

Infineon Device Configurator user guide

Device Configurator version 5.50

About this document

[A newer version of this document may be available on the web here](#)

Scope and purpose

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

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
Bold	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
File > New	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

	About this document	1
	Table of contents	2
1	Overview	4
2	Stand-alone operation	5
2.1	Install Infineon Device Configurator	5
2.2	Run Device Configurator	6
3	Using with ModusToolbox™ application	13
3.1	Installation	13
3.2	Launch the Device Configurator	13
4	Quick start	15
5	Code generation	17
6	GUI description	18
6.1	Menus	18
6.2	Device tabs	19
6.3	Resources tabs	19
6.4	Panes	19
6.5	Icons	19
7	Resources tabs	20
7.1	Tab features	20
7.2	Solutions tab	22
7.3	Peripherals	23
7.4	Analog tab	24
7.5	Pins	25
7.6	Analog-Routing tab	27
7.7	System tab	31
7.8	Memory tab	32
7.9	MMIO-Clocks tab	48
7.10	Peripheral-Clocks tab	49
7.11	DMA tab	50
8	Panes	51
8.1	Parameters pane	51
8.2	Notice List	53
8.3	Code Preview pane	54
9	Version changes	55
	Revision history	57



Table of contents

Disclaimer 58

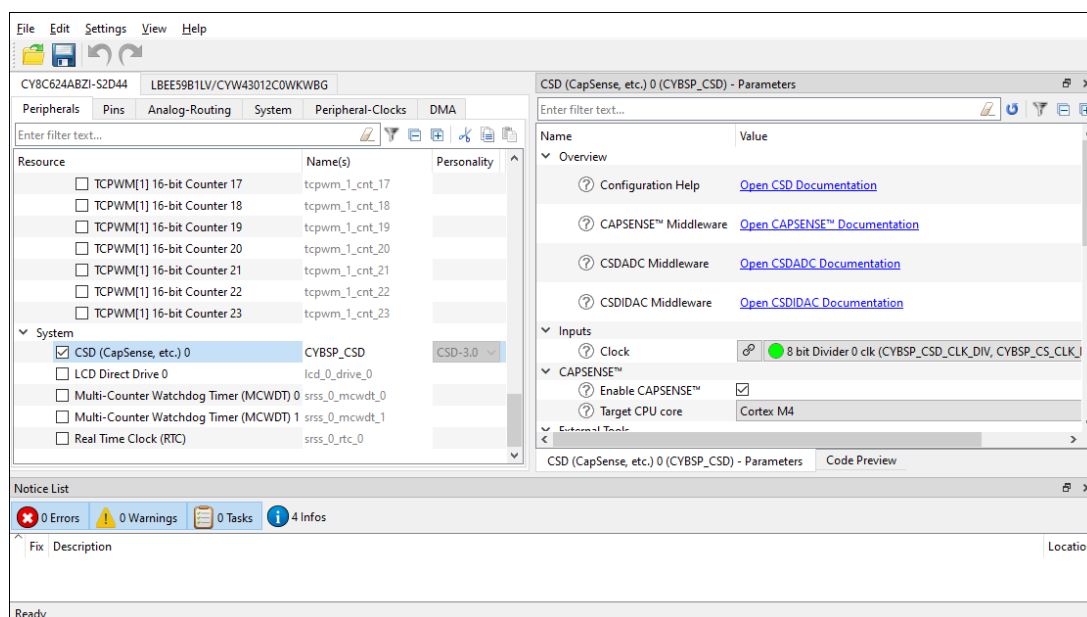
1 Overview

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 Keil µVision and IAR Embedded Workbench. The Device Configurator can be run as a GUI and via the command line.

- [Install Infineon Device Configurator](#)
- [Run Device Configurator](#)

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 or IAR Embedded Workbench outside of the ModusToolbox™ environment is only supported on Windows, as these are 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. By default, the tool is installed here:

`C:\Users\[user-home]\Infineon\Tools\InfineonDeviceConfigurator-[version]`

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 or IAR Embedded Workbench, 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 IDE in the context of a supported project. See the [Run Device Configurator](#) section.*

2.1.1 CMSIS Pack

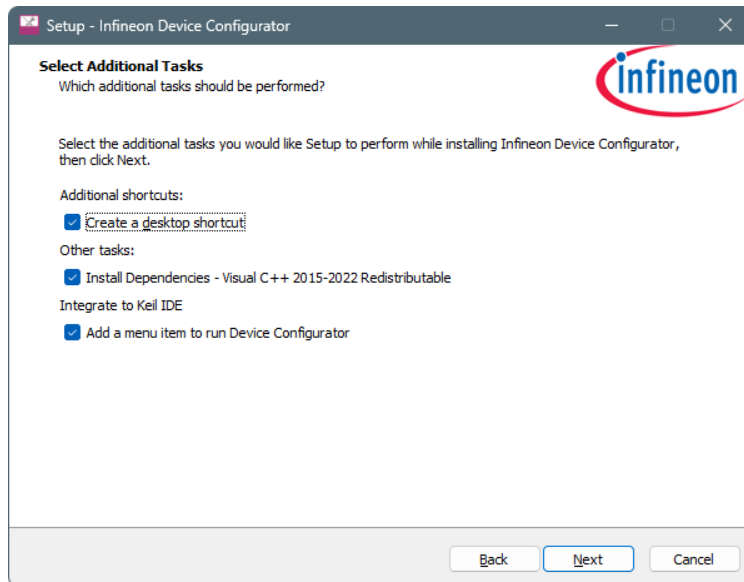
In order to use devices such as TLE994x_5x, you need to install the appropriate CMSIS pack. For example, the `MOTIX_TLE994x_5x_PDL.pack`. You can find the pack on the Infineon Developer Center webpage: <http://www.infineon.com/idc>

2 Stand-alone operation

2.2 Run Device Configurator

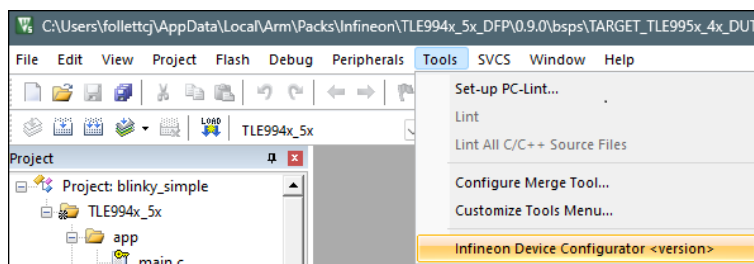
2.2.1 Keil μ Vision

The installer adds the **Run Device Configurator [version]** item to the **Tools** menu 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 \$\mu\$ Vision \(script\)](#) after the installer finishes.



2.2.1.1 Using the GUI

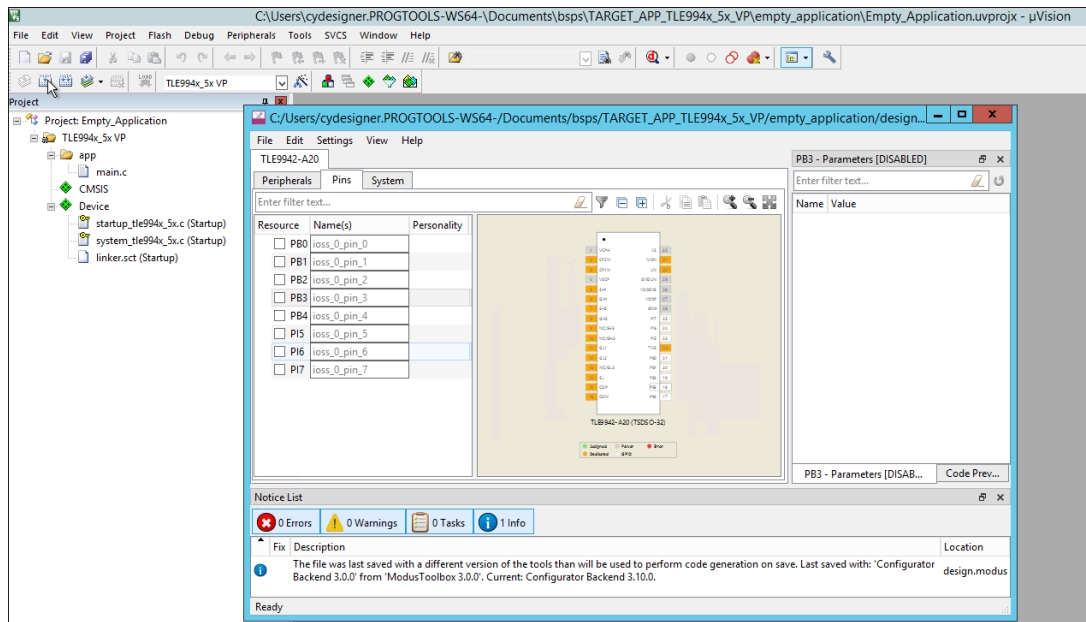
To run the Device Configurator GUI tool, open the Keil μ Vision IDE with a project. Then, select the **Tools > Run Infineon Device Configurator [version]**. If you don't see the menu item, refer to [section 2.4](#).



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 μ 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:

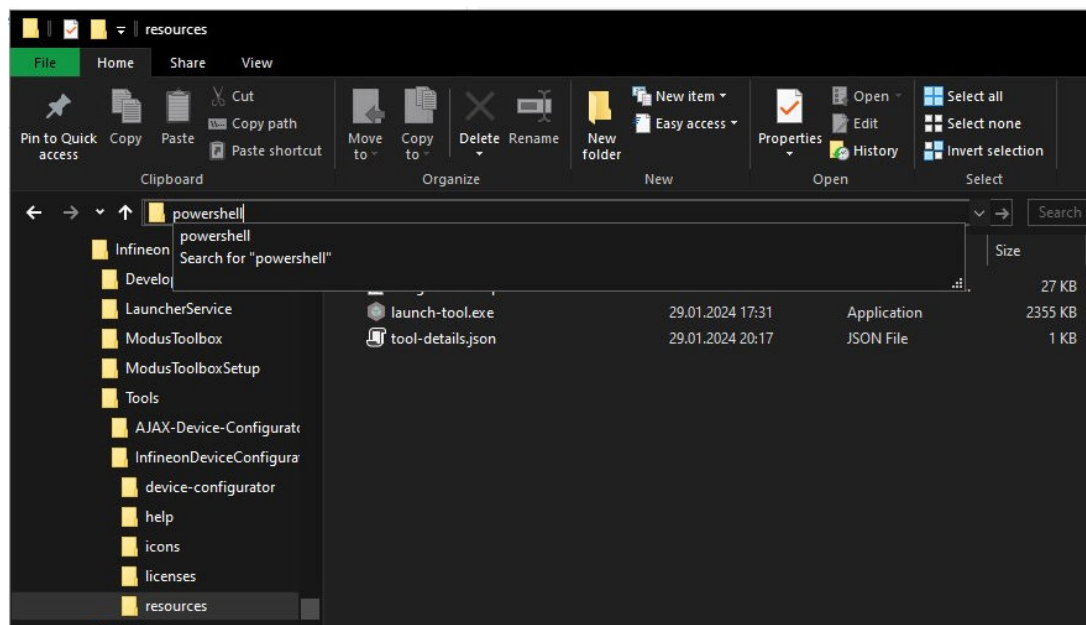
2 Stand-alone operation



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.

2.2.1.1.1 Add menu item to µVision (script)

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 (for example, *C:\Users\[user-home]\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
```

2 Stand-alone operation

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

```
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.

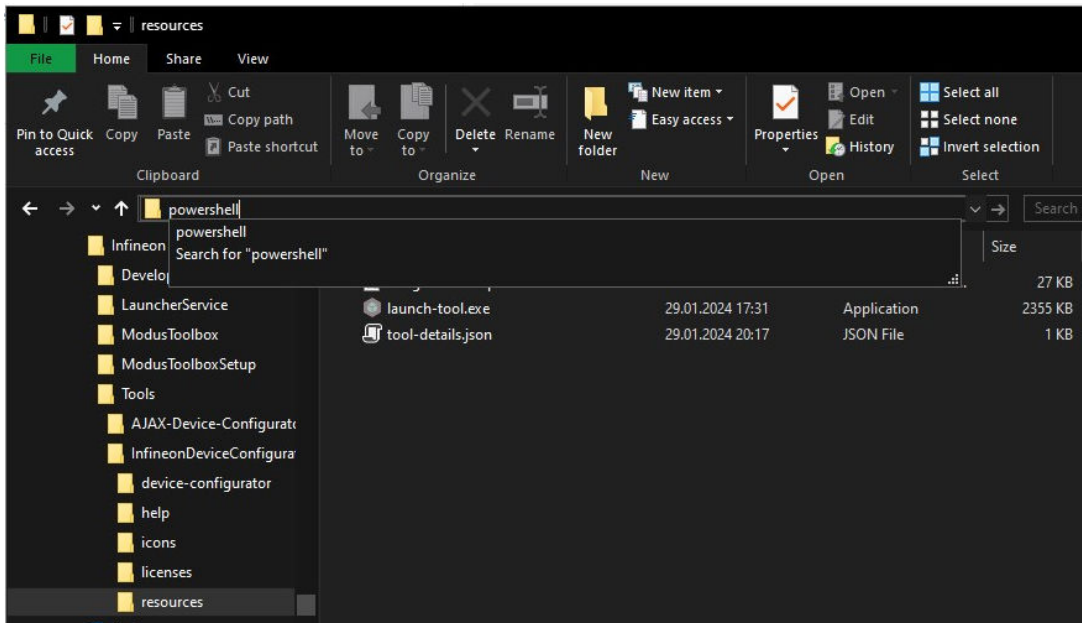
Please verify if the added Mag<Id>, Mfg<Id>, Mtx<Id>, Mid<Id> and Mex<Id> registry values are in the script output.

