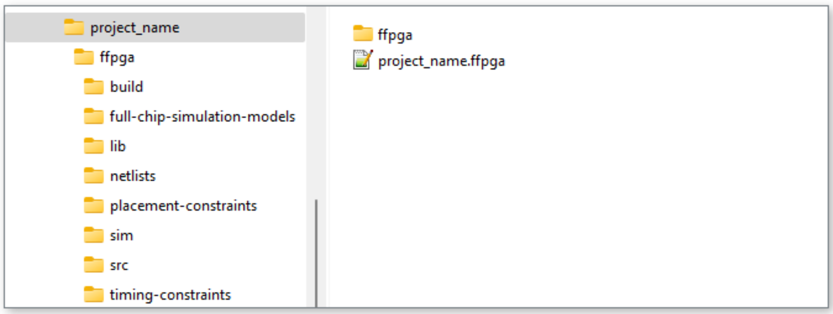
Note 2: To include all cells or ports from the design, use the * wildcard character instead of listing specific names. See the example:

- `add_to_placement_group name_of_the_group [get_cells *]`

- `add_to_placement_group name_of_the_group [get_ports *]`

2.5.8 Project directory structure

Once you create 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:



```
project_name
  ffpga
    build
    full-chip-simulation-models
    lib
    netlists
    placement-constraints
    sim
    src
    timing-constraints
```

```
ffpga
project_name.ffpga
```

FPGA project structure

- *lib* — modules from the *Modules Library*

- *sim* — testbench files

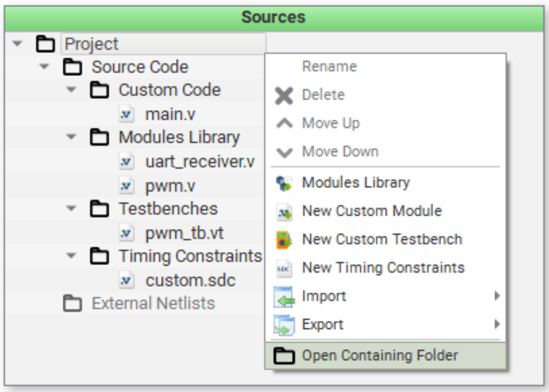
- *src* — *main*, *mcm_generated* (created by clicking *Generate Verilog* in *Macrocell Editor*), and imported modules

- *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.



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, you can see the description of the most important files present 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 to program the flash
  - *FPGA_bitstream_MCU* — the bitstream sequence that can be used for MCU programming
- *ta* — stores log files containing potential warnings encountered during the 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 upon performing the chip/flash procedures
- *io_spec_in.txt* — lists the I/O ports specification added in the *I/O Planner*, which is created right upon clicking *Generate Bitstream*
- *PNR_IO.log* — I/O ports mapping file generated after successful bitstream generation procedure
- *netlist.edif* — the result of Yosys data processing, describing the components and connectivity within the source design. You can also import an external netlist from another software/synthesis tool
- *PNR_PACK_PLACE.log* — source data for building a floorplan

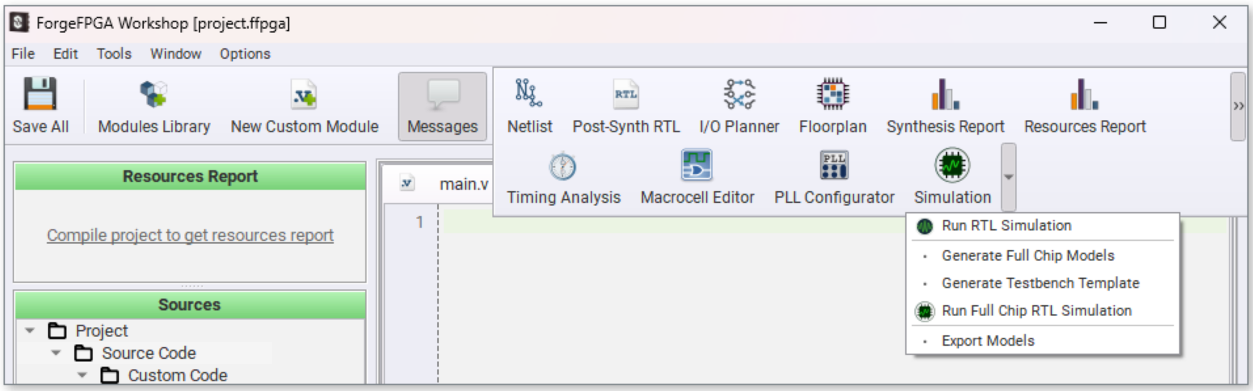
- *post_synth_results.v* — Yosys output data in the Verilog code format
- *resource-utilization-report.log* — information about the resources amount required for your design
- *simulation* — folder containing simulation results

2.5.9 Simulation

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

FPGA Editor works in correspondence with 3rd party software for simulating the testbench, called Icarus Verilog, and for verifying the functionality of the design by viewing simulation results we use GTKWave. Please refer to the How to section for the installation and quick start guide of the additional software.

Run the process via *Simulation* button on the toolbar or via *Tools* → *Simulation*.

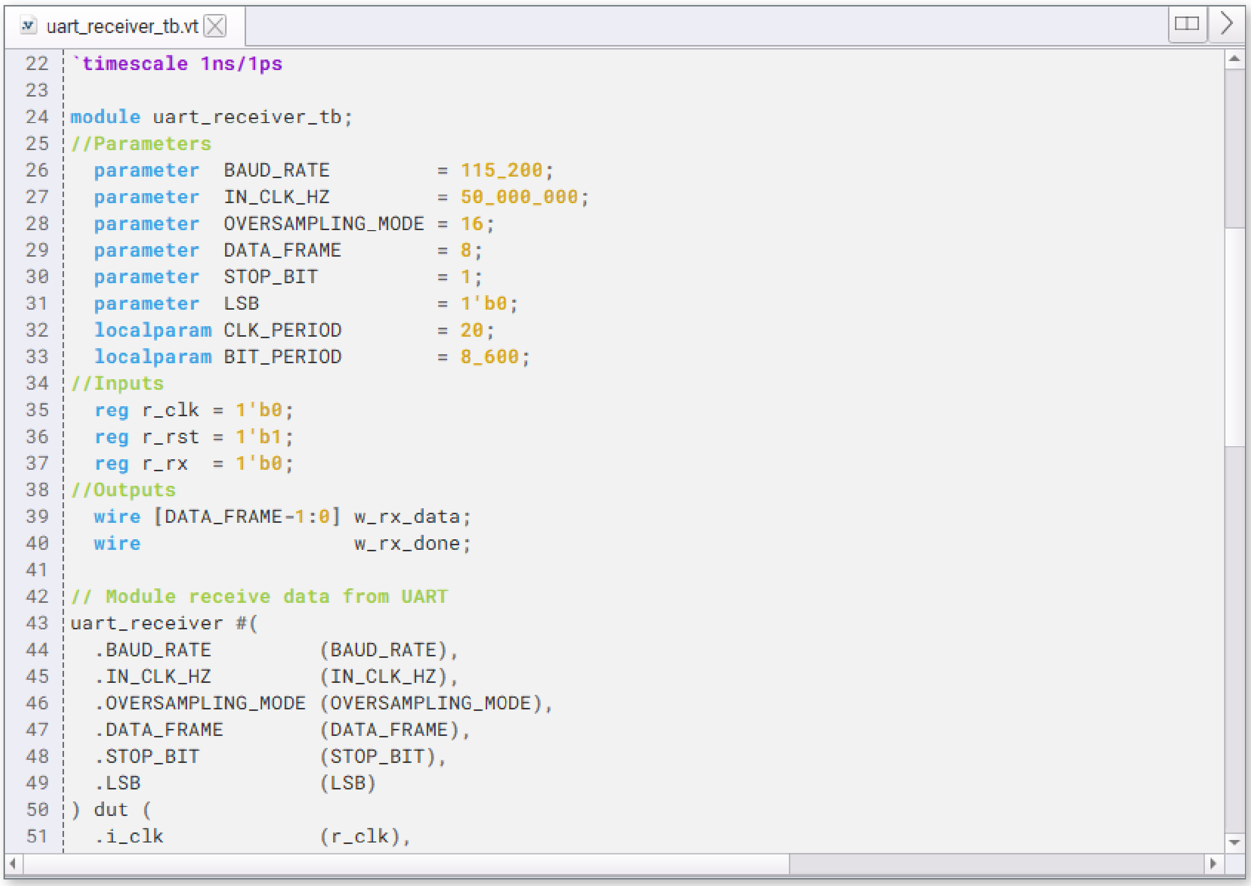


FPGA simulation on the toolbar

RTL Simulation executes the RTL only. *Full Chip RTL Simulation* allows you to perform extended behavioral simulation of your design, which simulates not only the logic behavior of the code but also the behavior of the FPGA Core's periphery (including GPIOs, BRAMs, LVDS, OSC, PLLs, etc.). This simulation type also includes basic delays of peripheral circuits, producing results that more accurately reflect the device's real behavior.

RTL Simulation

- Writing a testbench — to work with the *RTL Simulation* you should have a testbench module for the FPGA design. You can add a testbench from the *Modules Library*, or import a module from the main menu, *File → New Custom Testbench*.



```
`timescale 1ns/1ps

module uart_receiver_tb;
//Parameters
  parameter  BAUD_RATE          = 115_200;
  parameter  IN_CLK_HZ          = 50_000_000;
  parameter  OVERSAMPLING_MODE  = 16;
  parameter  DATA_FRAME         = 8;
  parameter  STOP_BIT           = 1;
  parameter  LSB                = 1'b0;
  localparam CLK_PERIOD         = 20;
  localparam BIT_PERIOD         = 8_600;
//Inputs
  reg r_clk = 1'b0;
  reg r_rst = 1'b1;
  reg r_rx  = 1'b0;
//Outputs
  wire [DATA_FRAME-1:0] w_rx_data;
  wire                  w_rx_done;

// Module receive data from UART
uart_receiver #(
  .BAUD_RATE         (BAUD_RATE),
  .IN_CLK_HZ         (IN_CLK_HZ),
  .OVERSAMPLING_MODE (OVERSAMPLING_MODE),
  .DATA_FRAME        (DATA_FRAME),
  .STOP_BIT          (STOP_BIT),
  .LSB               (LSB)
) dut (
  .i_clk             (r_clk),
```

UART Receiver Testbench example from Modules Library

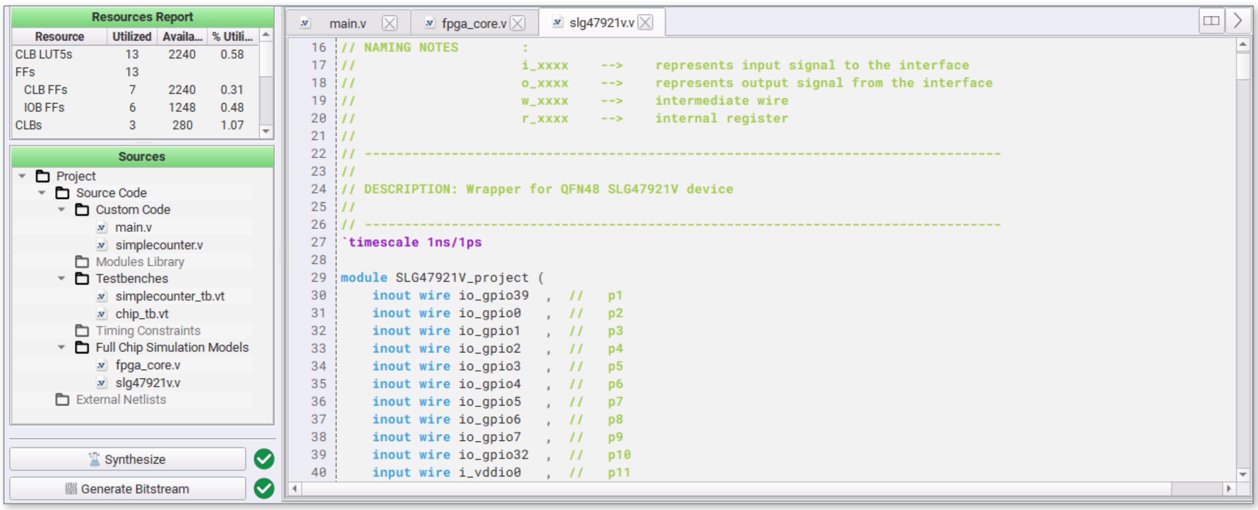
- Simulating a testbench — after adding the testbench, click the *Simulation → Run RTL Simulation* option on the toolbar to launch the Icarus Verilog and GTKWave or from the main menu, *Tools → Simulation → Run RTL Simulation*. The simulation stage handled by Icarus Verilog is performed in the background, while the GTKWave has a visual representation. If the selected testbench is correct and contains no syntax errors, then the GTKWave software will launch automatically.

Full Chip RTL Simulation

Performing a valid Full Chip Simulation requires the following steps and conditions:

- Your project should contain error-free RTL code
- Perform successful sythesis stage (netlist is used for top module ports)
- Synthesis results should be up to date with your design
- Perform successful bitstream generation stage (post PnR results are utilized to assign correct delays in the generated models)

- Perform models generation (*Simulation* → *Generate Full Chip Models*). Find them added in the *Sources* tree



Resource	Utilized	Availa...	% Utili...
CLB LUT5s	13	2240	0.58
FFs	13		
CLB FFs	7	2240	0.31
IOB FFs	6	1248	0.48
CLBs	3	280	1.07

```
// NAMING NOTES         :
//                      i_xxxx    -->    represents input signal to the interface
//                      o_xxxx    -->    represents output signal from the interface
//                      w_xxxx    -->    intermediate wire
//                      r_xxxx    -->    internal register
//
// ------------------------------------------------------------------------------
//
// DESCRIPTION: Wrapper for QFN48 SLG47921V device
//
// ------------------------------------------------------------------------------
`timescale 1ns/1ps

module SLG47921V_project (
    inout wire io_gpio39  ,  //   p1
    inout wire io_gpio0   ,  //   p2
    inout wire io_gpio1   ,  //   p3
    inout wire io_gpio2   ,  //   p4
    inout wire io_gpio3   ,  //   p5
    inout wire io_gpio4   ,  //   p6
    inout wire io_gpio5   ,  //   p7
    inout wire io_gpio6   ,  //   p8
    inout wire io_gpio7   ,  //   p9
    inout wire io_gpio32  ,  //   p10
    input wire i_vddio0   ,  //   p11
```

Full Chip Simulation models in the Sources tree

Note: Generated models are not the part of the design and are not synthesized. They are used for full chip simulation only.

- Optional: perform template testbench generation (*Simulation* → *Generate Testbench Template*)

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

Models generated for currently opened project can be exported to one dump file. It can be imported and used, for example, to perform two chips simulation in another projects (*Simulation* → *Export Models*).

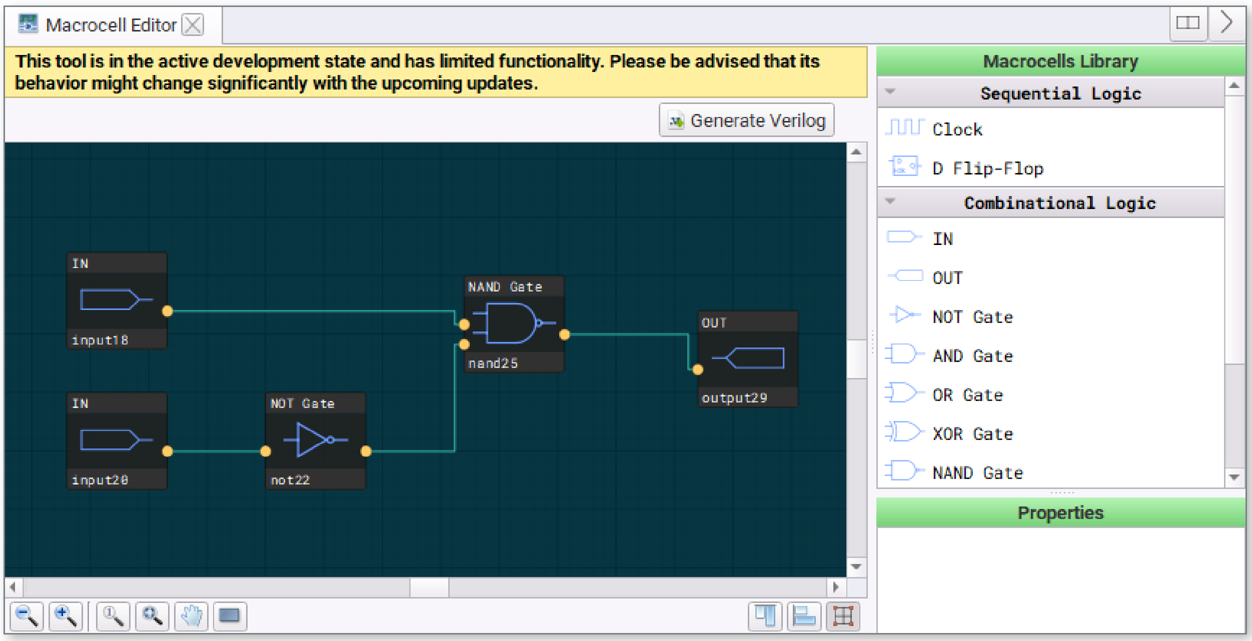
In case of simulation failure, check the *Messages* panel (*General Log* or *Issues* tab) and make necessary changes.

FPGA Editor ensures you can keep working from the last saved state of simulation results. Once you make necessary configurations (e.g. add signals, adjust their graphical representation, etc.), save the progress in the GTKWave tool. Next time you simulate the same testbench, you will start from where you left off the previous time.

2.5.10 Macrocell Editor

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

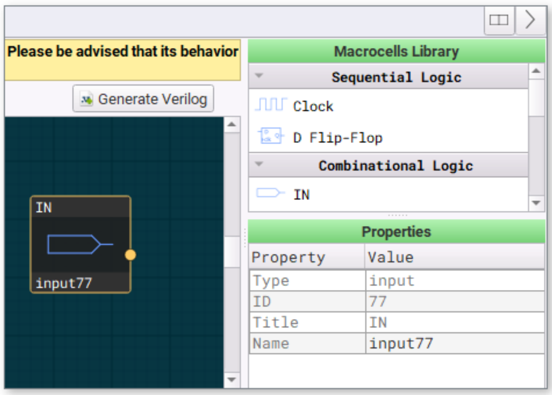
Editor.



