

ModusToolbox™ Library Manager user guide

ModusToolbox[™] tools package version 3.2.0

Library Manager version 2.20

A newer version of this document may be available on the web here.

About this document

Scope and purpose

This document provides information and instructions for how to use the ModusToolbox[™] Library Manager.

Intended audience

Read this document to learn how to manage ModusToolbox[™] BSPs and libraries for your application.

Document conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, menus and sub-menus
Italics	Denotes file names and paths.
Courier New	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

Reference documents

Refer to the following documents for more information as needed:

• <u>ModusToolbox™ tools package user guide</u>



Table of contents

Table of contents

1	Overview	3
1.1	ModusToolbox™ 3.x vs. 2.x BSPs	3
1.2	Specifying dependencies	3
1.3	Including libraries and dependencies	5
2	Launch the Library Manager	9
2.1	make command	9
2.2	VS Code and Eclipse	9
2.3	Stand-alone GUI mode	9
2.4	Local Content mode	
2.5	Non-GUI command line interface (CLI)	10
3	Working with BSPs	11
3.1	Select Active BSP	11
3.2	Add BSPs	12
3.3	Rename BSP	14
3.4	Remove BSP	15
4	Working with libraries	16
4 4.1	Working with libraries Add library (single-core application)	
-	-	17
4.1	Add library (single-core application)	17 17
4.1 4.2	Add library (single-core application) Add library (multi-core application)	17 17 18
4.1 4.2 4.3	Add library (single-core application) Add library (multi-core application) Update indirect dependency libraries	17 17 18 19
4.1 4.2 4.3 4.4	Add library (single-core application) Add library (multi-core application) Update indirect dependency libraries Change library version	17 17 18 19 19
4.1 4.2 4.3 4.4 4.5	Add library (single-core application) Add library (multi-core application) Update indirect dependency libraries Change library version Share/unshare libraries	17 17 18 19 19 20
4.1 4.2 4.3 4.4 4.5 5	Add library (single-core application) Add library (multi-core application) Update indirect dependency libraries Change library version Share/unshare libraries.	17 17 18 19 19 20
4.1 4.2 4.3 4.4 4.5 5 5.1	Add library (single-core application) Add library (multi-core application) Update indirect dependency libraries Change library version Share/unshare libraries GUI description Menus.	17 17 18 19 19 19 20 20 20
4.1 4.2 4.3 4.4 4.5 5 5.1 5.2	Add library (single-core application) Add library (multi-core application) Update indirect dependency libraries Change library version Share/unshare libraries GUI description Menus Application directory.	17 17 18 19 19 20 20 20 21
4.1 4.2 4.3 4.4 4.5 5 5.1 5.2 5.3	Add library (single-core application) Add library (multi-core application) Update indirect dependency libraries. Change library version Share/unshare libraries. GUI description Menus Application directory BSP/library controls	17 17 18 19 19 20 20 21 21
4.1 4.2 4.3 4.4 4.5 5 5.1 5.2 5.3 5.4	Add library (single-core application) Add library (multi-core application) Update indirect dependency libraries Change library version Share/unshare libraries GUI description . Menus Application directory BSP/library controls BSP context menus	17 17 19 19 20 20 20 21 21
4.1 4.2 4.3 4.4 4.5 5 5.1 5.2 5.3 5.4 5.5	Add library (single-core application) Add library (multi-core application) Update indirect dependency libraries Change library version Share/unshare libraries GUI description Menus Application directory BSP/library controls BSP context menus Tabs	17 18 19 19 20 20 21 21 21 23



1 Overview

The Library Manager provides a GUI to select which Board Support Package (BSP) should be used when building a ModusToolbox™ application. The tool collects a list of available and currently selected BSPs and libraries, as well as all the necessary metadata from a webservice. The tool allows you to add and remove BSPs and libraries, as well as change their versions.

1.1 ModusToolbox[™] 3.x vs. 2.x BSPs

One of the major changes made for ModusToolbox[™] version 3.x is that most BSPs are no longer Git repos when you create a version 3.x application. Instead, the BSP becomes owned by the application. This means you can change various aspects of the BSP as you see fit without having to create a custom BSP, because creating a version 3.x application effectively creates a custom BSP. As such, you will typically check the BSP into source control with the application.

