

# Infineon Device Configurator user guide

Device Configurator version 5.30

## About this document

### Scope and purpose

This guide provides information and instructions for using the Infineon Device Configurator to update various parameters for your device.

[A newer version of this document may be available on the web here.](#)

### Intended audience

This document helps application developers understand how to use the Device Configurator as part of creating a ModusToolbox™ application.

### Document conventions

Convention	Explanation
<b>Bold</b>	Emphasizes heading levels, column headings, menus and sub-menus
<i>Italics</i>	Denotes file names and paths.
Monospace	Denotes APIs, functions, interrupt handlers, events, data types, error handlers, file/folder names, directories, command line inputs, code snippets
<b>File &gt; New</b>	Indicates that a cascading sub-menu opens when you select a menu item

### Abbreviations and definitions

- **Resource** – Includes peripherals, pins, clocks, etc. used in an application.
- **Configurator** – A GUI-based tool used to configure a resource.
- **Application** – One or more projects related to each other.
- **Personality** – A file that defines a resource behavior.
- **Device Support Library** – A device support library provides critical firmware and device data files to configurators. Device support libraries are identified with a file named *props.json*. It is used to find things like other tools, devices, and personalities.

### Reference documents

Refer to the following documents for more information as needed:

- [ModusToolbox™ tools package user guide](#)
- [ModusToolbox™ tools package release notes](#)
- API reference guides

## Table of contents

## Table of contents

	<b>About this document</b> .....	1
	<b>Table of contents</b> .....	2
<b>1</b>	<b>Overview</b> .....	4
<b>2</b>	<b>Stand-alone operation</b> .....	5
2.1	Install Infineon Device Configurator .....	5
2.2	CMSIS Pack .....	6
2.3	Run Device Configurator .....	6
2.4	Add menu item to µVision manually .....	7
2.5	Remove menu item from µVision manually .....	8
<b>3</b>	<b>Using with ModusToolbox™ application</b> .....	11
3.1	Installation .....	11
3.2	Launch the Device Configurator .....	11
<b>4</b>	<b>Quick start</b> .....	13
<b>5</b>	<b>Code generation</b> .....	15
<b>6</b>	<b>GUI description</b> .....	16
6.1	Menus .....	16
6.2	Device tabs .....	17
6.3	Resources tabs .....	17
6.4	Panes .....	17
6.5	Icons .....	17
<b>7</b>	<b>Resources tabs</b> .....	18
7.1	Solutions tab .....	20
7.2	Peripherals .....	21
7.3	Pins .....	22
7.4	Analog-Routing tab .....	24
7.5	System tab .....	28
7.6	Memory tab .....	29
7.7	MMIO-Clocks tab .....	42
7.8	Peripheral-Clocks tab .....	43
7.9	DMA tab .....	44
<b>8</b>	<b>Panes</b> .....	45
8.1	Parameters pane .....	45
8.2	Notice List .....	47
8.3	Code Preview pane .....	48
<b>9</b>	<b>Version changes</b> .....	49
	<b>Revision history</b> .....	51



Table of contents

---

Disclaimer ..... 52

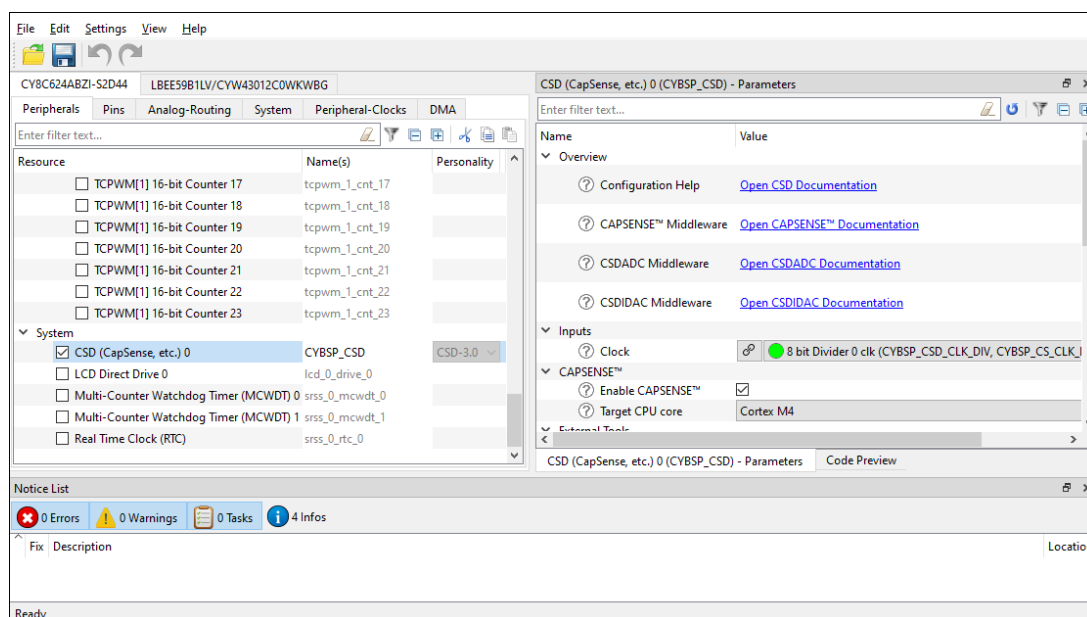
## 1 Overview

### 1 Overview

The Infineon Device Configurator allows you to enable and configure device peripherals, such as clocks and pins, as well as standard MCU peripherals. When working in the ModusToolbox™ environment, this tool is included as part of the tools package, and it is required when working on devices such as PSoC™ MCUs.

This tool is also provided as an optional stand-alone configuration utility for you to use with the Keil µVision IDE when working on devices such as TLE994x\_5x. This version is distributed via Infineon Developer Center: <http://www.infineon.com/idc>. Refer to the specific hardware documentation for your device to determine if this tool is supported.

After configuring and saving a particular device's settings, the Device Configurator generates firmware for use in your application (see [Code generation](#)).



## 2 Stand-alone operation

## 2 Stand-alone operation

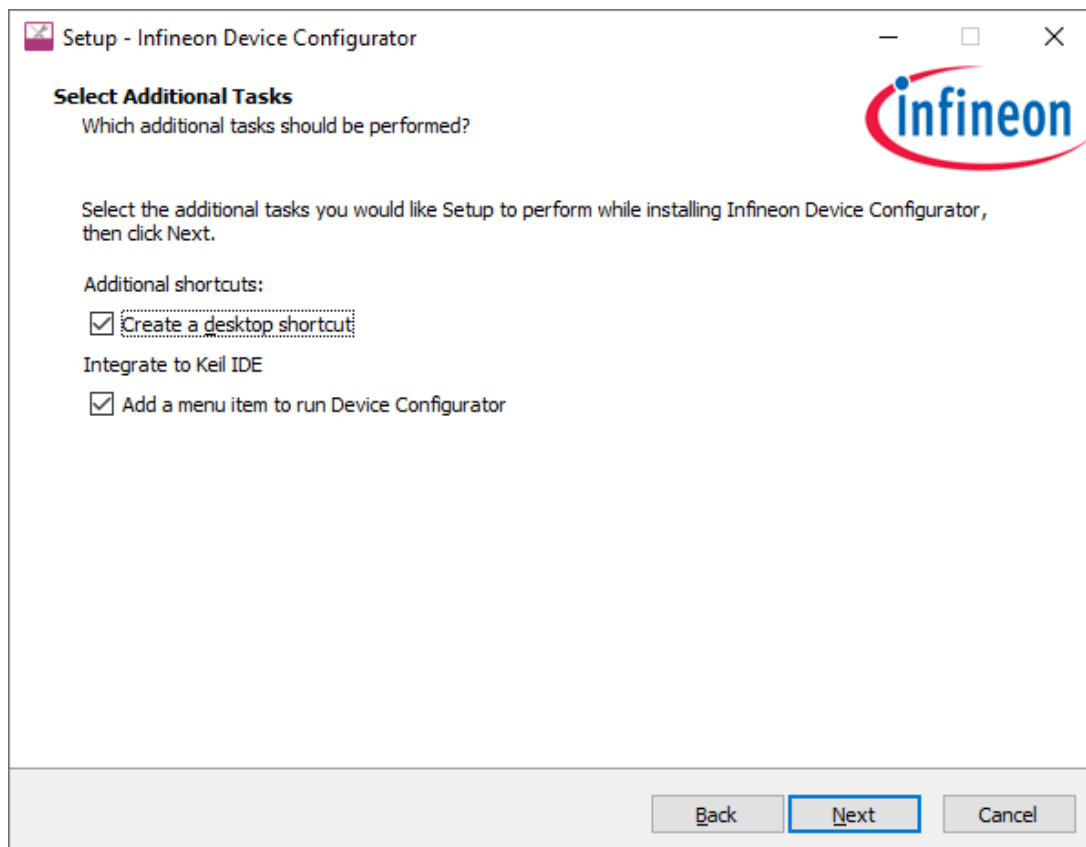
When the Device Configurator is used outside of the ModusToolbox™ environment, such as TLEXXX, it will not have the ModusToolbox™ application context. In the current stand-alone use case, the personalities and device-db are intended to be shipped in a CMSIS pack, typically distributed via Infineon Developer Center, that is used in µVision. The Device Configurator can be run as a GUI and via the command line.

### 2.1 Install Infineon Device Configurator

The installer for the Infineon Device Configurator is provided on Linux, macOS, and Windows. At this time, using the tool with Keil µVision IDE outside of the ModusToolbox™ environment is only supported on Windows as the Keil µVision IDE is only available for that operating system. The installer can be downloaded here: <http://www.infineon.com/idc>

Launch the `deviceconfigurator_<version>.<build #>_windows_x64.exe` installer and follow the installer wizard steps. We recommend installing as current user.

In most cases, the installer adds the **Run Device Configurator <version>** item to the **Tools** menu in the Keil µVision IDE when the **Add a menu item to run Device Configurator** task (by default) is selected on the Custom Installation page. If needed, you can [Add menu item to µVision manually](#). after the installer finishes.



**Note:** The installer has options to open the Device Configurator tool and the text file release notes. Opening the tool this way will not be in the context of a supported project loaded in Keil µVision, and tool will open without any configuration information with a message to that effect. We recommend deselecting this option on the installer, and opening the tool from the Keil µVision IDE in the context of a supported project. See the [Run Device Configurator](#) section.

## 2 Stand-alone operation

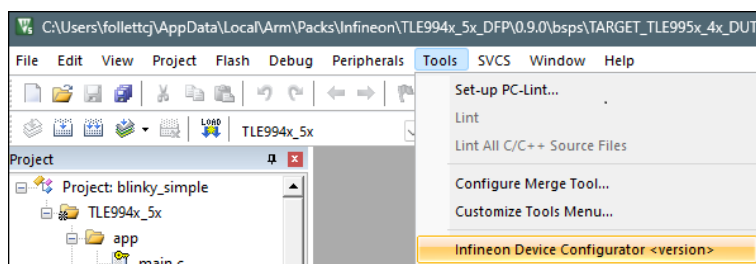
### 2.2 CMSIS Pack

In order to use devices such as TLE994x\_5x in the Keil  $\mu$ Vision IDE, you need to install the appropriate CMSIS pack. For example, the *MOTIX\_TLE994x\_5x\_PDL.pack*. You can find the pack on the Infineon Developer Center webpage: <http://www.infineon.com/idc>

### 2.3 Run Device Configurator

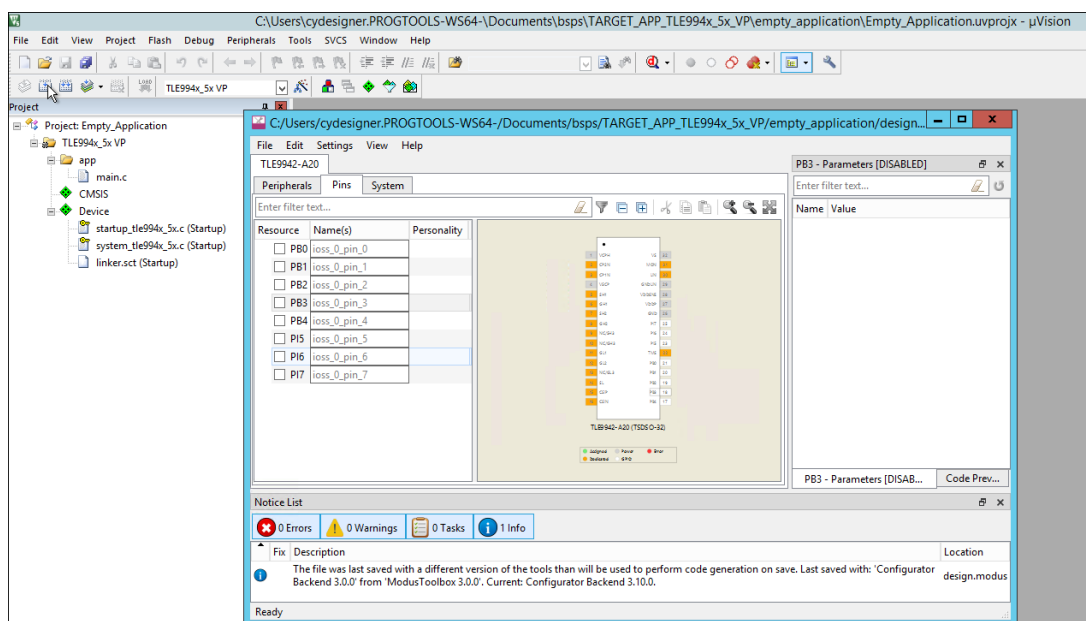
#### 2.3.1 Using the GUI

To run the Device Configurator GUI tool, open the Keil  $\mu$ Vision IDE with a project. Then, select the **Tools > Run Infineon Device Configurator <version>**. If you don't see the menu item, refer to [section 2.4](#).



**Note:** *This menu entry is for use with projects created out of supported CMSIS packs and **not** for applications exported from the ModusToolbox™ ecosystem. The CMSIS packs include the relevant files to be used via that menu entry. If you launch the Device Configurator without having a supported project loaded in Keil  $\mu$ Vision, it may fail to open, and there may not be any warning or message.*

With a supported project loaded, the IDE launches the Device Configurator with the opened *design.modus* file located in the current project:



When you save updates, the Device Configurator generates/updates source code in the *GeneratedSource* directory next to the *design.modus* file. See the [Code generation](#) section for more details.

## 2 Stand-alone operation

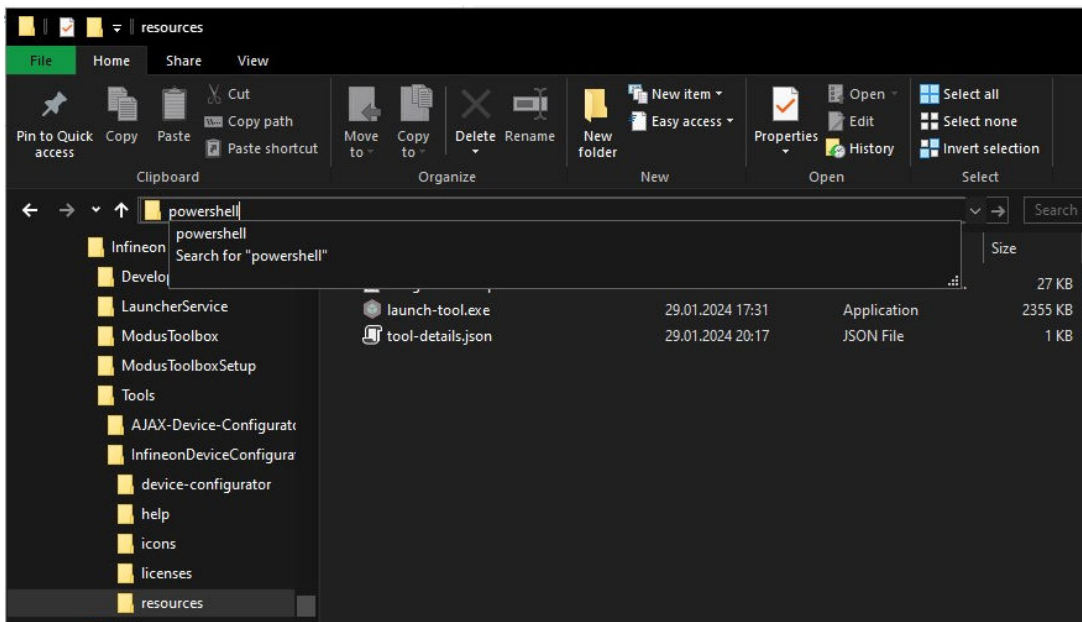
### 2.3.2 Using the command line

The Device Configurator executable can be run from the command line, and it also has a "cli" version of the executable as well. Running the executable from the command line can be useful as part of batch files or shell scripts to re-generate the source code based on the latest configuration settings. The exit code for the executable is zero if the operation is successful, or non-zero if the operation encounters an error. For more information about the command-line options, run the executable using the -h option.