Macrocell Editor

The *Macrocells Library* panel contains the list of components, which you can use for circuit design. The components are grouped as follows: *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.

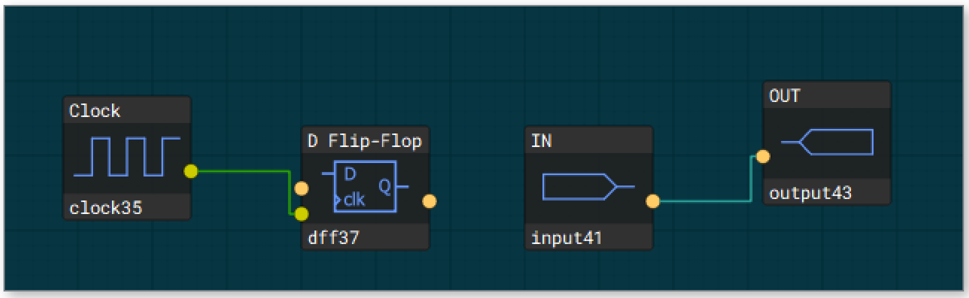
The *Properties* panel shows the details for one selected macrocell or connection. The *Name* field is editable (double-click the field or the macrocell).



Property	Value
Type	input
ID	77
Title	IN
Name	input77

Macrocells Library and Properties panels

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.



Connections between different port types

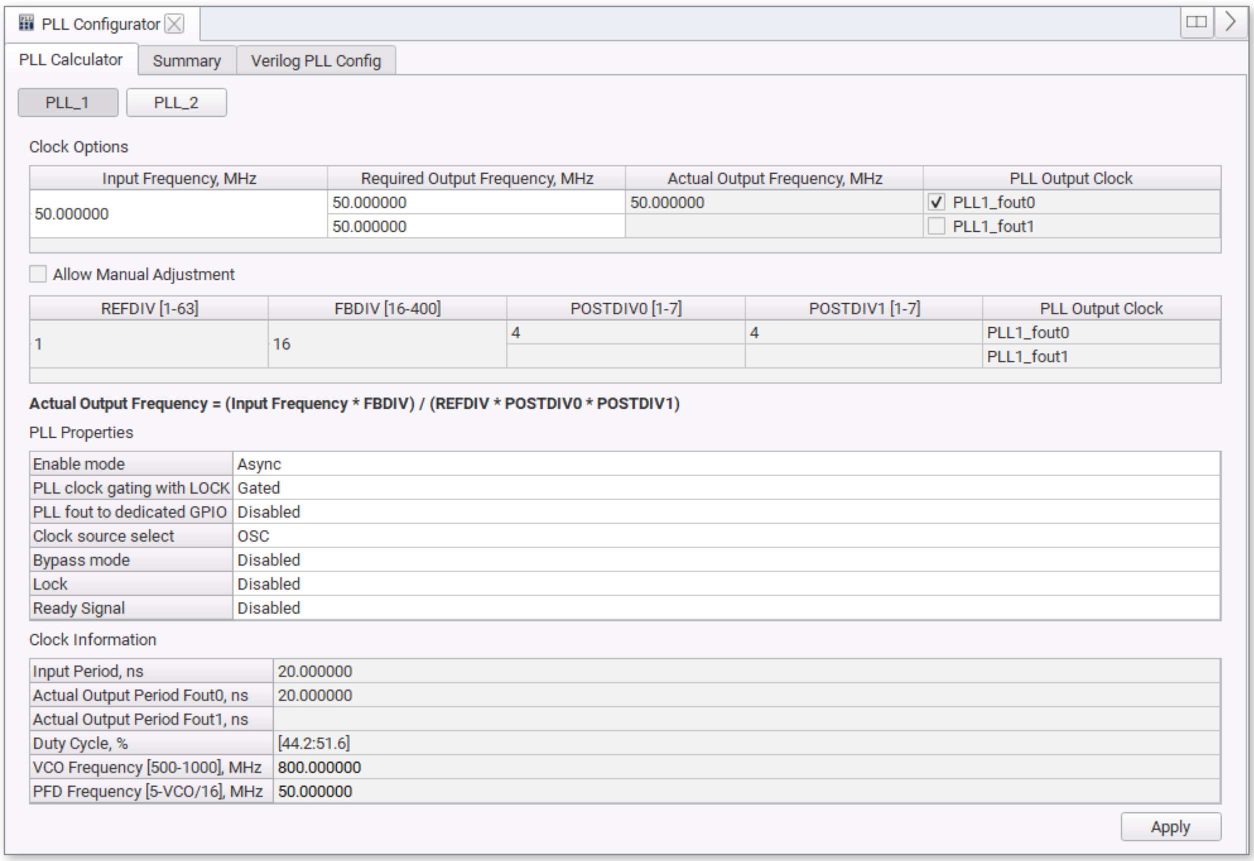
To generate the Verilog code based on the created design, click the *Generate Verilog* button. The created Verilog code is listed as *mcm_generated* under the *Custom Code* section of the control panel. You can use this code for the synthesis along with other modules added in the design.

2.5.11 PLL Configurator

The *PLL Configurator* helps to determine the parameters for the desired output signal or dividers. You can access it through the *FPGA Editor* → *PLL Configurator*.

PLL Calculator tab

The main tab allows you to configure the parameters and set them 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.


PLL Configurator

PLL Calculator | Summary | Verilog PLL Config

PLL_1 | PLL_2

Clock Options

Input Frequency, MHz	Required Output Frequency, MHz	Actual Output Frequency, MHz	PLL Output Clock
50.000000	50.000000	50.000000	✓ PLL1_fout0
	50.000000		☐ PLL1_fout1

☐ Allow Manual Adjustment

REFDIV [1-63]	FBDIV [16-400]	POSTDIV0 [1-7]	POSTDIV1 [1-7]	PLL Output Clock
1	16	4	4	PLL1_fout0
				PLL1_fout1

Actual Output Frequency = (Input Frequency * FBDIV) / (REFDIV * POSTDIV0 * POSTDIV1)

PLL Properties

Enable mode	Async
PLL clock gating with LOCK	Gated
PLL fout to dedicated GPIO	Disabled
Clock source select	OSC
Bypass mode	Disabled
Lock	Disabled
Ready Signal	Disabled

Clock Information

Input Period, ns	20.000000
Actual Output Period Fout0, ns	20.000000
Actual Output Period Fout1, ns	
Duty Cycle, %	[44.2:51.6]
VCO Frequency [500-1000], MHz	800.000000
PFD Frequency [5-VCO/16], MHz	50.000000

PLL Calculator tab

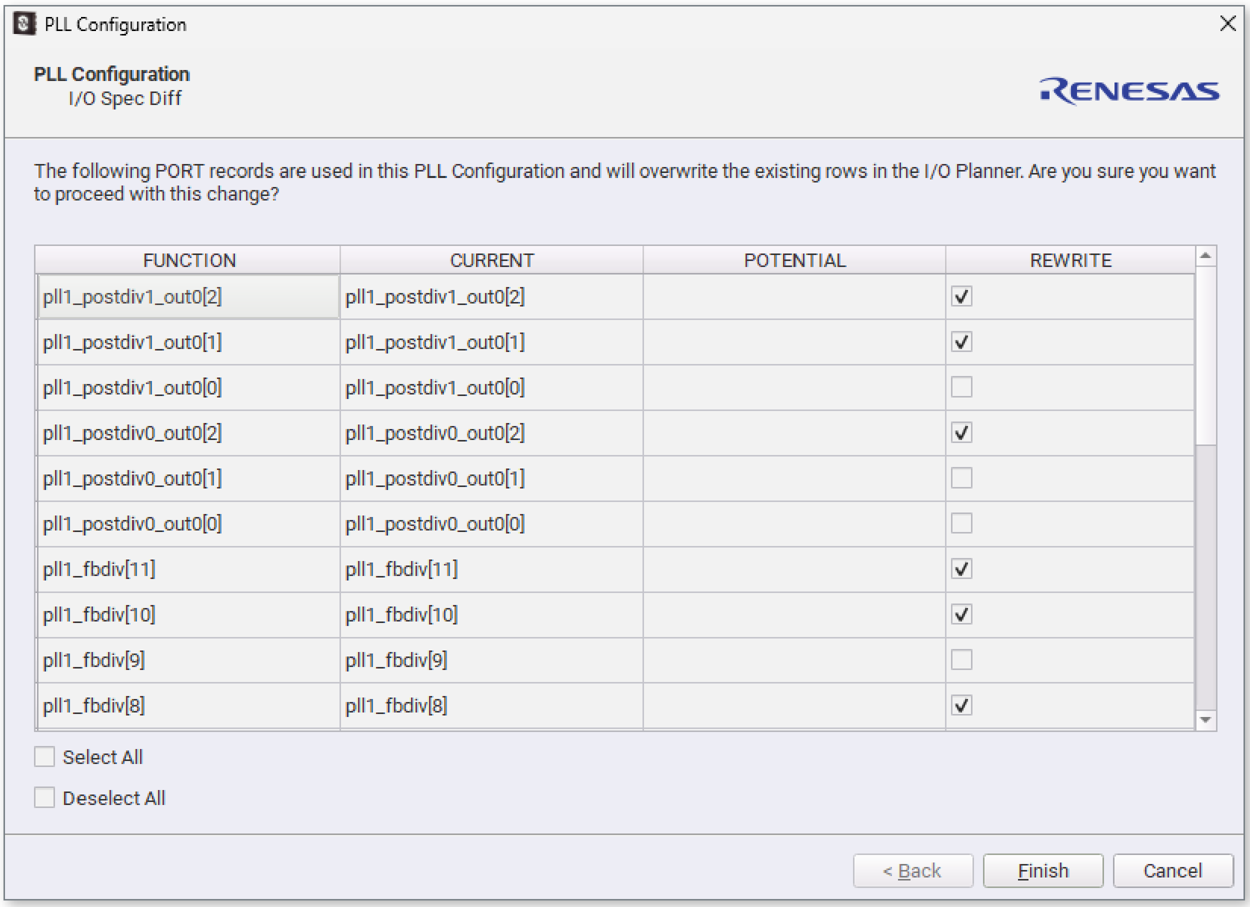
The *PLL Configurator* tool has the following calculation modes:

- 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 via 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.



The following PORT records are used in this PLL Configuration and will overwrite the existing rows in the I/O Planner. Are you sure you want to proceed with this change?

FUNCTION	CURRENT	POTENTIAL	REWRITE
pll1_postdiv1_out0[2]	pll1_postdiv1_out0[2]		✓
pll1_postdiv1_out0[1]	pll1_postdiv1_out0[1]		✓
pll1_postdiv1_out0[0]	pll1_postdiv1_out0[0]		☐
pll1_postdiv0_out0[2]	pll1_postdiv0_out0[2]		✓
pll1_postdiv0_out0[1]	pll1_postdiv0_out0[1]		☐
pll1_postdiv0_out0[0]	pll1_postdiv0_out0[0]		☐
pll1_fbdiv[11]	pll1_fbdiv[11]		✓
pll1_fbdiv[10]	pll1_fbdiv[10]		✓
pll1_fbdiv[9]	pll1_fbdiv[9]		☐
pll1_fbdiv[8]	pll1_fbdiv[8]		✓

☐ Select All

☐ Deselect All

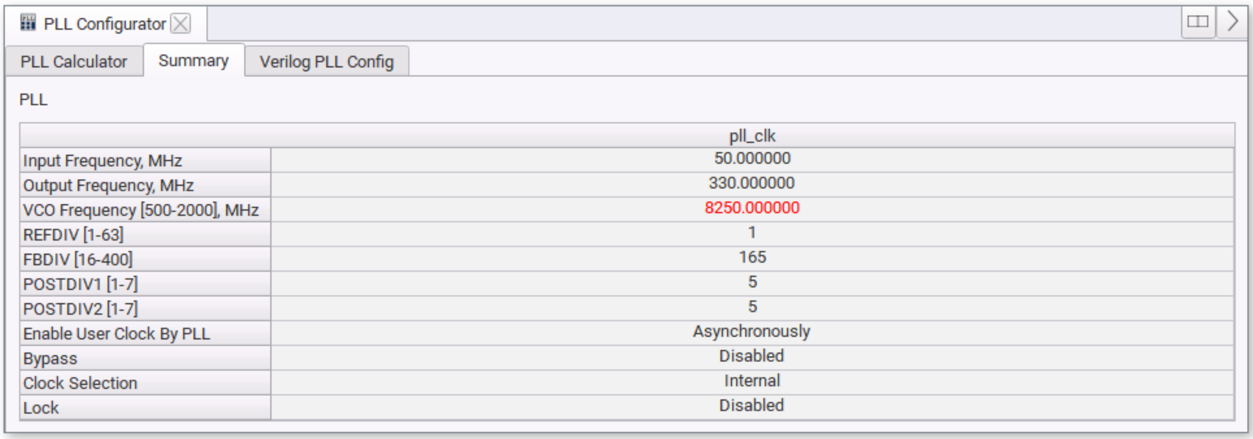
Conflict resolution window

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

The *Clock Information* table values are based on the set data. The results exceeding the range are highlighted in red.

Summary tab

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



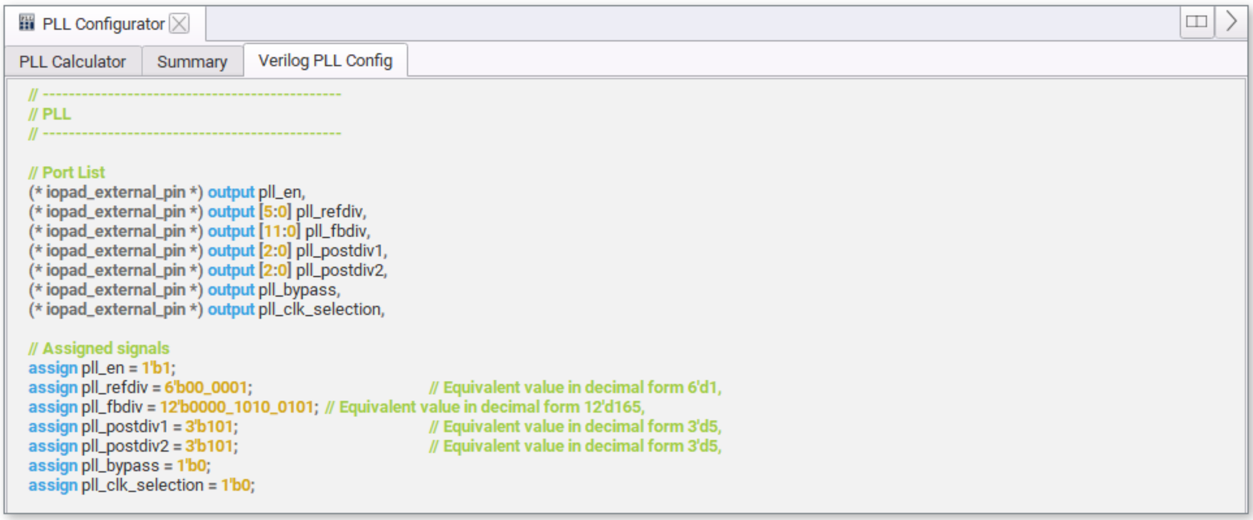
PLL

	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

Summary tab

Verilog PLL Config tab

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

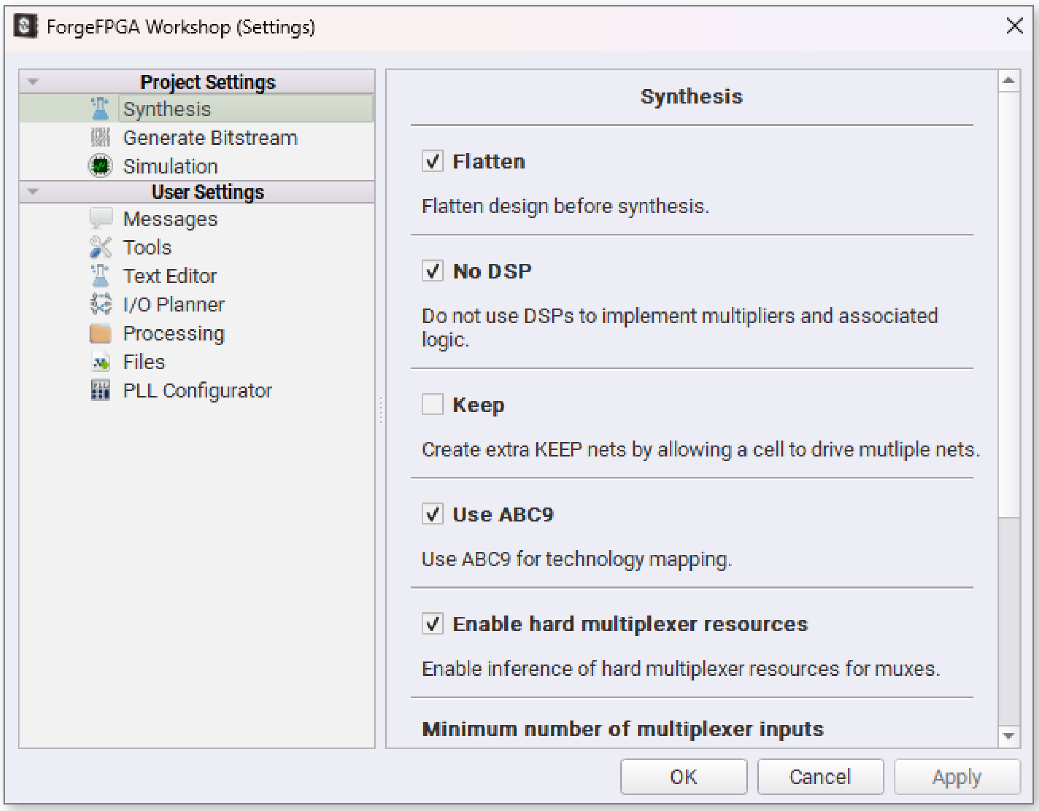


```
// ---------------------------------------------
// PLL
// ---------------------------------------------

// Port List
(* iopad_external_pin *) output pll_en,
(* iopad_external_pin *) output [5:0] pll_refdiv,
(* iopad_external_pin *) output [11:0] pll_fbdiv,
(* iopad_external_pin *) output [2:0] pll_postdiv1,
(* iopad_external_pin *) output [2:0] pll_postdiv2,
(* iopad_external_pin *) output pll_bypass,
(* iopad_external_pin *) output pll_clk_selection,

// Assigned signals
assign pll_en = 1'b1;
assign pll_refdiv = 6'b00_0001;                      // Equivalent value in decimal form 6'd1,
assign pll_fbdiv = 12'b0000_1010_0101; // Equivalent value in decimal form 12'd165,
assign pll_postdiv1 = 3'b101;                        // Equivalent value in decimal form 3'd5,
assign pll_postdiv2 = 3'b101;                        // Equivalent value in decimal form 3'd5,
assign pll_bypass = 1'b0;
assign pll_clk_selection = 1'b0;
```

Verilog PLL Config tab

2.5.12 Settings

The *ForgeFPGA Workshop Settings* window allows you to adjust multiple project and user settings.