2 Stand-alone operation

2.2.1.1.2 Remove menu item from µVision (script)

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:

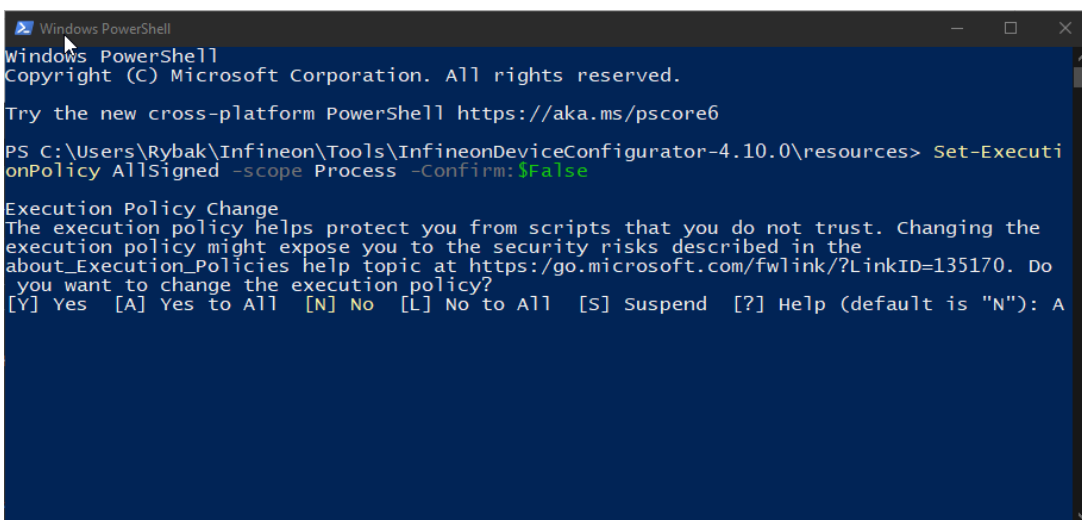
Open Windows Explorer and navigate to the resources folder (for example, `C:\Users\[user-home]\Infineon\Tools\InfineonDeviceConfigurator-[version]`), type "powershell" in address bar.



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]**.

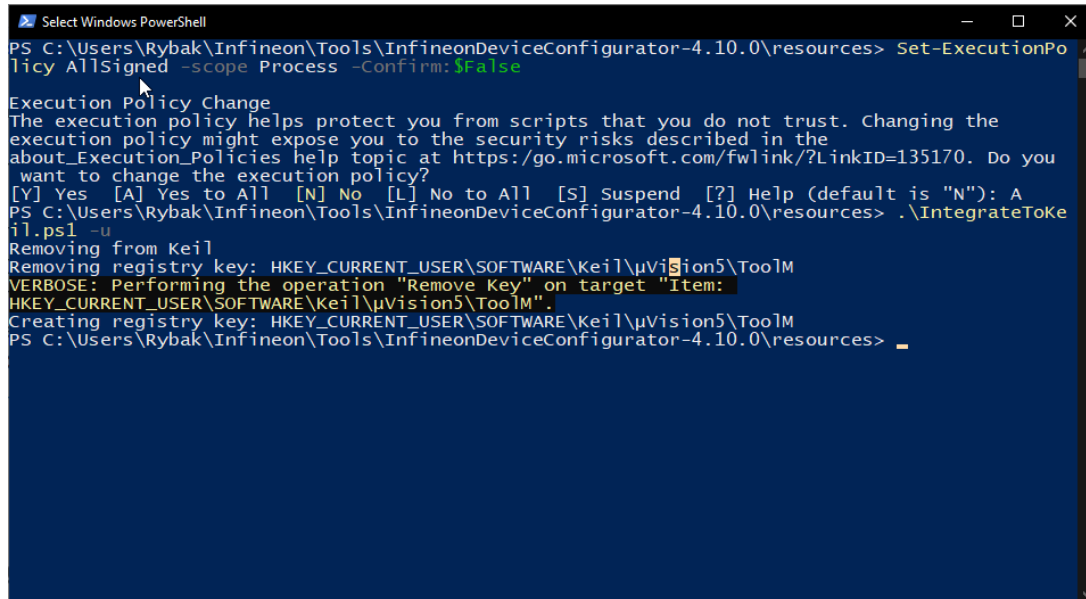


Then run the script:

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

2 Stand-alone operation

Please verify the script output:



```
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>
```

2.2.1.2 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.

2.2.2 IAR Embedded Workbench

The installer does not add any settings for IAR Embedded Workbench. You need to configure the GUI to add a menu item. You can also run the Device Configurator from the command line.

Create *design.modus* file

If you have a project without a *design.modus* file, follow these steps to allow configuration of your project with the Device Configurator:

1. Create a new text file called "design.modus" inside your project directory with the following minimal configuration:

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration app="BACKEND" formatVersion="14" lastSavedWith="Configurator Backend"
lastSavedWithVersion="3.50.0" xmlns="http://cypress.com/xsd/cydesignfile_v5">
  <Devices>
    <Device mpn="TLE9954EQ440">
      <BlockConfig/>
      <Netlist/>
    </Device>
  </Devices>
</Configuration>
```

2. Replace the "mpn" attribute with the part number of the used device in your project.
3. Save the "design.modus" file.

2 Stand-alone operation

Run Device Configurator

Run the tool as described in [Using the GUI](#) or [Using the command line](#).

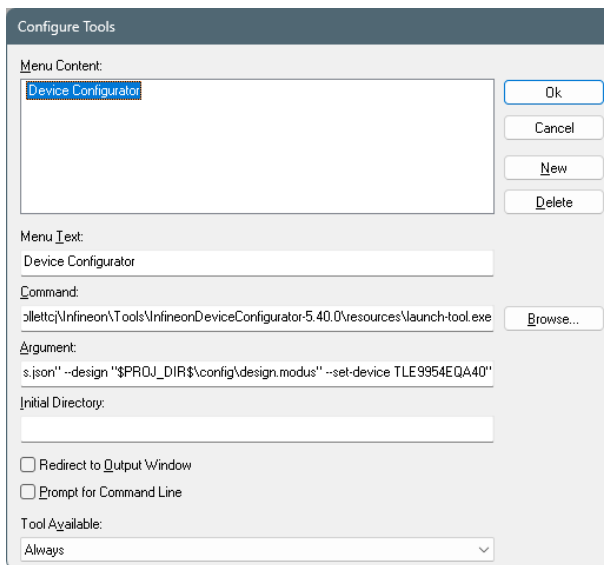
1. Upon first opening the Device Configurator for a project, resolve all pending tasks listed in the Notice List to achieve a valid configuration for the selected device. See [Notice List](#) for more details.
2. Set a configuration for your device by enabling desired blocks from the [Peripherals](#), [Pins](#), and [System tab](#).
3. Save the configuration.
4. Add the newly created *GeneratedSource* folder to your build environment in the same location as your *design.modus* file.
5. Include the *ifxcfg.h* header file from the *GeneratedSource* folder, and insert an `init_ifxcfg_all()` call inside your application start-up routine.

2.2.2.1 Using the GUI

Configure menu

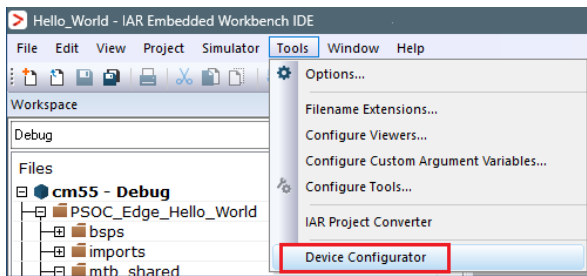
You need to configure IAR Embedded Workbench GUI manually to add Device Configurator to the menu, as follows:

1. On the main menu, select **Tools > Configure Tools...** to open the dialog.



2. On the dialog, enter the following:
 - **Menu Text:** Device Configurator
 - **Command:** `[path-to-device-configurator]\resources\launch-tool.exe`
You can navigate to the executable using the **[Browse...]** button.
 - **Argument:** `--library "[path-to-IAR-CMSIS-Packs]\Infineon\TLE994x_5x_DFP\1.0.0\props.json", "[path-to-IAR-CMSIS-Packs]\Infineon\TLE994x_5x_DFP\1.0.0\device-info\device-db\props.json" --design "$PROJ_DIR$\config\design.modus" --set-device TLE9954EQ440`
This passes a command to the Device Configurator to use the specified library, device-db, and *design.modus* file, and set the device.
The variable `$PROJ_DIR$` is the IAR Embedded Workbench project directory.
3. Click **OK** to close the dialog, and the menu item will be added:

2 Stand-alone operation



2.2.2.2 Using the command line

You can use the command line to run the Device Configurator with project context by providing specific arguments. For example (replace "[...]" with real values):

```
C:\Users\[user-home]\Infineon\Tools\InfineonDeviceConfigurator-[version]\resources\launch-
tool.exe --library "C:\Users\[user-
home]\IAR-CMSIS-Packs\Infineon\TLE994x_5x_DFP\1.0.0\props.json", "C:\Users\[user-home]\IAR-CMSIS-
Packs\Infineon\TLE994x_5x_DFP\1.0.0\device-info\device-db\props.json" --design "[IAR EW
project path]\config\design.modus" --set-device TLE9954EQ440
```

The following describe the arguments in detail:

- Mandatory: --library "[lib-file1]", "[lib-file2]"

Replace [lib-file1] with the local path to the top-level CMSIS pack *props.json* file. For example:

```
"C:\Users\[user-home]\IAR-CMSIS-Packs\Infineon\TLE994x_5x_DFP\1.0.0\props.json"
```

Replace [lib-file2] with the local path to the device-db *props.json* file in the CMSIS pack. For example:

```
"C:\Users\[user-home]\IAR-CMSIS-Packs\Infineon\TLE994x_5x_DFP\1.0.0\device-info\device-
db\props.json"
```

- Mandatory: --design "[design-file]"

Replace [design-file] with the local path to the *design.modus* file, usually located inside the project folder. For example:

```
"[your IAR EW project path]\config\design.modus"
```

For a template of a minimal *design.modus* file, see ____.

- Optional: --set-device [part-no]

Providing a valid [part-no] will update the device in the provided the *design.modus* file. For example, TLE9954EQ440

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)

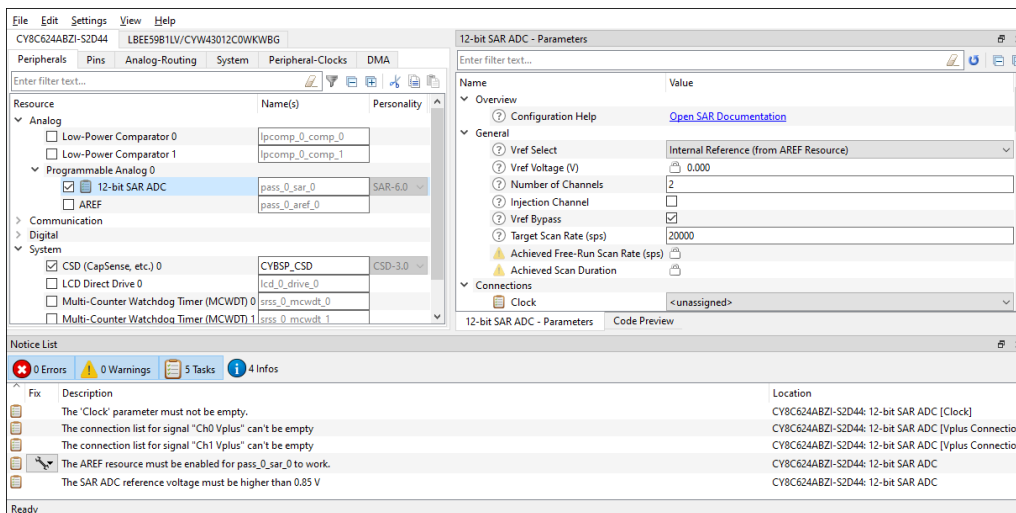
The Device Configurator executable can be run from the command line, and it also has a "cli" version of the executable as well. Running configurator executables from the command line can be useful as part of batch files or shell scripts to re-generate the source code based on the latest configuration 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

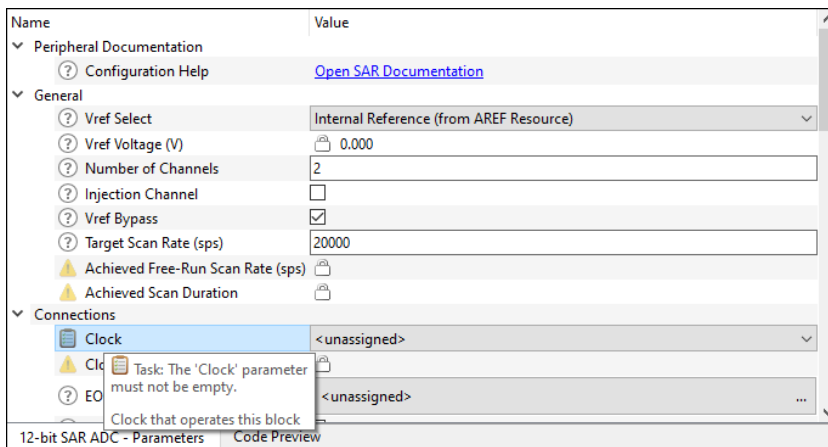
4 Quick start

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 μ 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.
4. 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/follettj/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_FWR_100 | (uint32_t)CY_SAR_VREF_SEL_BGR | (uint
#define pass_0_sar_0_SAMPLE ((uint32_t)SAR_SAMPLE_CTRL_EOS_DSI_OUT_EN_Msk | (uint32_t)CY_SAR_RI
#define pass_0_sar_0_CH0_CONFIG (((uint32_t)SAR0_VPLUS0_PORT << SAR_CHAN_CONFIG_POS_PORT_ADDR_P
#define pass_0_sar_0_CH1_CONFIG (((uint32_t)SAR0_VPLUS1_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 =
{