### 2.4 Add menu item to µVision manually

In most cases, the installer adds the menu item to the Keil µVision IDE automatically. It also extracts a *IntegrateToKeil.ps1* script to the <installation folder>\resources folder. If the **Tools** menu in the Keil µVision IDE does not contain the items to launch the Infineon Device Configurator, you can execute the script manually.

To run the script, open Windows Explorer and navigate to the resources folder (e.g., *C:\Users\UserName\Infineon\Tools\InfineonDeviceConfigurator-<version>*), type "powershell" in address bar.



In the Windows PowerShell window, run the following command to change the Restricted (by default) execution policy to AllSigned to allow running signed script:

```
Set-ExecutionPolicy AllSigned -scope Process -Confirm:$False
```

And confirm changes by **[A]** and **[Enter]**.

## 2 Stand-alone operation

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Rybak\Infineon\Tools\InfineonDeviceConfigurator-4.10.0\resources> Set-ExecutionPolicy AllSigned -scope Process -Confirm:$False

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose you to the security risks described in the about_Execution_Policies help topic at https://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): A
```

Then run the script:

```
.\IntegrateToKeil.ps1
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\Rybak\Infineon\Tools\InfineonDeviceConfigurator-4.10.0\resources> Set-ExecutionPolicy AllSigned -scope Process -Confirm:$False

Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose you to the security risks described in the about_Execution_Policies help topic at https://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): A
PS C:\Users\Rybak\Infineon\Tools\InfineonDeviceConfigurator-4.10.0\resources> .\IntegrateToKeil.ps1

Do you want to run software from this untrusted publisher?
File
C:\Users\Rybak\Infineon\Tools\InfineonDeviceConfigurator-4.10.0\resources\IntegrateToKeil.ps1 is published by CN=Infineon Technologies AG, O=Infineon Technologies AG, L=Neubiberg, C=DE and is not trusted on your system. Only run scripts from trusted publishers.
[V] Never run [D] Do not run [R] Run once [A] Always run [?] Help (default is "D"): A
Adding registry value: 'Mag1 = --library=$Sprops.json,$Sdevice-info\device-db\props.json --design $Pdesign.modus'
Adding registry value: 'Mfg1 = 2304'
Adding registry value: 'Mtx1 = Run Device Configurator 4.10'
Adding registry value: 'Mid1 = '
Adding registry value: 'Mex1 = C:\Users\Rybak\Infineon\Tools\InfineonDeviceConfigurator-4.10.0\resources\launch-tool.exe'
PS C:\Users\Rybak\Infineon\Tools\InfineonDeviceConfigurator-4.10.0\resources>
```

The CLI can require confirmation of script run owned by Infineon Technologies publisher.

Please verify if the added Mag<Id>, Mfg<Id>, Mtx<Id>, Mid<Id> and Mex<Id> registry values are in the script output.

### 2.5 Remove menu item from µVision manually

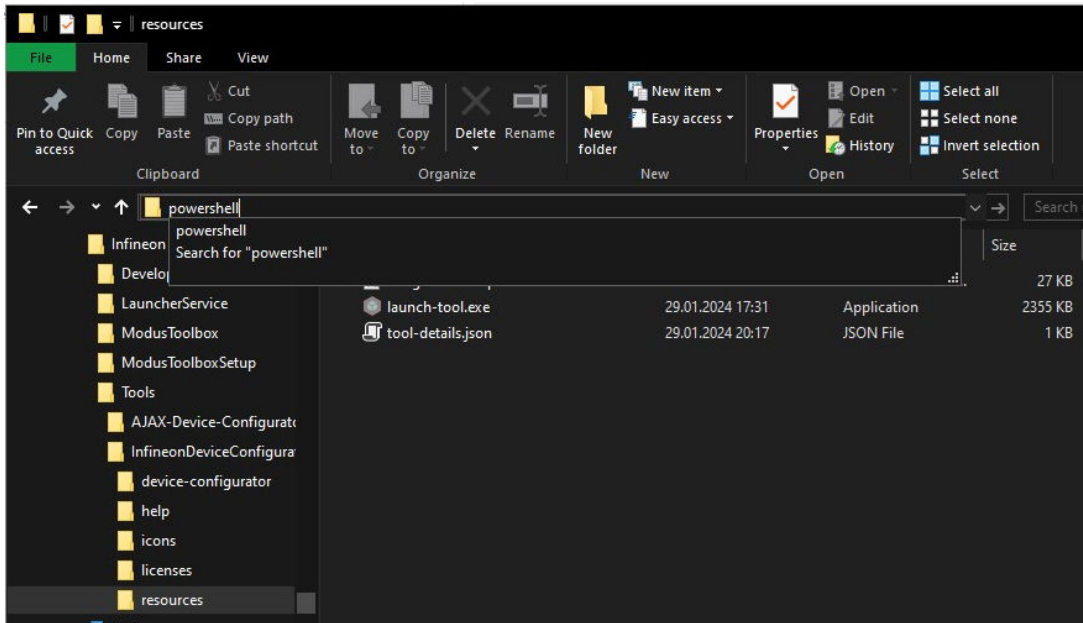
If you use the Device Configurator uninstaller, it will remove the **Run Device Configurator** menu item from the Keil µVision IDE automatically. To remove the item manually, follow these steps:

Open Windows Explorer and navigate to the resources folder (e.g.,

C:\Users\<UserName>\Infineon\Tools\InfineonDeviceConfigurator-<version>), type "powershell" in address bar.



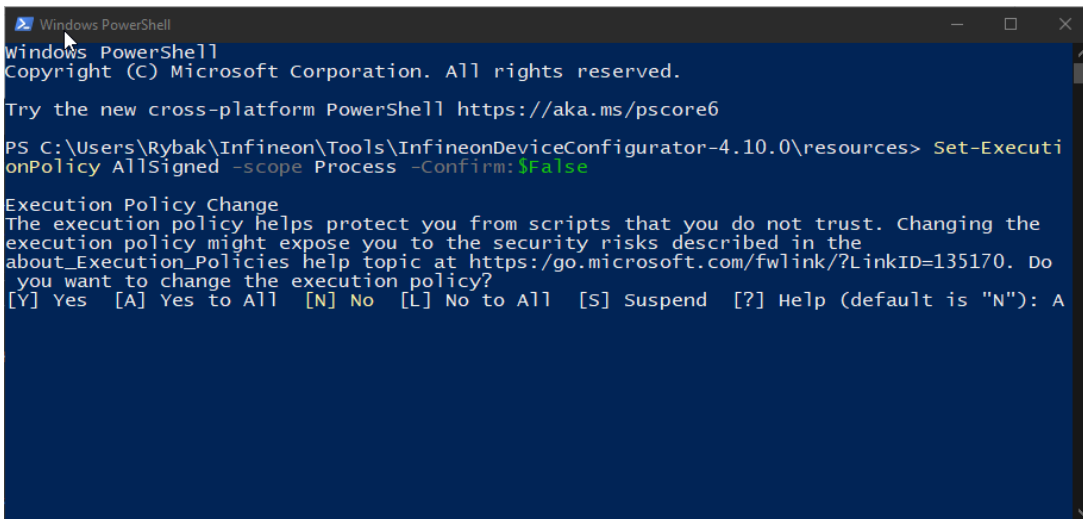
## 2 Stand-alone operation



In the Windows PowerShell window, run the following command to change the Restricted (by default) execution policy to AllSigned to allow running signed script:

```
Set-ExecutionPolicy AllSigned -scope Process -Confirm:$False
```

And confirm changes by **[A]** and **[Enter]**.

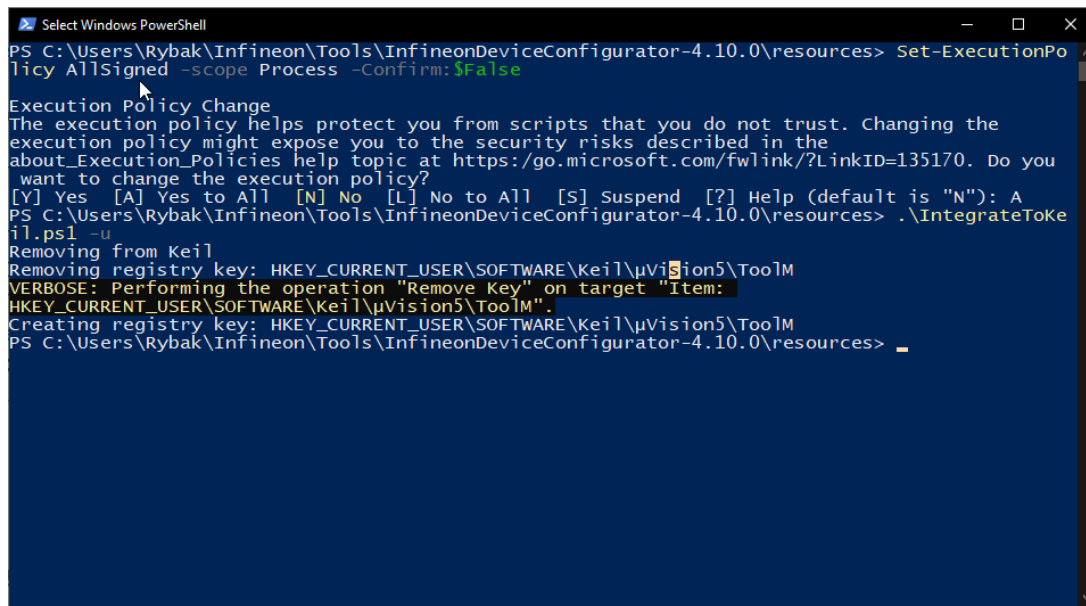


Then run the script:

```
.\IntegrateToKeil.ps1 -u
```

Please verify the script output:

## 2 Stand-alone operation



```
Select Windows PowerShell
PS C:\Users\Rybak\Infineon\Tools\InfineonDeviceConfigurator-4.10.0\resources> Set-ExecutionPolicy AllSigned -scope Process -Confirm:$False
Execution Policy Change
The execution policy helps protect you from scripts that you do not trust. Changing the execution policy might expose you to the security risks described in the about_Execution_Policies help topic at https://go.microsoft.com/fwlink/?LinkID=135170. Do you want to change the execution policy?
[Y] Yes [A] Yes to All [N] No [L] No to All [S] Suspend [?] Help (default is "N"): A
PS C:\Users\Rybak\Infineon\Tools\InfineonDeviceConfigurator-4.10.0\resources> .\IntegrateToKeil.ps1 -u
Removing from Keil
Removing registry key: HKEY_CURRENT_USER\SOFTWARE\Keil\µVision5\ToolM
VERBOSE: Performing the operation "Remove Key" on target "Item: HKEY_CURRENT_USER\SOFTWARE\Keil\µVision5\ToolM".
Creating registry key: HKEY_CURRENT_USER\SOFTWARE\Keil\µVision5\ToolM
PS C:\Users\Rybak\Infineon\Tools\InfineonDeviceConfigurator-4.10.0\resources>
```

### 3 Using with ModusToolbox™ application

## 3 Using with ModusToolbox™ application

As part of a ModusToolbox™ application, the Device Configurator *design.modus* file is an integral part of the board support package (BSP) and requires an association to an application in order to obtain device configuration information.

### 3.1 Installation

The Infineon Device Configurator is installed as part of the ModusToolbox™ tools package. There are no other requirements for installing this tool.

### 3.2 Launch the Device Configurator

You can launch the Device Configurator in various ways as described in this section; however, the tool's configuration (*design.modus*) file requires an association to an application in order to obtain device configuration information.

The *design.modus* file contains all the required hardware configuration information about the device for the application. When you save updates, the Device Configurator generates/updates source code in the *GeneratedSource* directory next to the *design.modus* file. Applications use the *design.modus* file and generated source code in future application builds.

#### 3.2.1 make command

As described in the [ModusToolbox™ tools package user guide](#) build system chapter, you can run numerous make commands in the application directory, such as launching the Device Configurator. After you have created a ModusToolbox™ application, navigate to the application directory and type the following command in the appropriate bash terminal window:

```
make device-configurator
```

This opens the Device Configurator using the application's *design.modus* file.

#### 3.2.2 VS Code and Eclipse

VS Code and Eclipse have tools to launch the Device Configurator from within an open application. Refer to the applicable user guide for more details:

- [VS Code for ModusToolbox™ user guide](#)
- [Eclipse IDE for ModusToolbox™ user guide](#)

#### 3.2.3 Executable (GUI)

If you don't have an application or if you just want to see what the configurator looks like, you can launch the Device Configurator GUI by running its executable as appropriate for your operating system (for example, double-click it or select it using the Windows **Start** menu). By default, it is installed here:

```
<install_dir>/ModusToolbox/tools_<version>/device-configurator-<version>
```

When launched this way, the Device Configurator opens without any configuration information. You can open a specific \*.modus file using the "Open" link or clicking **File > Open**, and then navigating to the location of the \*.modus file to open.

---

## 3 Using with ModusToolbox™ application

### 3.2.4 Executable (CLI)

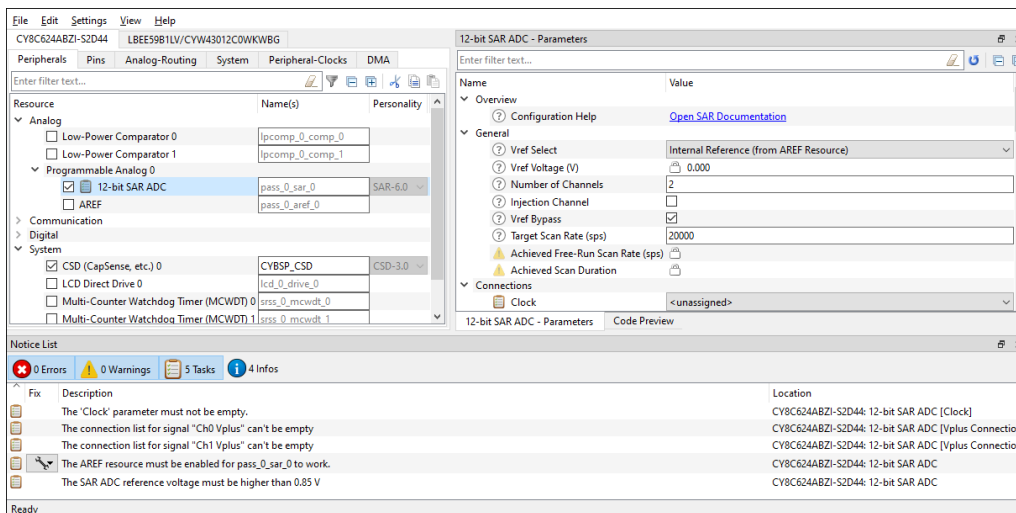
The Device Configurator executable can be run from the command line, and it also has a "cli" version of the executable as well. Running configurator executables from the command line can be useful as part of batch files or shell scripts to re-generate the source code based on the latest configuration settings. The exit code for the executable is zero if the operation is successful, or non-zero if the operation encounters an error. For more information about the command-line options, run the executable using the -h option.

## 4 Quick start

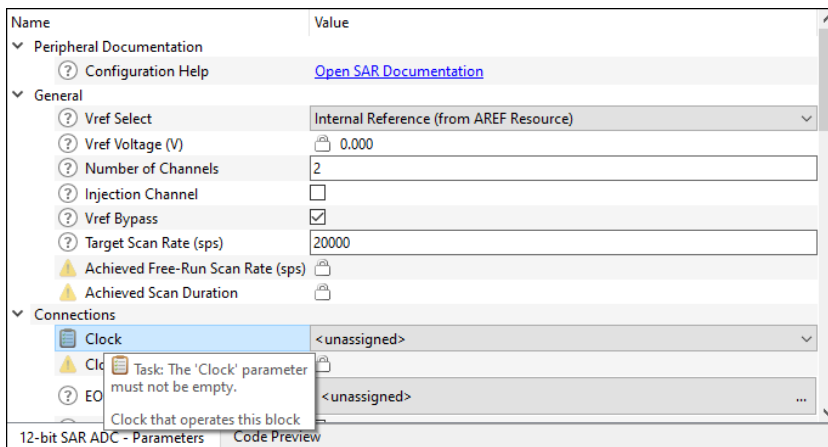
### 4 Quick start

This section provides a simple workflow for how to use the Device Configurator.

1. Open the Device Configurator as applicable to your environment; that is, in stand-alone operation in Keil  $\mu$ Vision or as part of a ModusToolbox™ application.
2. Enable a desired peripheral on the **Peripherals** tab by clicking the enable check box. Notice the **Parameters** pane becomes populated with fields.



3. Notice also that a new task may appear in the **Notice List** pane.
4. See **Icons** for descriptions of the various icons displayed in the Device Configurator.
- Double-click on a task to jump to the parameter that needs to be addressed.



5. Select appropriate **Parameter values** and the task should be removed from the Notice List.
6. When all tasks have been completed, select the **Code Preview** pane to see a preview of the code that will be generated upon saving.

## 4 Quick start

```
Code Preview
Enter search text...