Synthesis

✓ Flatten

Flatten design before synthesis.

✓ No DSP

Do not use DSPs to implement multipliers and associated logic.

Keep

Create extra KEEP nets by allowing a cell to drive mutliple nets.

✓ Use ABC9

Use ABC9 for technology mapping.

✓ Enable hard multiplexer resources

Enable inference of hard multiplexer resources for muxes.

Minimum number of multiplexer inputs

ForgeFPGA Workshop Settings

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

Project Settings

Within the *Project Settings* section, you can find a range of options to optimize your workflow. These options include:

- *Synthesize* — enables you to flatten your design before synthesis, manage DSP usage, generate additional KEEP nets, or control the use of ABC9. You can also add additional arguments to the synthesis procedure

- *Generate Bitstream* — involves several settings and processes. These include option to choose between the Place-and-Route compiler versions (where applicable). Another settings are *High-Density IO Packing*, *High-Density Packing Logic*, *Clock-Concurrent Optimization (CCopt)*, the *Place-and-Trial routing*, adjustments to the *Place-and-Trial routing iteration count*, and the *Maximum Routing Iterations* parameter. Here, you can also set the number of CPU cores used for the routing stage. Additional arguments can be also added to the Generate Bitstream procedure

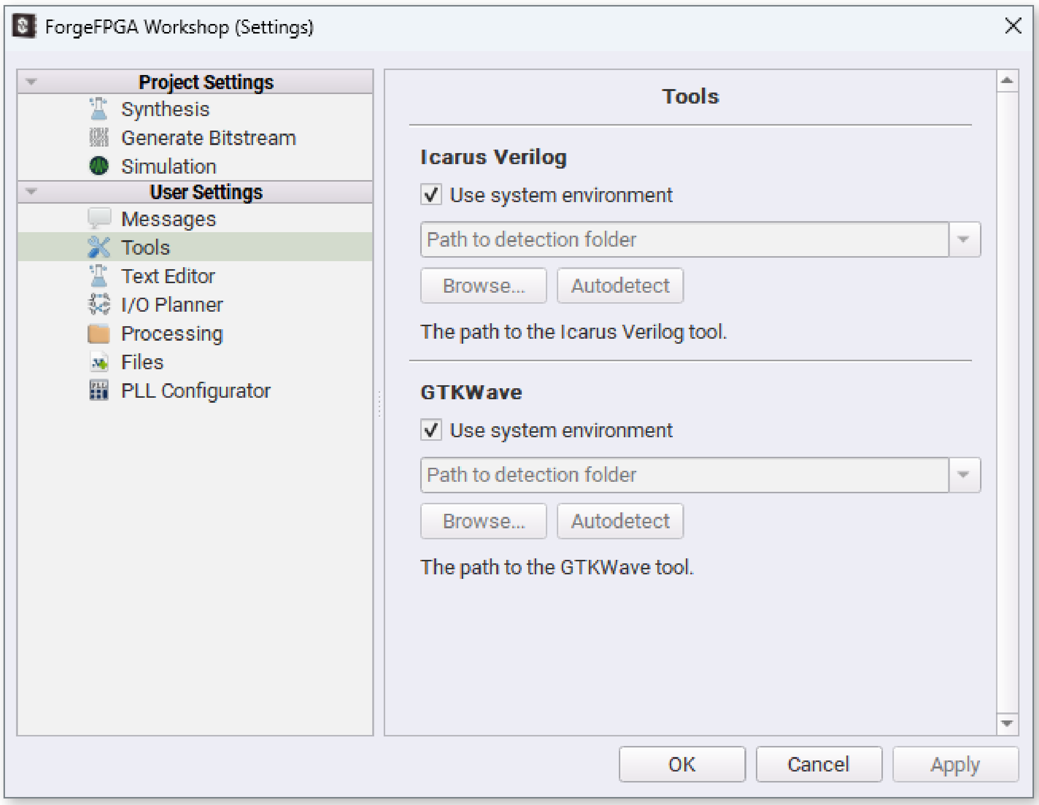
- *Simulation* — allows you to configure the Wavedump output format to either .vcd or .fst

User Settings

In the *User Settings* category, you can customize:

- *Messages Panel* — offers you the flexibility to decide whether to clear *Synthesis/Bitsteam Log* each time you start a new procedure

- *Tools* — gives you the ability to modify the accessibility settings for the *Icarus Verilog tool* and the *GTKWave tool*. Here, you can either specify the path to the folders for detection or choose Autodetect to let the software locate the files automatically. You can also switch to using the system environment



The path to the Icarus Verilog tool.

The path to the GTKWave tool.

ForgeFPGA Workshop Settings Tools Tab

- *Text Editor* — allows you to define 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 you wish to save the latest changes before starting the procedures

- *Files* — allows you to choose 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 descriptions of the options to avoid errors and improve your design.

3 Devices

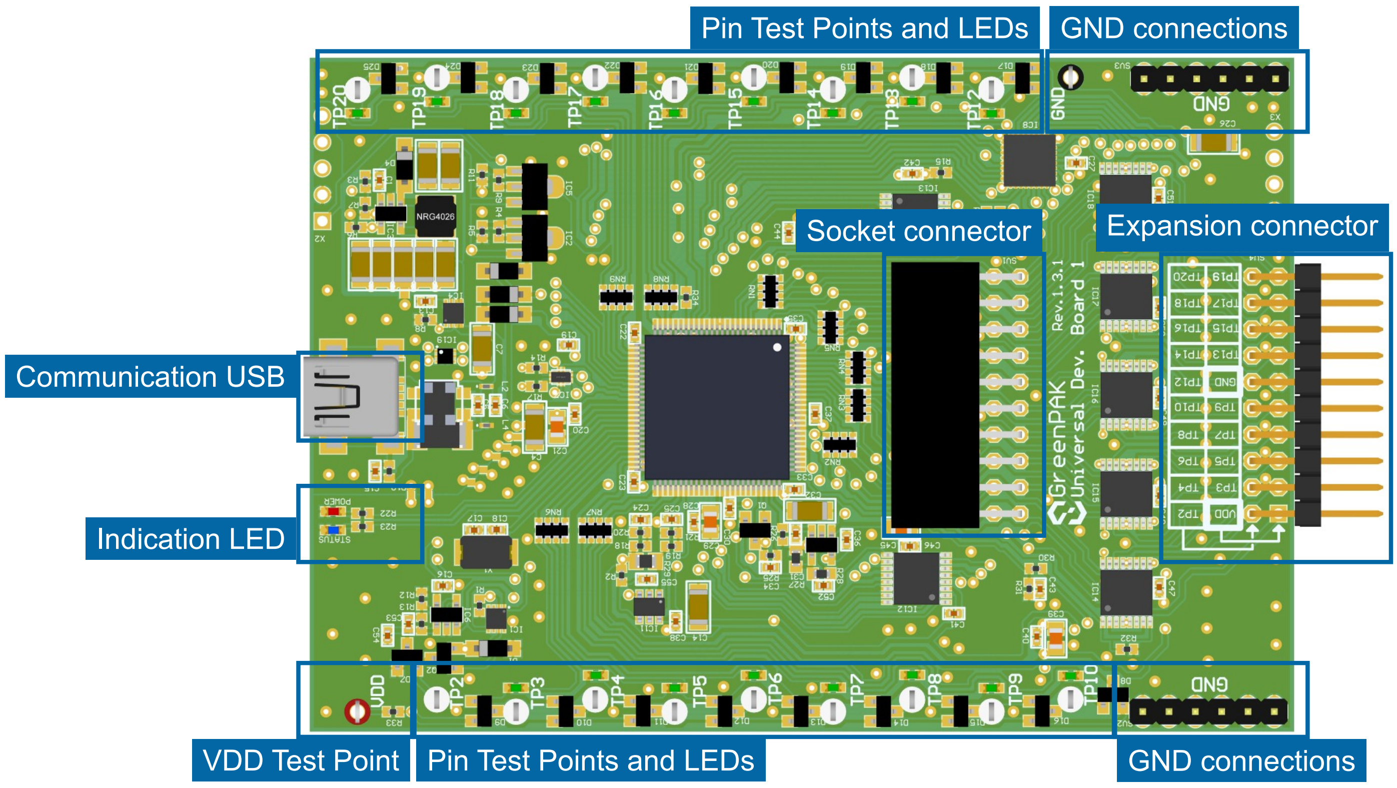
The *Go Configure Software Hub* allows project debugging using diverse hardware platforms to provide communication between a specific chip and the software. Depending on the device you choose for your project, the software adapts to support different platforms.

In this chapter, you will be introduced to the supported development platforms tailored for *GreenPAK*, *ForgeFPGA*, and *Power GreenPAK* devices. Each platform will be presented with an overview, functional descriptions, and insights into its software representation.

3.1 GreenPAK devices

3.1.1 GreenPAK Advanced Development Platform

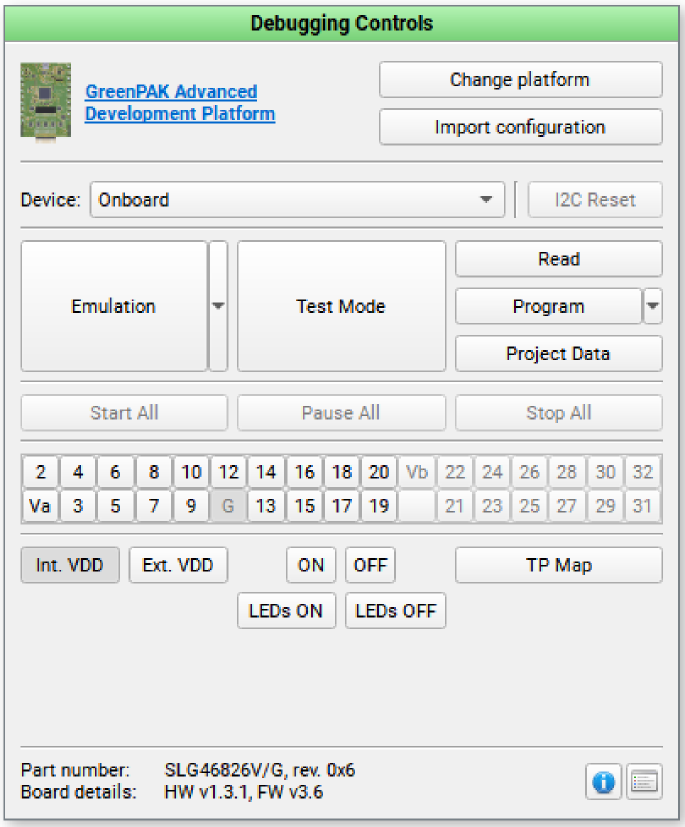
- Programming and emulation for *GreenPAK* devices
- 18 individually configurable Test Points (TPs):
  - Analog, digital and I2C generators support
  - Onboard LED state indication
  - Pull-up, Pull-down, GND, VDD, Hi-Z
  - Programmable software button
  - Floating hooks on each Test Point for oscilloscope 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)



GreenPAK Advanced Development Platform

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

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



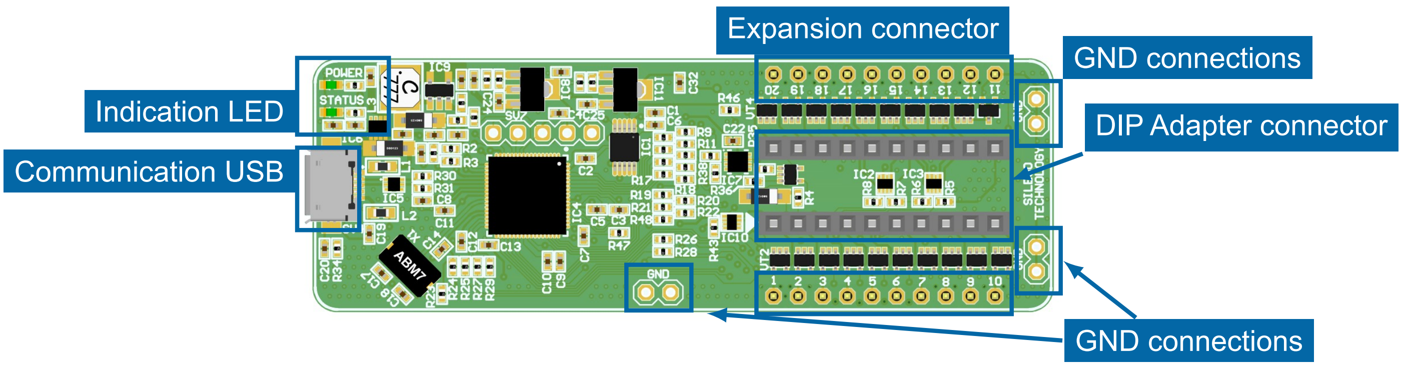
Part number: SLG46826V/G, rev. 0x6
Board details: HW v1.3.1, FW v3.6

GreenPAK Advanced Development Platform 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 Platform

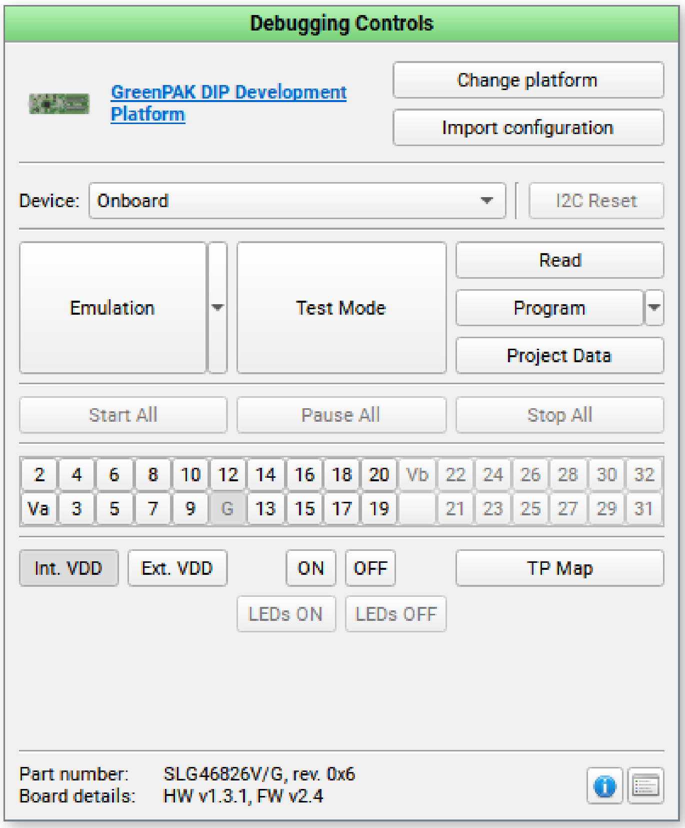
- Programming and emulation for *GreenPAK* devices
- 18 individually configurable Test Points (TP):
  - Pull-up, Pull-down, GND, VDD, Hi-Z
  - Programmable software button
- Dual VDD IC's support
- DIP adapter support
- USB interface for power and communication (micro-B connector)
- In-System Debugging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



GreenPAK DIP (Dual In-Line Package) Development Platform

The *GreenPAK DIP Development Platform* *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



Device: Onboard

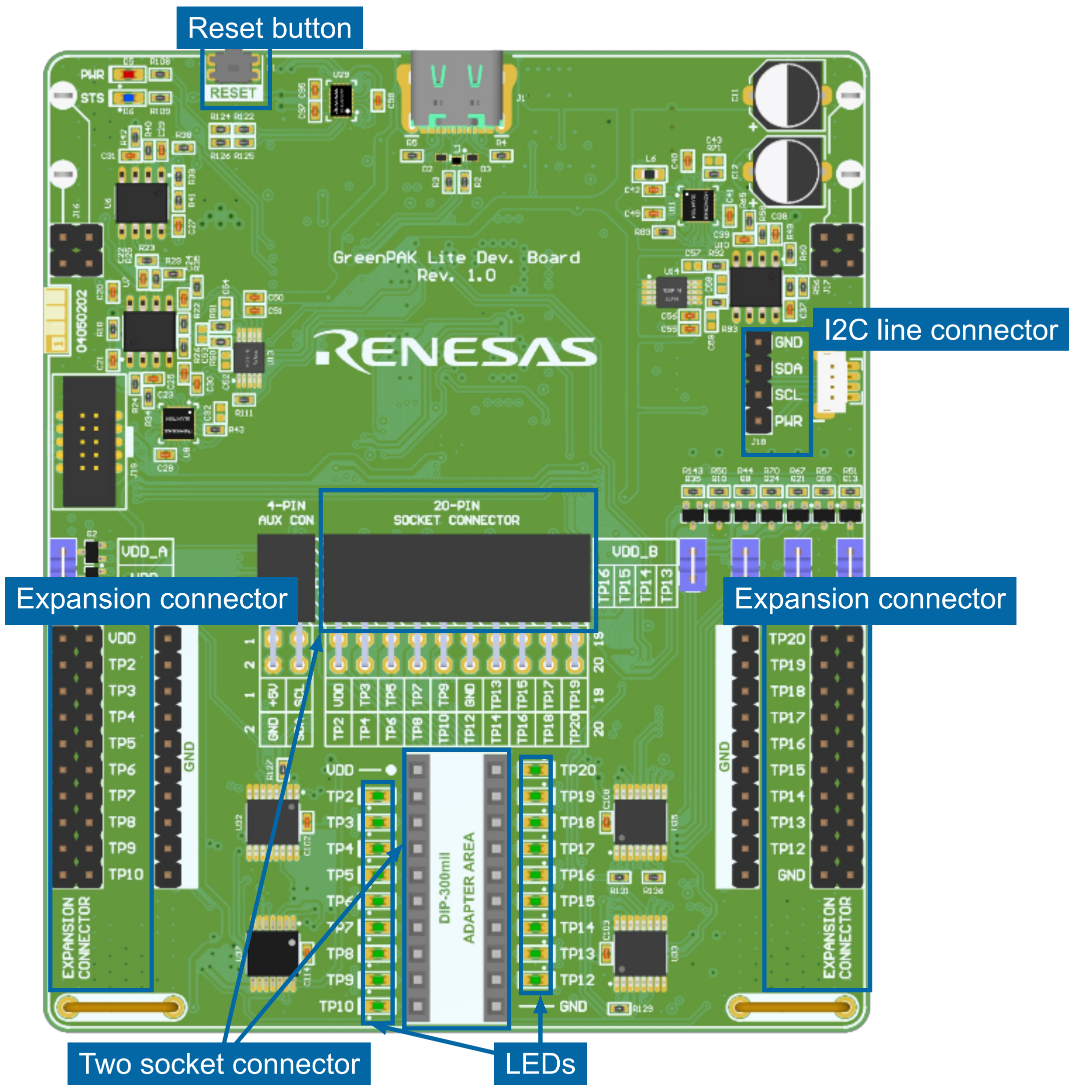
Part number: SLG46826V/G, rev. 0x6
Board details: HW v1.3.1, FW v2.4