ModusToolbox[™] version 2.x BSPs are still Git repos, by default, and may be used by more than one application in a given workspace. In general, this means any change you make to a BSP creates a "dirty" repo, and those changes affect all the related applications. In this case, you will typically not check the version 2.x BSP into source control, since you can regenerate the BSP. For this reason, it is best to create a "custom BSP" for your needs.

Note: BTSDK applications still default to shared BSPs, due to the nature of how BTSDK applications are structured and typical use cases.

Refer to the <u>ModusToolbox[™] tools package user guide</u> for more details about ModusToolbox[™] BSPs and applications.

This version of the Library Manager supports both ModusToolbox version 3.x and 2.x BSPs. The main difference is that the **Version** and **Type** fields are not applicable and cannot be changed for a 3.x BSP. See the <u>Properties</u> <u>tab</u> section later in this document for descriptions of these fields.

1.2 Specifying dependencies

This document only applies to applications that use .mtb files to specify their dependencies, not .lib files. To find information for applications that that use .lib files, refer to an <u>older version of this document</u>.

Applications can share libraries. If needed, different applications can use different versions of the same library. Sharing resources reduces the number of files on your computer and speeds up subsequent application creation time. Shared libraries and versions are located in a "*mtb_shared*" directory adjacent to your application directories. You can easily switch a shared library to become local to a specific application, or back to being shared.

The Library Manager runs the make getlibs command to process the *.mtb* files, pull the libraries from the specified git repos, and store them in the specified location. The system also creates a file called *mtb.mk* in the application's *libs* subdirectory. The build system uses that file to find all the libraries required by the application.



The following shows the Library Manager for a typical PSoC[™] 6 BSP. The key items are the lock symbols for libraries that are included because they are dependendies of other libraries. See <u>Working with libraries</u> for more details about using the tool.

nter filter text		+ Properties R	EADME.md RELEASE.md	
Name Name	Update Available Rem	Property Name Version Type Location Source *.mtb file location	Value APP_CV8CKIT-06252-43012 4.2.0.33410 App Owned C:/Users/Public/Examples/mtw3.2/2-12/Hello_World/bsps/ on	~
Add BSP Add Library rting to refresh dependencies for reshed all dependencies for pro rror(s), 0 warning(s) ccessfully acquired the informat	ject 'mtb-example-hal-hello-w			

The following concepts are key to understanding dependencies:

1.2.1 .mtb file

This file provides information about the associated BSP/library. It contains:

- A URL to a git repository somewhere that is accessible by your computer, such as GitHub.
- A git Commit Hash or Tag that tells which version of the library that you want.
- A path to where the library should be stored.

A typical .*mtb* file looks like this:

https://github.com/Infineon/core-make#release-v3.0.0#\$\$ASSET_REPO\$\$core-make/release-v3.0.0

The variable <code>\$\$ASSET_REPO\$\$</code> points to the root of the shared location. If you want a library to be local to the application instead of shared, use <code>\$\$LOCAL\$\$</code> instead of <code>\$\$ASSET_REPO\$\$</code>.

1.2.2 Direct dependency

This is a library that is directly referenced by an application. In some applications, there will be direct dependencies such as Wi-Fi or Bluetooth libraries. You may also have changed the location or version of an indirect library, which makes that library a direct dependency.

You manage the *.mtb* files for direct dependencies, and *.mtb* files should be checked into source control along with your application source code. A *.mtb* file for a direct dependency is typically stored in the application's *deps* subdirectory by default. You can store it anywhere inside the application except in the application's *libs* subdirectory.



1.2.3 Indirect dependency

This is a library that was included indirectly as a dependency of a BSP or another library. The system stores the code in the appropriate directory and resolves any potential conflicts with different library versions.

The system finds indirect dependencies for each library using information that is stored in a manifest file. For each indirect dependency found, the Library Manager places an *.mtb* file in the application's *libs* subdirectory. The tool also creates a locking_commit.log file in the application's *deps* subdirectory to record the locked release version of those libraries.