/* NOTE: This is a preview only. It combines elements of the
 * cycfg_peripherals.c and cycfg_peripherals.h files located in the folder
 * C:/Users/folettetj/mtw2.4/5645/hw/Hello_World/libs/TARGET_Cy8CKIT-062-WIFI-BT/COMPONENT_BSP
 */

#include "cy_sar.h"
#include "cycfg_routing.h"
#include "cy_sysclk.h"
#if defined(CY_USING_HAL)
    #include "cyhal_hwmgr.h"
#endif //defined(CY_USING_HAL)

#define pass_0_sar_0_HW SAR
#define pass_0_sar_0_IRQ pass_interrupt_sar_IRQn
#define pass_0_sar_0_CTL ((uint32_t)CY_SAR_VREF_PWR_100 | (uint32_t)CY_SAR_VREF_SEL_BGR | (uint32_t)CY_SAR_SAMPLE_CTRL_EOS_DSI_OUT_EN_Msk | (uint32_t)CY_SAR_CH0_CONFIG ((uint32_t)SAR0_VELU0_PORT << SAR_CHAN_CONFIG_POS_PORT_ADDR_P
#define pass_0_sar_0_CH0_CONFIG ((uint32_t)SAR0_VELU0_PORT << SAR_CHAN_CONFIG_POS_PORT_ADDR_P
#define pass_0_sar_0_CH1_CONFIG ((uint32_t)SAR0_VELU0_PORT << SAR_CHAN_CONFIG_POS_PORT_ADDR_P
#define pass_0_sar_0_VREF_MV 1200UL