GreenPAK DIP Development Platform 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 Platform

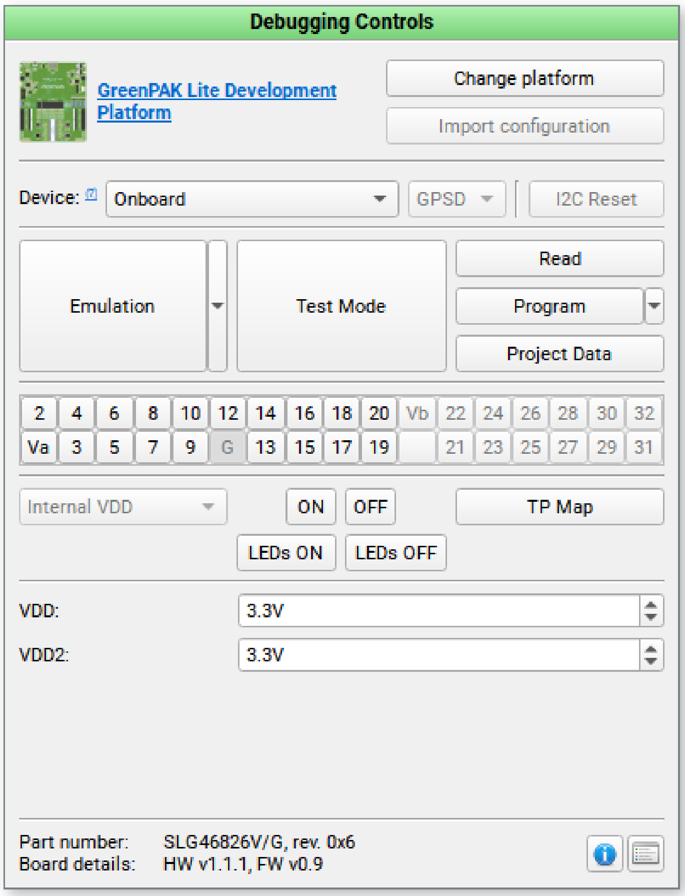
- Programming and emulation for *GreenPAK* devices
- 18 individually configurable Test Points (TP):
  - Onboard LED state indication
  - Pull-up, Pull-down, GND, VDD, Hi-Z
  - Programmable software button
  - Four floating hooks for oscilloscope connection
- Dual VDD IC's support
- DIP and socket adapters 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)



GreenPAK Lite Development Platform

The *GreenPAK Lite Development Platform* *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



VDD: 3.3V
VDD2: 3.3V

Part number: SLG46826V/G, rev. 0x6
Board details: HW v1.1.1, FW v0.9

GreenPAK Lite Development Platform 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

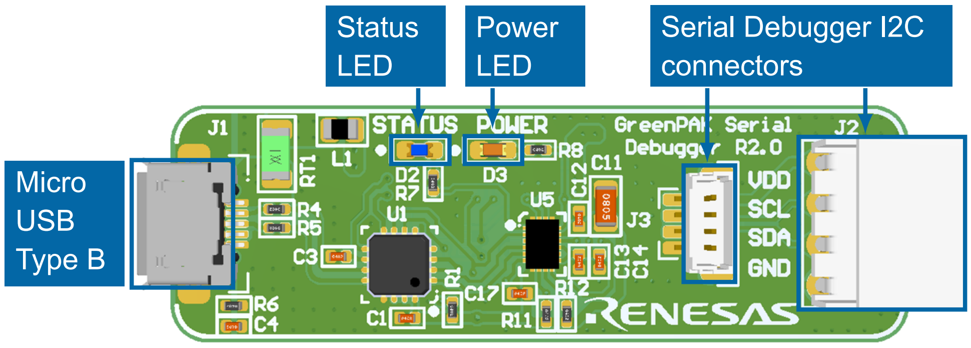
➤ In-System Programming for Multiple Programmable *GreenPAK* devices (e.g. SLG46824, SLG46826, SLG47004)

➤ 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

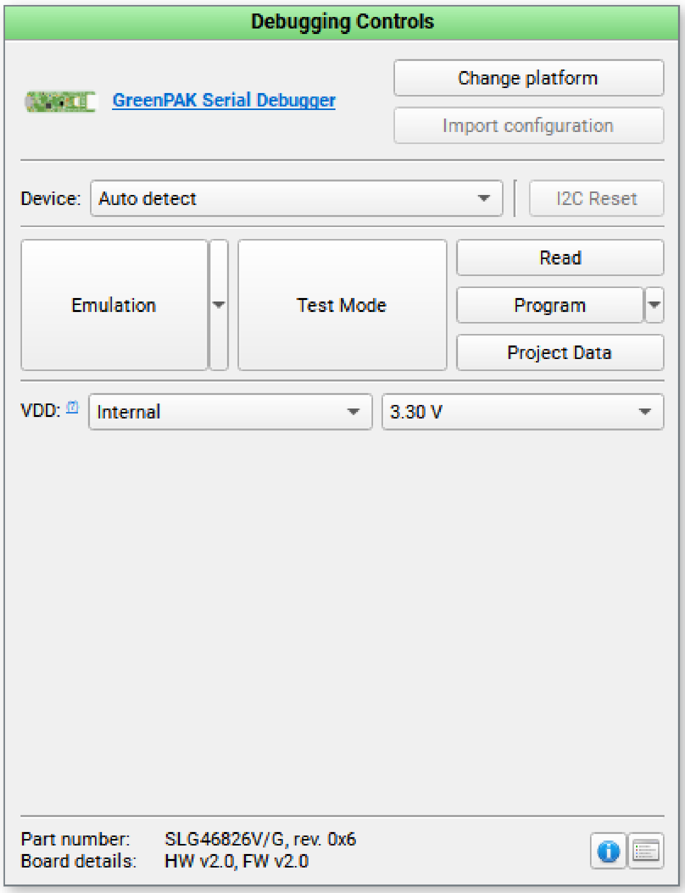
➤ Internal and external power supply for I2C



GreenPAK Serial Debugger

The *GreenPAK Serial Debugger Platform* *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



Device: Auto detect

VDD: Internal 3.30 V

Part number: SLG46826V/G, rev. 0x6
Board details: HW v2.0, FW v2.0

GreenPAK Serial Debugger Platform 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

Advanced, DIP, or Lite platforms

- ➤ Connect the socket with an external chip to a *Development Board* via pins:
  - Corresponding *I2C* pins (*SCL*, *SDA*)
  - *VDD* pin
  - *GND* pin

  For the *DIP/Advanced/Lite Development Board*: connect a socket with an external chip to the *Expansion Connector* pins:
- ➤ 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:** 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

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

Note: *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/Test mode* operations for a chip with applied external voltage are running.

Serial Debugger platform

- ➤ Connect a socket with an external chip to a *Development Board* via 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 *Serial Debugger platform*
- ➤ 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 *Serial Debugger* board
- *External* — chip is powered from an external power source

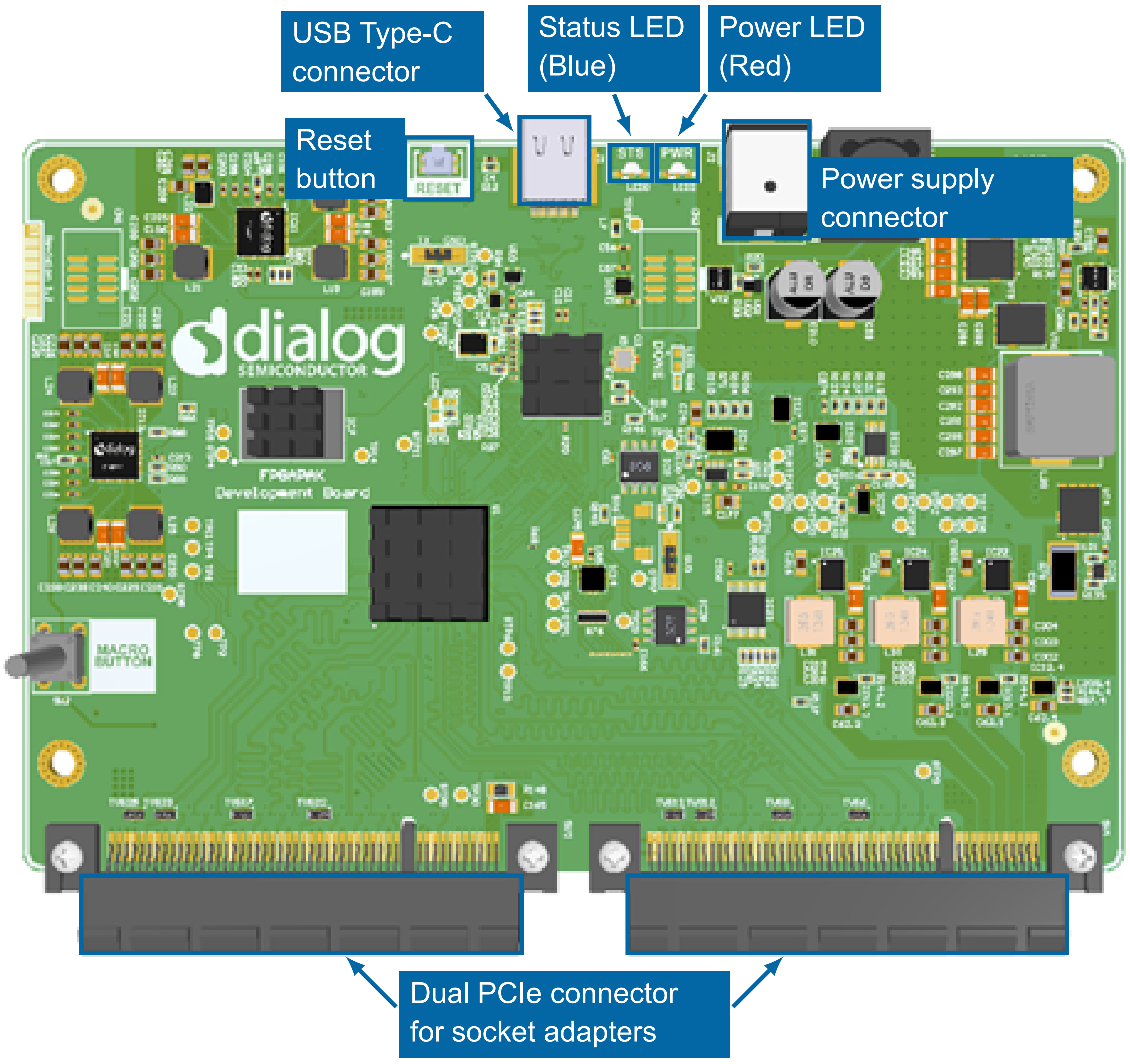
Once all steps are completed, the debugging controls become active.

3.2 ForgeFPGA devices

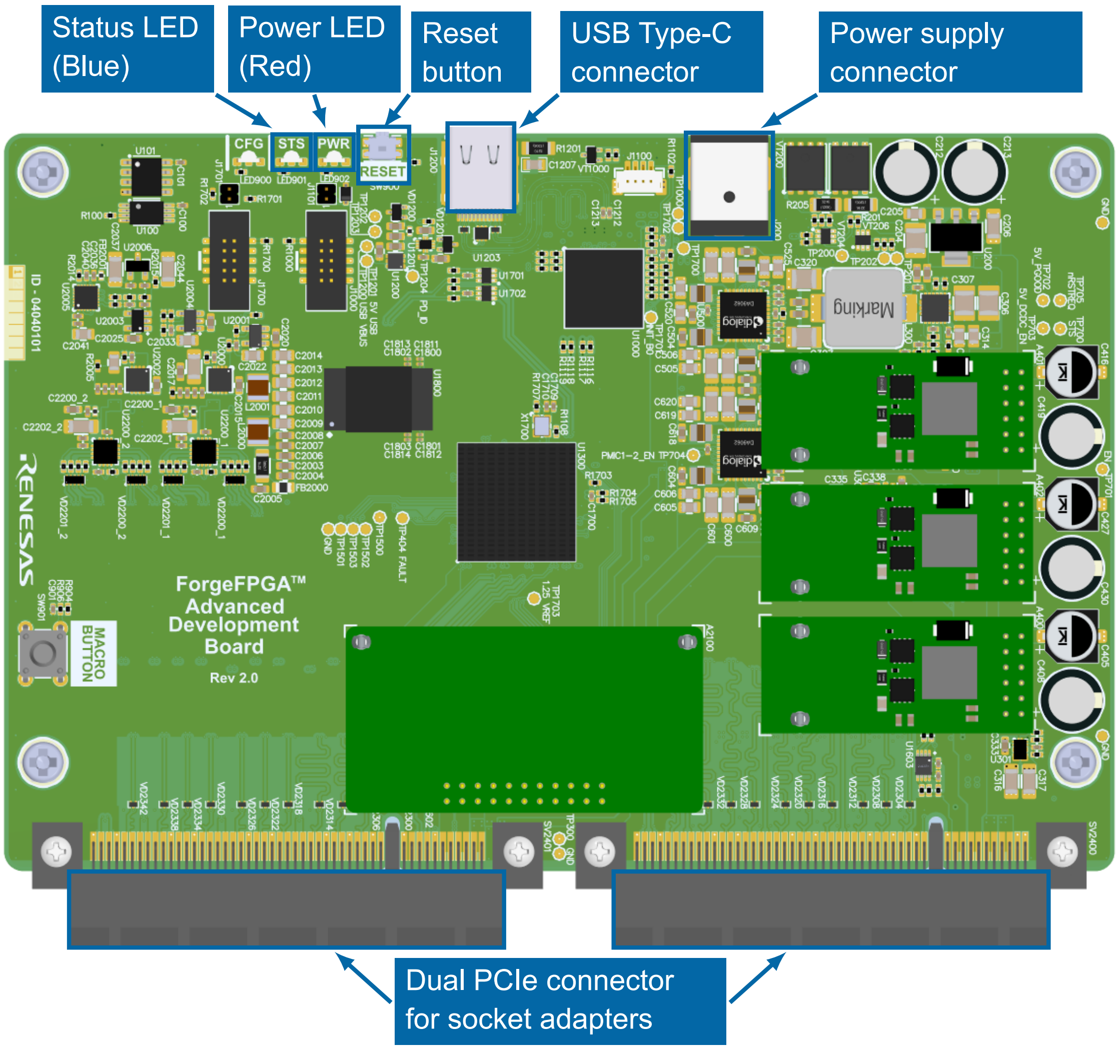
3.2.1 ForgeFPGA™ Deluxe Development Boards

- 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

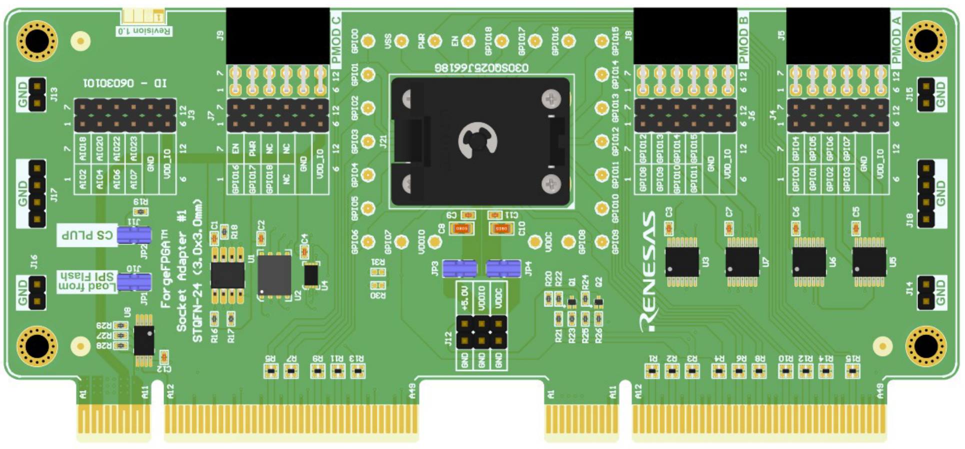
Note: The *ForgeFPGA Deluxe Development Board Rev. 2.0* is now accessible alongside the platform revision described above. Make sure you select the appropriate platform revision while working with *Debug tools*. You can load the preset configurations between the *ForgeFPGA Deluxe Development Board* and *ForgeFPGA Deluxe Development Board Rev. 2.0* (currently excluding *Logic Analyzer* settings) using the *Import configuration* option on the *Debugging Controls*.