```

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:

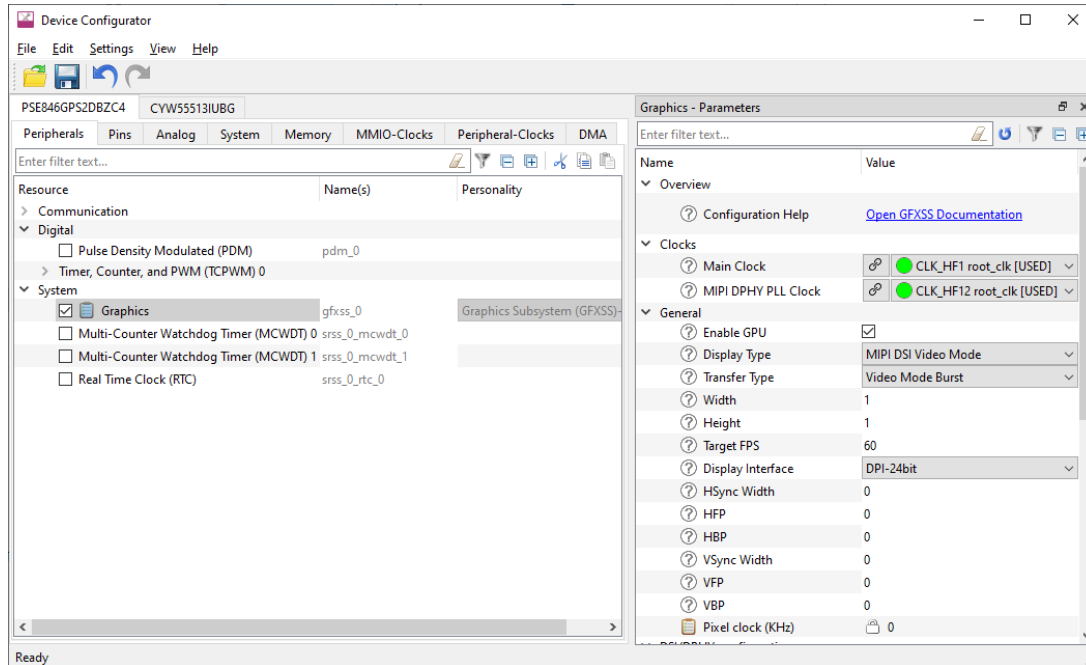
- Clocks: Peripheral clocks
- Connectivity: Configuration of the programmable analog and digital routing resources
- Peripherals: Fixed function analog and digital peripherals
- System: Overall configuration function to setup all power and clock options

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

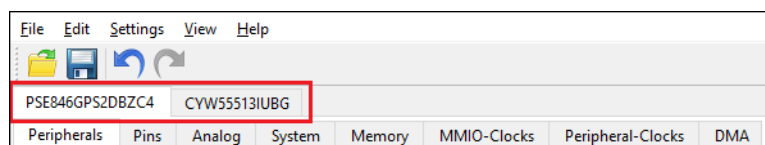
Contains toggles to hide or show different [Panels](#). All panels are shown by default. There is also a command to show or hide the Toolbar (hidden by default) and reset the view of the configurator to the default.

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 Panels

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 Notice List 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.

- [Tab features](#)
- [Solutions tab](#) – Options to configure multiple hardware blocks.
- [Peripherals](#) – Options to enable any of the analog (if available), digital, system, and communication hardware capabilities of the chip that are not related to the platform.
- [Analog](#) – For newer devices, options to enable and configure analog resources
- [Pins](#) – Options for all the pin related resources.
- [Analog-Routing tab](#) – For older devices, 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.

7.1 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.

7 Resources tabs

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.

Personality

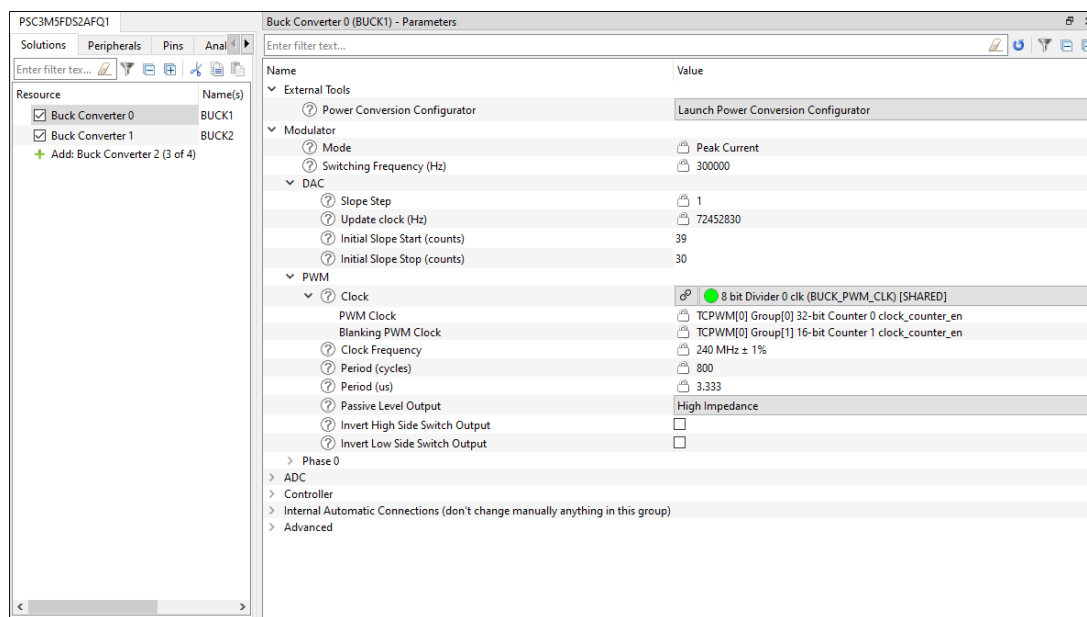
Each resource has a "Personality" file that contains the information for the given resource.

- 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 I²C.
- 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

7.2 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.

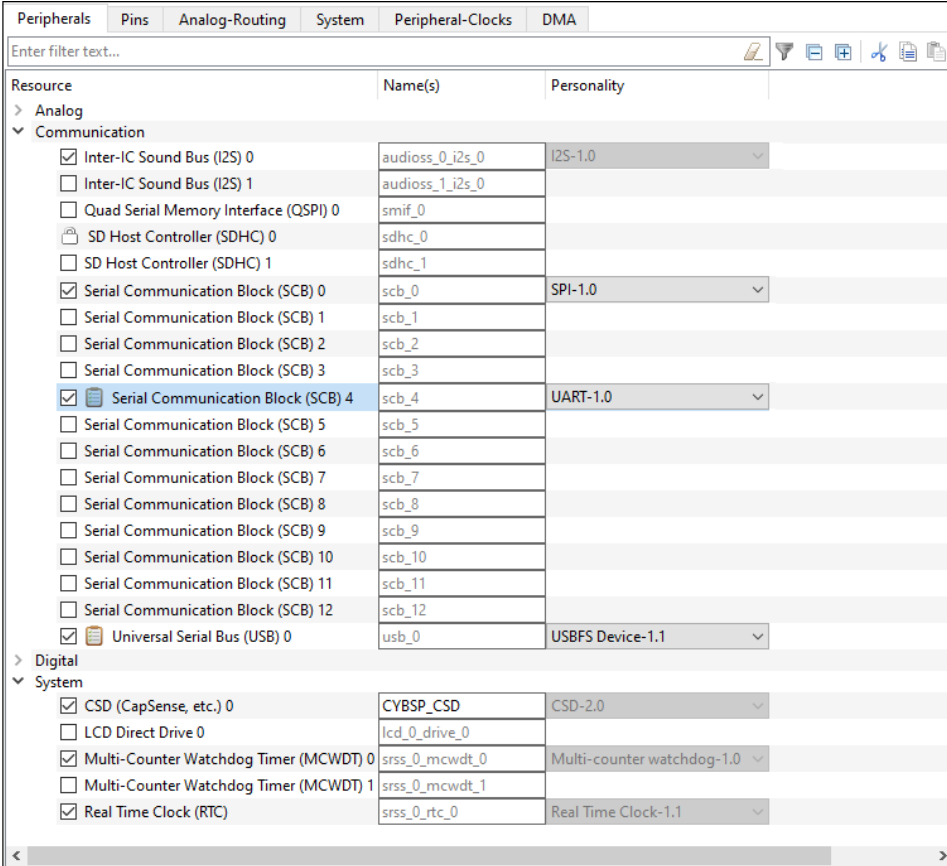


7 Resources tabs

7.3 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.

7.3.1 Device families with tabs



The screenshot shows the 'Peripherals' tab in the Infineon Device Configurator. The interface includes a filter box at the top and a tree view of resources. The resources are categorized into Analog, Communication, Digital, and System. The 'Communication' category is expanded, showing various serial communication blocks (SCB) and a USB device. The 'System' category is also expanded, showing various system peripherals like CSD, LCD, MCWDT, and RTC.

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

7.3.2 Device families without tabs

Enter filter text...

Resource	Name(s)	Personality
Peripherals		
<input checked="" type="checkbox"/> ADC	adc_0	ADC-1.0
<input type="checkbox"/> ARM IO	armio_0	
<input type="checkbox"/> Audio	audio_0	
<input checked="" type="checkbox"/> Bluetooth	bluetooth_0	Bluetooth-1.0
<input type="checkbox"/> Clock	clock_0	
<input type="checkbox"/> GCI-SECI	coex_0	
<input checked="" type="checkbox"/> I2C	i2c_0	I2C-1.0
<input type="checkbox"/> Keyboard Scanner	keyscan_0	
<input type="checkbox"/> Low Noise/External Power Amplifiers	amplifiers_0	
<input type="checkbox"/> PDM	pdm_0	
<input type="checkbox"/> PWM 0	pwm_0	
<input type="checkbox"/> PWM 1	pwm_1	
<input type="checkbox"/> PWM 2	pwm_2	
<input type="checkbox"/> PWM 3	pwm_3	
<input type="checkbox"/> PWM 4	pwm_4	
<input type="checkbox"/> PWM 5	pwm_5	
<input type="checkbox"/> Quadrature Decoder	quadrature_0	
<input type="checkbox"/> SPI 1	spi_0	
<input checked="" type="checkbox"/> SPI 2	spi_1	SPI-1.0
<input type="checkbox"/> SWD	swd_0	
<input type="checkbox"/> UART 1 (HCI UART)	uart_0	
<input checked="" type="checkbox"/> UART 2 (PUART)	uart_1	UART-1.0
<input checked="" type="checkbox"/> Pins	ioss_0	Pins-1.0