const cy_stc_sar_config_t pass_0_sar_0_config = {
    <
12-bit SAR ADC - Parameters    Code Preview
>
```

7. Use the [Resources tabs](#) to enable and configure other resources as needed in the same manner as peripherals.
8. Save the \*.modus file to generate source code.

The Device Configurator generates code into a "GeneratedSource" directory in your Eclipse IDE application, or in the same location you saved the \*.modus file for non-Eclipse IDE applications. That directory contains the necessary source (.c) and header (.h) files that use the relevant driver APIs to configure the hardware. Application code then uses this code to configure the system.

9. Use the appropriate API in your application code.

---

## 5 Code generation

### 5 Code generation

The Device Configurator generates structures, defines, and initialization code for the blocks on the chip. All generated code is located in the *GeneratedSource* folder next to the \*.*modus* file. Refer to the Peripheral Driver Library (PDL) API Reference for more information about this code. Each enabled resource has a link to the specific driver documentation in the [Parameters](#) pane.

**Note:** *The Device Configurator generates code based on the hardware resources that are enabled. If a resource is not enabled, no configuration will be generated for it. This means the resource will retain its default reset state. In most cases, this is powered off. However, some features are enabled by default, such as debug connectivity. To disconnect these features, you must call the appropriate API functions to turn the feature off.*

The defines and structures are all named based on the resource that created it. In general, these have the form <resource-name>\_config. These structures can be passed to the PDL functions that are responsible for configuring the hardware block.

The functions are specific to a resource category and have names of the form init\_cycfg\_<resource-category>. The init function for a particular resource type is located in *GeneratedSource/cycfg\_<resource-category>.h*. There are also the *cycfg.h* and *cycfg.c* files. Include the *cycfg.h* file in your application to access the generated header files. The *cycfg.c* file implements init\_cycfg\_all(), which calls all other generated functions, for example init\_cycfg\_pins().

The resource types include:

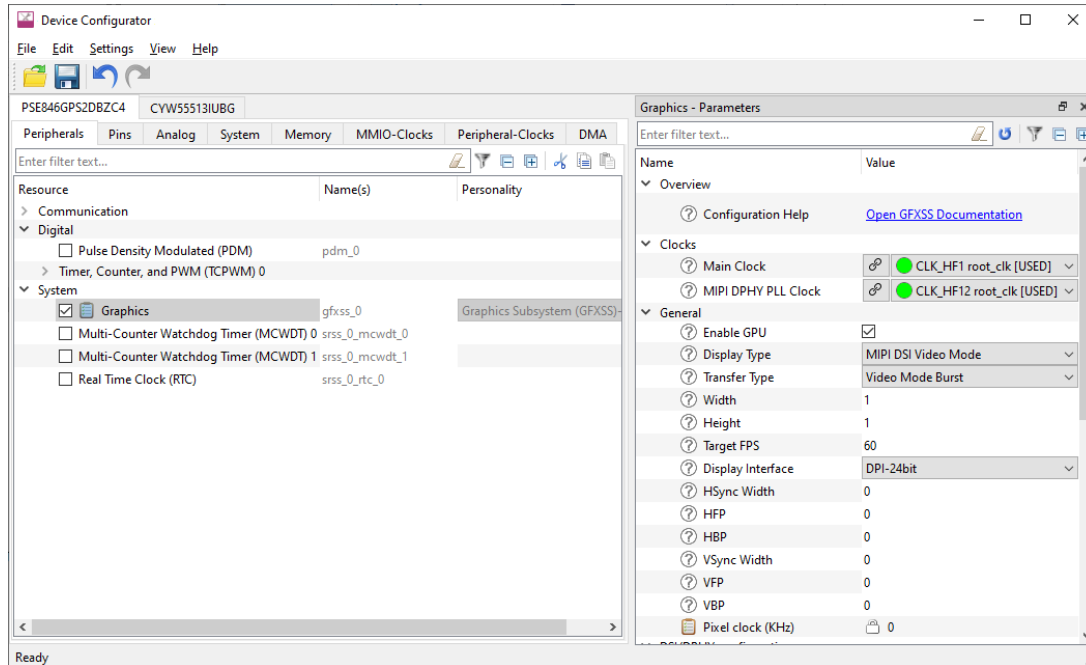
- Clocks: Peripheral clocks
- Connectivity: Configuration of the programmable analog and digital routing resources
- Peripherals: Fixed function analog and digital peripherals
- System: Overall configuration function to setup all power and clock options

It is up to you to make use of the generated code based on the application's needs. This can be done as part of the application's main() loop.

## 6 GUI description

### 6 GUI description

The Device Configurator GUI contains [Menus](#), [Icons](#), [Resources tabs](#), and several [Panels](#) used to configure MCU peripherals.



#### 6.1 Menus

##### File

- **Open** – Opens an existing \*.modus file. The current file, if any, will be closed.
- **Close** – Closes the current file. If there are pending changes, you will be prompted to save the file.
- **Save** – Saves the current file and generates code for the related application. If there are errors in the application, a dialog will indicate such. The file will still be saved.
- **Open in System Explorer** – This opens your computer's file explorer tool to the folder that contains the \*.modus file.
- **Recent Files** – Shows up to five recent files that you can open directly.
- **Update All Personalities** – Use this item to update all resource Personalities (see [Resources tabs](#)). This opens a dialog showing all the personalities in the design.  
For example, if you load a \*.modus file made with an older device support library, there might be many warnings in the [Notice List](#) to update personalities or that a personality is no longer supported. Each warning must be addressed, and doing so one at a time can be annoying. The Update All Personalities menu item addresses them all at once.
- **Exit** – Closes the tool. You will be prompted to save any pending changes.

##### Edit

- **Undo** – Undoes the last action or sequence of actions.
- **Redo** – Redoes the last undone action or sequence of undone actions.

##### Settings

- **ModusToolbox™ Settings:** This opens the Centralized Settings tool, an editor that allows you to configure a wide range of settings for your environment, such as proxy settings, content modes, and manifest DB settings. See the Settings tool user guide ([insert link](#)) for more details on specific features.



## 6 GUI description

### View

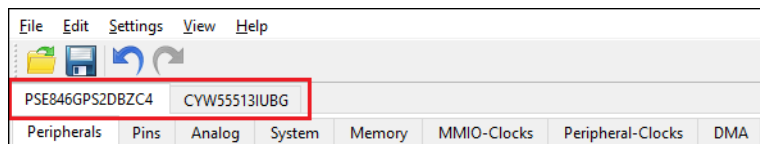
Contains toggles to hide or show different [Panels](#). All panels are shown by default. There is also a command to show or hide the Toolbar (hidden by default) and reset the view of the configurator to the default.

### Help

- **View Help** – Opens this document.
- **About** – Opens the About box for version information, with links to open Infineon.com and the current session log file.

## 6.2 Device tabs

The Device Configurator can be used to configure multiple devices. All devices in the BSP display in top-level tabs above the [Resources tabs](#).



All the settings for each **Device** tab are configured separately.

**Note:** *If you need to update or change devices, the best approach is to close the Device Configurator and then use the BSP Assistant. Refer to the [BSP Assistant user guide](#) for details.*

## 6.3 Resources tabs







See [Resources tabs](#).

## 6.4 Panes

See [Panels](#).

## 6.5 Icons

When configuring various options with this tool, you will see the following icons:

Icon	Description
	Indicates there is a tooltip. Hover over the icon to display a brief message about the setting.
	Enables or disables a specific resource.
	There may be occasions where an error, warning, task, or info icon displays for an enabled resource. See <a href="#">Notice List</a> pane for more details.
	When shown in Parameters, this indicates that it is a read-only field. When shown for a Resource, this indicates the resource is locked and disabled. There is a tooltip explaining why the resource is locked.
	When shown for a Resource, this indicates the resource is locked and enabled. There is a tooltip explaining why the resource is locked.
	After assigning a signal, clicking this icon jumps to the linked resource(s).

## 7 Resources tabs

### 7 Resources tabs

For some device families, the Device Configurator contains several tabs, each of which provides access to specific resources. Different devices have different resources tabs. However, for some device families, there are no separate tabs; resources are shown in a single pane, sometimes under collapsible trees.

When you enable a resource, or select an enabled resource, the [Parameters pane](#) displays various configuration options. As described under [Icons](#), some enabled resources may contain errors, warnings, tasks, or infos that indicate some action might be required to resolve the issue. See [Notice List](#) for more details.

**Note:** *Only the tabs relevant for a selected device are displayed, so some of the tabs may not be included for some devices.*

- [Solutions tab](#) – Options to configure multiple hardware blocks.
- [Peripherals](#) – Options to enable any of the analog, digital, system, and communication hardware capabilities of the chip that are not related to the platform.
- [Pins](#) – Options for all the pin related resources.
- [Analog-Routing tab](#) – This tab shows all the analog resources, whether enabled or not, and how they connect. It also allows you to edit routes.
- [System tab](#) – Options for chip-wide configuration settings such as system clocks, power management, and debug interfaces.
- [Memory tab](#) – Options to allocate memory to various cores.
- [MMIO-Clocks tab](#) – Options for all the MMIO clocks.
- [Peripheral-Clocks tab](#) – Options for all the peripheral clocks.
- [DMA tab](#) – Provides configuration of the DMA channel and transaction descriptors.

#### Tab Features

Sections with diagrams, such as Pins, Analog-Routing, and System, include **Zoom** and **Fit to size** commands to resize the diagram as needed. You can also press and hold the **[Ctrl]** key and use the mouse scroll wheel to zoom in and out.

If you zoom the image larger than the frame area, scroll bars appear to move to different area of the diagram. You can also press and hold the **[Alt]** key with the mouse button to use the pan tool.

Each of the tabs (except the Analog-Routing tab) also has the following features:

- **Filter** – The Resource column shows all available resources in an expandable tree. The filter box above the list of peripherals allows you to limit the peripherals shown in the tree as well as a Hide disabled resources filter button. There are also Expand and Collapse commands.
- **Cut, Copy, Paste** – Use these commands to move and copy settings from one resource of the same type to another.
  - When you use **Cut**, the settings will be copied to the clipboard, and the selected resource will be disabled.
  - When you use **Copy**, the settings will just be copied to the clipboard.
  - When you use **Paste**, the selected resource will be enabled if needed. The selected resource must support the same Personality name and version as the cut/copied resource.
- **Name(s)** – This displays the current resource name(s). This is an editable field where you can specify optional, alternate names for this resource. This is also used in generated code.

---

## 7 Resources tabs

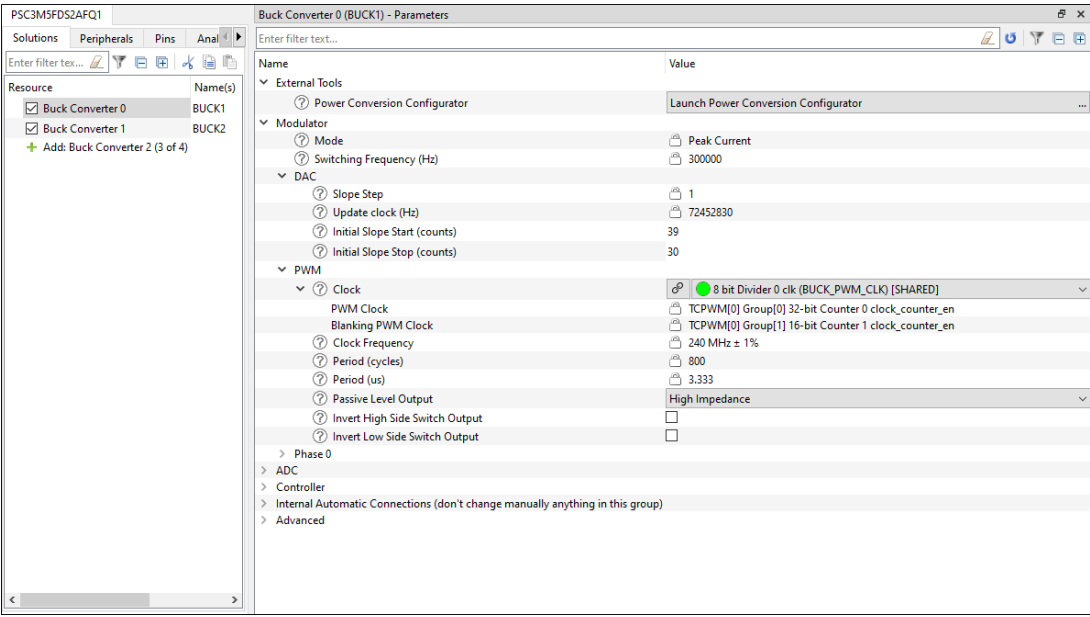
**Note:** *Enter any string in this field. The tool converts the name into a legal C identifier and replaces non-legal characters with underscores. If entering more than one name, use a coma-separated list.*

- **Personality** – Each resource has a "Personality" file that contains the information for the given resource.
  - Some peripherals, such as Serial Communication Block (SCB) and Timer, Counter, Pulse Width Modulator (TCPWM), have a pull-down menu to select a specific personality, such as UART, SPI, or I<sup>2</sup>C.
  - Some peripherals have multiple personality versions from which you can select.
  - Some peripherals have a read-only field that only shows the name of this resource's personality file.

7 Resources tabs

7.1 Solutions tab

The **Solutions** tab displays for supported devices that provide configurations for a multi-resource solution. This tab provides configurable elements that generally consist of multiple hardware blocks. These elements pre-configure many of the low-level details and just present a higher-level interface to configure the solution-level element itself. All hardware that makes up the solution is configured in one **Parameters** pane, instead of having individual blocks on different panes. This tab allows you to enter one or more **Name(s)** for the resource. It also shows the selected **Personality**, where applicable.

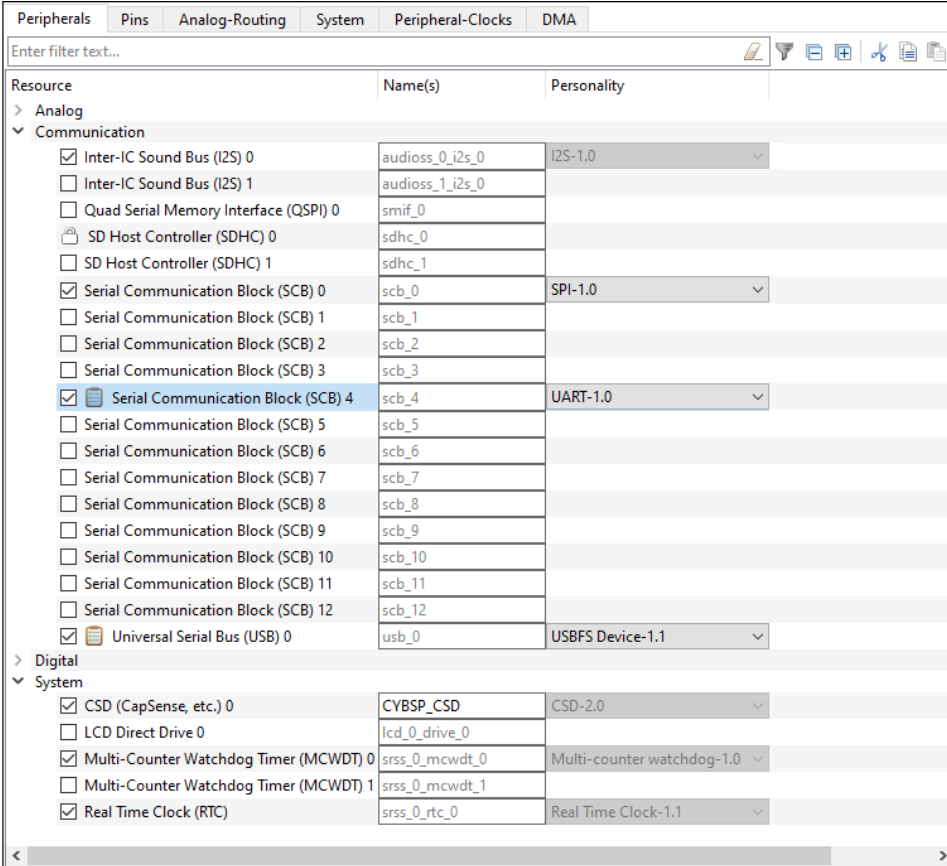


## 7 Resources tabs

### 7.2 Peripherals

The **Peripherals** tab/tree is where you enable various analog, digital, system, and communication peripherals for the device to include in your application. The filter box and the hide disabled button above the list of peripherals allows you to limit the resources shown in the tree. This tab allows you to enter one or more **Name(s)** for the resource. It also shows the selected **Personality**, where applicable.

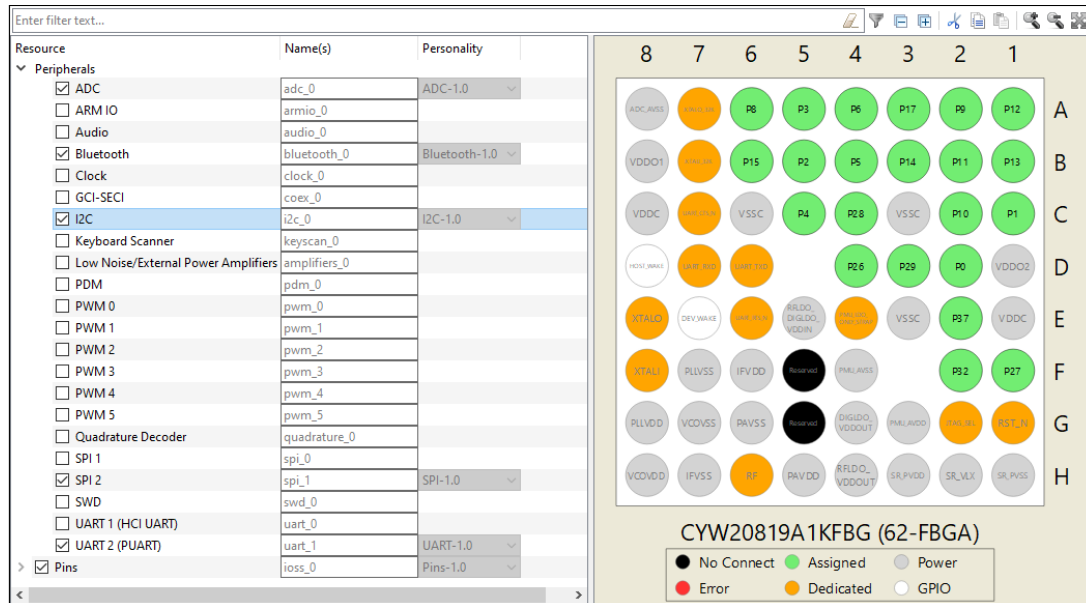
#### 7.2.1 Device families with tabs



Resource	Name(s)	Personality
> Analog		
> Communication		
<input checked="" type="checkbox"/> Inter-IC Sound Bus (I2S) 0	audioss_0_i2s_0	I2S-1.0
<input type="checkbox"/> Inter-IC Sound Bus (I2S) 1	audioss_1_i2s_0	
<input type="checkbox"/> Quad Serial Memory Interface (QSPI) 0	smif_0	
<input type="checkbox"/> SD Host Controller (SDHC) 0	sdhc_0	
<input type="checkbox"/> SD Host Controller (SDHC) 1	sdhc_1	
<input checked="" type="checkbox"/> Serial Communication Block (SCB) 0	scb_0	SPI-1.0
<input type="checkbox"/> Serial Communication Block (SCB) 1	scb_1	
<input type="checkbox"/> Serial Communication Block (SCB) 2	scb_2	
<input type="checkbox"/> Serial Communication Block (SCB) 3	scb_3	
<input checked="" type="checkbox"/> Serial Communication Block (SCB) 4	scb_4	UART-1.0
<input type="checkbox"/> Serial Communication Block (SCB) 5	scb_5	
<input type="checkbox"/> Serial Communication Block (SCB) 6	scb_6	
<input type="checkbox"/> Serial Communication Block (SCB) 7	scb_7	
<input type="checkbox"/> Serial Communication Block (SCB) 8	scb_8	
<input type="checkbox"/> Serial Communication Block (SCB) 9	scb_9	
<input type="checkbox"/> Serial Communication Block (SCB) 10	scb_10	
<input type="checkbox"/> Serial Communication Block (SCB) 11	scb_11	
<input type="checkbox"/> Serial Communication Block (SCB) 12	scb_12	
<input checked="" type="checkbox"/> Universal Serial Bus (USB) 0	usb_0	USBFS Device-1.1
> Digital		
> System		
<input checked="" type="checkbox"/> CSD (CapSense, etc.) 0	CYBSP_CSD	CSD-2.0
<input type="checkbox"/> LCD Direct Drive 0	lcd_0_drive_0	
<input checked="" type="checkbox"/> Multi-Counter Watchdog Timer (MCWDT) 0	srss_0_mcwdt_0	Multi-counter watchdog-1.0
<input type="checkbox"/> Multi-Counter Watchdog Timer (MCWDT) 1	srss_0_mcwdt_1	
<input checked="" type="checkbox"/> Real Time Clock (RTC)	srss_0_rtc_0	Real Time Clock-1.1

## 7 Resources tabs

### 7.2.2 Device families without tabs



## 7.3 Pins

The **Pins** tab/tree is where you enable all the pin related resources. All available pins are shown in an expandable tree, arranged by port number. The filter box and the hide disabled button above the list of pins allows you to limit the pins shown in the tree. This tab allows you to enter one or more **Name(s)** for the resource. It also shows the selected **Personality**, where applicable.

The interactive pin package diagram shows the different states of the pins; there is a legend on the diagram. You can enable/disable a pin by double-clicking it in the diagram.

Pin states are shown in different colors:

- Black – No connect
- White – Disabled
- Green – Enabled
- Grey – Power/ground pins
- Orange – Fixed function pins
- Red – Error state
- Semi-transparent – The hardware resource's enabled state has been locked.

7 Resources tabs

7.3.1 Device families with tabs

Device Configurator interface for CY8C624ABZI-S2D44 (124-BGA) showing resource assignment for Port 0 and Port 1. The interface includes a filter text field, a list of resources with checkboxes, and a pin assignment table.

Resource	Name(s)	Personality
Port 0		
<input checked="" type="checkbox"/> P0[0]	CYBSP_WCO_IN	Pin-1.1
<input checked="" type="checkbox"/> P0[1]	CYBSP_WCO_OUT	Pin-1.1
<input type="checkbox"/> P0[2]	ioss_0_port_0_pin_2	
<input type="checkbox"/> P0[3]	ioss_0_port_0_pin_3	