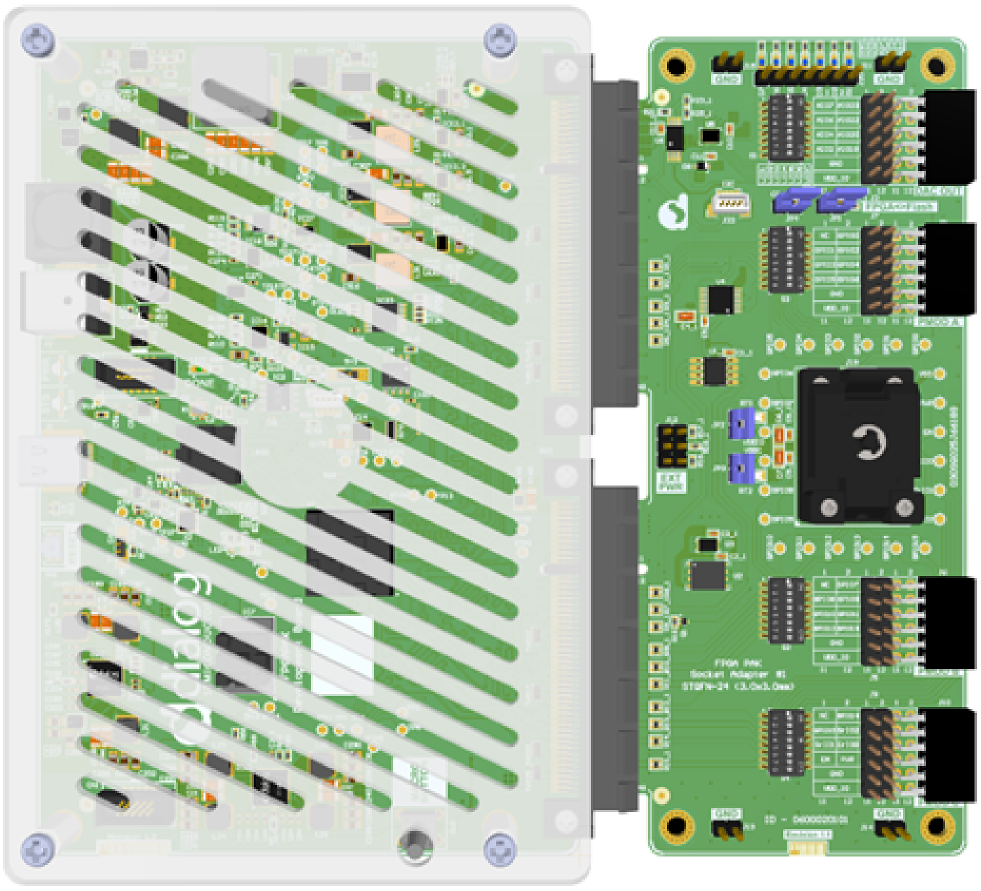
ForgeFPGA Deluxe Development Board



ForgeFPGA Deluxe Development Board Rev. 2.0



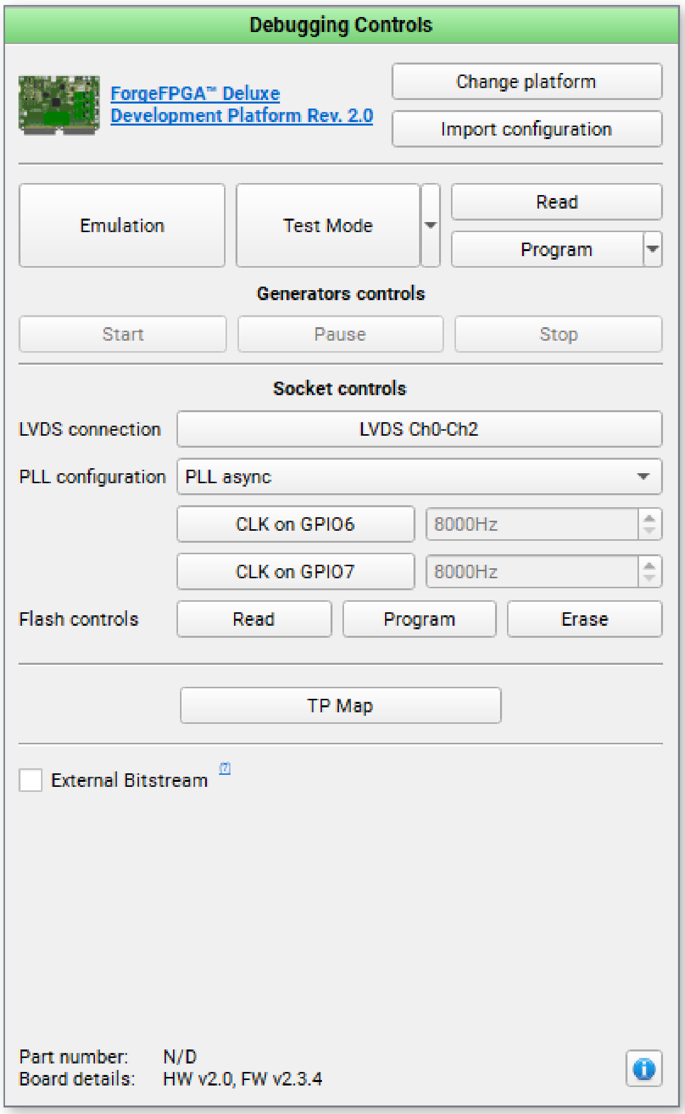
ForgeFPGA socket adapter



Assembled equipment for working with a chip

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

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



Part number: N/D
Board details: HW v2.0, FW v2.3.4

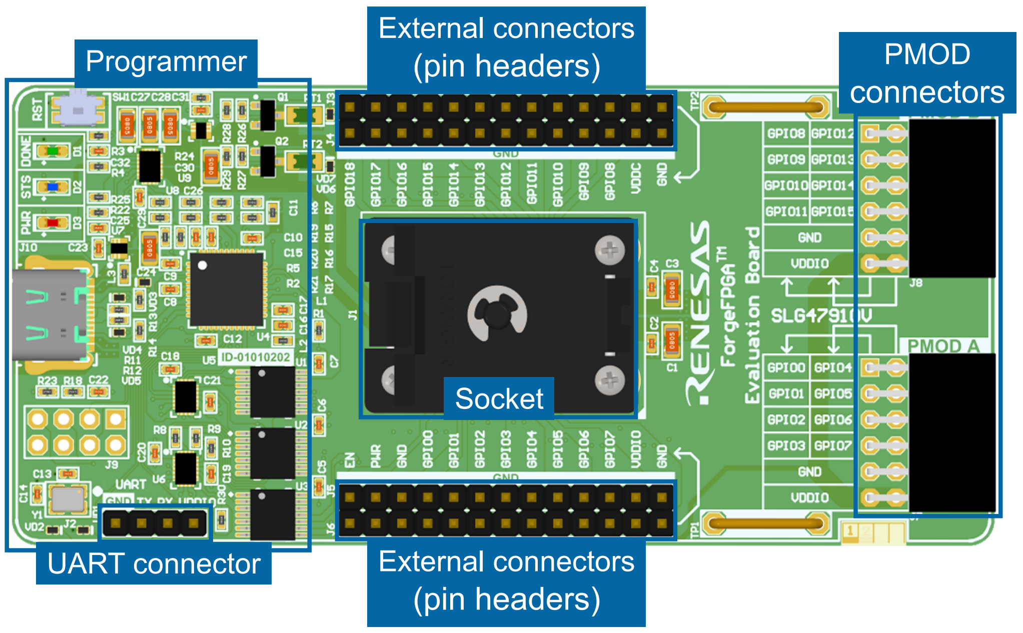
ForgeFPGA Deluxe Development Board controls

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

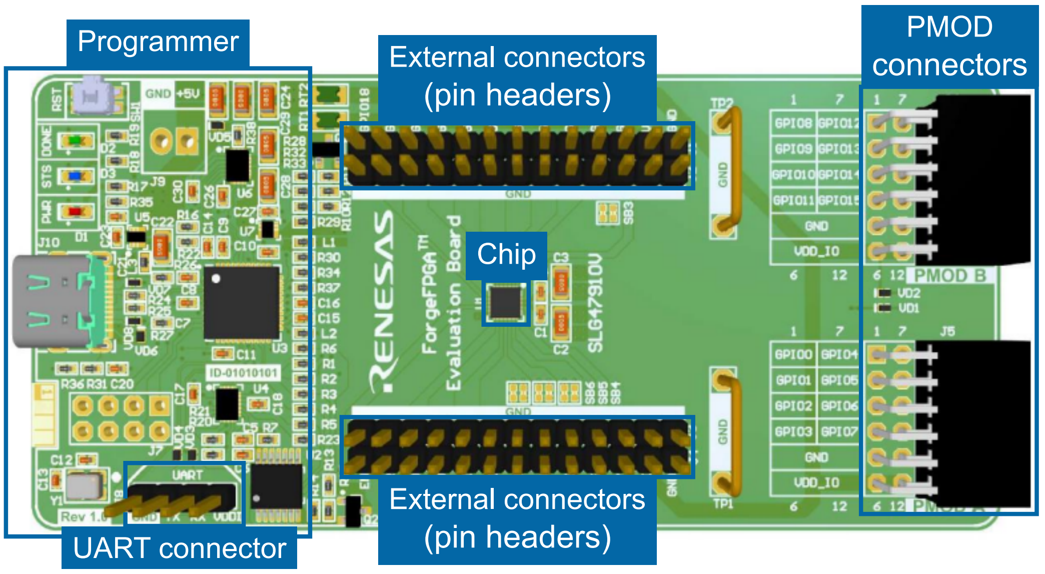
3.2.2 ForgeFPGA™ Evaluation Board

- 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.


Programmer
External connectors (pin headers)
PMOD connectors
Socket
UART connector
External connectors (pin headers)

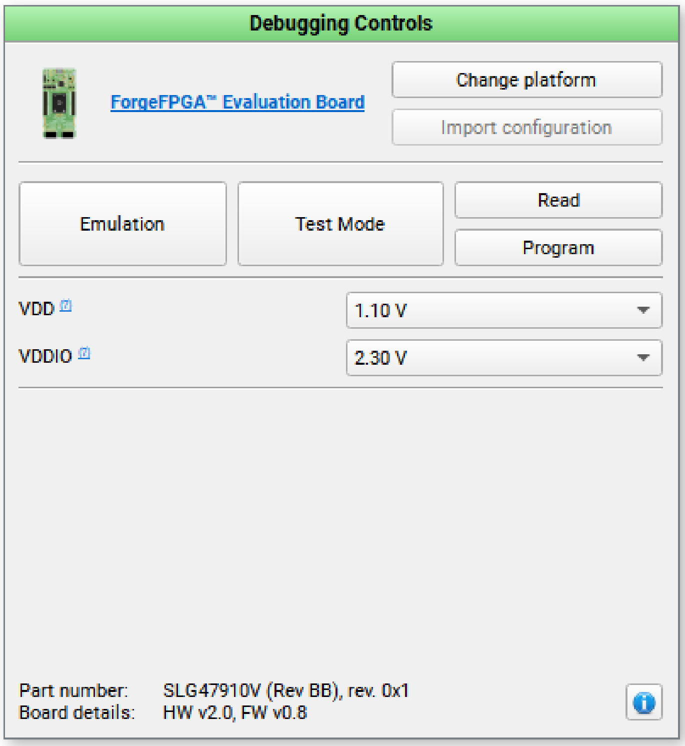
ForgeFPGA Evaluation Board 2.0


Programmer
External connectors (pin headers)
PMOD connectors
Chip
UART connector
External connectors (pin headers)

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



VDD 1.10 V

VDDIO 2.30 V

Part number: SLG47910V (Rev BB), rev. 0x1

Board details: HW v2.0, FW v0.8

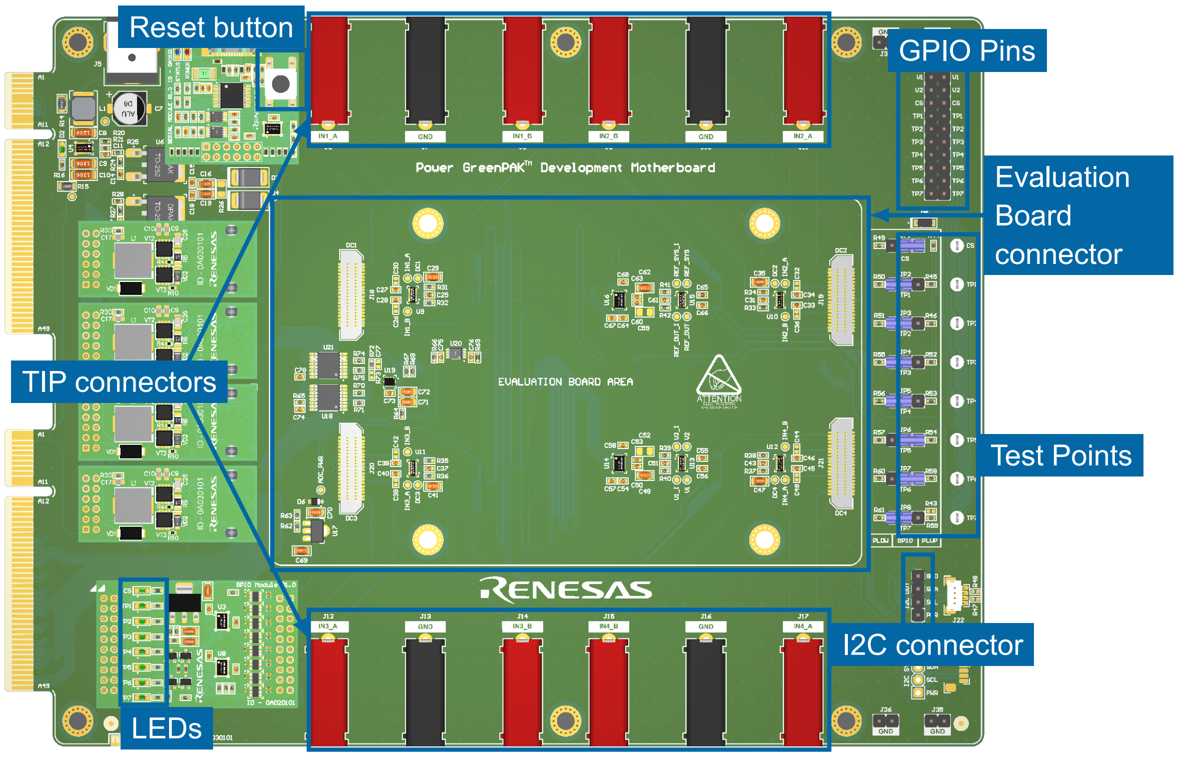
ForgeFPGA Evaluation Board controls

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

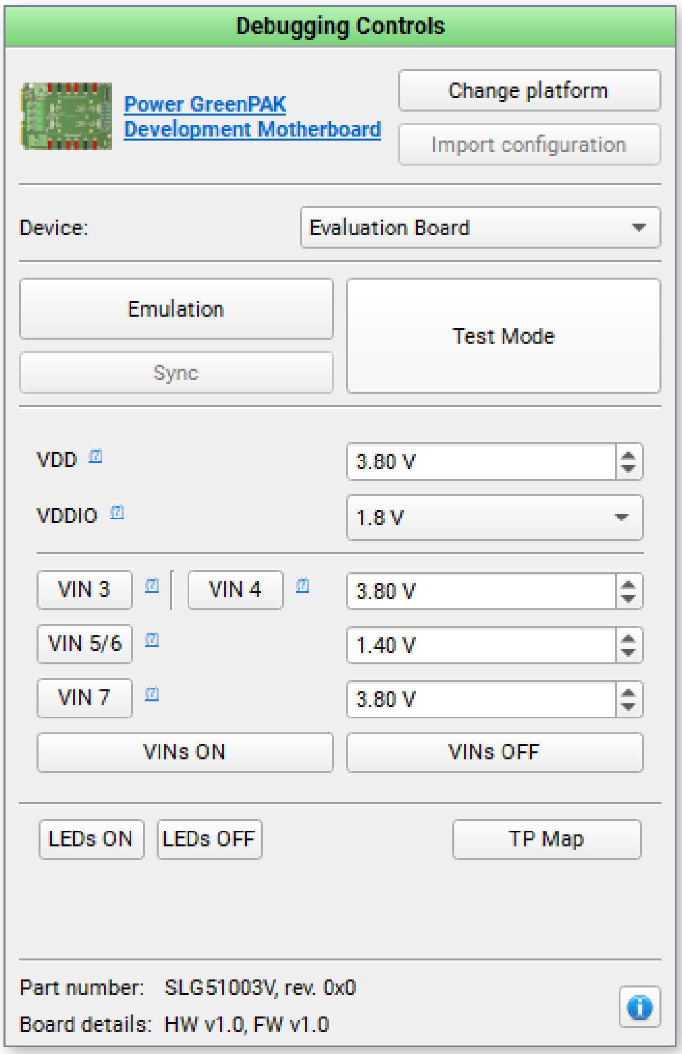
- Programming and emulation for *Power GreenPAK* devices (SLG51000, SLG51001, SLG51002, and SLG51003)
- Individually configurable Test Points (TPs):
  - Onboard LED state indication
  - Programmable software button (VDD, GND, Hi-Z)
  - Pull Up/Pull down jumpers
  - Floating hooks on each TP for oscilloscope connection
- Internal (onboard DC modules) and external power supply (TIP connectors) for IC's LDOs Vin
- 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