8 7 6 5 4 3 2 1

A

B

C

D

E

F

G

H

ADC_0

CTB0_0

P8

P3

P6

P17

P9

P12

VDDO1

CTB0_1

P15

P2

P5

P14

P11

P13

VDDC

CTB0_2

VSSC

P4

P8

VSSC

P10

P1

HOST_WAKE

CTB0_3

CTB0_4

P26

P29

P0

VDDO2

CTA0

DEV_WAKE

CTB0_5

CTB0_6

VSSC

P37

VDDC

CTA1

PLVSS

IFVDD

CTB0_7

P32

P27

CTA2

PLVDD

VCOVSS

PAVSS

CTB0_8

CTB0_9

CTB0_10

CTB0_11

CTB0_12

CTB0_13

CTB0_14

CTB0_15

CTB0_16

CTB0_17

CTB0_18

CTB0_19

CTB0_20

CTB0_21

CTB0_22

CTB0_23

CTB0_24

CTB0_25

CTB0_26

CTB0_27

CTB0_28

CTB0_29

CTB0_30

CTB0_31

CTB0_32

CTB0_33

CTB0_34

CTB0_35

CTB0_36

CTB0_37

CTB0_38

CTB0_39

CTB0_40

CTB0_41

CTB0_42

CTB0_43

CTB0_44

CTB0_45

CTB0_46

CTB0_47

CTB0_48

CTB0_49

CTB0_50

CTB0_51

CTB0_52

CTB0_53

CTB0_54

CTB0_55

CTB0_56

CTB0_57

CTB0_58

CTB0_59

CTB0_60

CTB0_61

CTB0_62

CTB0_63

CTB0_64

CTB0_65

CTB0_66

CTB0_67

CTB0_68

CTB0_69

CTB0_70

CTB0_71

CTB0_72

CTB0_73

CTB0_74

CTB0_75

CTB0_76

CTB0_77

CTB0_78

CTB0_79

CTB0_80

CTB0_81

CTB0_82

CTB0_83

CTB0_84

CTB0_85

CTB0_86

CTB0_87

CTB0_88

CTB0_89

CTB0_90

CTB0_91

CTB0_92

CTB0_93

CTB0_94

CTB0_95

CTB0_96

CTB0_97

CTB0_98

CTB0_99

CTB0_100

CTB0_101

CTB0_102

CTB0_103

CTB0_104

CTB0_105

CTB0_106

CTB0_107

CTB0_108

CTB0_109

CTB0_110

CTB0_111

CTB0_112

CTB0_113

CTB0_114

CTB0_115

CTB0_116

CTB0_117

CTB0_118

CTB0_119

CTB0_120

CTB0_121

CTB0_122

CTB0_123

CTB0_124

CTB0_125

CTB0_126

CTB0_127

CTB0_128

CTB0_129

CTB0_130

CTB0_131

CTB0_132

CTB0_133

CTB0_134

CTB0_135

CTB0_136

CTB0_137

CTB0_138

CTB0_139

CTB0_140

CTB0_141

CTB0_142

CTB0_143

CTB0_144

CTB0_145

CTB0_146

CTB0_147

CTB0_148

CTB0_149

CTB0_150

CTB0_151

CTB0_152

CTB0_153

CTB0_154

CTB0_155

CTB0_156

CTB0_157

CTB0_158

CTB0_159

CTB0_160

CTB0_161

CTB0_162

CTB0_163

CTB0_164

CTB0_165

CTB0_166

CTB0_167

CTB0_168

CTB0_169

CTB0_170

CTB0_171

CTB0_172

CTB0_173

CTB0_174

CTB0_175

CTB0_176

CTB0_177

CTB0_178

CTB0_179

CTB0_180

CTB0_181

CTB0_182

CTB0_183

CTB0_184

CTB0_185

CTB0_186

CTB0_187

CTB0_188

CTB0_189

CTB0_190

CTB0_191

CTB0_192

CTB0_193

CTB0_194

CTB0_195

CTB0_196

CTB0_197

CTB0_198

CTB0_199

CTB0_200

CTB0_201

CTB0_202

CTB0_203

CTB0_204

CTB0_205

CTB0_206

CTB0_207

CTB0_208

CTB0_209

CTB0_210

CTB0_211

CTB0_212

CTB0_213

CTB0_214

CTB0_215

CTB0_216

CTB0_217

CTB0_218

CTB0_219

CTB0_220

CTB0_221

CTB0_222

CTB0_223

CTB0_224

CTB0_225

CTB0_226

CTB0_227

CTB0_228

CTB0_229

CTB0_230

CTB0_231

CTB0_232

CTB0_233

CTB0_234

CTB0_235

CTB0_236

CTB0_237

CTB0_238

CTB0_239

CTB0_240

CTB0_241

CTB0_242

CTB0_243

CTB0_244

CTB0_245

CTB0_246

CTB0_247

CTB0_248

CTB0_249

CTB0_250

CTB0_251

CTB0_252

CTB0_253

CTB0_254

CTB0_255

CTB0_256

CTB0_257

CTB0_258

CTB0_259

CTB0_260

CTB0_261

CTB0_262

CTB0_263

CTB0_264

CTB0_265

CTB0_266

CTB0_267

CTB0_268

CTB0_269

CTB0_270

CTB0_271

CTB0_272

CTB0_273

CTB0_274

CTB0_275

CTB0_276

CTB0_277

CTB0_278

CTB0_279

CTB0_280

CTB0_281

CTB0_282

CTB0_283

CTB0_284

CTB0_285

CTB0_286

CTB0_287

CTB0_288

CTB0_289

CTB0_290

CTB0_291

CTB0_292

CTB0_293

CTB0_294

CTB0_295

CTB0_296

CTB0_297

CTB0_298

CTB0_299

CTB0_300

CTB0_301

CTB0_302

CTB0_303

CTB0_304

CTB0_305

CTB0_306

CTB0_307

CTB0_308

CTB0_309

CTB0_310

CTB0_311

CTB0_312

CTB0_313

CTB0_314

CTB0_315

CTB0_316

CTB0_317

CTB0_318

CTB0_319

CTB0_320

CTB0_321

CTB0_322

CTB0_323

CTB0_324

CTB0_325

CTB0_326

CTB0_327

CTB0_328

CTB0_329

CTB0_330

CTB0_331

CTB0_332

CTB0_333

CTB0_334

CTB0_335

CTB0_336

CTB0_337

CTB0_338

CTB0_339

CTB0_340

CTB0_341

CTB0_342

CTB0_343

CTB0_344

CTB0_345

CTB0_346

CTB0_347

CTB0_348

CTB0_349

CTB0_350

CTB0_351

CTB0_352

CTB0_353

CTB0_354

CTB0_355

CTB0_356

CTB0_357

CTB0_358

CTB0_359

CTB0_360

CTB0_361

CTB0_362

CTB0_363

CTB0_364

CTB0_365

CTB0_366

CTB0_367

CTB0_368

CTB0_369

CTB0_370

CTB0_371

CTB0_372

CTB0_373

CTB0_374

CTB0_375

CTB0_376

CTB0_377

CTB0_378

CTB0_379

CTB0_380

CTB0_381

CTB0_382

CTB0_383

CTB0_384

CTB0_385

CTB0_386

CTB0_387

CTB0_388

CTB0_389

CTB0_390

CTB0_391

CTB0_392

CTB0_393

CTB0_394

CTB0_395

CTB0_396

CTB0_397

CTB0_398

CTB0_399

CTB0_400

CTB0_401

CTB0_402

CTB0_403

CTB0_404

CTB0_405

CTB0_406

CTB0_407

CTB0_408

CTB0_409

CTB0_410

CTB0_411

CTB0_412

CTB0_413

CTB0_414

CTB0_415

CTB0_416

CTB0_417

CTB0_418

CTB0_419

CTB0_420

CTB0_421

CTB0_422

CTB0_423

CTB0_424

CTB0_425

CTB0_426

CTB0_427

CTB0_428

CTB0_429

CTB0_430

CTB0_431

CTB0_432

CTB0_433

CTB0_434

CTB0_435

CTB0_436

CTB0_437

CTB0_438

CTB0_439

CTB0_440

CTB0_441

CTB0_442

CTB0_443

CTB0_444

CTB0_445

CTB0_446

CTB0_447

CTB0_448

CTB0_449

CTB0_450

CTB0_451

CTB0_452

CTB0_453

CTB0_454

CTB0_455

CTB0_456

CTB0_457

CTB0_458

CTB0_459

CTB0_460

CTB0_461

CTB0_462

CTB0_463

CTB0_464

CTB0_465

CTB0_466

CTB0_467

CTB0_468

CTB0_469

CTB0_470

CTB0_471

CTB0_472

CTB0_473

CTB0_474

CTB0_475

CTB0_476

CTB0_477

CTB0_478

CTB0_479

CTB0_480

CTB0_481

CTB0_482

CTB0_483

CTB0_484

CTB0_485

CTB0_486

CTB0_487

CTB0_488

CTB0_489

CTB0_490

CTB0_491

CTB0_492

CTB0_493

CTB0_494

CTB0_495

CTB0_496

CTB0_497

CTB0_498

CTB0_499

CTB0_500

CTB0_501

CTB0_502

CTB0_503

CTB0_504

CTB0_505

CTB0_506

CTB0_507

CTB0_508

CTB0_509

CTB0_510

CTB0_511

CTB0_512

CTB0_513

CTB0_514

CTB0_515

CTB0_516

CTB0_517

CTB0_518

CTB0_519

CTB0_520

CTB0_521

CTB0_522

CTB0_523

CTB0_524

CTB0_525

CTB0_526

CTB0_527

CTB0_528

CTB0_529

CTB0_530

CTB0_531

CTB0_532

CTB0_533

CTB0_534

CTB0_535

CTB0_536

CTB0_537

CTB0_538

CTB0_539

CTB0_540

CTB0_541

CTB0_542

CTB0_543

CTB0_544

CTB0_545

CTB0_546

CTB0_547

CTB0_548

CTB0_549

CTB0_550

CTB0_551

CTB0_552

CTB0_553

CTB0_554

CTB0_555

CTB0_556

CTB0_557

CTB0_558