<input checked="" type="checkbox"/> P0[4]	CYBSP_SW2,CYBSP_USER_BTN1,CYBSP_USER_BTN	Pin-1.1
<input type="checkbox"/> P0[5]	CYBSP_LED_RGB_GREEN,CYBSP_USER_LED4	
Port 1		
<input checked="" type="checkbox"/> P1[0]	CYBSP_CSD_RX,CYBSP_CS_RX,CYBSP_CS_TX_RX	Pin-1.1
<input type="checkbox"/> P1[1]	CYBSP_LED_RGB_RED,CYBSP_USER_LED3	
<input type="checkbox"/> P1[2]	ioss_0_port_1_pin_2	
<input type="checkbox"/> P1[3]	ioss_0_port_1_pin_3	
<input type="checkbox"/> P1[4]	CYBSP_SW4,CYBSP_USER_BTN2	
<input type="checkbox"/> P1[5]	CYBSP_LED8,CYBSP_USER_LED1,CYBSP_USER_LED	

Pin assignment table (13-1):

Pin	Resource	Personality
13	VDD0A	Power
12	VDD0A	Power
11	P1001	Assigned
10	P1004	Assigned
9	P1007	Assigned
8	P1010	Assigned
7	P1013	Assigned
6	P1016	Assigned
5	P1019	Assigned
4	P1022	Assigned
3	P1025	Assigned
2	P1028	Assigned
1	P1031	Assigned

7.3.2 Device families without tabs

Device Configurator interface for CYW20819A1KFBG (62-FBGA) showing resource assignment for Pins. The interface includes a filter text field, a list of resources with checkboxes, and a pin assignment table.

Resource	Name(s)	Personality
Peripherals		
<input checked="" type="checkbox"/> Pins	ioss_0	Pins-1.0
<input type="checkbox"/> DEV_WAKE	ioss_0_pin_40	
<input type="checkbox"/> HOST_WAKE	ioss_0_pin_41	
<input checked="" type="checkbox"/> P0	CYBSP_D2,SW3,USER_BUTTON1	Pin-1.0
<input checked="" type="checkbox"/> P1	CYBSP_RST	Pin-1.0
<input checked="" type="checkbox"/> P2	CYBSP_D4	Pin-1.0
<input checked="" type="checkbox"/> P3	CYBSP_D5	Pin-1.0
<input checked="" type="checkbox"/> P4	CYBSP_D6	Pin-1.0
<input checked="" type="checkbox"/> P5	CYBSP_D7	Pin-1.0
<input checked="" type="checkbox"/> P6	CYBSP_D11,SPI2_MOSI	Pin-1.0
<input checked="" type="checkbox"/> P8	CYBSP_A0,CYBSP_THERM_TEMP_SENSE	Pin-1.0
<input checked="" type="checkbox"/> P9	CYBSP_D13,SPI2_CLK	Pin-1.0
<input checked="" type="checkbox"/> P10	CYBSP_A2	Pin-1.0
<input checked="" type="checkbox"/> P11	CYBSP_RSVD9,SPI2_CS	Pin-1.0
<input checked="" type="checkbox"/> P12	CYBSP_A4	Pin-1.0
<input checked="" type="checkbox"/> P13	CYBSP_A5	Pin-1.0
<input checked="" type="checkbox"/> P14	CYBSP_D8	Pin-1.0
<input checked="" type="checkbox"/> P15	CYBSP_D10	Pin-1.0
<input checked="" type="checkbox"/> P17	CYBSP_D12,SPI2_MISO	Pin-1.0
<input checked="" type="checkbox"/> P26	LED2	Pin-1.0
<input checked="" type="checkbox"/> P27	LED1	Pin-1.0
<input checked="" type="checkbox"/> P28	I2C_SCL	Pin-1.0
<input checked="" type="checkbox"/> P29	I2C_SDA	Pin-1.0
<input checked="" type="checkbox"/> P32	CYBSP_D1,UART_TX	Pin-1.0
<input checked="" type="checkbox"/> P37	CYBSP_D0,UART_RX	Pin-1.0

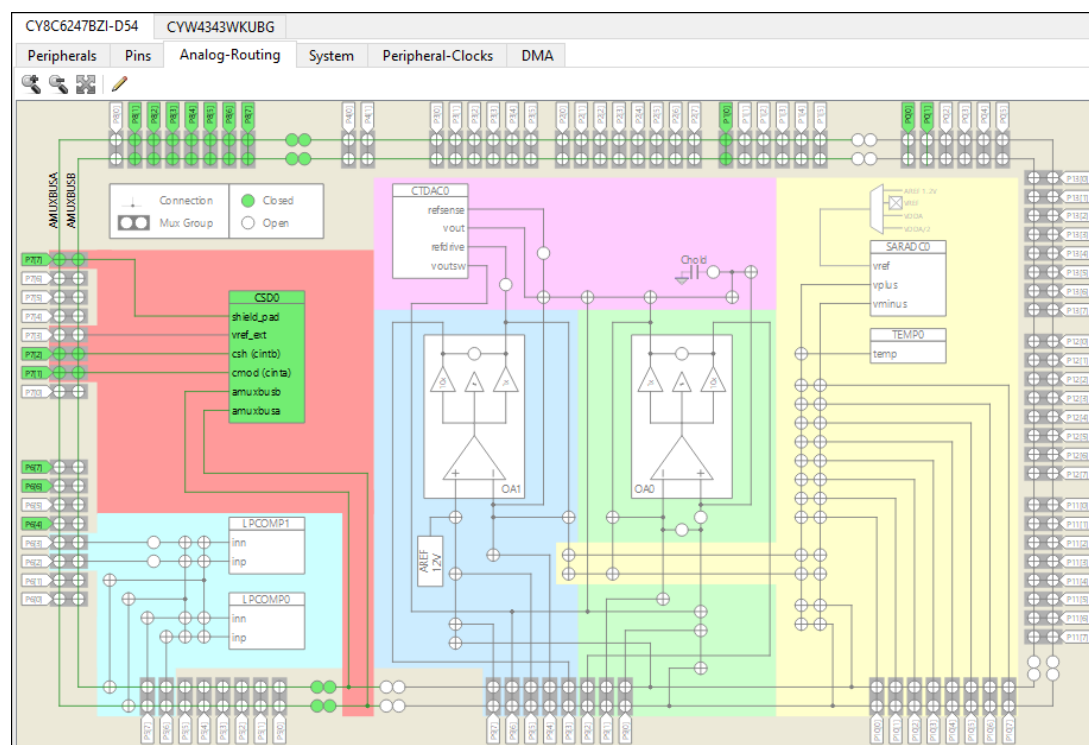
Pin assignment table (8-1):


Pin	Resource	Personality
8	ADC_AVSS	Power
7	P8	Assigned
6	P3	Assigned
5	P6	Assigned
4	P17	Assigned
3	P9	Assigned
2	P12	Assigned
1	P13	Assigned

## 7 Resources tabs

### 7.4 Analog-Routing tab

The Analog-Routing tab shows the various analog resources in your application. Enabled resources are green.



The **Edit**  command opens the [Analog Route Editor](#).

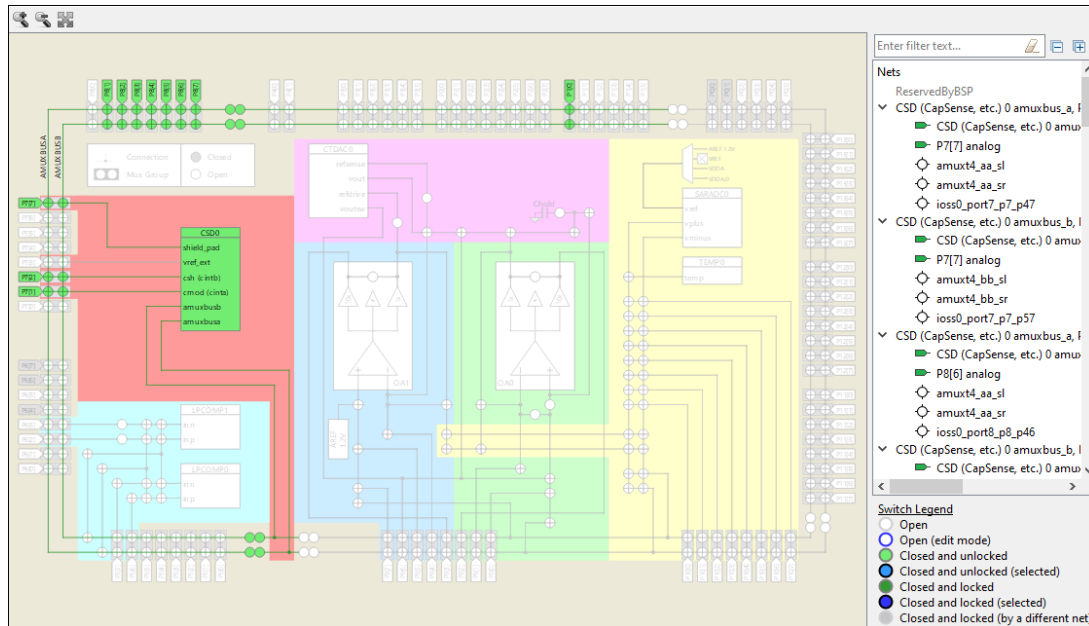


## 7 Resources tabs

### 7.4.1 Analog Route Editor

The Analog Route Editor allows you to manually edit the routing of analog resources in your application. It also provides the ability to lock-down all or some of the results.

**Note:** *The Analog Editor can some times be unresponsive during route recalculation. Please wait for it to finish algorithm execution.*

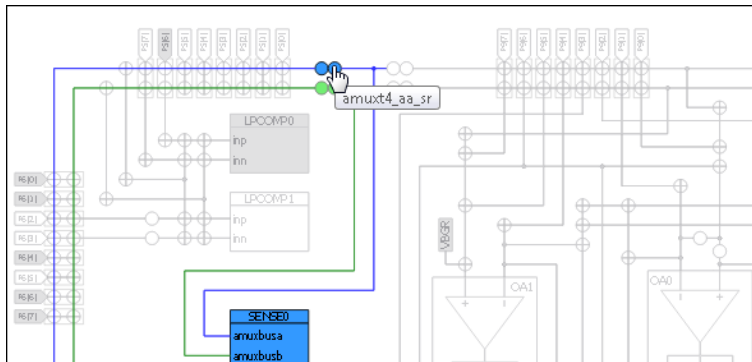


**Note:** *If there are configuration errors, complete routing results will not be available; only locked resources. If you open the Analog Editor in this error state, a warning message will display. You can still lock and unlock switches, but you won't get complete routing results as long as the configuration has errors.*

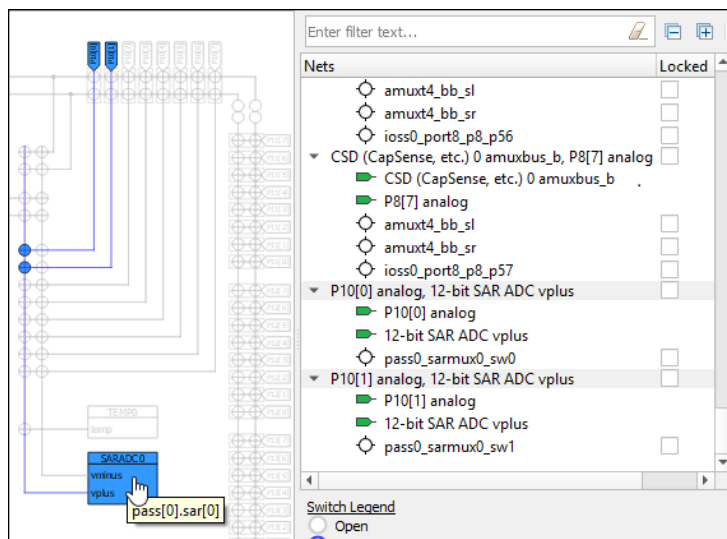
## 7 Resources tabs

### 7.4.1.1 Select a resource

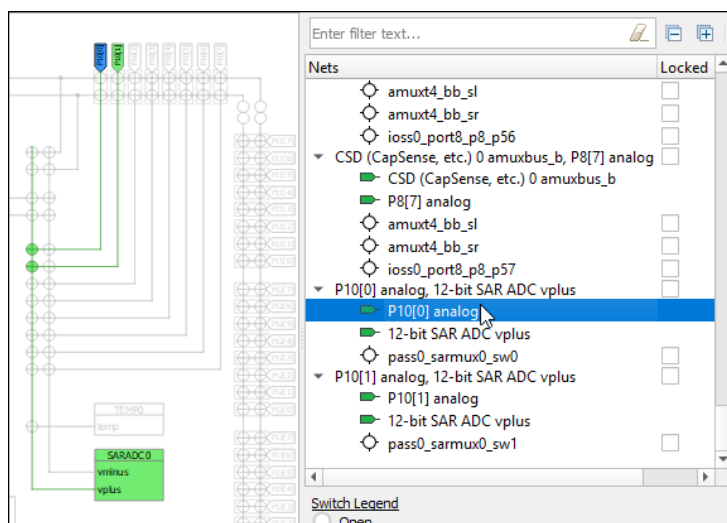
To select an analog resource, click on it. Any enabled (green) element in the tree can be selected. The resource and the associated route(s) become blue. Also, the **Edit Route** command appears on the toolbar. See [Edit Route](#).



At the same time, the selected analog resource(s) is highlighted in the Nets tree.



You can also select items in the tree to highlight them in the diagram.



---

## 7 Resources tabs

### 7.4.1.2 Edit Route

With an editable analog resource selected, click the **Edit Route** command to enable edit mode. If multiple routes are selected, a pull-down menu displays to select the route to edit. You cannot edit multiple routes at the same time.

In edit mode, the net tree shows only the applicable route entries, and you cannot select resources using the tree. However, the lock/unlock check boxes remain enabled for use. The inactive switches change color to indicate they can be selected to use for the route being edited.

Route changes are live with updates applied automatically as you make changes. Selecting a switch adds it to the current route in a locked state and the route tree is updated to reflect the modifications.

If a change results in an error, a message displays. The routes are automatically rolled back to the previous state, so you will lose at most the last invalid change.

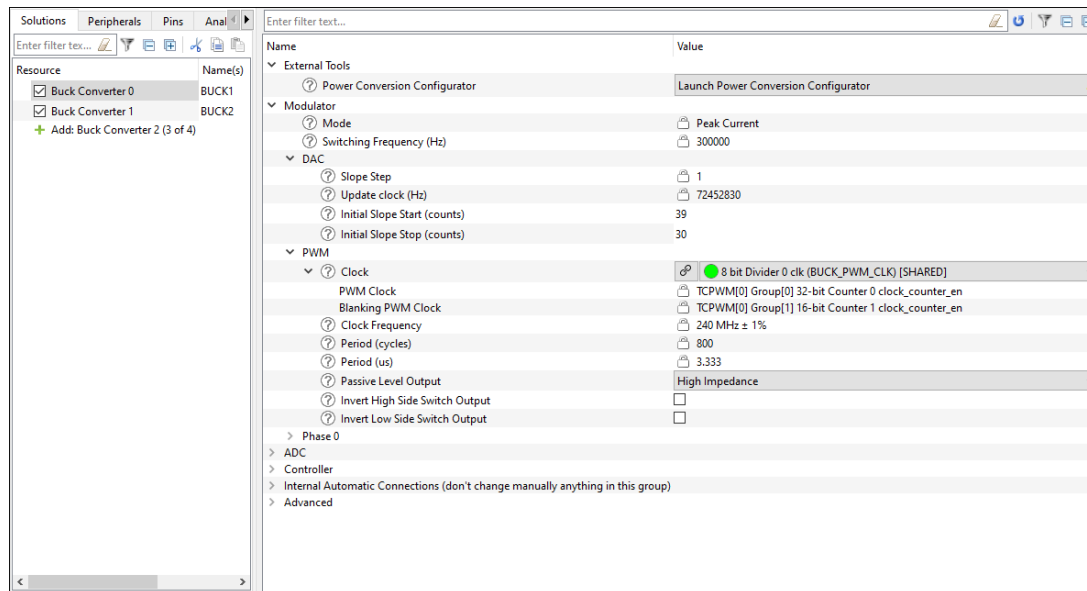
The toolbar shows the **Finish edit** command to return the editor to selection mode.

**Note:** *If a route is edited so that it uses switches associated with a location where no personalities are instantiated, you must manually power on the containing block at startup in order for the switches to function. Refer to the PDL API Reference Guide and the Device Technical Reference Manual for more details.*

## 7 Resources tabs

### 7.5 System tab

The **System** tab provides access to system-level items, such as system clocks, power management, and debug interfaces. All available resources are shown in an expandable tree. The filter box and the hide disabled button above the list of resources allows you to limit the items shown in the tree. This tab allows you to enter one or more **Name(s)** for the resource. It also shows the selected **Personality**, where applicable.



The interactive clock diagram shows all the system clocks and how they connect to each other. You can enable/disable a clock by double-clicking it in the diagram. Enabled clocks are green, disabled clocks are white, and clocks in error state are red.

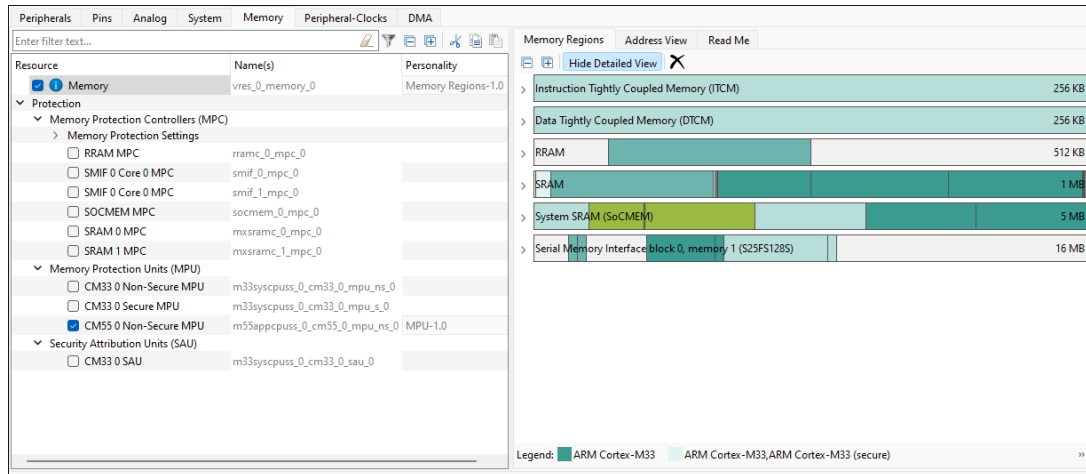
**Note:** The semi-transparent (faded) elements in the diagram indicate that their enabled state is locked.

## 7 Resources tabs

### 7.6 Memory tab

The **Memory** tab becomes visible for supported devices to configure and visualize memory regions for each core in the device. Similar to other tabs, there is a **Resource** section to enable/disable the Memory, as well as the **Protection** section for the Memory Protection Controllers (MPC), Memory Protection Units (MPU), and Security Attribution Units (SAU). This tab allows you to enter one or more **Name(s)** for the resource. It also shows the selected **Personality**, where applicable.

The main section of this tab contains three subtabs: **Memory Regions**, **Address View**, and **Read Me**

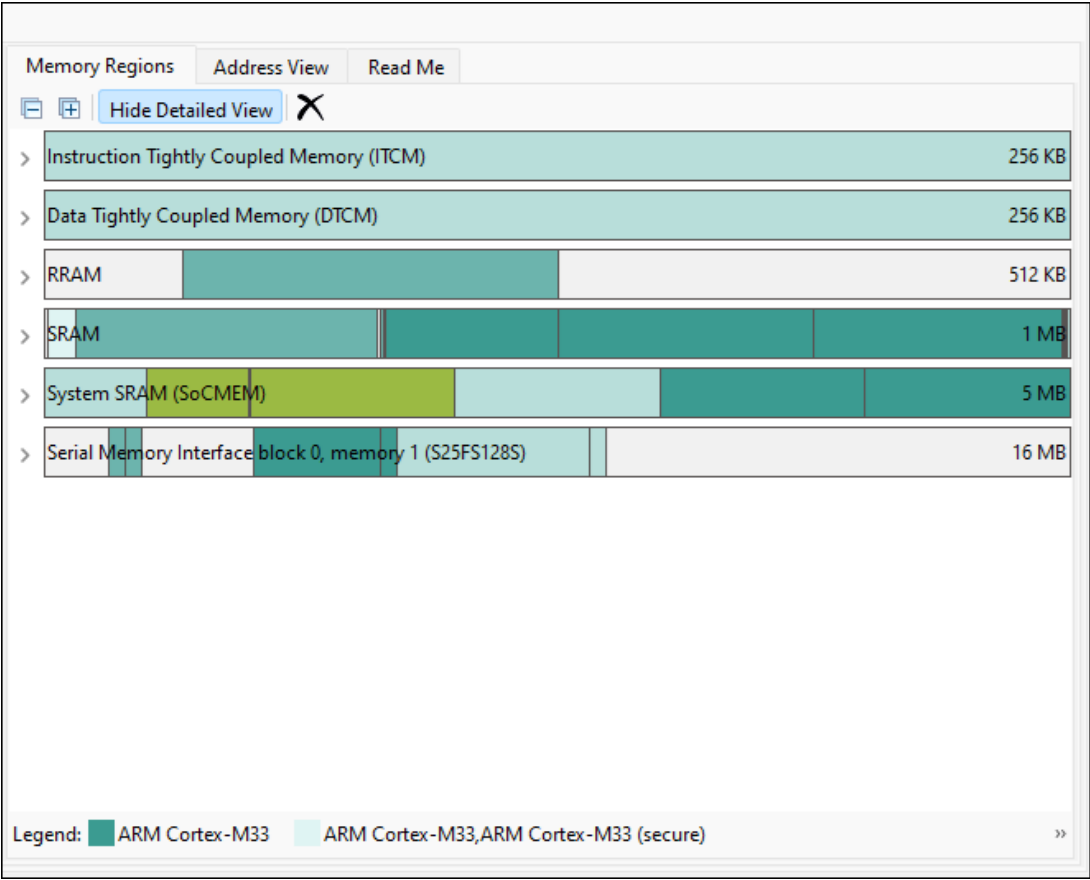


**Note:** The **Parameters** pane for this tab is similar to other tabs, except this area contains mostly information, as well as a button to launch the QSPI Configurator.

#### 7.6.1 Memory Regions subtab

The **Memory Regions** subtab displays if the Memory resource has been enabled. It consists of a toolbar, table, and legend. The toolbar allows you to expand and collapse the available memory bars in the table area, as well as **Show/Hide Detailed View** of the memories and **Delete All** memory regions.

7 Resources tabs



Each memory shows as a usage bar that when expanded displays the details of how it is being used. The bar itself shows the display name of the memory along with its overall size. Hover the cursor over a used section to displays a tooltip that corresponds to the same information provided in the rows under the memory.

Core (s)	Id: gpu_buf Offset: 0x00080000 Size: 0x00080000 (512 KB)	Region Id	Settings	Offset	Size
ARM		code_6	<NONE>	0x00000000	0x00080000 (512 KB)
ARM Cortex-M33,ARM Cortex-M55	gpu_buf		<NONE>	0x00080000	0x00080000 (512 KB)
ARM Cortex-M33,ARM Cortex-M55	shared_mem1_ns		<NONE>	0x00100000	0x00100000 (1 MB)
ARM Cortex-M55	data_3		<NONE>	0x00200000	0x00100000 (1 MB)
ARM Cortex-M33	code_5_ns		<NONE>	0x00300000	0x00100000 (1 MB)
ARM Cortex-M33	data_3_ns		<NONE>	0x00400000	0x00100000 (1 MB)
					USED: (5 MB)
					0x00500000 FREE: (0 Bytes)

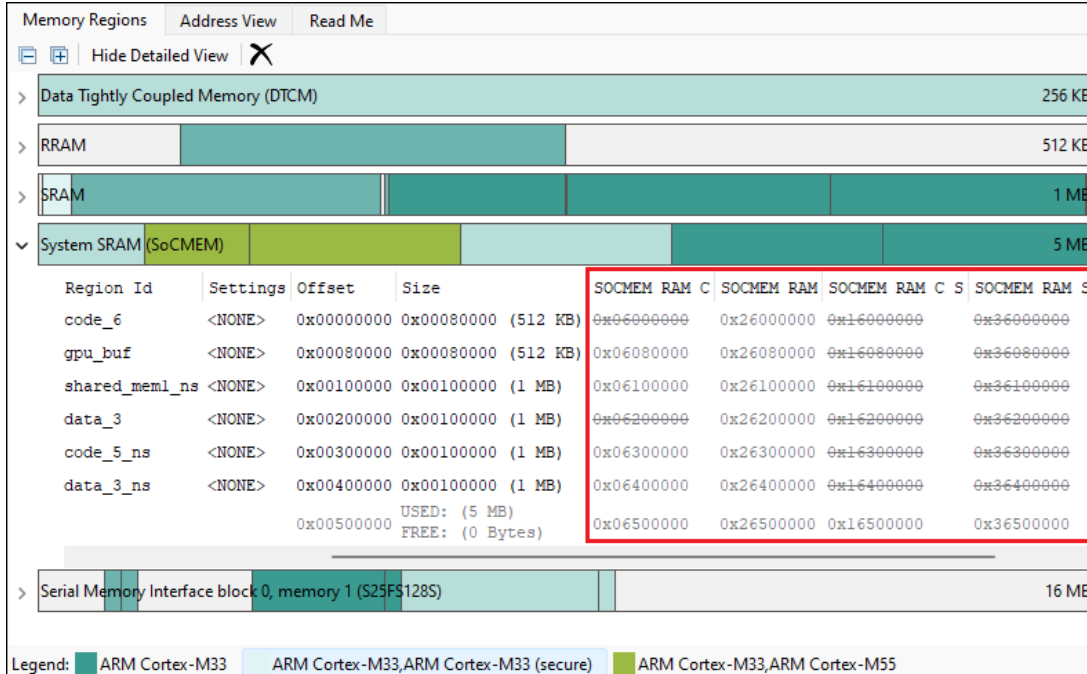
The rows show allocated and unallocated ranges of the memory. The final row shows the end offset of the memory along with a summary of used/free space. If the ending offset is outside the allowed range, the text become red and display the maximum allowed offset along with the currently set offset that is in violation of the max. If more size is allocated then is available, that text will be in red as well.

Core (s)	Region Id	Settings	Offset	Size
ARM Cortex-M55	code_6	<NONE>	0x00000000	0x00080000 (512 KB)
ARM Cortex-M33,ARM Cortex-M55	gpu_buf	<NONE>	0x00080000	0x00080000 (512 KB)
ARM Cortex-M33,ARM Cortex-M55	shared_mem1_ns	<NONE>	0x00100000	0x00100000 (1 MB)
ARM Cortex-M55	data_3	<NONE>	0x00200000	0x00100000 (1 MB)
ARM Cortex-M33	code_5_ns	<NONE>	0x00300000	0x00100000 (1 MB)
ARM Cortex-M33	data_3_ns	<NONE>	0x00400000	0x00500000 (5 MB)
				MAX: 0x00500000 USED: (9 MB)
				ACT: 0x00900000 FREE: (-4 MB)

## 7 Resources tabs

### 7.6.1.1 Show/hide detailed view

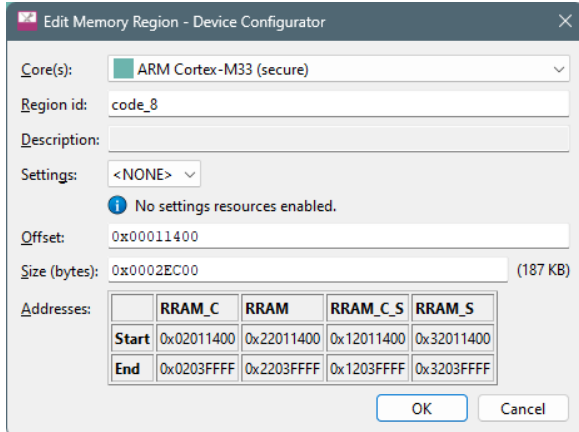
Click the **Hide Detailed View** button to toggle it off and on to show/hide detailed address maps. If any of the addresses do not apply (that is, the core that uses that map is not selected), the values will be stricken out.



Region Id	Settings	Offset	Size	SOC MEM RAM C	SOC MEM RAM S
code_6	<NONE>	0x00000000	0x00080000 (512 KB)	0x06000000	0x26000000
gpu_buf	<NONE>	0x00080000	0x00080000 (512 KB)	0x06080000	0x26080000
shared_mem1_ns	<NONE>	0x00100000	0x00100000 (1 MB)	0x06100000	0x26100000
data_3	<NONE>	0x00200000	0x00100000 (1 MB)	0x06200000	0x26200000
code_5_ns	<NONE>	0x00300000	0x00100000 (1 MB)	0x06300000	0x26300000
data_3_ns	<NONE>	0x00400000	0x00100000 (1 MB)	0x06400000	0x26400000
		0x00500000	USED: (5 MB) FREE: (0 Bytes)	0x06500000	0x26500000

### 7.6.1.2 Add/edit memory allocations

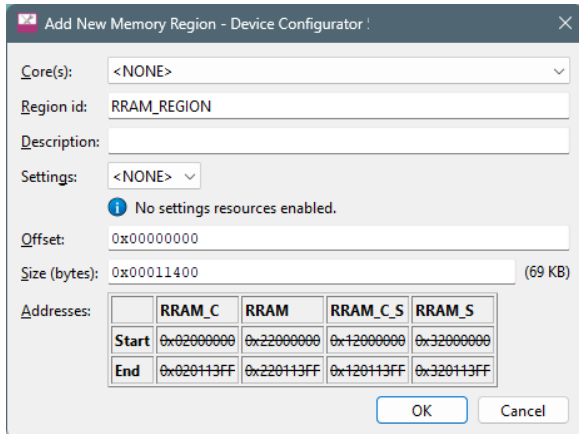
To edit allocations, double-click an allocated row to open Edit Memory Region dialog.



	RRAM_C	RRAM	RRAM_C_S	RRAM_S
Start	0x02011400	0x22011400	0x12011400	0x32011400
End	0x0203FFFF	0x2203FFFF	0x1203FFFF	0x3203FFFF

To add an allocation, double-click an unallocated row to open the Add New Memory Region dialog.

## 7 Resources tabs



Core(s): <NONE>

Region id: RRAM\_REGION

Description:

Settings: <NONE>

Offset: 0x00000000

Size (bytes): 0x00011400 (69 KB)

	RRAM_C	RRAM	RRAM_C_S	RRAM_S
Start	0x02000000	0x22000000	0x12000000	0x32000000
End	0x020113FF	0x220113FF	0x120113FF	0x320113FF

OK Cancel

Once the dialog is open, enter values:

1. Select one or more **Core(s)** from the pull-down menu by clicking the corresponding check boxes.
2. As needed, type the name of the **Region id**.
3. If present, select **Settings**.

**Note:** The **Settings** option is only available if there is a corresponding MPC available on the hardware for the memory. If no memory protection settings resource has been enabled, an info message will be added stating this to make you aware of what needs to be done to populate the drop-down.

4. Enter the **Offset** and **Size**.

The **Size** entry allows for hex or decimal entry. It also accepts specifying units, which will always be converted to bytes. Units are case insensitive and can be entered as follows:

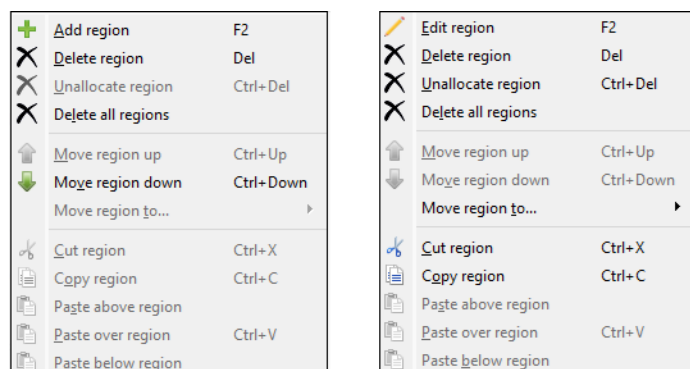
- bytes or b = byte
- kb or k = kilobyte (1 KB = 1,024 Bytes)
- mb or m = megabyte (1 MB = 1024KB = 1,048,576 Bytes)
- gb or g = gigabyte (1 GB = 1024MB = 1,048,576 KB = 1,073,741,824 Bytes)
- tb or t = terabyte (1 TB = 1024 GB = 1,048,576 MB = 8,388,608 KB = 1,099,511,627,776 Bytes)
- pb or p = petabyte (1 PB = 1024 TB = 1,048,576 GB = 1,073,741,824 MB = 1,099,511,627,776 KB = 1,125,899,906,842,624 Bytes)
- eb or e = exabyte (1 EB = 1024 PB = 1,048,576 TB = 1,073,741,824 GB = 1,099,511,627,776 MB = 1,125,899,906,842,624 KB = 1,152,921,504,606,846,976 Bytes)
- zb or z = zettabyte (1 ZB = 1024 EB = 1,048,576 PB = 1,073,741,824 TB = 1,099,511,627,776 GB = 1,125,899,906,842,624 MB = 1,152,921,504,606,846,976 KB = 1,180,591,620,717,411,303,424 Bytes)
- yb or y = yottabyte (1 YB = 1024 ZB = 1,048,576 EB = 1,073,741,824 PB = 1,099,511,627,776 TB = 1,125,899,906,842,624 GB = 1,152,921,504,606,846,976 MB = 1,180,591,620,717,411,303,424 KB = 1,208,925,819,614,629,174,706,176 Bytes)

If there are obvious errors, the dialog will not allow you to click **OK**. There may also be an error message.

You can also open the dialog using the drop-down button at the beginning of a row or right-clicking on a row. Unallocated regions will start with an **Add region** option while allocated regions will have the **Edit region** option. Both options launch the same edit dialog.



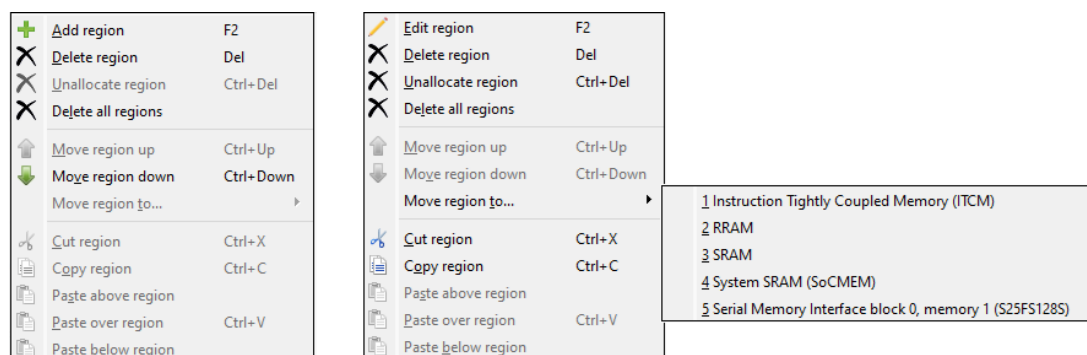
## 7 Resources tabs



If you prefer, use the keyboard shortcuts defined on the menus.

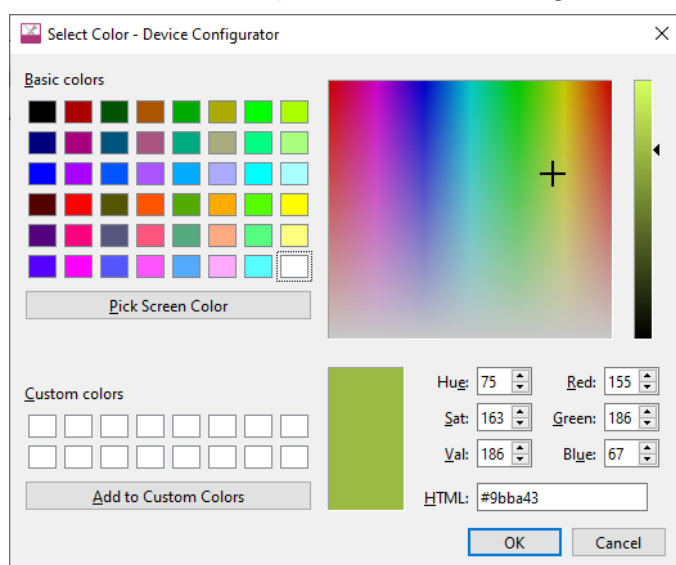
### 7.6.1.3 Rearrange memory regions

On the context menu, there are various options to move regions, as well as cut, copy, and paste regions.



### 7.6.1.4 Use the Legend

The legend shows you the color for each core. You can change the color associated with a used combination of cores. Just click on any of the colors in the legend to open the color picker.



## 7 Resources tabs

### 7.6.1.5 How to find out which region the main goes in from the main ld

Code that doesn't have an explicit section name goes in .text (or .text.function\_name when using -ffunction-sections). Search for .text in the linker script or search the .map file for the function name.

### 7.6.1.6 How to read the build output and associate it with the regions defined in the tool

For each project in an application, “make build” will display a summary of the output sections:

Section Name	Address	Size
.appText_1	0x60580400	8048
.ARM.exidx	0x60582370	8
.copy.table	0x60582378	48
.data	0x20000000	36
.appText_2	0x00000000	680
.zero.table	0x605823a8	8
.ram_vectors	0x00000400	852
.noinit	0x20000028	152
.bss	0x200000c0	116
.heap	0x26200000	1048576
.reserved_socmem	0x26100000	1048576
.cy_sharedmem	0x240ff000	500

The “.map” file generated by the linker contains details about which input sections the linker placed into each output section. To find the memory region corresponding to each output section, search for the memory region that contains the section's address.

Example of “.map” file from GNU ld:

```
.appText_1      0x60580400      0x1f70
...
.text.main      0x6058094c      0x14  build/APP_KIT_PSE84_EVAL/Debug/main.o
               0x6058094c                      main
.text.Cy_PDL_Init
               0x60580960      0xc   build/APP_KIT_PSE84_EVAL/.../cy_device.o
               0x60580960                      Cy_PDL_Init
```

- .appText\_1 is the output section name. It is followed by the output section start address (0x60580400) and the output section total size (0x1f70).
- .text.main is the input section name. It is followed by its address (0x6058094c) and size (0x14).

Initialized data in volatile memory also consumes space in a non-volatile memory.

```
.data           0x20000000      0x24  load address 0x60582658
```

The .data section consumes 0x24 bytes at both its writable address 0x20000000 and its non-volatile address 0x60582658.

## 7 Resources tabs

### 7.6.1.7 How to define a new region for some data buffer variable

#### For uninitialized data:

1. In the memory tab, add a new region named "CUSTOM\_DATA" in a writable memory such as SRAM.
2. In the linker script, add a section for the region:

#### GNU ld or LLVM lld

```
.custom_data :
{
    *(.custom_data*)
} >CUSTOM_DATA
```

#### armlink

```
LOAD_CUSTOM_DATA CYMEM_CM55_0_CUSTOM_DATA_START CYMEM_CM55_0_CUSTOM_DATA_SIZE
{
    EXEC_CUSTOM_DATA CYMEM_CM55_0_CUSTOM_DATA_START UNINIT CYMEM_CM55_0_CUSTOM_DATA_SIZE
    {
        *(.custom_data*)
    }
}
```

#### ilinkarm

```
do not initialize { section .custom_data* }
place in CUSTOM_DATA { section .custom_data* }
```

3. In the source code, assign the variable to the section (GCC, armclang, or iccarm)

```
__attribute__((section ".custom_data"))
uint32_t data_buffer[DATA_BUFFER_COUNT];
```

#### For data in volatile memory

For data in volatile memory that requires run-time initialization, the linker script must configure a non-volatile location to initialize the data. For this example, create a region named "CUSTOM\_CODE" in a non-volatile memory.

## 7 Resources tabs

### GNU ld or LLVM lld

```
.custom_data :
{
    *(.custom_data*)
} >CUSTOM_DATA AT>CUSTOM_CODE

.copy.table :
{
    /* ... */
    LONG(LOADADDR(.custom_data))
    LONG(ADDR(.custom_data))
    LONG(SIZEOF(.custom_data)/4)
    /* ... */
}
```

**Note:** If `ADDR(.custom_data)` is read-only, the `.copy.table` entry must use a writable address.

### armlink

```
LOAD_CUSTOM_CODE CYMEM_CM55_0_CUSTOM_CODE_START CYMEM_CM55_0_CUSTOM_CODE_SIZE
{
    EXEC_CUSTOM_DATA CYMEM_CM55_0_CUSTOM_DATA_START CYMEM_CM55_0_CUSTOM_DATA_SIZE
    {
        *(.custom_data*)
    }
}
```

### ilinkarm

```
initialize by copy { section .custom_data* }
place in CUSTOM_DATA { rw section .custom_data* }
place in CUSTOM_CODE { ro section .custom_data* }
```

### 7.6.1.8 How to define a new region for the code and then assign code to the particular region

1. In the memory tab, add a new region named “CUSTOM\_CODE” in a non-volatile memory such as RRAM.
2. In the linker script, add a section for the region:

#### GNU ld or LLVM lld

```
a. .custom_code :
{
    *(.custom_code*)
} >CUSTOM_CODE
```

## 7 Resources tabs

### armlink

```
LOAD_CUSTOM_CODE CYMEM_CM55_0_CUSTOM_CODE_START CYMEM_CM55_0_CUSTOM_CODE_SIZE
{
    EXEC_CUSTOM_CODE +0
    {
        *(.custom_code*)
    }
}
```

### ilinkarm

```
place in CUSTOM_CODE { section .custom_code* }
```

3. In the source code, assign the variable to the section (GCC, armclang, or iccarm)

```
__attribute__((section ".custom_code"))
void custom_code_function1(void);
```

### 7.6.1.9 How to assign a function or variable into a custom memory section

Variables and functions may be assigned to a section using the section attribute. The linker script must assign the section to an output section.