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



Device: Evaluation Board

VDD 3.80 V

VDDIO 1.8 V

VIN 3 | VIN 4: 3.80 V

VIN 5/6: 1.40 V

VIN 7: 3.80 V

Part number: SLG51003V, rev. 0x0

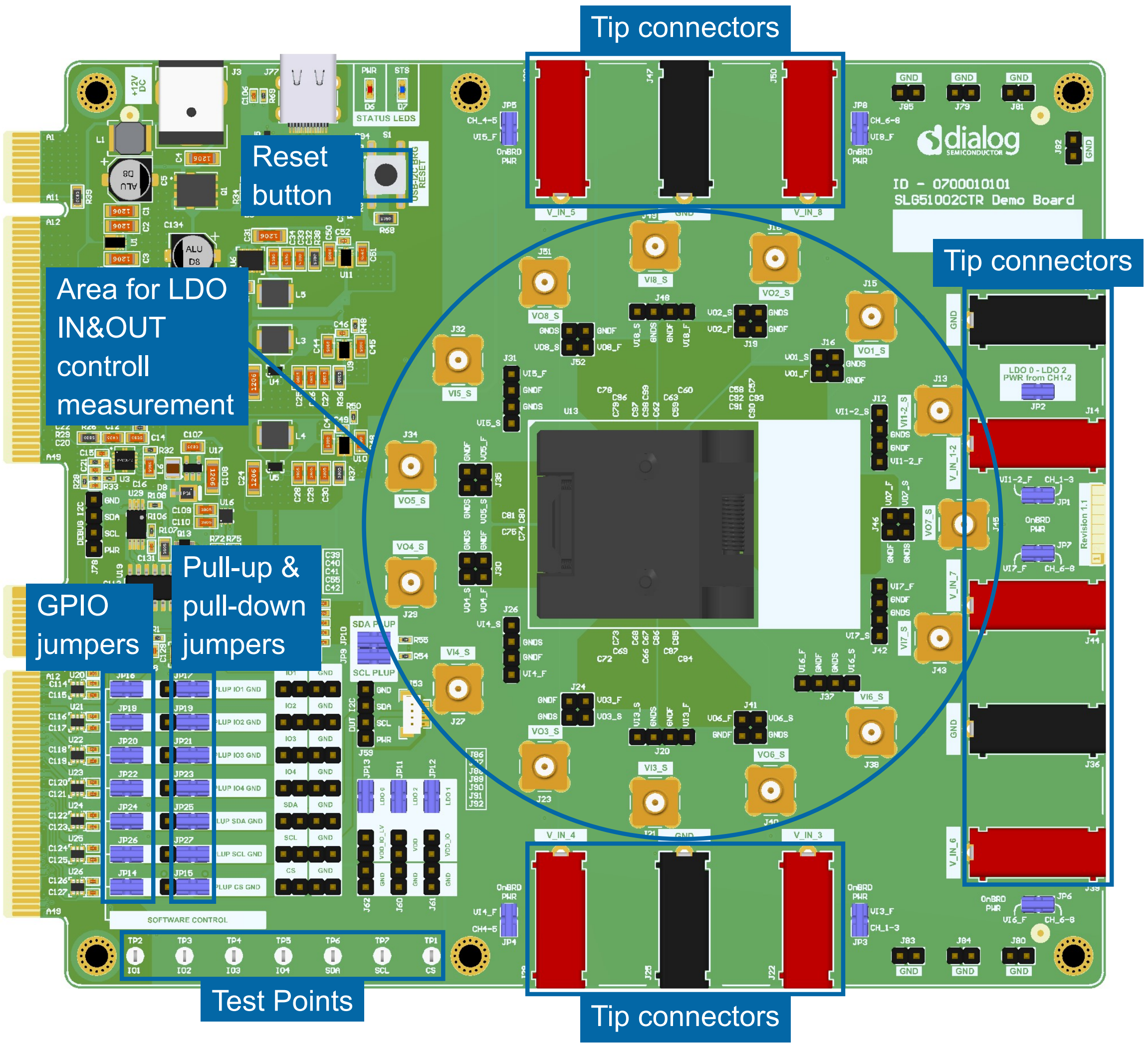
Board details: HW v1.0, FW v1.0

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

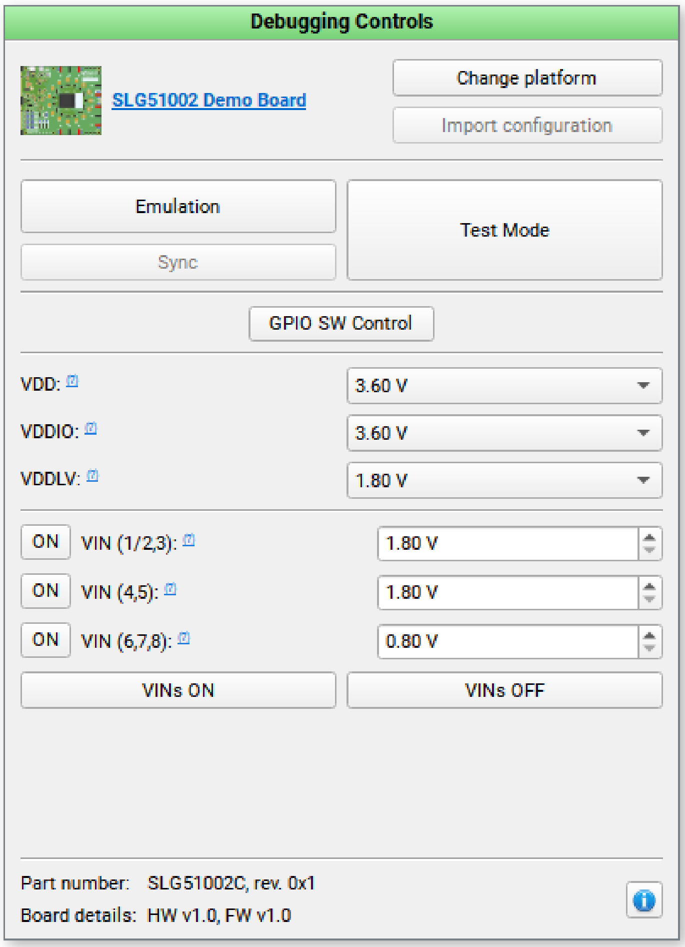
- Programming and emulation for SLG51002
- Individually configurable Test Points (TPs):
  - Programmable software button (VDD, GND, Hi-Z)
  - Pull Up jumpers
  - Floating hooks on each TP for oscilloscope connection
- SMB connectors for measurement LDO parameters
- USB interface for communication (USB-C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



Power GreenPAK SLG51002C Demo board

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

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



VDD: 3.60 V
VDDIO: 3.60 V
VDDLV: 1.80 V

ON VIN (1/2,3): 1.80 V
ON VIN (4,5): 1.80 V
ON VIN (6,7,8): 0.80 V

Part number: SLG51002C, rev. 0x1
Board details: HW v1.0, FW v1.0

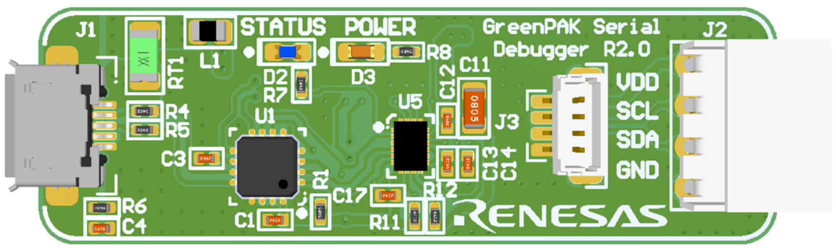
Power GreenPAK 002 Demo Board controls

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

Serial Debugger platform

You can use *GreenPAK Serial Debugger* for an external chip debugging of *Power GreenPAK* part numbers.

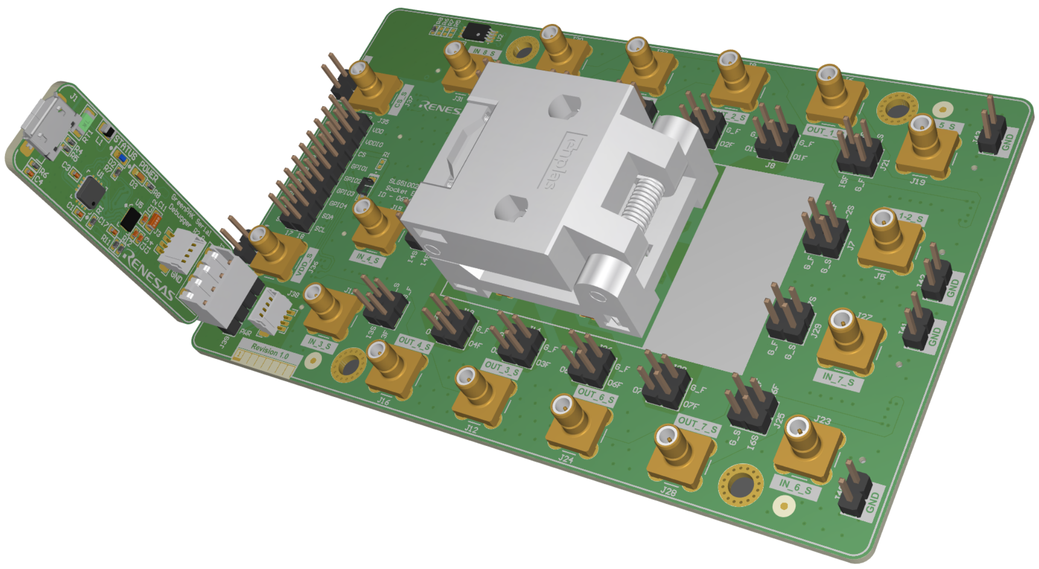


GreenPAK Serial debugger

To start working with an external chip on the *Serial Debugger* platform:

- Connect a chip with wires to the *Serial Debugger* pins (*CLK*, *SDA*, *GND*). If you use *SLG5100xCRT Socket Eval.*, use the DUT I2C connector.
- Ensure the Evaluation board's CS pin has been pulled-up to VDD or VDDIO. Connect the external power supply to VDD and VDDIO inputs.
- Start the *Go Configure Software Hub* and select the Part Number of the connected external chip
- Start the *Debug tool* and select the *Serial Debugger platform*
- Select *I2C Device address* or a corresponding address programmed to the chip
- Specify *VDD* configuration — *VDD* applied for I2C
- Connect the chip power sources externally

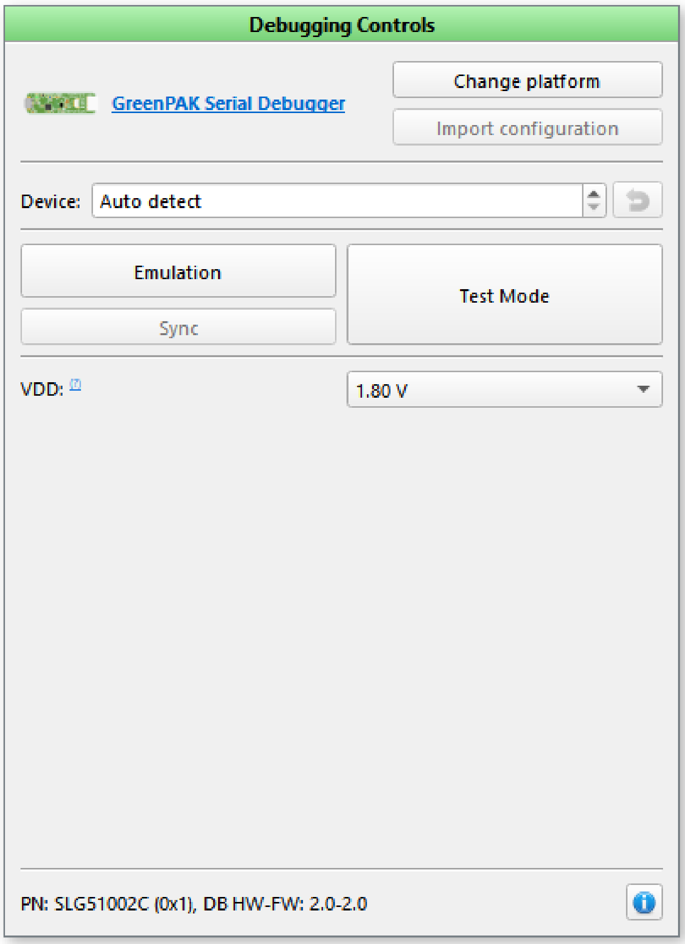
Once all steps are completed, the debugging controls become active.



GreenPAK Serial Debugger with Power GreenPAK socket

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

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



Device: Auto detect

VDD: 1.80 V

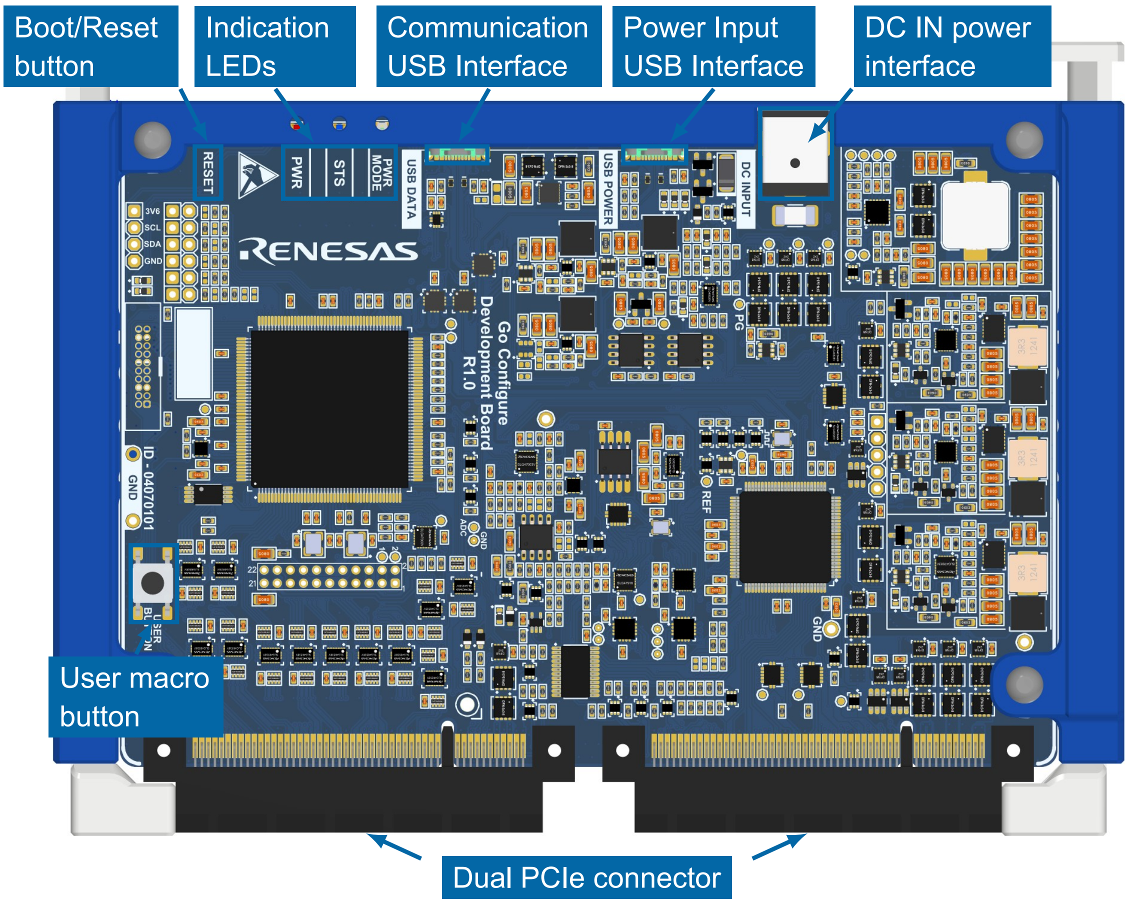
PN: SLG51002C (0x1), DB HW-FW: 2.0-2.0

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

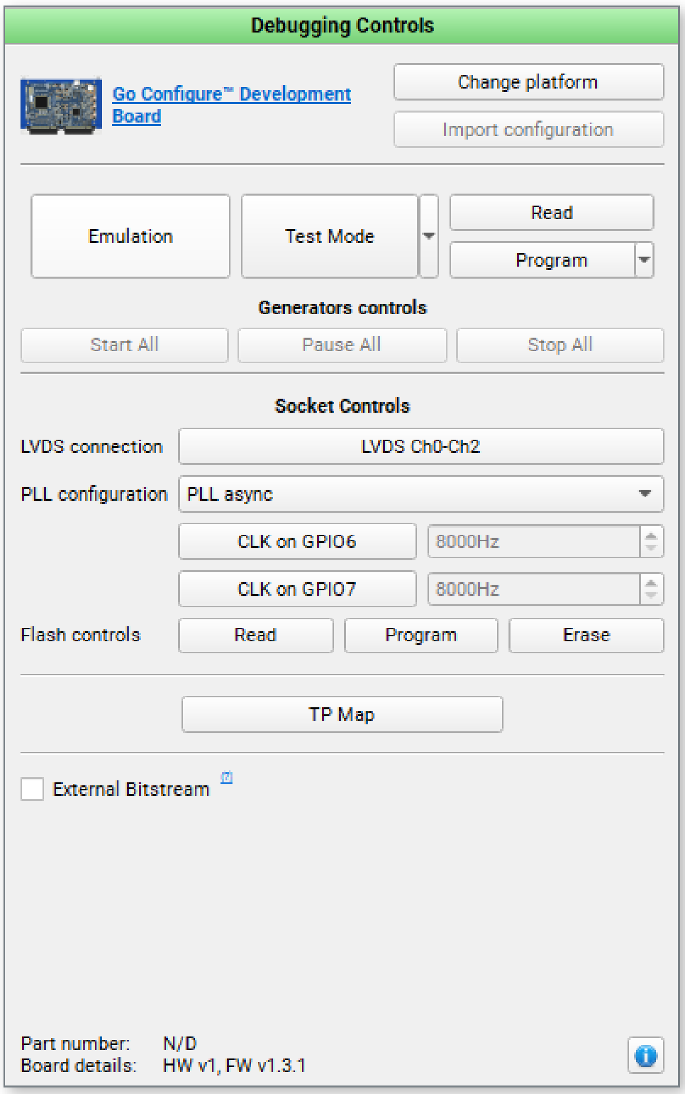
- Programming and emulation (currently *ForgeFPGA* devices are supported)
- 80 multi-function configurable I/O lines Test Points (TPs):
  - 32 channels Digital Pattern Generator
  - 16 channels Analog Waveform Generator (AWG)
- Dual PCIe connector
- Flexible power management with software-configurable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



Go Configure Development Board

The *Go Configure Development Board* *Debugging Controls* User Interface includes the following sections:

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



Part number: N/D
Board details: HW v1, FW v1.3.1

Go Configure Development Board

4 How to

In addition to facilitating chip configuration, the *Go Configure Software Hub* offers a range of specific features designed to enhance your workflow efficiency. While these features may not be directly tied to your project, they can greatly assist you in streamlining various aspects of your work process.

This chapter presents the methods to keep your software version up to date and introduces various approaches for updating software. Furthermore, you will learn how to create and save block diagrams, delve into debug tool-related instructions, and explore the integration of external software within your project.