CTB0_559

CTB0_560

CTB0_561

CTB0_562

CTB0_563

CTB0_564

CTB0_565

CTB0_566

CTB0_567

CTB0_568

CTB0_569

CTB0_570

CTB0_571

CTB0_572

CTB0_573

CTB0_574

CTB0_575

CTB0_576

CTB0_577

CTB0_578

CTB0_579

CTB0_580

CTB0_581

CTB0_582

CTB0_583

CTB0_584

CTB0_585

CTB0_586

CTB0_587

CTB0_588

CTB0_589

CTB0_590

CTB0_591

CTB0_592

CTB0_593

CTB0_594

CTB0_595

CTB0_596

CTB0_597

CTB0_598

CTB0_599

CTB0_600

CTB0_601

CTB0_602

CTB0_603

CTB0_604

CTB0_605

CTB0_606

CTB0_607

CTB0_608

CTB0_609

CTB0_610

CTB0_611

CTB0_612

CTB0_613

CTB0_614

CTB0_615

CTB0_616

CTB0_617

CTB0_618

CTB0_619

CTB0_620

CTB0_621

CTB0_622

CTB0_623

CTB0_624

CTB0_625

CTB0_626

CTB0_627

CTB0_628

CTB0_629

CTB0_630

CTB0_631

CTB0_632

CTB0_633

CTB0_634

CTB0_635

CTB0_636

CTB0_637

CTB0_638

CTB0_639

CTB0_640

CTB0_641

CTB0_642

CTB0_643

CTB0_644

CTB0_645

CTB0_646

CTB0_647

CTB0_648

CTB0_649

CTB0_650

CTB0_651

CTB0_652

CTB0_653

CTB0_654

CTB0_655

CTB0_656

CTB0_657

CTB0_658

CTB0_659

CTB0_660

CTB0_661

CTB0_662

CTB0_663

CTB0_664

CTB0_665

CTB0_666

CTB0_667

CTB0_668

CTB0_669

CTB0_670

CTB0_671

CTB0_672

CTB0_673

CTB0_674

CTB0_675

CTB0_676

CTB0_677

CTB0_678

CTB0_679

CTB0_680

CTB0_681

CTB0_682

CTB0_683

CTB0_684

CTB0_685

CTB0_686

CTB0_687

CTB0_688

CTB0_689

CTB0_690

CTB0_691

CTB0_692

CTB0_693

CTB0_694

CTB0_695

CTB0_696

CTB0_697

CTB0_698

CTB0_699

CTB0_700

CTB0_701

CTB0_702

CTB0_703

CTB0_704

CTB0_705

CTB0_706

CTB0_707

CTB0_708

CTB0_709

CTB0_710

CTB0_711

CTB0_712

CTB0_713

CTB0_714

CTB0_715

CTB0_716

CTB0_717

CTB0_718

CTB0_719

CTB0_720

CTB0_721

CTB0_722

CTB0_723

CTB0_724

CTB0_725

CTB0_726

CTB0_727

CTB0_728

CTB0_729

CTB0_730

CTB0_731

CTB0_732

CTB0_733

CTB0_734

CTB0_735

CTB0_736

CTB0_737

CTB0_738

CTB0_739

CTB0_740

CTB0_741

CTB0_742

CTB0_743

CTB0_744

CTB0_745

CTB0_746

CTB0_747

CTB0_748

CTB0_749

CTB0_750

CTB0_751

CTB0_752

CTB0_753

CTB0_754

CTB0_755

CTB0_756

CTB0_757

CTB0_758

CTB0_759

CTB0_760

CTB0_761

CTB0_762

CTB0_763

CTB0_764

CTB0_765

CTB0_766

CTB0_767

CTB0_768

CTB0_769

CTB0_770

CTB0_771

CTB0_772

CTB0_773

CTB0_774

CTB0_775

CTB0_776

CTB0_777

CTB0_778

CTB0_779

CTB0_780

CTB0_781

CTB0_782

CTB0_783

CTB0_784

CTB0_785

CTB0_786

CTB0_787

CTB0_788

CTB0_789

CTB0_790

CTB0_791

CTB0_792

CTB0_793

CTB0_794

CTB0_795

CTB0_796

CTB0_797

CTB0_798

CTB0_799

CTB0_800

CTB0_801

CTB0_802

CTB0_803

CTB0_804

CTB0_805

CTB0_806

CTB0_807

CTB0_808

CTB0_809

CTB0_810

CTB0_811

CTB0_812

CTB0_813

CTB0_814

CTB0_815

CTB0_816

CTB0_817

CTB0_818

CTB0_819

CTB0_820

CTB0_821

CTB0_822

CTB0_823

CTB0_824

CTB0_825

CTB0_826

CTB0_827

CTB0_828

CTB0_829

CTB0_830

CTB0_831

CTB0_832

CTB0_833

CTB0_834

CTB0_835

CTB0_836

CTB0_837

CTB0_838

CTB0_839

CTB0_840

CTB0_841

CTB0_842

CTB0_843

CTB0_844

CTB0_845

CTB0_846

CTB0_847

CTB0_848

CTB0_849

CTB0_850

CTB0_851

CTB0_852

CTB0_853

CTB0_854

CTB0_855

CTB0_856

CTB0_857

CTB0_858

CTB0_859

CTB0_860

CTB0_861

CTB0_862

CTB0_863

CTB0_864

CTB0_865

CTB0_866

CTB0_867

CTB0_868

CTB0_869

CTB0_870

CTB0_871

CTB0_872

CTB0_873

CTB0_874

CTB0_875

CTB0_876

CTB0_877

CTB0_878

CTB0_879

CTB0_880

CTB0_881

CTB0_882

CTB0_883

CTB0_884

CTB0_885

CTB0_886

CTB0_887

CTB0_888

CTB0_889

CTB0_890

CTB0_891

CTB0_892

CTB0_893

CTB0_894

CTB0_895

CTB0_896

CTB0_897

CTB0_898

CTB0_899

CTB0_900

CTB0_901

CTB0_902

CTB0_903

CTB0_904

CTB0_905

CTB0_906

CTB0_907

CTB0_908

CTB0_909

CTB0_910

CTB0_911

CTB0_912

CTB0_913

CTB0_914

CTB0_915

CTB0_916

CTB0_917

CTB0_918

CTB0_919

CTB0_920

CTB0_921

CTB0_922

CTB0_923

CTB0_924

CTB0_925

CTB0_926

CTB0_927

CTB0_928

CTB0_929

CTB0_930

7 Resources tabs

7.5 Pins

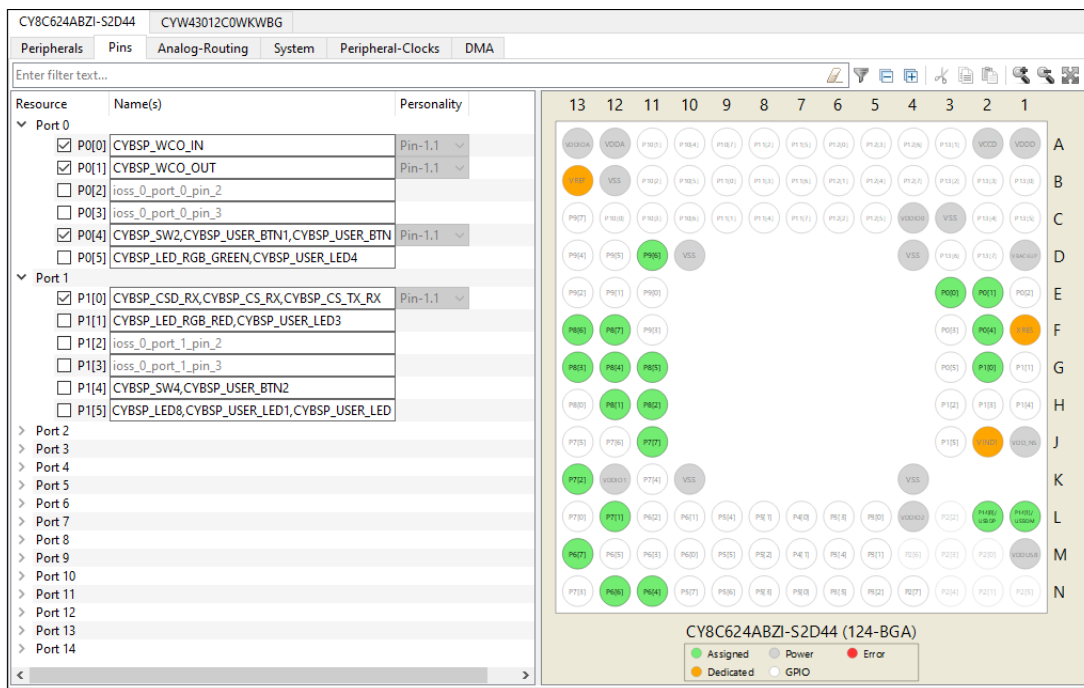
The **Pins** tab/tree is where you enable all the pin related resources. All available pins are shown in an expandable tree, arranged by port number. The filter box and the hide disabled button above the list of pins allows you to limit the pins shown in the tree. This tab allows you to enter one or 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.

7.5.1 Device families with tabs



Resource

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	

CY8C624ABZI-S2D44 (124-BGA)

Legend:
● Assigned
● Power
● Error
● Dedicated
● GPIO

7 Resources tabs

7.5.2 Device families without tabs

Enter filter text...

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

8 7 6 5 4 3 2 1

A B C D E F G H

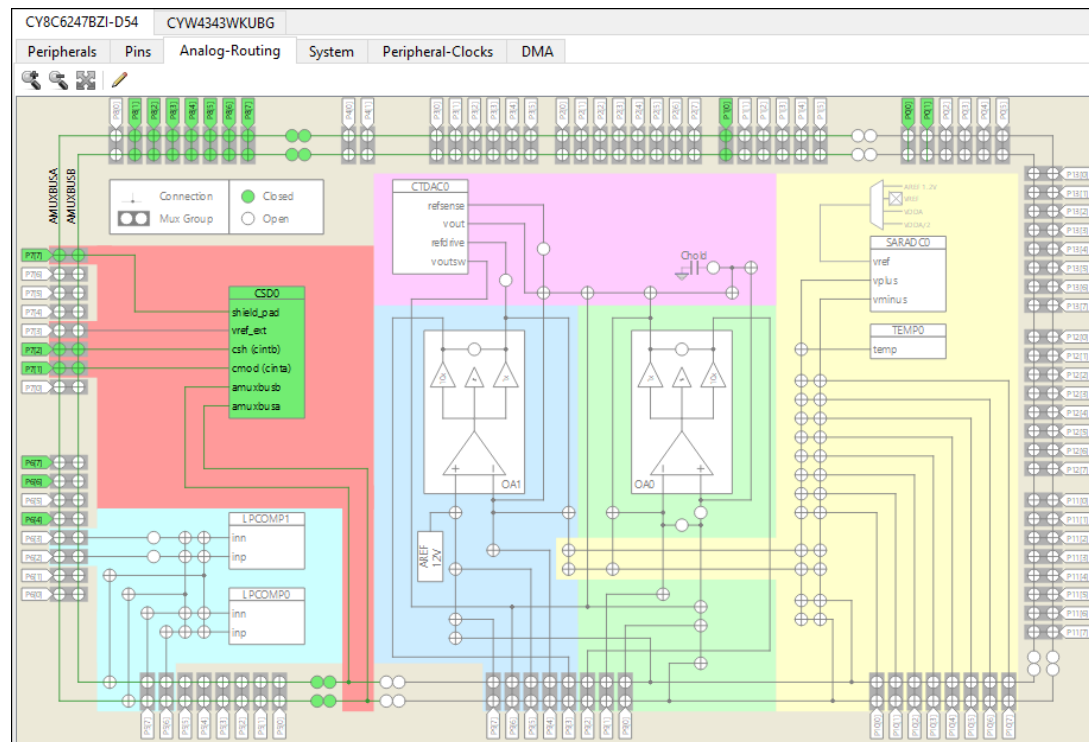
CYW20819A1KFBG (62-FBGA)


● No Connect ● Assigned ● Power
● Error ● Dedicated ● GPIO

7 Resources tabs

7.6 Analog-Routing tab

For older devices, the **Analog-Routing** tab shows the various analog resources in your application. Enabled resources are green.



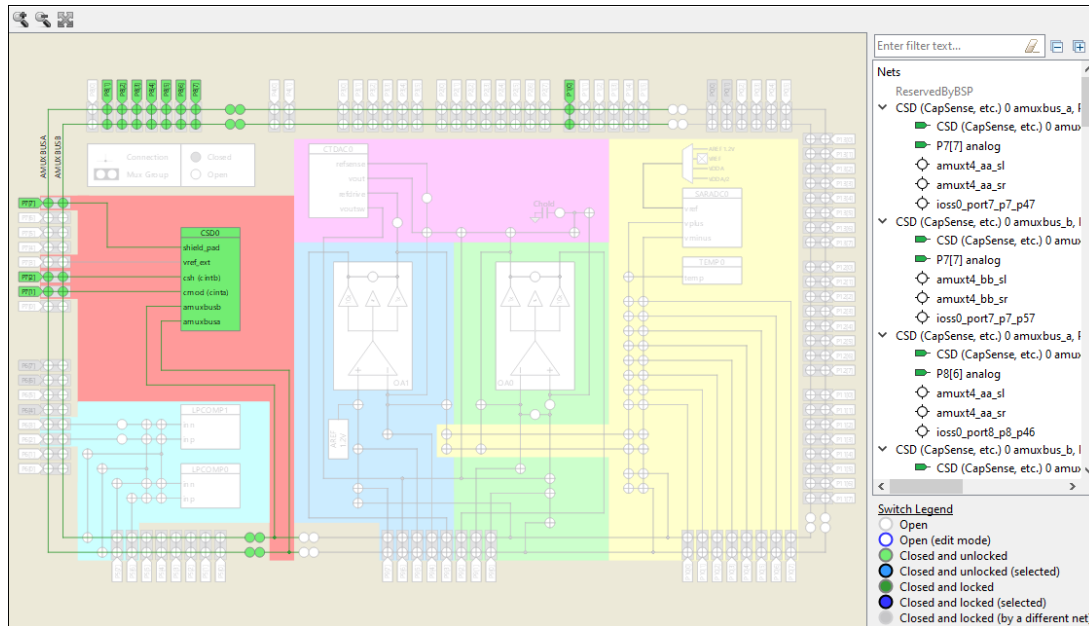
The **Edit**  command opens the [Analog Route Editor](#).

7 Resources tabs

7.6.1 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.*

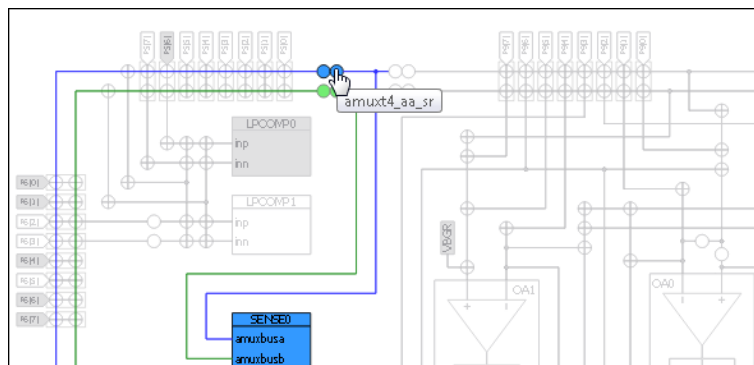


Note: *If there are configuration errors, complete routing results will not be available; only locked resources. If you open the Analog Editor in this error state, a warning message will display. You can still lock and unlock switches, but you won't get complete routing results as long as the configuration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

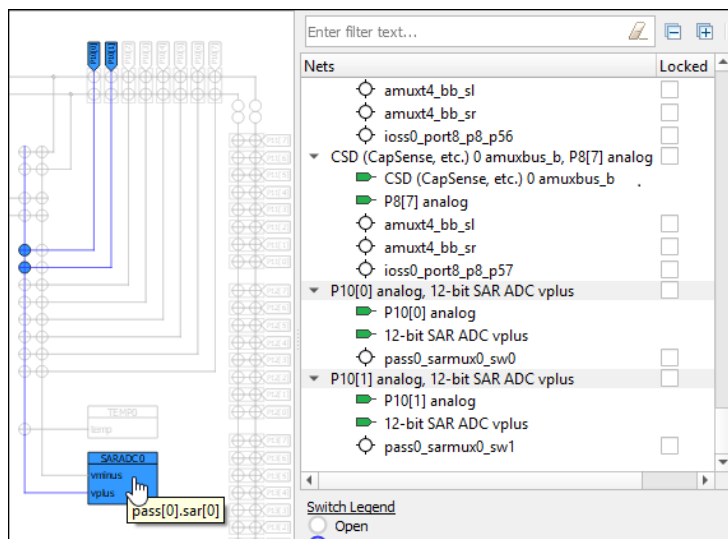
7 Resources tabs

7.6.1.1 Select a resource

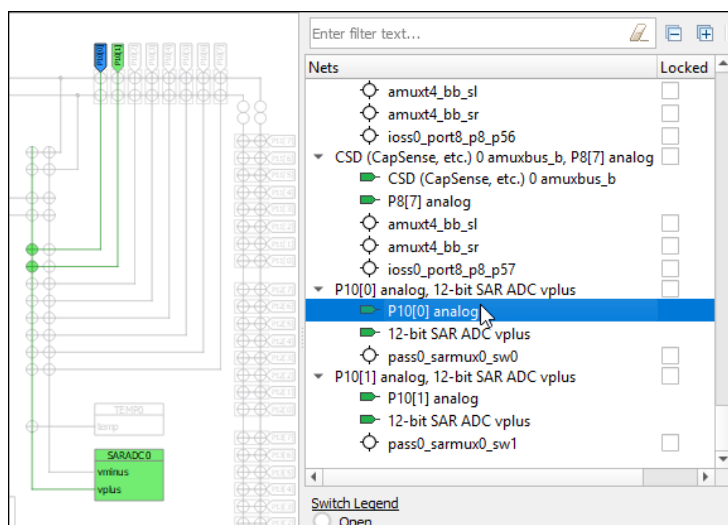
To select an analog resource, click on it. Any enabled (green) element in the tree can be selected. The resource and the associated route(s) become blue. Also, the **Edit Route** command appears on the toolbar. See [Edit Route](#).