```
__attribute__((section ".custom_data"))
uint32_t data_buffer[DATA_BUFFER_COUNT];

__attribute__((section ".custom_code"))
void custom_code_function1(void);
```

### 7.6.1.10 How to modify the linker to include a specific file into a custom memory section

#### GNU ld or LLVM lld

```
.custom_code :
{
    main.o(.text* .rodata*)
} >CUSTOM_CODE
.custom_data :
{
    main.o(.data*)
    main.o(.bss*)
} >CUSTOM_DATA AT>CUSTOM_CODE
```

## 7 Resources tabs

**Note:** *The toolchain does not automatically initialize data. Sections that require initialization must be added to the .zero.table and/or .copy.table manually.*

### armlink

```
LOAD_CUSTOM_CODE CYMEM_CM55_0_CUSTOM_CODE_START CYMEM_CM55_0_CUSTOM_CODE_SIZE
{
    EXEC_CUSTOM_CODE +0
    {
        main.o(+RO)
    }
    EXEC_CUSTOM_DATA CYMEM_CM55_0_CUSTOM_DATA_START CYMEM_CM55_0_CUSTOM_DATA_SIZE
    {
        main.o(+RW,+ZI)
    }
}
```

### ilinkarm

```
place in CUSTOM_CODE { ro object main.o }
place in CUSTOM_DATA { rw object main.o }
```

### 7.6.1.11 How to modify the linker to include an entire middleware library into a custom memory section

The linker script example assigns all of the code and data from object files matching `mtb_hal_*.o` to the `.custom_code` and `.custom_data` sections respectively. If the middleware's object files cannot be described in a single pattern, it may be necessary to use multiple patterns.

#### GNU ld or LLVM lld

```
.custom_code :
{
    mtb_hal_*.o(.text* .rodata*)
} >CUSTOM_CODE
.custom_data :
{
    mtb_hal_*.o(.data*)
    mtb_hal_*.o(.bss*)
} >CUSTOM_DATA AT>CUSTOM_CODE
```

**Note:** *The toolchain does not automatically initialize data. Sections that require initialization must be added to the .zero.table and/or .copy.table manually.*

## 7 Resources tabs

### armlink

```
LOAD_CUSTOM_CODE CYMEM_CM55_0_CUSTOM_CODE_START CYMEM_CM55_0_CUSTOM_CODE_SIZE
{
    EXEC_CUSTOM_CODE +0
    {
        mtb_hal_*.o(+RO)
    }
    EXEC_CUSTOM_DATA CYMEM_CM55_0_CUSTOM_DATA_START CYMEM_CM55_0_CUSTOM_DATA_SIZE
    {
        mtb_hal_*.o(+RW,+ZI)
    }
}
```

### ilinkarm