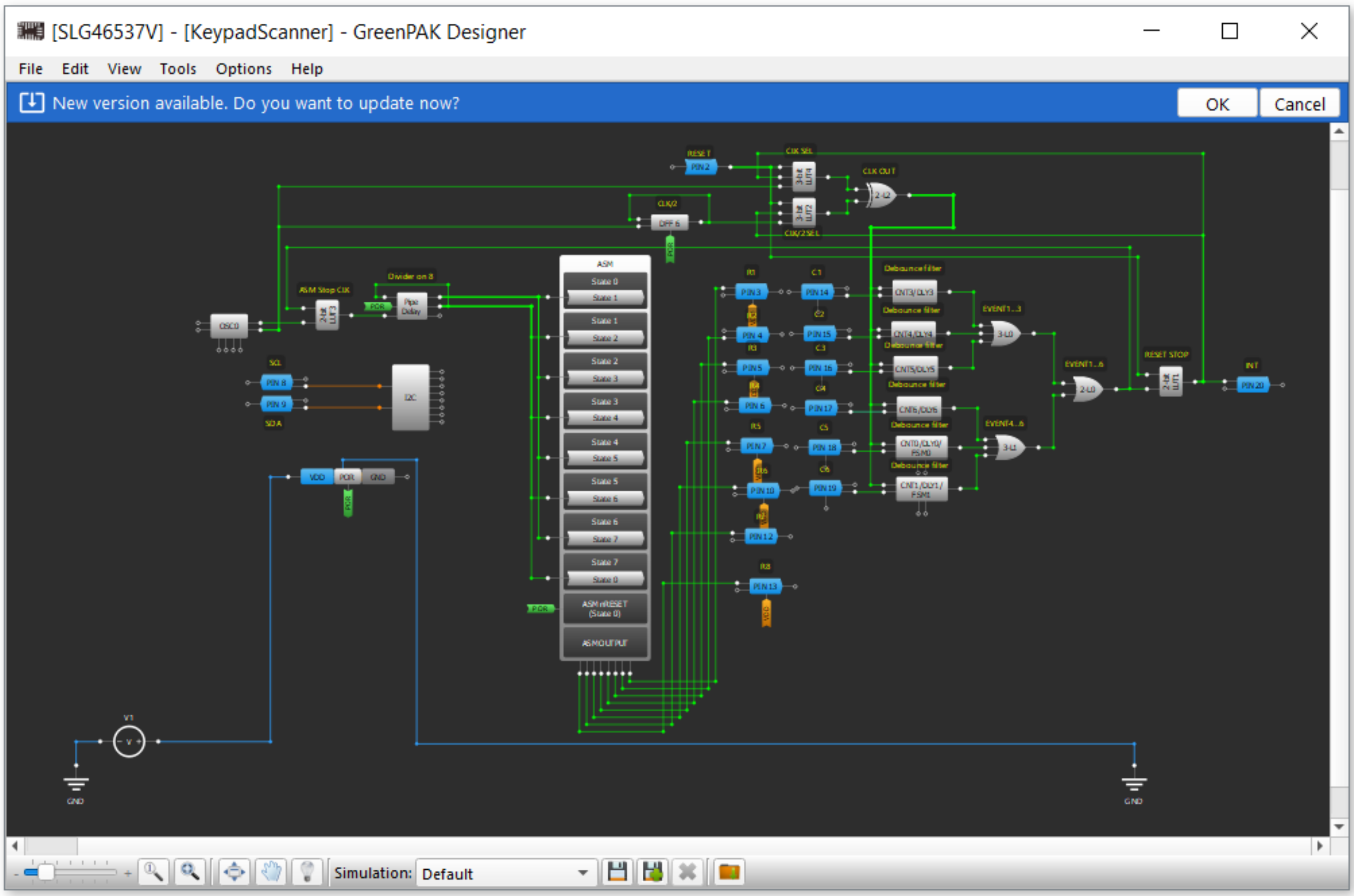
4.1 Software update

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

There are two ways of updating the *Go Configure Software Hub*:

➤ When updates are available, a notification appears. You can either download the latest version or postpone the update until the program restarts. After the download is complete, the *Go Configure Software Hub* will handle the automatic installation of the software. **Note**: it is important to restart the software after the installation process is done.

➤ You can also find the latest version of the *Go Configure Software Hub* on the Software page on the Renesas website. To ensure the best user experience, keep your software up to date.


[SLG46537V] - [KeypadScanner] - GreenPAK Designer
File Edit View Tools Options Help
OK
Cancel
Simulation: Default

New version available. Do you want to update now?

Software update notification

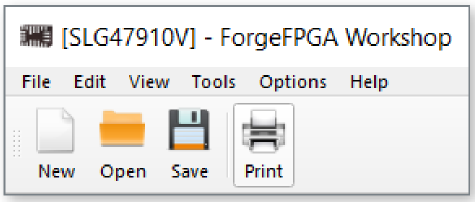
Adjust the *Updater* preferences in the *Designer Settings* window, *Updater* section (refer to Section 2.1.7 Settings).

Feel free to email suggested updates to the developer team to improve the software (click *Help* → *About Go Configure Software Hub*).

4.2 Printing

4.2.1 Print

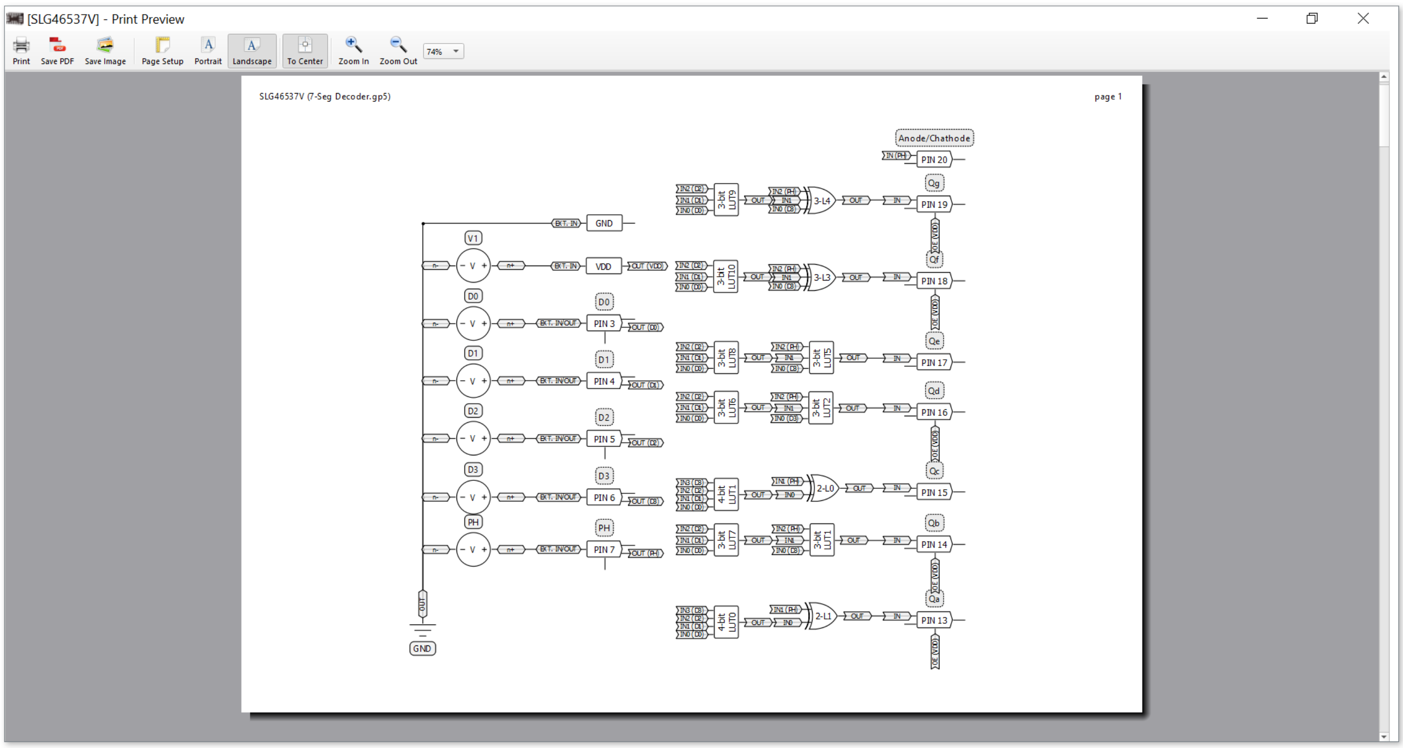
You can use the *Print tool* to present and save a block diagram in a graphic format to use these images in datasheets or other project documentation showing the interconnections between blocks and their configurations. Find the *Print* tool on the toolbar.



Print tool on the toolbar

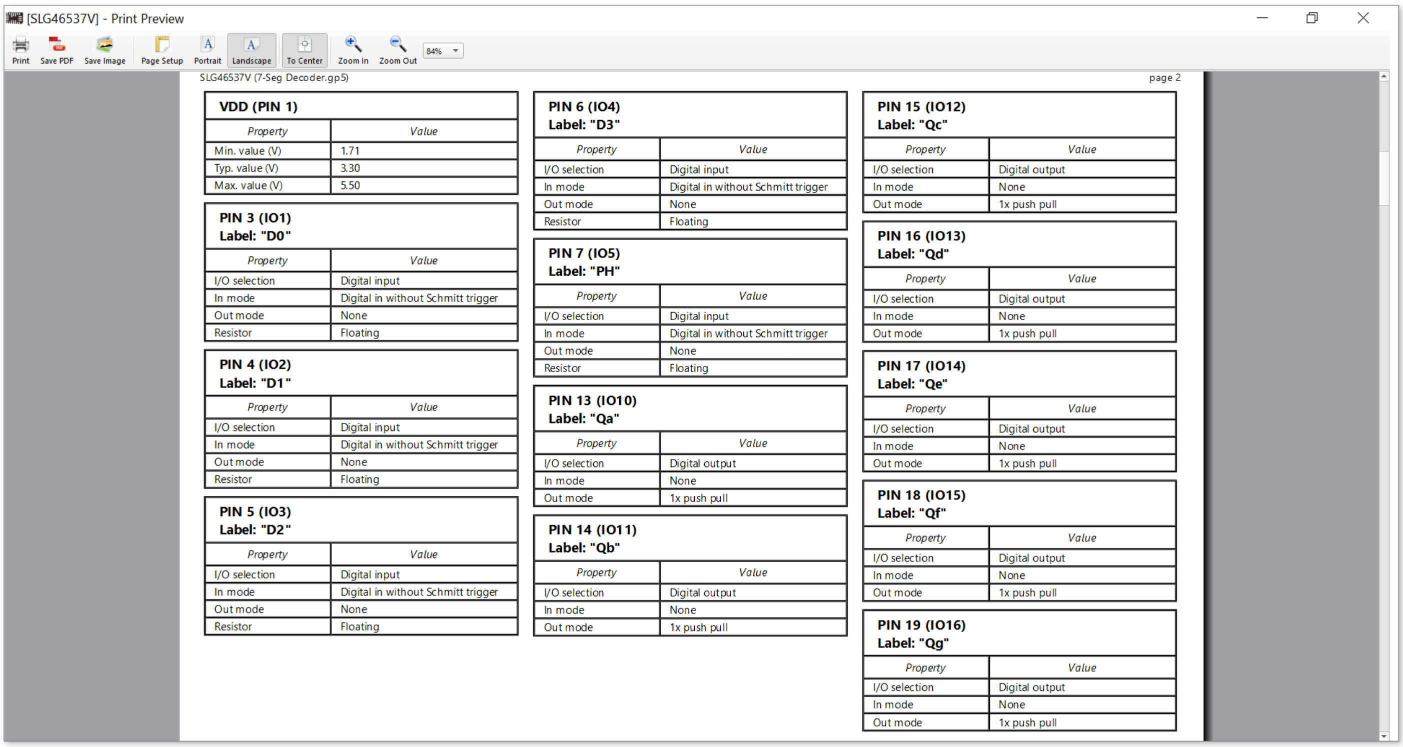
Also, you can open the *Print tool* by clicking *File* → *Print* in the main menu.

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, you can't change the position of elements or lines on the diagram. To reposition components or wires, you should return to the working area, introduce the changes, and launch the *Print Tool* again.



Print Preview window

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



SLG46537V (7-Seg Decoder.gp5)

page 2

VDD (PIN 1)

Property	Value
Min. value (V)	1.71
Typ. value (V)	3.30
Max. value (V)	5.50

PIN 3 (IO1)
Label: "D0"

Property	Value
I/O selection	Digital input
In mode	Digital in without Schmitt trigger
Out mode	None
Resistor	Floating

PIN 4 (IO2)
Label: "D1"

Property	Value
I/O selection	Digital input
In mode	Digital in without Schmitt trigger
Out mode	None
Resistor	Floating

PIN 5 (IO3)
Label: "D2"

Property	Value
I/O selection	Digital input
In mode	Digital in without Schmitt trigger
Out mode	None
Resistor	Floating

PIN 6 (IO4)
Label: "D3"

Property	Value
I/O selection	Digital input
In mode	Digital in without Schmitt trigger
Out mode	None
Resistor	Floating

PIN 7 (IO5)
Label: "PH"

Property	Value
I/O selection	Digital input
In mode	Digital in without Schmitt trigger
Out mode	None
Resistor	Floating

PIN 13 (IO10)
Label: "Qa"

Property	Value
I/O selection	Digital output
In mode	None
Out mode	1x push pull

PIN 14 (IO11)
Label: "Qb"

Property	Value
I/O selection	Digital output
In mode	None
Out mode	1x push pull

PIN 15 (IO12)
Label: "Qc"

Property	Value
I/O selection	Digital output
In mode	None
Out mode	1x push pull

PIN 16 (IO13)
Label: "Qd"

Property	Value
I/O selection	Digital output
In mode	None
Out mode	1x push pull

PIN 17 (IO14)
Label: "Qe"

Property	Value
I/O selection	Digital output
In mode	None
Out mode	1x push pull

PIN 18 (IO15)
Label: "Qf"

Property	Value
I/O selection	Digital output
In mode	None
Out mode	1x push pull

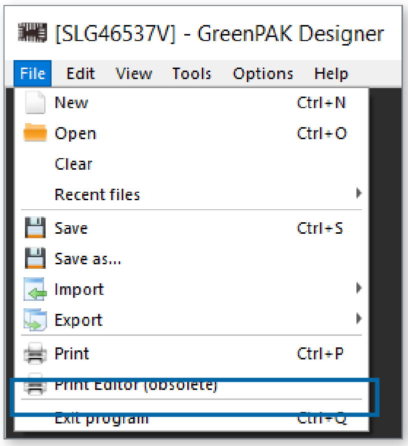
PIN 19 (IO16)
Label: "Qg"

Property	Value
I/O selection	Digital output
In mode	None
Out mode	1x push pull

Table with block properties and values in Print Preview window

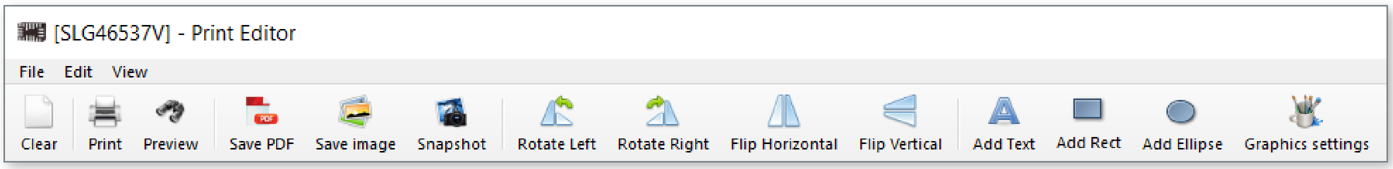
4.2.2 Print Editor (obsolete)

In the older versions of the software updates, the *Print editor (obsolete)* is available. You can find this tool by clicking *File* → *Print Editor (obsolete)*.



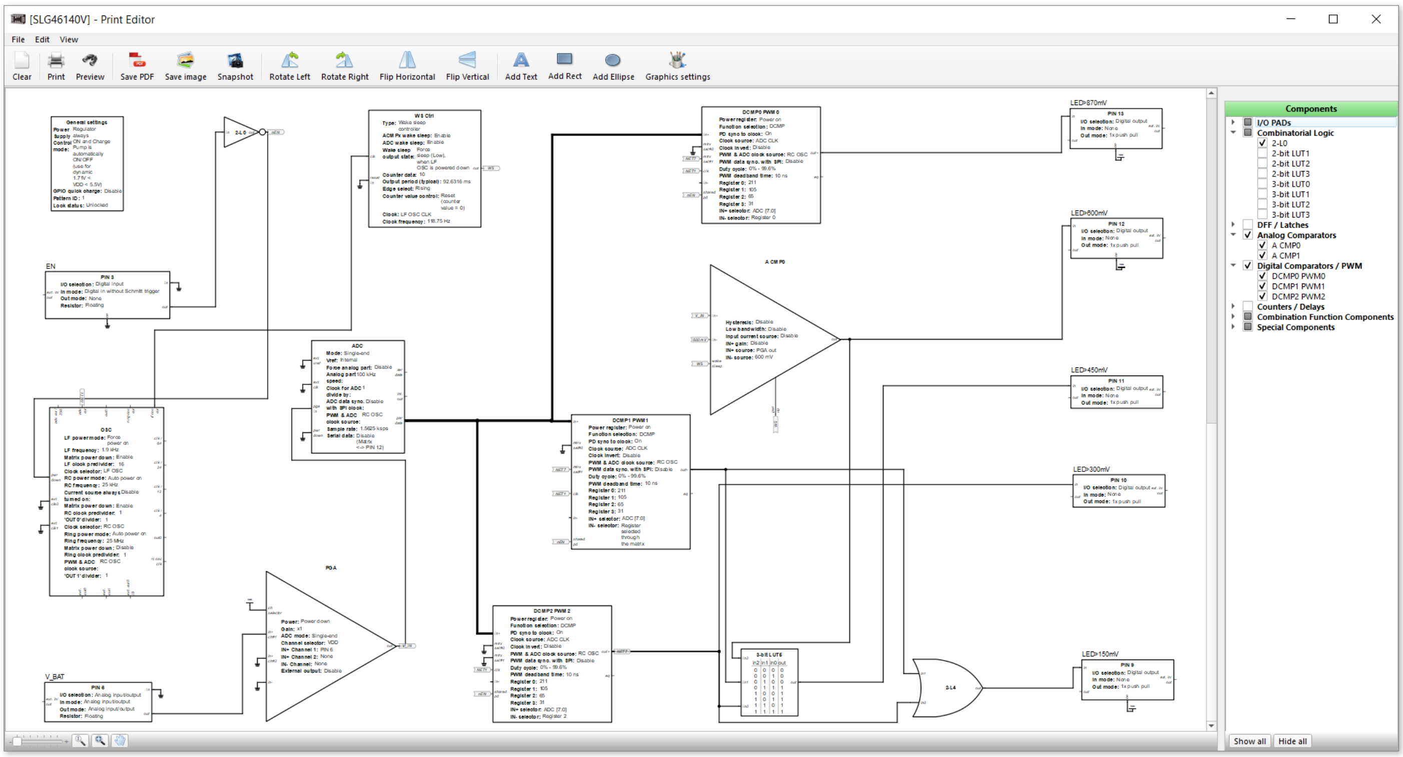
Print Editor (obsolete)

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



Print Editor toolbar

An editable working area of the *Print Editor (obsolete)* contains all components used in the design. It enables customizing positions, views of components, and wires.