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

7.6.1.2 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.

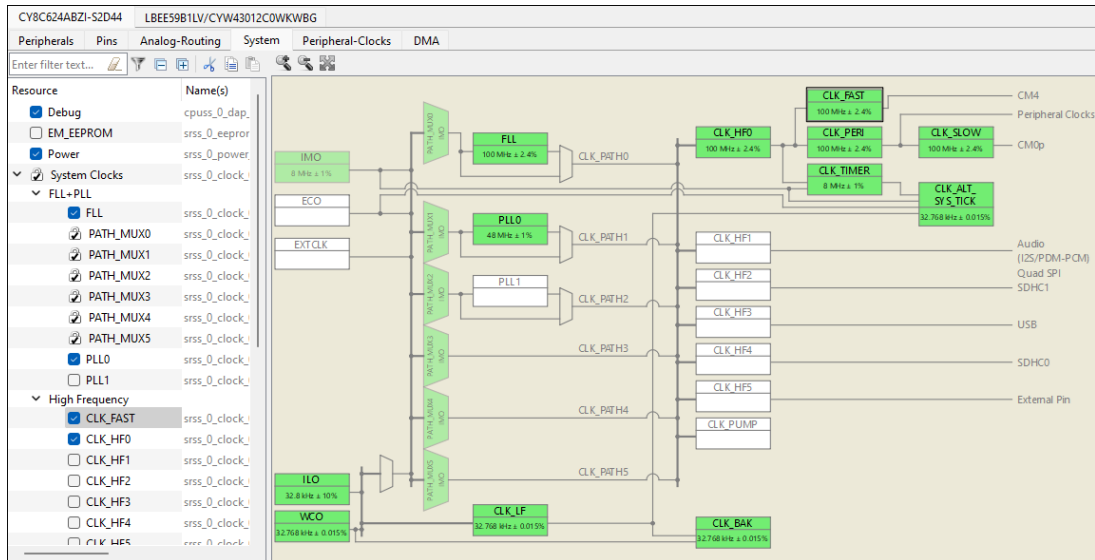
The toolbar shows the **Finish edit** command to return the editor to selection mode.

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

7.7 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.

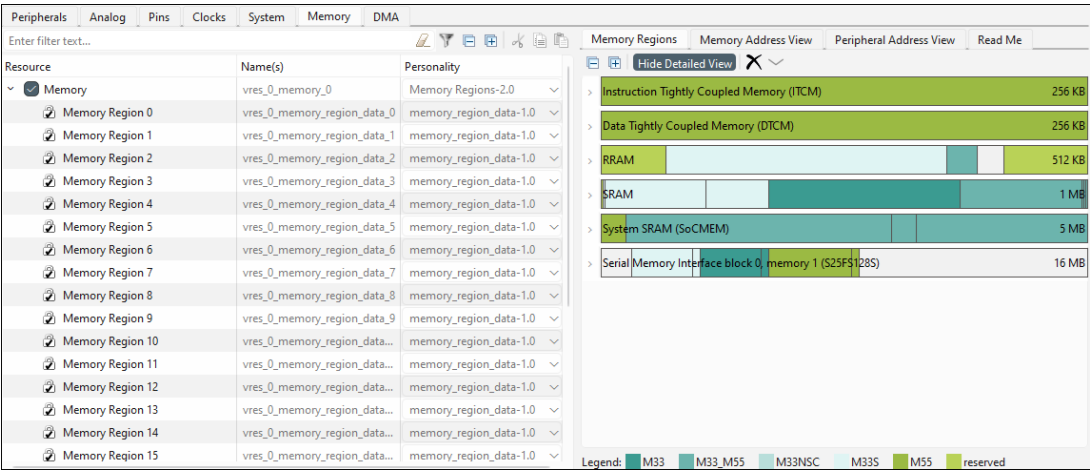
Note: The semi-transparent (faded) elements in the diagram indicate that their enabled state is locked.

7 Resources tabs

7.8 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 various subtabs: **Memory Regions**, **Memory Address View**, **Peripheral 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 Resources tabs

7.8.1 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, **Delete All** memory regions, and **Autocorrect All Overlaps**.

Memory Regions	Memory Address View	Peripheral Address View	Read Me
<div>Hide Detailed View</div>			
<div>Instruction Tightly Coupled Memory (ITCM) 256 KB</div>			
Region Id	Domain	Offset	Size
m55_code	M55	0x00000000	0x00040000 (256 KB)
USED: (256 KB) FREE: (0 Bytes)			
<div>Data Tightly Coupled Memory (DTCM) 256 KB</div>			
<div>RRAM 512 KB</div>			
Region Id	Domain	Offset	Size
extended_boot_reserved	reserved	0x00000000	0x00011000 (68 KB)
user_programmable	M33S	0x00011000	0x0004A000 (296 KB)
user_nvm	M33_M55	0x0005B000	0x00008000 (32 KB)
UNALLOCATED	<NONE>	0x00063000	0x00007000 (28 KB)
reserved_region	reserved	0x0006A000	0x00016000 (88 KB)
USED: (484 KB) FREE: (28 KB)			
<div>SRAM 1 MB</div>			
<div>System SRAM (SoCMEM) 5 MB</div>			
<div>Serial Memory Interface block 0, memory 1 (S25FS128S) 16 MB</div>			
Legend: M33 M33_M55 M33NSC M33S M55 reserved			

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.

Serial Memory Interface block 0, memory 1 (S25FS128S) 16 MB			
Region Id	Domain	Offset	Size
UNALLOCATED	<NONE>	Id: m33_nvm 0x000 Offset: 0x00340000 Size: 0x00200000 (2 MB)	MB)
m33a_nvm	M33S	0x00100000	0x00200000 (2 MB)
m33a_trailer	M33S	0x00300000	0x00040000 (256 KB)
m33_nvm	M33	0x00340000	0x00200000 (2 MB)
m33_trailer	M33	0x00540000	0x00040000 (256 KB)
m55_nvm	M55	0x00580000	0x00200000 (2.75 MB)
m55_trailer	M55	0x00840000	0x00040000 (256 KB)
UNALLOCATED	<NONE>	0x00880000	0x00780000 (7.5 MB)
USED: (7.5 MB) FREE: (8.5 MB)			

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 becomes red and displays the maximum allowed offset along with the currently set offset that is in violation of the max. If more size is allocated than is available, that text will be in red as well.

RRAM 512 KB			
Region Id	Domain	Offset	Size
extended_boot_reserved	reserved	0x00000000	0x00011000 (68 KB)
whatever	reserved	0x00011000	0x00006000 (24 KB)
user_programmable	M33	0x00017000	0x00052000 (328 KB)
user_nvm	M33S	0x00069000	0x00016000 (88 KB)
reserved_region	reserved	0x0006A000	0x00016000 (88 KB)
RRAM_REGION	M33NSC	0x00080000	0x00007000 (28 KB)
MAX: 0x00080000 USED: (540 KB) ACT: 0x00087000 FREE: (-28 KB)			

7 Resources tabs

7.8.1.1 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.

Memory Regions

Memory Address View

Peripheral Address View

Read Me

Hide Detailed View

Instruction Tightly Coupled Memory (ITCM)256 KB

Region Id	Domain	Offset	Size	CM55_ITCM_CM33	CM55_ITCM_CM33_S	CM55_ITCM_INTERNAL	Description
m55_code	M55	0x00000000	0x00040000 (256 KB)	0x48000000	0x58000000	0x60000000	CM55 code region
		0x00040000	USED: (256 KB) FREE: (0 Bytes)	0x48040000	0x58040000	0x60004000	

Data Tightly Coupled Memory (DTCM)256 KB

RRAM512 KB

Region Id	Domain	Offset	Size	RRAM_C	RRAM	RRAM_C_S	RRAM_S	Description
extended_boot_reserved	reserved	0x00000000	0x00011000 (68 KB)	0x02000000	0x22000000	0x12000000	0x32000000	CM33 secure user programmable region
user_programmable	M33S	0x00011000	0x0004A000 (296 KB)	0x02011000	0x22011000	0x12011000	0x32011000	
user_nvm	M33_M55	0x0005B000	0x00008000 (32 KB)	0x0205B000	0x2205B000	0x1205B000	0x3205B000	CM33 non-secure and CM55 user programmable region
UNALLOCATED	<NONE>	0x00063000	0x00007000 (28 KB)	0x02063000	0x22063000	0x12063000	0x32063000	
reserved_region	reserved	0x0006A000	0x00016000 (88 KB)	0x0206A000	0x2206A000	0x1206A000	0x3206A000	
		0x00080000	USED: (484 KB) FREE: (28 KB)	0x02080000	0x22080000	0x12080000	0x32080000	

SRAM1 MB

System SRAM (SoC MEM)5 MB

Serial Memory Interface block 0, memory 1 (S2FS128S)16 MB

Legend: M33 M33_M55 M33NSC M33S M55 reserved

7.8.1.2 Add/edit memory allocations

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

Edit Memory Region - Device Configurator

Region id: user_nvm

Description: CM33 non-secure and CM55 user programmable region

Domain: M33_M55

Offset: 0x0005B000

Size (bytes): 0x00008000 (32 KB)

Addresses:

	RRAM_C	RRAM	RRAM_C_S	RRAM_S
Start	0x0205B000	0x2205B000	0x1205B000	0x3205B000
End	0x02062FFF	0x22062FFF	0x12062FFF	0x32062FFF

OK

Cancel

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

Add New Memory Region - Device Configurator

Region id: RRAM_REGION

Description:

Domain: M33_M55

Offset: 0x00063000

Size (bytes): 0x00007000 (28 KB)

Addresses:

	RRAM_C	RRAM	RRAM_C_S	RRAM_S
Start	0x02063000	0x22063000	0x12063000	0x32063000
End	0x02069FFF	0x22069FFF	0x12069FFF	0x32069FFF

OK

Cancel

Once the dialog is open, enter values:

- As needed, type the name of the **Region id**.
- Select a **Domain** from the pull-down menu.
- Enter the **Offset** and **Size**.

7 Resources tabs

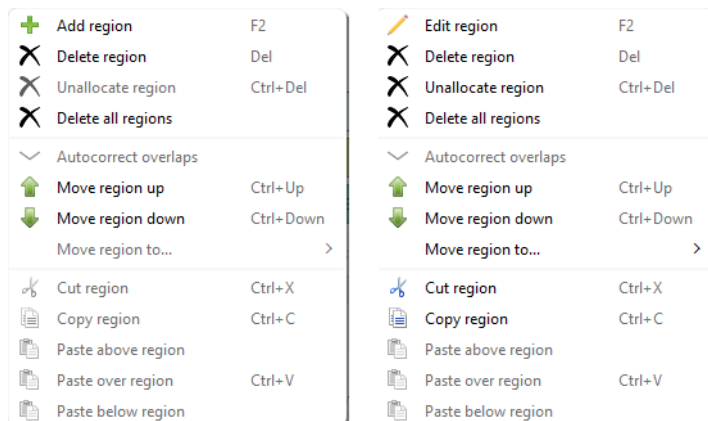
The **Size** entry allows for hex or decimal entry. It also accepts specifying units, which will always be converted to bytes. Units are case insensitive and can be entered as follows:

- 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.