```
place in CUSTOM_CODE { ro object mtb_hal_*.o }
place in CUSTOM_DATA { rw object mtb_hal_*.o }
```

### 7.6.1.12 How to re-balance the memory between the cores based on the build output

This is highly dependent on the needs of the customer's application. Solutions may include:

- Resizing existing regions
  - Double-click on the region to open the Edit dialog.
  - Update the Size. If the region needs to be moved, also update the Offset.
- [Moving existing regions to another memory](#)
- [Updating linker scripts to place code or data in different regions](#)

### 7.6.1.13 How to move a region to another memory

In the **Memory Regions** tab, use the context menu to move the memory region to another memory.

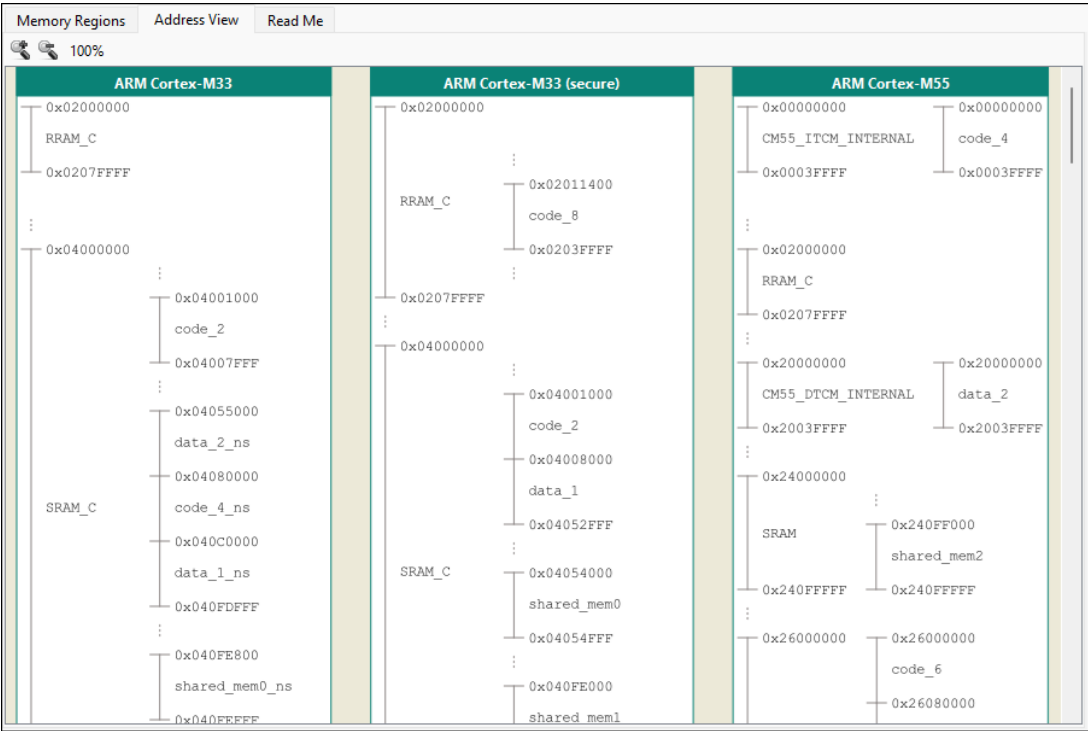
Core(s)	Region Id	Offset	Size
<NONE>	UNALLOCATED	0x00000000	0x00100000 (1 MB)
ARM Cortex-M33 (secure)	cm33_s_header	0x00100000	0x00000400 (1 KB)
ARM Cortex-M33 (secure)	code_1	0x00100400	0x0003FB00 (254.75 KB)
	codeNsc	0x0013FF00	0x00000100 (256 Bytes)
	cm33_s_signature	0x00140000	0x00040000 (256 KB)
	UNALLOCATED	0x00180000	0x001C0000 (1.75 MB)
	cm33_ns_header	0x00340000	0x00000400 (1 KB)
	code_1_ns	0x00340400	0x001FFC00 (2 MB)
	cm33 ns signature	0x00540000	0x00040000 (256 KB)
	1 Instruction Tightly Coupled Memory (ITCM)		0 (1 KB)
	2 Data Tightly Coupled Memory (DTCM)		0 (3 MB)
	3 RRAM		0 (256 KB)
	4 SRAM		0 (7.25 MB)
	5 System SRAM (SoC MEM)		0 (KB)
	0x01000000		FREE: (10 MB)

7 Resources tabs

- Note:** The regions required for initial application startup must be in a non-volatile memory.
- Note:** Some toolchains including GNU and LLVM do not initialize data automatically. If a region is moved to a volatile memory, entries must be added to .zero.table and/or .copy.table manually.

7.6.2 Address View subtab

The **Address View** subtab displays a generated diagram of defined regions per core. The first column is based off when the hardware defines are available. The defined regions are displayed next (if applicable). Then SAU and MPU columns are displayed if defined. ... is added whenever there is a gap in the addresses.



This subtab contains a toolbar with the standard zoom commands. Hovering the cursor over any of the regions displays a tooltip with a summary of the region’s info.



## 7 Resources tabs

### 7.6.3 Read Me subtab

This subtab displays an HTML generated document with information from the device-db's *memories.cydata* file, combined with memory data exported from the QSPI configurator.

Memory Regions

Address View

Read Me

Overview

The PSOC Edge MCU features two CPU subsystems. The CM33 CPU comes with a 32KB instruction cache while the CM55 CPU comes with a 32KB instruction cache along with a 32KB data cache. The MCU has two 512KB SRAM sections in the low-power domain and 5120KB of System SRAM (SoCMEM) in the high-performance domain. Additionally, it includes 512KB of RRAM for non-volatile storage, primarily for code and selective application data. The CM55 core's performance is further enhanced by 256KB of tightly coupled memory for data and instructions. For more information, refer to AN239774.

[Additional documentation](#)

Instruction Tightly Coupled Memory (ITCM) (256 KB)

ITCM stores program instructions requiring low latency and minimal contention for access, often housing critical parts of the program like interrupt service routines (ISRs), time-critical control loops, or frequently used functions. ITCM instructions do not pass through the cache and do not impose any restriction on wait states. This allows for direct and deterministic access to instructions from this TCM, improving execution speed for critical, or time-sensitive code. Accessing instructions from ITCM typically incurs fewer or zero wait states compared to accessing instructions from slower memory regions like flash memory. This reduction in wait states enhances the real-time performance and responsiveness of the MCU.

Address	Cores	Executable	Writable	Readable	Secure
CM55_ITCM_CM33 0x48000000-0x4803FFFF	ARM Cortex-M33 ARM Cortex-M33 (secure)	✓	✓	✓	
CM55_ITCM_CM33_S 0x58000000-0x5803FFFF	ARM Cortex-M33 (secure)	✓	✓	✓	✓
CM55_ITCM_INTERNAL 0x00000000-0x0003FFFF	ARM Cortex-M55	✓	✓	✓	
CM55_ITCM 0x48000000-0x4803FFFF	ARM Cortex-M55	✓	✓	✓	

This file provides links to "how to" HTML topics to guide you how to configure memories.

7 Resources tabs

7.7 MMIO-Clocks tab

For supported devices, the **MMIO-Clocks** tab lists all the clocks in a design used to drive the various Memory Mapped IOs. All available clocks are shown in an expandable tree. The filter box and the hide disabled button above the list of resources allows you to limit the items shown in the tree. This tab allows you to enter one or more **Name(s)** for the resource. It also shows the selected **Personality**, where applicable.

Peripherals	Pins	Analog	System	Memory	MMIO-Clocks	Peripheral-Clocks	DMA
Enter filter text...							
Resource		Name(s)	Personality				
▼ APP MMIO							
<input type="checkbox"/>	APP MMIO 0	peri_1_peri_group_0					
<input checked="" type="checkbox"/>	APP MMIO 1	peri_1_peri_group_1	Peripheral Group-1.0 ▼				
<input type="checkbox"/>	APP MMIO 2	peri_1_peri_group_2					
<input checked="" type="checkbox"/>	APP MMIO 3	peri_1_peri_group_3	Peripheral Group-1.0 ▼				
<input type="checkbox"/>	APP MMIO TCM	peri_1_peri_group_4					
▼ SYS MMIO							
<input type="checkbox"/>	SYS MMIO 0	peri_0_peri_group_0					
<input type="checkbox"/>	SYS MMIO 1	peri_0_peri_group_1					
<input checked="" type="checkbox"/>	SYS MMIO 2	peri_0_peri_group_2	Peripheral Group-1.0 ▼				
<input type="checkbox"/>	SYS MMIO 3	peri_0_peri_group_3					
<input type="checkbox"/>	SYS MMIO 4	peri_0_peri_group_4					
<input type="checkbox"/>	SYS MMIO 5	peri_0_peri_group_5					

7 Resources tabs

7.8 Peripheral-Clocks tab

The Peripheral-Clocks tab lists all the clocks in a design used to drive the various peripherals. All available clocks are shown in an expandable tree. The filter box and the hide disabled button above the list of resources allows you to limit the items shown in the tree. This tab allows you to enter one or more **Name(s)** for the resource. It also shows the selected **Personality**, where applicable.

PeripheralsPinsAnalog-RoutingSystemPeripheral-ClocksDMA

Enter filter text...

Resource	Name(s)	Personality
8 bit		
<input checked="" type="checkbox"/> 8 bit Divider 0	CYBSP_CSD_CLK_DIV,CYBSP_CS_CLK_DIV	Peripheral Clock-1.0
<input checked="" type="checkbox"/> 8 bit Divider 1	peri_0_div_8_1	Peripheral Clock-1.0
<input type="checkbox"/> 8 bit Divider 2	peri_0_div_8_2	
<input type="checkbox"/> 8 bit Divider 3	peri_0_div_8_3	
<input type="checkbox"/> 8 bit Divider 4	peri_0_div_8_4	
<input type="checkbox"/> 8 bit Divider 5	peri_0_div_8_5	
<input type="checkbox"/> 8 bit Divider 6	peri_0_div_8_6	
<input type="checkbox"/> 8 bit Divider 7	peri_0_div_8_7	
16 bit		
16.5 bit		
<input type="checkbox"/> 16.5 bit Divider 0	peri_0_div_16_5_0	
<input type="checkbox"/> 16.5 bit Divider 1	peri_0_div_16_5_1	
<input type="checkbox"/> 16.5 bit Divider 2	peri_0_div_16_5_2	
<input type="checkbox"/> 16.5 bit Divider 3	peri_0_div_16_5_3	
24.5 bit		

7 Resources tabs

7.9 DMA tab

The DMA tab lists all the DMA resources in the design. All available DMA channels are shown in an expandable tree. The filter box and the hide disabled button above the list of resources allows you to limit the items shown in the tree. This tab allows you to enter one or more **Name(s)** for the resource. It also shows the selected **Personality**, where applicable.

PeripheralsPinsAnalog-RoutingSystemPeripheral-ClocksDMA

Enter filter text...

Resource	Name(s)	Personality
▼ DMA Controller		
<input checked="" type="checkbox"/> DMA Channel 0	cpuss_0_dmac_0_chan_0	DMAC-1.0 ▼
<input type="checkbox"/> DMA Channel 1	cpuss_0_dmac_0_chan_1	
<input type="checkbox"/> DMA Channel 2	cpuss_0_dmac_0_chan_2	
<input type="checkbox"/> DMA Channel 3	cpuss_0_dmac_0_chan_3	
▼ DMA DataWire 0		
<input type="checkbox"/> DMA DataWire 0: Channel 0	cpuss_0_dw0_0_chan_0	
<input type="checkbox"/> DMA DataWire 0: Channel 1	cpuss_0_dw0_0_chan_1	
<input type="checkbox"/> DMA DataWire 0: Channel 2	cpuss_0_dw0_0_chan_2	
<input type="checkbox"/> DMA DataWire 0: Channel 3	cpuss_0_dw0_0_chan_3	
<input type="checkbox"/> DMA DataWire 0: Channel 4	cpuss_0_dw0_0_chan_4	
<input type="checkbox"/> DMA DataWire 0: Channel 5	cpuss_0_dw0_0_chan_5	
<input type="checkbox"/> DMA DataWire 0: Channel 6	cpuss_0_dw0_0_chan_6	
<input type="checkbox"/> DMA DataWire 0: Channel 7	cpuss_0_dw0_0_chan_7	
<input type="checkbox"/> DMA DataWire 0: Channel 8	cpuss_0_dw0_0_chan_8	
<input type="checkbox"/> DMA DataWire 0: Channel 9	cpuss_0_dw0_0_chan_9	
<input type="checkbox"/> DMA DataWire 0: Channel 10	cpuss_0_dw0_0_chan_10	
<input type="checkbox"/> DMA DataWire 0: Channel 11	cpuss_0_dw0_0_chan_11	
<input type="checkbox"/> DMA DataWire 0: Channel 12	cpuss_0_dw0_0_chan_12	
<input type="checkbox"/> DMA DataWire 0: Channel 13	cpuss_0_dw0_0_chan_13	
<input type="checkbox"/> DMA DataWire 0: Channel 14	cpuss_0_dw0_0_chan_14	
<input type="checkbox"/> DMA DataWire 0: Channel 15	cpuss_0_dw0_0_chan_15	
<input type="checkbox"/> DMA DataWire 0: Channel 16	cpuss_0_dw0_0_chan_16	
<input type="checkbox"/> DMA DataWire 0: Channel 17	cpuss_0_dw0_0_chan_17	
<input type="checkbox"/> DMA DataWire 0: Channel 18	cpuss_0_dw0_0_chan_18	

## 8 Panes

### 8 Panes

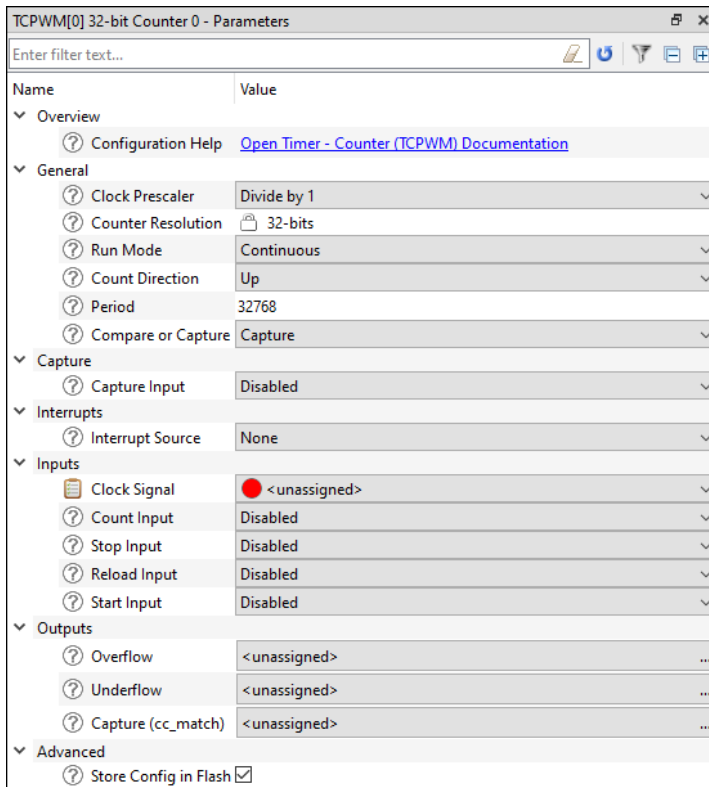
The Device Configurator tool contains the following primary panes that display information based on what is selected in a particular [Resources](#) tabs:

- [Parameters pane](#) – This pane shows the various parameters for any specific resource enabled in one of the tabs.
- [Notice List](#) – This pane shows any errors, warnings, tasks, and infos for the application.
- [Code Preview pane](#) – This pane shows a preview of the code that will be generated for the selected resource when you save the \*.modus file.

#### 8.1 Parameters pane

The Parameters pane contains all the parameters for a selected, enabled resource. This pane will show different parameters for each resource, grouped by various categories. For example, the parameters for the TCPWM peripheral are completely different than those for a pin resource. Some resources also provide a link to [Launch other configurators](#). This pane contains a few tools:

- The filter box above the list of parameters allows you to limit the items shown in the pane.
- The reset button resets all parameter values to their defaults.
- The filter button toggles hides or shows non-editable parameters.
- The - and + buttons expand or collapse all nodes.



Name	Value
<b>Overview</b>	
Configuration Help	<a href="#">Open Timer - Counter (TCPWM) Documentation</a>
<b>General</b>	
Clock Prescaler	Divide by 1
Counter Resolution	32-bits
Run Mode	Continuous
Count Direction	Up
Period	32768
Compare or Capture	Capture
<b>Capture</b>	
Capture Input	Disabled
<b>Interrupts</b>	
Interrupt Source	None
<b>Inputs</b>	
Clock Signal	<unassigned>
Count Input	Disabled
Stop Input	Disabled
Reload Input	Disabled
Start Input	Disabled
<b>Outputs</b>	
Overflow	<unassigned>
Underflow	<unassigned>
Capture (cc_match)	<unassigned>
<b>Advanced</b>	
Store Config in Flash	<input checked="" type="checkbox"/>

##### 8.1.1 Configuration help

Nearly all resources provide a link to open the Peripheral Driver Library (PDL) documentation to the specific driver. This is the Doxygen-generated HTML file located in the installation directory. To see links to the documentation, simply highlight a resource; you do not need to enable it.

## 8 Panes

### 8.1.2 Parameter descriptions

As described under [Icons](#), all parameters have a tooltip icon (?) to indicate there is information about the parameter. Hover the mouse cursor over the icon to display a description of the parameter.

### 8.1.3 Parameter values

Different parameter types have different ways to specify a value, as follows:

- Pull-down Menu – For parameters with a specific set of values, use the pull-down menu to select the appropriate value.
- Selection Box – For parameters with a variable set of values, click the ellipsis [...] button to open a selection box. There, use the check boxes to select one or more appropriate values for the parameter.

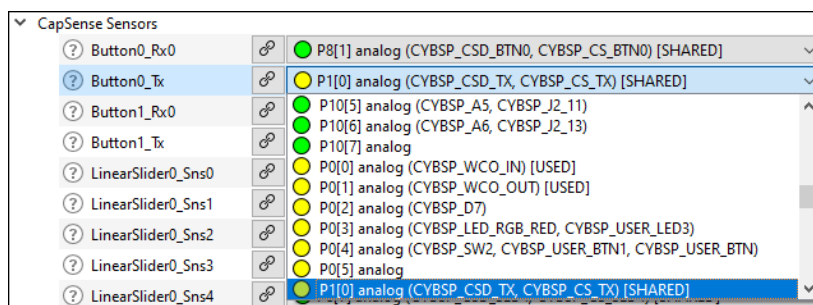
**Note:** After selecting these parameter types, use the **Go To** button to jump to the selected resource.

- Check Box – For parameters with a true or false value, use the check box to enable or disable the parameter.
- Text Box – For parameters with editable values, type the value in the text box.

**Note:** Values preceded by '0' are interpreted as octal; values preceded by '0x', '0X', and '#' are interpreted as hexadecimal.

### 8.1.4 Signal Select Indicators

For parameters where you select a signal, there is a pull-down menu for single-select signals and a button to open a dialog for multi-select signals.



The signals have guidance icons next to them to indicate the status of the signal, as follows:

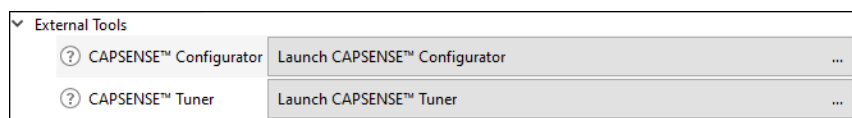
- Green – preferred
- Yellow – valid
- Red – constrained

After selecting one or more signals, use the **Go To** button to jump to the selected signal or open a dialog to select from multiple signals.

### 8.1.5 Launch other configurators

For peripherals with their own configuration tools (CapSense, SegLCD, etc.), the Device Configurator provides links to launch those separate configurators. After enabling the peripheral on the [Peripherals](#) tab, the **Parameters** pane contains a **<Configurator>** parameter, where <Configurator> is the name of the other configurator.

## 8 Panes

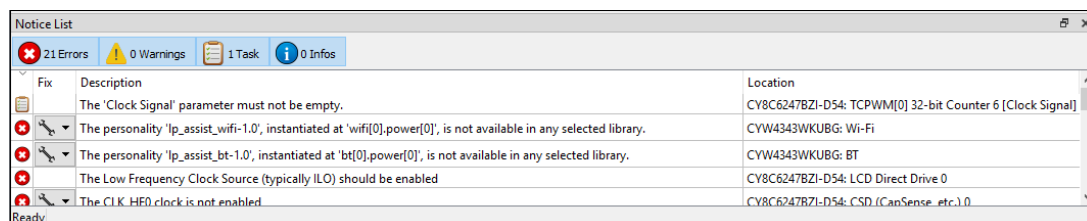


Click on **[Launch < Configurator > . . .]** to launch the configurator.

**Note:** When launching another configurator from the Device Configurator, it passes information, such as the location of the \*.modus file and any configuration data, to that other configurator. Those other configurators can be launched independently from the Device Configurator. When launched independently, you will need to either open or create the appropriate configuration file for that tool. If you want to use the configuration tools independently for the same application, make sure to save the source files in the correct "GeneratedSource" folder for the appropriate application.

## 8.2 Notice List

The Notice List pane combines notices (errors, warnings, tasks, and infos) from many places in your design into a centralized list. If a notice shows a location, you can double-click the entry to navigate to the error or warning.




Notices display in rows. Use the filters above the notices to show or hide different types of notices, as follows:

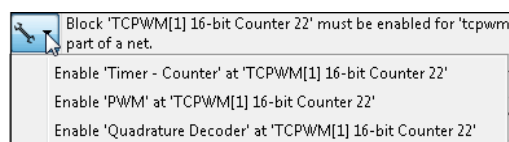
- **Errors** – These indicate there is at least one problem that must be addressed before you can build your application. Typical errors could include compiler build errors and connectivity errors.
- **Warnings** – These report unusual conditions that might indicate a problem, although you can usually build the application regardless.
- **Tasks** – These are actions you need to perform to resolve an issue, such as enabling a resource. If you save without resolving a task, it becomes an error.
- **Infos** – These are informational messages from the system to indicate something occurred.

The Notice List pane contains the following columns (each column header contains an arrow control to change the sorting of the notices in the table):

- **Icon** – Displays the icons for the error, warning, task, or info.
- **Fix** – This may display a wrench icon, which can be used to automatically address the required notice.
- **Description** – Displays a brief description of the notice.
- **Location** – Displays the specific line number or other location of the message, when applicable.

### 8.2.1 Fix a task/error

When a wrench icon  displays in the **Fix** column, click on it and select the appropriate action from the pull-down menu. When all related issues have been addressed, the notice will be removed from the Notice List pane.



## 8 Panes

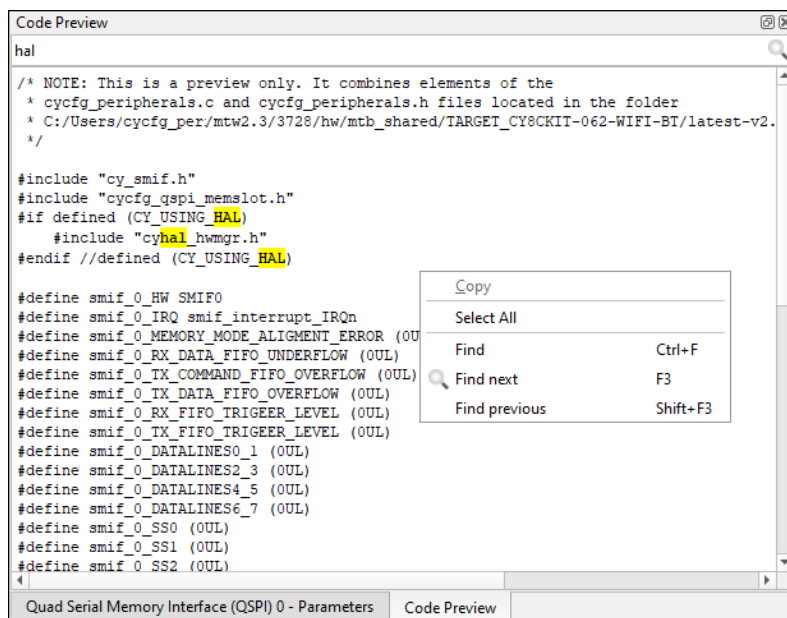
**Note:** The fixes listed are not necessarily the only way to fix the issue. They are merely common options. Also, if you save the \*.modus file with outstanding tasks, they become errors saved in the GeneratedSource/cycfg\_notices.h file.

### 8.2.2 Copy a notice

You can copy a notice to your system's clipboard using **[Ctrl]+[c]** or right-click and select **Copy**. Then, paste the notice text into an email, document, and so on.

### 8.3 Code Preview pane

The **Code Preview** pane is a read-only preview of the code that will be generated for the currently selected resource when you save the \*.modus file. As you update configuration options, the **Code Preview** pane updates the code shown. This code will be written to the appropriate file(s) located in the *GeneratedSource* folder of your application.



You can select and copy code from this pane using **[Ctrl]+[c]** key or using the right-click menu option. You can use the **Search** feature to find instances of specific terms.



## 9 Version changes

### 9 Version changes

This section lists and describes the changes for each version of this tool.

Version	Change Descriptions	Notes
1.0	New tool.	
1.1	Added support for WICED Bluetooth® devices.	
2.0	Changed the Platform tab to System tab.	This affects the file name generated during code generation. Older versions of the Device Configurator generated cycfg_platform.(c/h) files; it now generates cycfg_system.(c/h) files. If you are updating a design from a previous version, manually remove the old cycfg_platform.(c/h) files and update any references you created to use the new file names.
	Moved the Analog-Routing Editor to a tab.	
	Updated the File menu for library settings.	
	Added Update All Personalities menu item.	
	Added the ability to enter multiple resource Names using comma-separated list.	
2.1	Added Open System Explorer menu item. Added Change Devices menu item and dialog. Added Undo/Redo operations to the Edit menu. Added major/minor version number to the title bar.	
2.20	Added Copy feature to the Notice List. Added ability to see documentation links without enabling a resource. Added frequencies to clock diagram. Added feature to support incremental patch updates.	
2.21	Added Search feature to Code Preview. Implemented various performance improvements. Fixed an issue with multiple font sizes in the GUI. Fixed the PDL display name when using the MTB flow. Updated to allow analog routing from the SAR to any analog resource or pin.	
3.0	Removed Change Devices dialog. Updated the tool to recognize obsolete devices. Added path to files in the Code Preview Pane.	Use command line tools instead. Displays an error in the Notice List instead of blocking the file from being opened.

## 9 Version changes

Version	Change Descriptions	Notes
3.10	<p>Fixed the name conflict warning notices when the selected device is changed to not check against non-existent chip locations.</p> <p>Added a 'Fix' to name conflict warnings in the notice list.</p> <p>Locked down number entry to the English format (decimal place is always a period).</p> <p>Made various performance improvements.</p> <p>Added a search filter to the dialog that allows for the selection of multiple signals.</p> <p>Added additional MPN consistency checking preprocessor macros to cyf_notices.h.</p> <p>Added a log file that can be accessed from the About box. Debug messages are redirected to it.</p> <p>Added back the 'Fix' to the MPN consistency notice.</p> <p>Fixed error message displayed when the same *.modus file is opened in multiple instances of the Device-Configurator and mpn checking command line arguments are provided.</p>	
4.0	<p>Updated back-end to support multi-core applications.</p> <p>Updated Create Design dialog.</p> <p>Changed device library file from xml to props.json.</p>	
4.10	<p>Added Settings menu and items.</p> <p>Improved parameter view sizing.</p>	
4.20	<p>Removed the Save As and Change Libraries menu items.</p> <p>Removed New Design dialog and menu items.</p> <p>Added a Filter button to the Parameters pane to toggle off and on non-editable parameters.</p> <p>Deeper grouping is supported in the parameter view.</p> <p>Added the use of the back-end server, meaning that multiple editors can be open for the same design file at once.</p> <p>Added support for dynamically displayed resources. Used by some virtual locations to get a better user experience.</p> <p>Supports multiple "configurator_support" libraries being found by default to be included for a single design (i.e. personalities can be split across PDL and middleware libraries).</p>	
5.0	<p>Renamed to Infineon Device Configurator.</p> <p>Added section for stand-alone operation.</p>	
5.10	<p>Various bug fixes and back-end updates.</p> <p>Updates for future device support.</p>	
5.20	Update to the <b>Settings</b> menu; back-end changes.	
5.30	<p>Added HTML documentation in the tool.</p> <p>Created How to topics for memory configuration.</p> <p>Moved subtabs to the top of the <b>Memory</b> tab and renamed <b>Summary</b> to <b>Address View</b>.</p> <p>Added <b>Delete All</b> regions command to the <b>Memory</b> tab.</p>	

---

**Revision history****Revision history**

Revision	Date	Description
**	2018-11-21	New document.
*A	2019-03-01	Updated to version 1.1.
*B	2019-10-17	Updated to version 2.0.
*C	2020-03-27	Updated to version 2.1.
*D	2020-09-01	Updated to version 2.20.
*E	2020-12-10	Updated to version 2.21.
*F	2021-03-15	Updated to version 3.0.
*G	2021-09-24	Updated to version 3.10.
*H	2022-09-12	Updated to version 4.0.
*I	2023-05-08	Updated to version 4.10.
*J	2024-02-05	Updated to version 4.20.
*K	2024-03-22	Updated to version 5.0.
*L	2024-09-27	Updated to version 5.10.
*M	2024-11-22	Update to include <b>Solutions</b> and <b>Memory</b> tabs configuration for early access pack.
*N	2024-12-06	Updated to version 5.20.
*O	2025-03-26	Updated to version 5.30.

## Trademarks

All referenced product or service names and trademarks are the property of their respective owners.

**Edition 2025-03-26**

**Published by**

**Infineon Technologies AG**  
**81726 Munich, Germany**

**© 2025 Infineon Technologies AG**  
**All Rights Reserved.**

**Do you have a question about any aspect of this document?**

**Email:** [erratum@infineon.com](mailto:erratum@infineon.com)

**Document reference**  
**IFX-xpp1712178400749**

## Important notice

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

## Warnings

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.