Print Editor (obsolete) interface

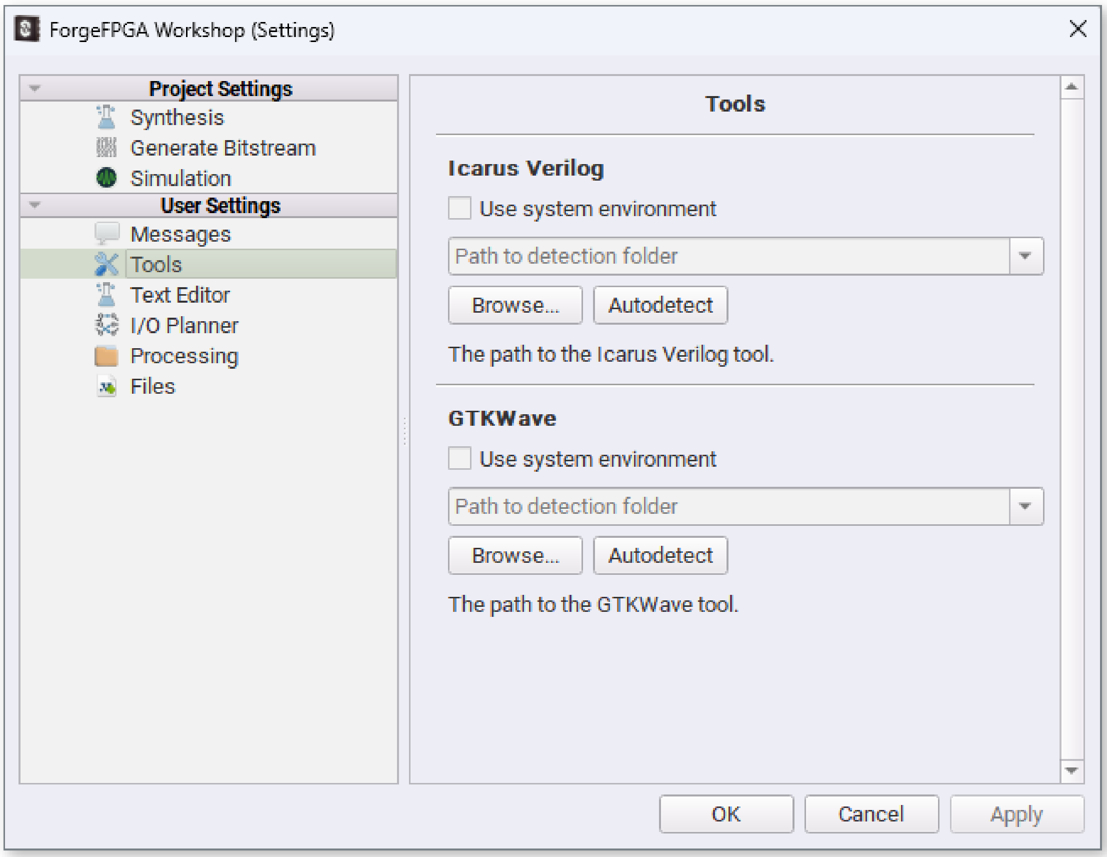
4.3 Icarus Verilog and GTKWave quick start

The *Icarus Verilog* and *GTKWave* software serve as simulation and simulation results visualizing tools for validating Verilog design code using a testbench. Within the testbench, you can instantiate the *Unit Under Test (UUT)* or *Device Under Test (DUT)* and systematically apply input combinations. You can observe and analyze the resulting outputs using the GTKWave software.

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 of these components to ensure proper functionality. Add iVerilog and GTKWave to the system paths during the installation process. If you overlook this step, you can add these paths later by navigating *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.



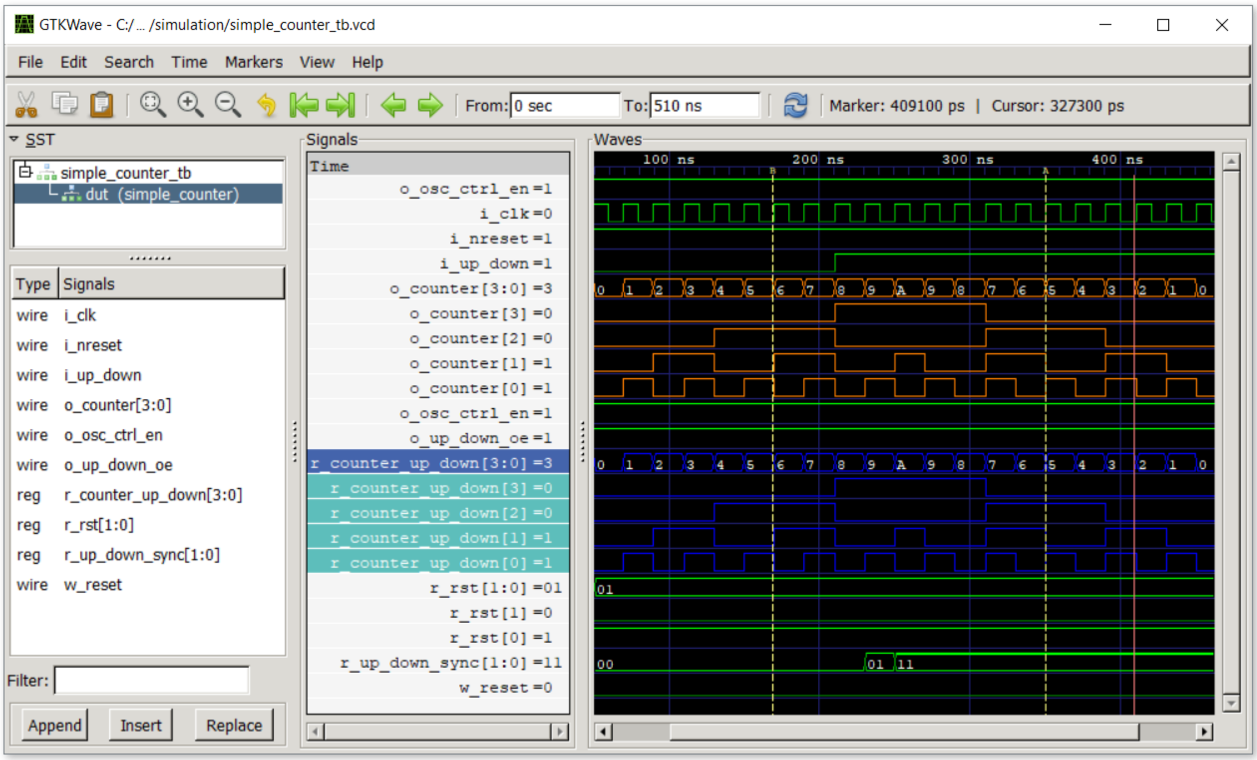
The path to the Icarus Verilog tool.

The path to the GTKWave tool.

FPGA Editor Settings

4.3.2 Quick start

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



Simple Counter in GTKWave software

- Access GTKWave by either clicking *Simulate Testbench* in the ForgeFPGA Workshop or entering `gtkwave` in the terminal. If you opt to launch the software externally, please 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, the 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.

```verilog
// Custom testbench

`timescale 1ns / 1ps

module ALU16_tb;

  initial begin

    $dumpfile ("ALU16_tb.vcd");
    $dumpvars (0, ALU16_tb);

  end

```

```
endmodule
```

The `initial` block is executed only once. Above the `initial` block you can find the declaration of input and output signals, along with the module instantiation.

➤ Start the simulation by clicking the *Simulate Testbench* button in the ForgeFPGA Workshop. This action triggers GTKWave to automatically open the corresponding .vcd file generated by Icarus Verilog. Upon the *Wave* window's launch, no waveforms are initially displayed, giving you the option to select the specific signals whose waveforms you wish to view.

➤ Select the desired signal by navigating to the *SST* window, showcasing your hardware hierarchy. You can reveal the signals associated with that specific instance in the bottom section. Either drag and drop the preferred signal or double-click on it to display it in the *Signals* window. Alternatively, you can select all signals (`Ctrl + A`) and insert them. Also, to observe the signals' respective values, click the *Reload* button on the toolbar.

➤ Customize signal properties in the context menu of a signal and selecting *Data Format* or *Color Format*. By default, signal values are in hexadecimal format, and all waves are colored green (if the simulation is running correctly). Additionally, you can *Insert Blank* signal to create sections between groups of signals.

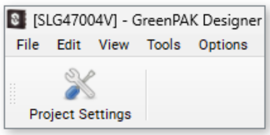
➤ Save settings by navigating to *File* → *Write Save File* once you achieved the desired visual configuration. Saving your progress ensures that when you simulate the same testbench next time, most of your previous adjustments will be automatically restored.

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 your project's 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 upon project launch

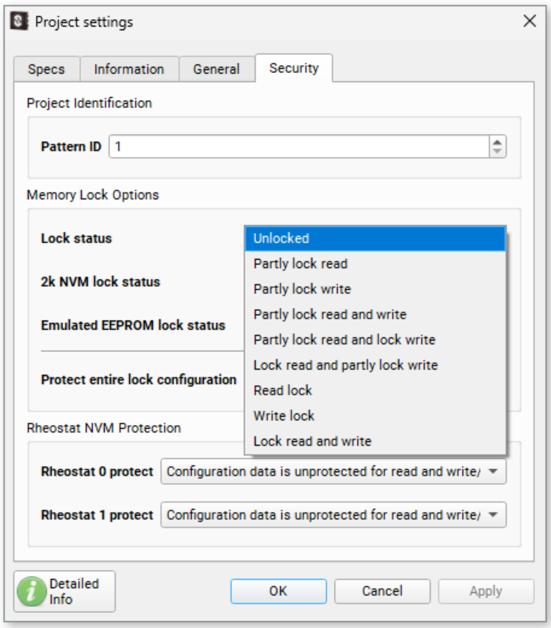


Project Settings on the toolbar

2. Reach *Security* tab in the *Project Settings* window

3. The settings may vary between different Part Numbers based on their specific characteristics, such as e.g. 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



Project Identification

Pattern ID 1

Memory Lock Options

Lock status

2k NVM lock status

Emulated EEPROM lock status

Protect entire lock configuration

Unlocked
Partly lock read
Partly lock write
Partly lock read and write
Partly lock read and lock write
Lock read and partly lock write
Read lock
Write lock
Lock read and write

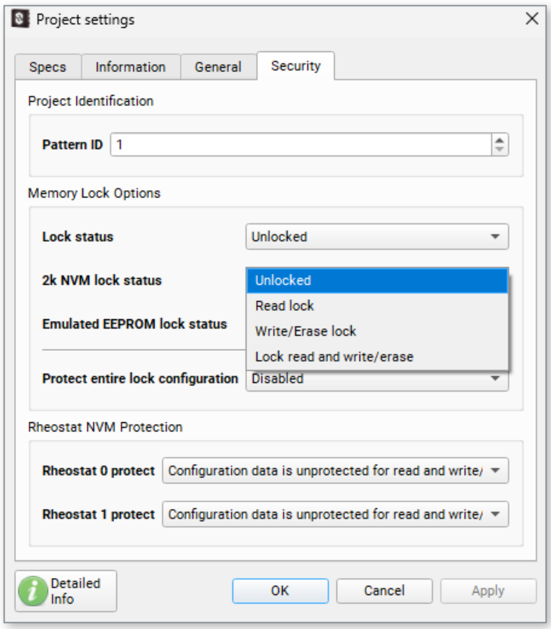
Rheostat NVM Protection

Rheostat 0 protect Configuration data is unprotected for read and write/

Rheostat 1 protect Configuration data is unprotected for read and write/

SLG47004V Lock status options

➤ *2k NVM lock status* — allows to lock MTP (Multi-Time Programmable) NVM


Project settings

Specs | Information | General | Security

Project Identification

Pattern ID: 1

Memory Lock Options

Lock status: Unlocked

2k NVM lock status: Unlocked
- Unlocked
- Read lock
- Write/Erase lock
- Lock read and write/erase

Emulated EEPROM lock status

Protect entire lock configuration: Disabled

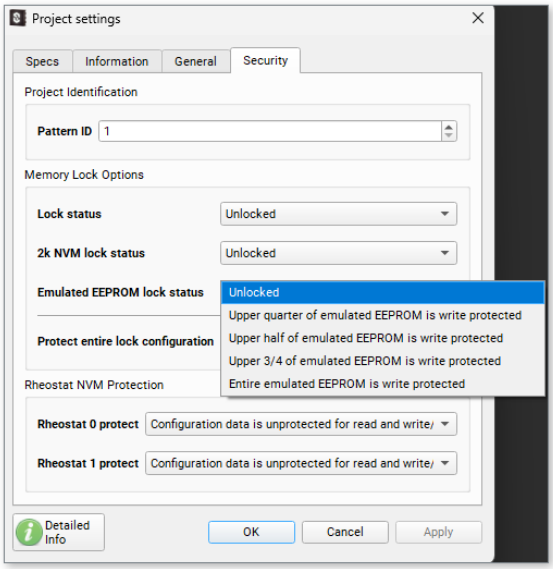
Rheostat NVM Protection

Rheostat 0 protect: Configuration data is unprotected for read and write/

Rheostat 1 protect: Configuration data is unprotected for read and write/

SLG47004V 2k NVM lock status options

➤ *Emulated EEPROM lock status* — locks EEPROM (Electrically Erasable Programmable Read-only Memory) memory type


Project settings

Specs | Information | General | Security

Project Identification

Pattern ID: 1

Memory Lock Options

Lock status: Unlocked

2k NVM lock status: Unlocked

Emulated EEPROM lock status: Unlocked
- Unlocked
- Upper quarter of emulated EEPROM is write protected
- Upper half of emulated EEPROM is write protected
- Upper 3/4 of emulated EEPROM is write protected
- Entire emulated EEPROM is write protected

Protect entire lock configuration

Rheostat NVM Protection

Rheostat 0 protect: Configuration data is unprotected for read and write/

Rheostat 1 protect: Configuration data is unprotected for read and write/

SLG47004V Emulated EEPROM lock status options

➤ *Protect entire lock configuration* — choose if all lock modes described above can be modified

4. Once 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 of great importance for us to help you overcome obstacles, you may encounter while using the software.

In this chapter, we'll help you tackle common problems you might face while using the software. We'll cover troubleshooting for issues like technical glitches, error messages, or unexpected behavior. By following these steps, you'll be able to solve problems easily and make the most out of your software experience.

5.1 Failed Socket Test

Here is the list of steps to solve the most common causes of the Socket Test procedure failure:

- Ensure the contacts connecting a socket and chip pins are clean and undamaged
- Disconnect any external circuits or signal sources (generator, voltage supply, etc) connected to a board or a socket itself

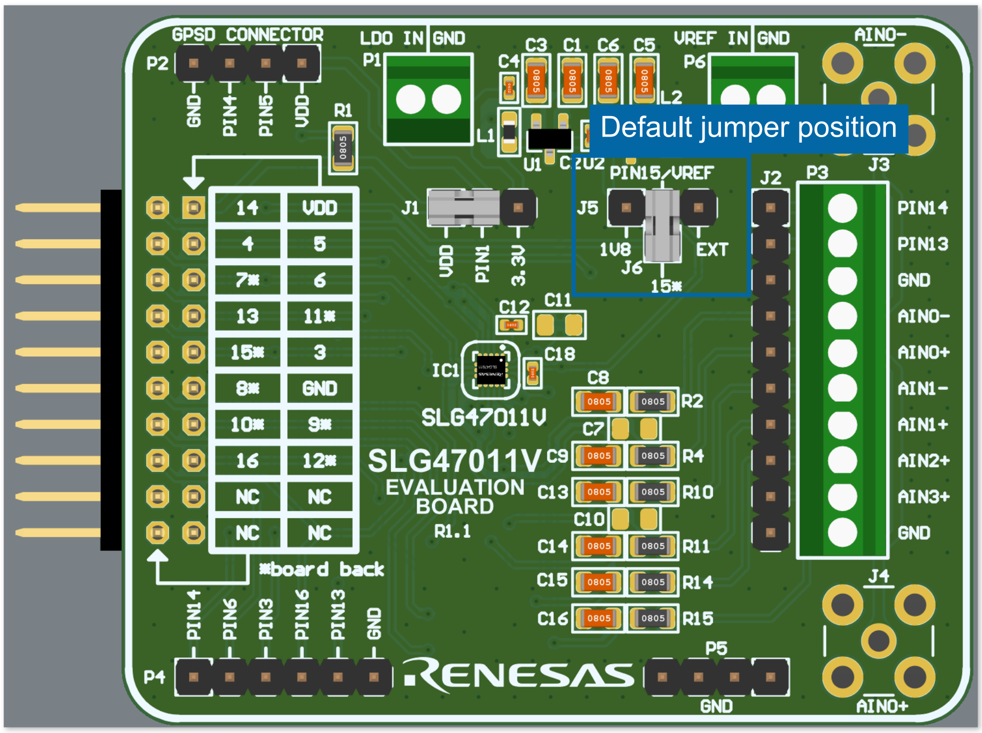
  Note: External devices connected to expansions connectors do not affect the *Socket Test* procedure.

- 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.

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:



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/disable the work area move in pan mode
	Show hints	enable/disable hints for blocks on the work area
	Properties	show/hide Properties panel
	Schematic Library	list of external components for Software Simulation
	Components	show/hide software tools blocks list
	NVM Viewer	show/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`

Software tool launcher		
Action	**Windows/Linux controls**	**macOS controls**
Welcome tab*	`Ctrl` + `1`	`Command` + `1`
Recent files tab*	`Ctrl` + `2`	`Command` + `2`
Develop tab*	`Ctrl` + `3`	`Command` + `3`
Demo tab*	`Ctrl` + `4`	`Command` + `4`
Recovery files tab*	`Ctrl` + `5`	`Command` + `5`
Datasheets*	`Ctrl` + `6`	`Command` + `6`
User guide*	`Ctrl` + `7`	`Command` + `7`

Note: Non-configurable controls are marked with an asterisk (*).

6.3 Debugging Controls feature availability

GreenPAK platforms

Feature	Advanced board	DIP board	Lite board	Serial Debugger
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	✓	✓	✓	✓

FPGA platforms

Feature	ForgeFPGA Deluxe Development Board	ForgeFPGA Evaluation Board
Debug Configuration	✓	✓
Chip procedures	✓	✓
Flash procedures	✓	
Generator controls	✓	
TP map	✓	
Voltage level controls		✓
Info details	✓	✓

Power GreenPAK platforms

Feature	Power GreenPAK Development Motherboard	SLG51002CTR Demo Board	Serial Debugger(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	Multi-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
2.12.0	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/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/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/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

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