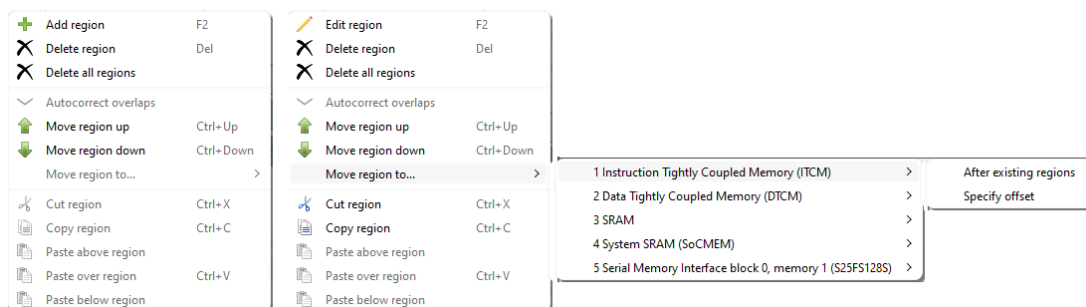
Unallocated regions will start with an **Add region** option while allocated regions will have the **Edit region** option. Both options launch the same edit dialog.



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

7.8.1.3 Rearrange memory regions

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



When moving a region, there are two options: **After existing regions** and **Specify offset**.

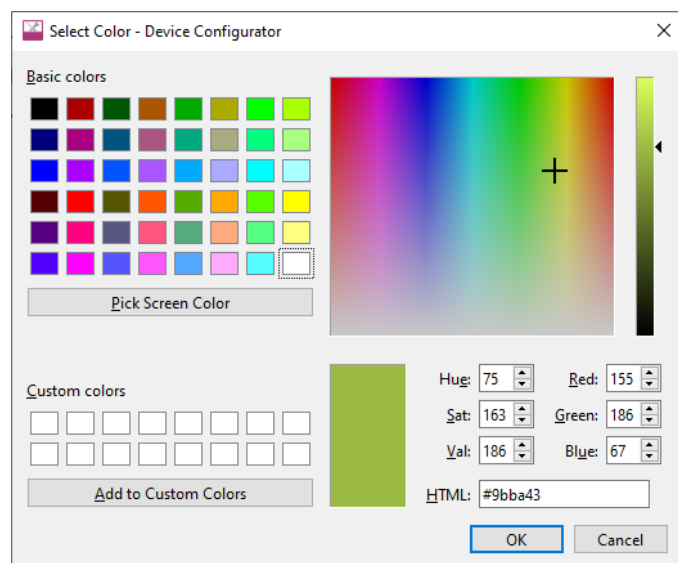
When you select **Specify offset**, this dialog displays to enter the offset.

7 Resources tabs



7.8.1.4 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.



7.8.1.5 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 Resources tabs

7.8.1.6 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:

Memory region	Used Size	Region Size	%age Used
m55_data_INTERNAL:	3796 B	256 KB	1.45%
m55_code_INTERNAL:	12024 B	256 KB	4.59%
user_nvm_C:	0 GB	32 KB	0.00%
m55_nvm:	24900 B	2816 KB	0.86%
m55_trailer:	0 GB	256 KB	0.00%
m55_code_secondary:	0 GB	256 KB	0.00%
m55_data_secondary:	2800 KB	2800 KB	100.00%
m33_m55_shared:	256 KB	256 KB	100.00%
gfx_mem:	0 GB	1808 KB	0.00%
m33_data:	0 GB	256 KB	0.00%
m33s_allocatable_shared:	0 GB	4 KB	0.00%
m33_allocatable_shared:	0 GB	4 KB	0.00%
m55_allocatable_shared:	1088 B	4 KB	26.56%

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

7.8.1.7 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* }
```

7.8.1.8 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);
```

7.8.1.9 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);
```

7.8.1.10 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 }
```

7.8.1.11 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(+R0)
    }
    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 }
```

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

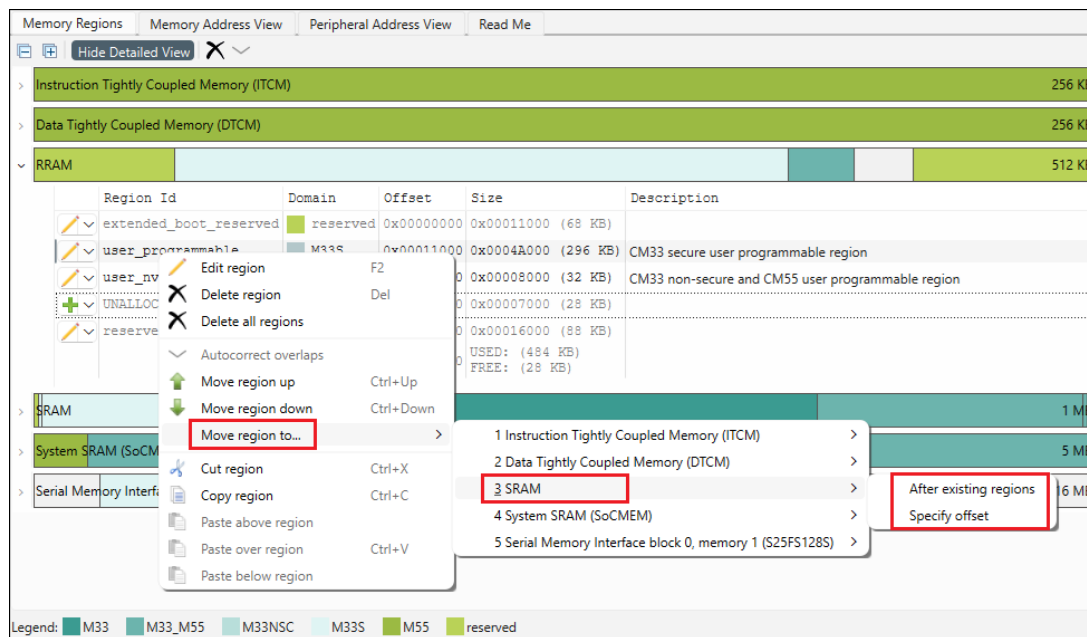
This is highly dependent on the needs of the customer's application. Solutions may include:

- 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](#)

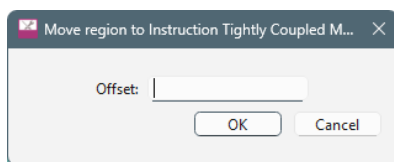
7.8.1.13 How to switch between internal and external memory

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

7 Resources tabs



- If you want to let the tool just place it at the end of the other regions in the destination, select **After existing regions**.
- If you want to specify exactly what offset to put it at, then select **Specify Offset**. Then enter the value in the dialog.



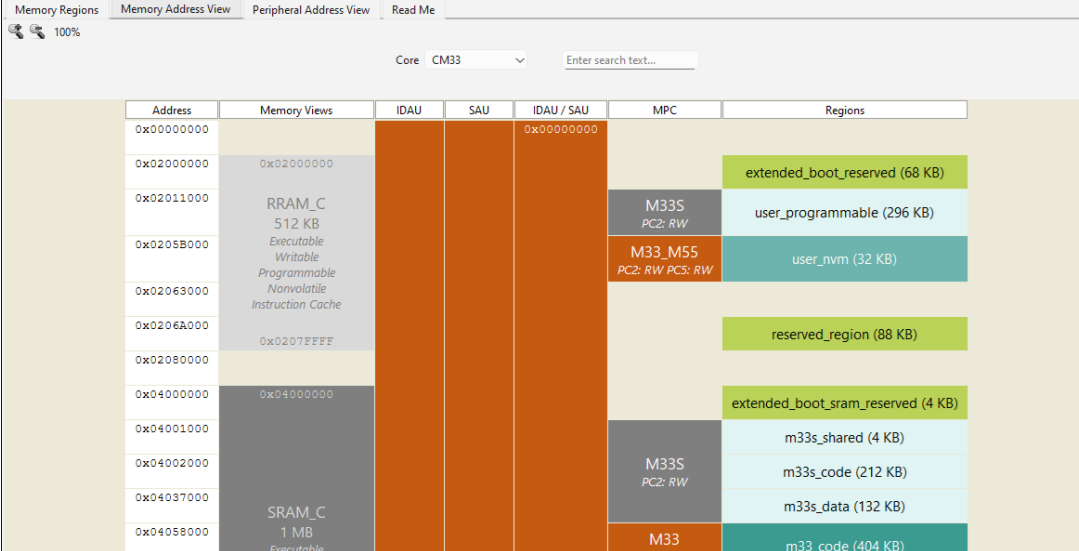
Note: The regions required for initial application start up must be in a non-volatile memory.

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 Resources tabs

7.8.2 Memory Address View subtab

The **Memory Address View** subtab displays the security status and access permissions enforced by the IDAU, SAU, and MPC at a given address or memory region. The **Core** pull-down allows you to switch to other cores, and the **Search** shows all occurrences of an entered term with **Forward** and **Back** buttons to jump to the next or previous term. This subtab also contains zoom commands.



Address	Memory Views	IDAU	SAU	IDAU / SAU	MPC	Regions
0x00000000				0x00000000		
0x02000000	0x02000000					extended_boot_reserved (68 KB)
0x02011000	RRAM_C 512 KB Executable Writable				M33S PC2: RW	user_programmable (296 KB)
0x0205B000	0x0205B000				M33_M55 PC2: RW PC5: RW	user_nvm (32 KB)
0x02063000	Programmable Nonvolatile Instruction Cache					
0x0206A000	0x0207FFFF					reserved_region (88 KB)
0x02080000						
0x04000000	0x04000000					extended_boot_sram_reserved (4 KB)
0x04001000						
0x04002000					M33S PC2: RW	m33s_shared (4 KB)
0x04037000						m33s_code (212 KB)
0x04058000	SRAM_C 1 MB Executable				M33	m33s_data (132 KB)
						m33_code (404 KB)

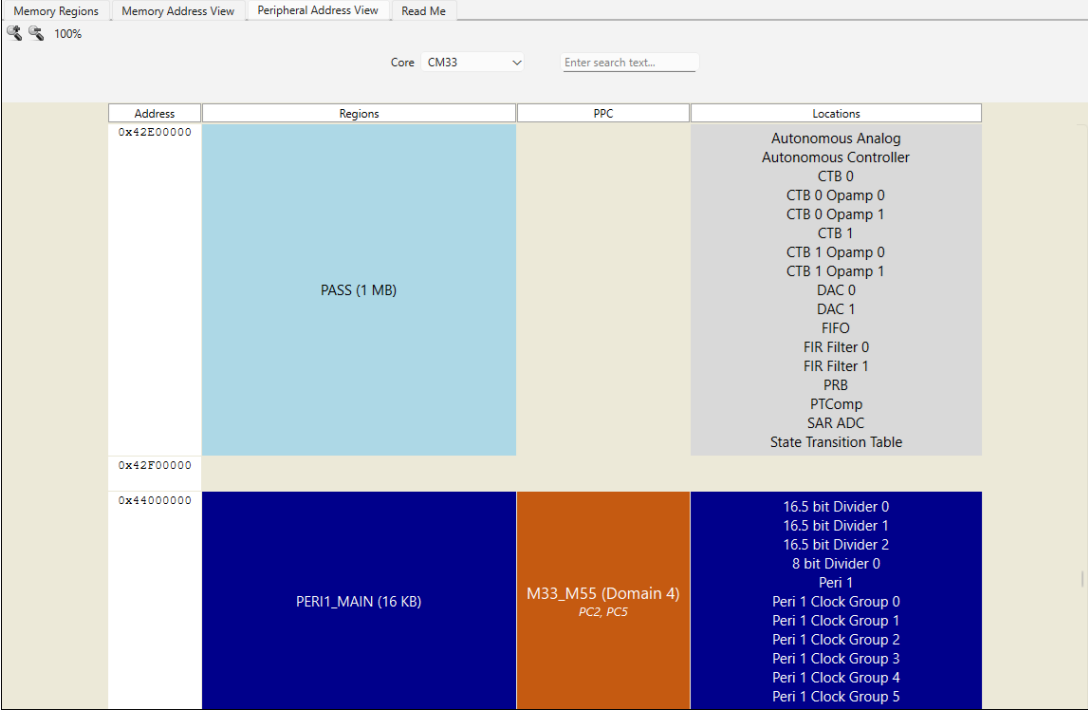
- The **Address** column shows the start address of every segment shown in the other columns.
- The **Memory Views** column depicts all the physical memories and external memory reservations for the device, along with the size and capabilities for each. Possible capabilities include Executable, Writable, Readable, Programmable, Nonvolatile, Secure, and Instruction Cache.
- The **IDAU** column shows the security across all address ranges determined by the device's IDAU. Each range lists a security status code and corresponding background color (S/green for Secure, NS/orange for Non-Secure, NSC/yellow for Non-Secure Callable, and X/gray for Exempted) and the address range size.
- The **SAU** column depicts the security across all address ranges as configured in the designs SAU entries. Each range lists a security status code and corresponding background color (S/green for Secure, NS/orange for Non-Secure, NSC/yellow for Non-Secure Callable) and the address range size.
- The **IDAU / SAU** column depicts the effective security across all address ranges combining the (fixed) device's IDAU and configured SAU entries. Each range lists a start address, end address, security status code and corresponding background color (S/green for Secure, NS/orange for Non-Secure, NSC/yellow for Non-Secure Callable, and X/gray for Exempted) and the address range size.
- The **MPC** column shows the address ranges that will be under MPC control according to the configured Protection Domains and access details for each. Each range lists the name of the governing Protection Domain and a background color corresponding the range's security (S/green for Secure in the context of a secure core or S/gray in the context of a non-secure core, NS/orange for Non-Secure, NSC/yellow for Non-Secure Callable) and the access permissions (R for Read, W for Write) for each Protection Context granted by the Protection Domain. E.g., the access permission string "PC2: RW PC5: RW" denotes read and write access is granted to Protection Contexts PC2 and PC5.
- The **Regions** column shows the memory regions that are configured in the [Memory Regions subtab](#). For each region, the region id and size is shown, and has the same user-configured color coding from the Memory Regions tab.

In each of the columns, if the described content is not applicable – either because there is no such configuration or the device doesn't support it – the column will be omitted.

7 Resources tabs

7.8.3 Peripheral Address View subtab

The **Peripheral Address View** subtab displays all peripheral regions and locations and (where used) the access permissions enforced by the PPC. The **Core** pull-down allows you to switch to other cores, and the **Search** shows all occurrences of an entered term with **Forward** and **Back** buttons to jump to the next or previous term. This subtab also contains zoom commands.

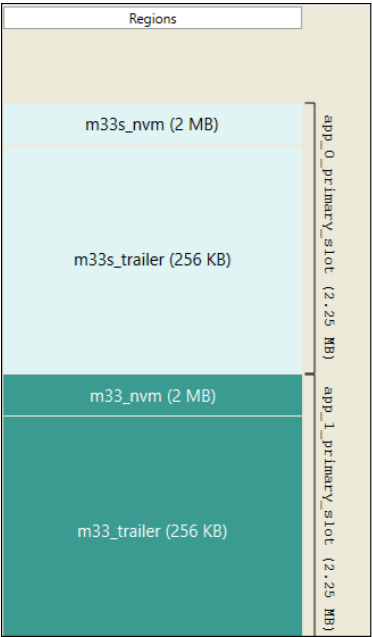


Address	Regions	PPC	Locations
0x42E00000	PASS (1 MB)		Autonomous Analog Autonomous Controller CTB 0 CTB 0 Opamp 0 CTB 0 Opamp 1 CTB 1 CTB 1 Opamp 0 CTB 1 Opamp 1 DAC 0 DAC 1 FIFO FIR Filter 0 FIR Filter 1 PRB PTComp SAR ADC State Transition Table
0x42F00000			
0x4000000	PERI1_MAIN (16 KB)	M33_M55 (Domain 4) PC2, PC5	16.5 bit Divider 0 16.5 bit Divider 1 16.5 bit Divider 2 8 bit Divider 0 Peri 1 Peri 1 Clock Group 0 Peri 1 Clock Group 1 Peri 1 Clock Group 2 Peri 1 Clock Group 3 Peri 1 Clock Group 4 Peri 1 Clock Group 5

- The **Address** column shows the start address of every segment shown in the proceeding columns.
- The **Regions** column depicts all of the peripheral regions defined for the device. There is no meaning in the background colors of these regions; the colors simply alternate between two shades of blue for visual contrast.
- If a PPC is configured, the **PPC** column shows the address ranges that will be under PPC control according to the configured Protection Domains and access details for each. Each range lists the name of the governing Protection Domain and a background color corresponding the range's security (S/green for Secure in the context of a secure core or S/gray in the context of a non-secure core, NS/orange for Non-Secure, NSC/yellow for Non-Secure Callable) and the Protection Contexts having access to the peripherals in the region per the Protection Domain.
- The **Locations** column show all of the peripheral locations in the corresponding region. If there is no configured access to the locations in this region, they will be depicted with a gray background; if at least one Protection Context is granted access to the region and its locations, it will have the same blue or light blue coloring in the corresponding region.

7 Resources tabs

Note: For applications using the MCUboot library, memory ranges used to define the primary slot will be shown on the right side of the view.



In the [Memory Regions subtab](#), there is a corresponding table with links to the [Solution](#) parameters where the range is defined.

7 Resources tabs

7.8.4 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
Memory Address View
Peripheral 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.

- [How to read the build output and associate it with the regions defined in the tool](#)
- [How to define a new region for some data buffer variable](#)
- [How to define a new region for the code and then assign code to the particular region using pragma directives](#)
- [How to assign a function or variable into a custom memory section](#)
- [How to modify the linker to include a specific file into a custom memory section](#)
- [How to modify the linker to include an entire middleware library into a custom memory section](#)
- [How to re-balance the memory between the cores based on the build output](#)
- [How to switch between internal and external memory](#)

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.

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

7 Resources tabs

7.9 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

7.10 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

7.11 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.

Peripherals Pins Analog-Routing System Peripheral-Clocks DMA			
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

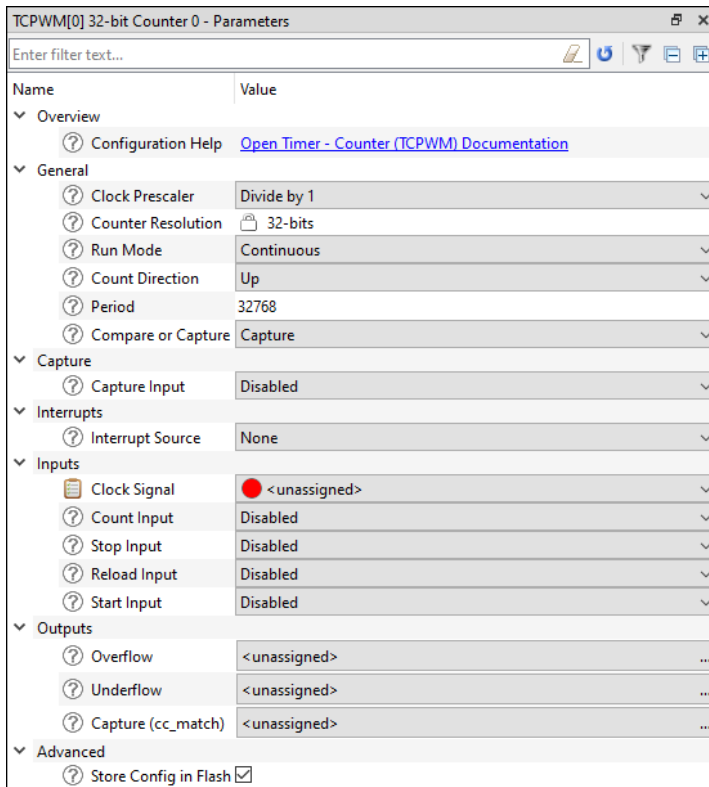
The Device Configurator tool contains the following primary panes that display information based on what is selected in a particular [Resources](#) tabs:

- [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
Overview	
Configuration Help	Open Timer - Counter (TCPWM) Documentation
General	
Clock Prescaler	Divide by 1
Counter Resolution	32-bits
Run Mode	Continuous
Count Direction	Up
Period	32768
Compare or Capture	Capture
Capture	
Capture Input	Disabled
Interrupts	
Interrupt Source	None
Inputs	
Clock Signal	<unassigned>
Count Input	Disabled
Stop Input	Disabled
Reload Input	Disabled
Start Input	Disabled
Outputs	
Overflow	<unassigned>
Underflow	<unassigned>
Capture (cc_match)	<unassigned>
Advanced	
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.

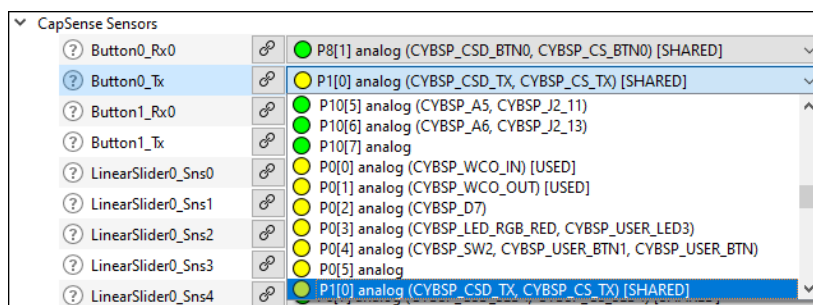
Note: After selecting these parameter types, use the Go To button to jump to the selected resource.

- 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:

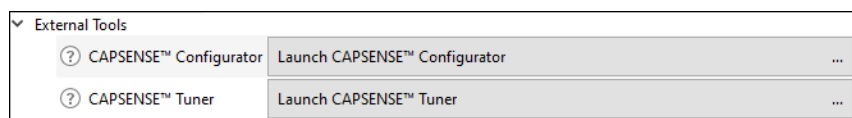
- Green – preferred
- Yellow – valid
- Red – constrained

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

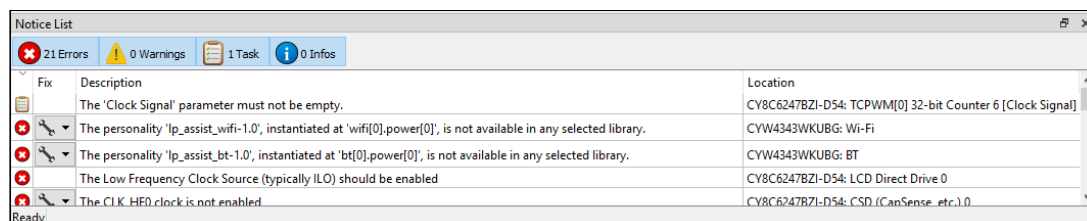


Click on **[Launch < Configurator > . . .]** to launch the configurator.

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:

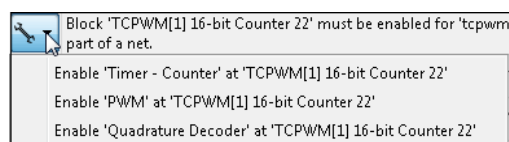
- **Errors** – These indicate there is at least one problem that must be addressed before you can build your application. Typical errors could include compiler build errors and connectivity errors.
- **Warnings** – These report unusual conditions that might indicate a problem, although you can usually 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

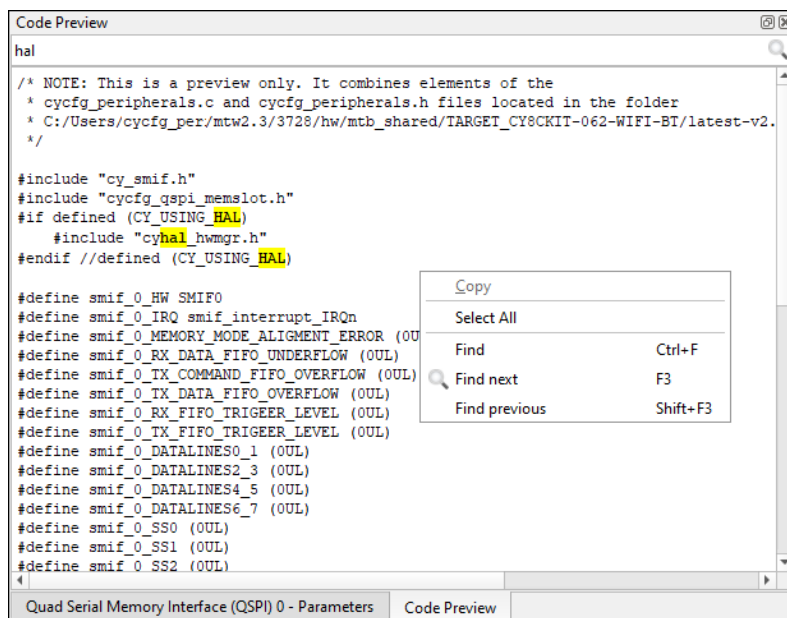
Note: The fixes listed are not necessarily the only way to fix the issue. They are merely common options. Also, if you save the *.modus file with outstanding tasks, they become errors saved in the GeneratedSource/cycfg_notices.h file.

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	<p>Update to the Settings menu; back-end changes.</p>	
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 Memory tab and renamed Summary to Address View.</p> <p>Added Delete All regions command to the Memory tab.</p>	
5.40	<p>Minor GUI updates for tooltips and improved display; minor back-end changes.</p>	
5.50	<p>Updates to the Memory tab; minor back-end changes.</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 Solutions and Memory tabs configuration for early access pack.
*N	2024-12-06	Updated to version 5.20.
*O	2025-03-26	Updated to version 5.30.
*P	2025-06-20	Updated to version 5.40.
*Q	2025-09-05	Updated to version 5.50.

Trademarks

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

Edition 2025-09-05

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

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.