You do not manage *.mtb* files for indirect dependencies, and they do not need to be checked into source control. The *.mtb* file for an indirect dependency is stored in the application's *libs* subdirectory and must not be moved.

1.2.4 Shared

A shared library includes the code for the associated *.mtb* file, and the library can be shared by multiple applications in the same directory structure or workspace. When creating a new application, BSPs/libraries can be shared and placed in an *mtb_shared* directory adjacent to the application directory(ies), based on the "default_location" in the manifest file. When you add a BSP/library and specify it as shared, it is also included in the *mtb_shared* directory. These BSPs/libraries do not need to be checked into source control. All the code can be re-downloaded at any time from the information in the *.mtb* file.

You can change the name and location of the shared directory using make variables CY_GETLIBS_SHARED_NAME and CY_GETLIBS_SHARED_PATH. For more details about the ModusToolbox[™] build system and make variables, refer to the ModusToolbox[™] tools package user guide.

1.2.5 Local

A local library includes the code for the associated *.mtb* file, and the BSP/library is used only for a specific application. The local BSP/library is stored in the *libs* subdirectory of the specific application. These BSPs/libraries do not need to be checked into source control. All the code can be re-downloaded at any time from the information in the *.mtb* file.

If you change libraries to be local to an application (see <u>Share/unshare libraries</u> later in this guide), then the source code for these are located in the *libs* subdirectory inside the application directory.

1.3 Including libraries and dependencies

There are two basic ways of including a library and its dependencies into your application: using a manifest file or using manual methods. There is also a third method that is only used for custom BSPs. Each method is discussed separately below. The method you should use depends on whether or not the library and its dependencies are specified in a manifest file.

1.3.1 Including libraries that are specified in a manifest

All libraries provided by Infineon are specified in manifest files that are automatically found by the tools. Manifests are XML files that tell the Library Manager how to discover the list of available BSPs, libraries, and library dependencies. The manifest can also specify specific versions of libraries and dependent libraries that are meant to work together. For more information, refer to the Manifest chapter in the <u>ModusToolbox[™] tools</u> <u>package user guide</u>.



You can create your own manifest file(s) for your libraries so that they will show up in the Library Manager and will work just like the Infineon libraries. More information on manifest files and how to create and use your own manifests is covered in the ModusToolbox[™] tools package user guide.

In order to manage libraries that are specified in a manifest file in the application, files with the extension *mtb* are used. Their location and behavior depend on whether the library is a direct dependency or an indirect dependency. Normally the Library Manager is used to create/modify *mtb* files so you won't need to change them or even view them, but it is worthwhile to understand what is in them.

1.3.1.1 Direct dependencies

For direct dependencies, there will be one or more *mtb* files somewhere in your project (typically in the *deps* directory but could be anywhere except the *libs* directory). An *mtb* file is simply a text file with the extension *mtb* that has three fields separated by #:

- A URL to a Git repository somewhere that is accessible by your computer, such as GitHub
- A Git Commit Hash or Tag that tells which version of the library that you want
- A path to where the library should be stored in the shared location (i.e. the directory path underneath *mtb_shared*).

A typical *mtb* file looks like this:

```
https://github.com/cypresssemiconductorco/retarget-io/#latest-v1.X#$$ASSET_REPO$$/retarget-
io/latest-v1.X
```

The variable <code>\$\$ASSET_REPO\$\$</code> points to the root of the shared location - it is specified in the application's *Makefile*. If you want a library to be local to the app instead of shared, you can use <code>\$\$LOCAL\$\$</code> instead of <code>\$\$ASSET_REPO\$\$</code> in the *mtb* file before downloading the libraries. Typically, the version is excluded from the path for local libraries since there can only be one local version used in a given application. Using the above example, a library local to the app would normally be specified like this:

```
https://github.com/cypresssemiconductorco/TARGET_CY8CKIT-062S2-43012/#latest-v1.X#$$LOCAL$$/TARGET_CY8CKIT-062S2-43012
```

Note: The examples above specify dynamic library versions (e.g. latest-v1.X). Normally in your application, you will want fixed library versions (e.g. release-v1.0.0). This behavior can be controlled using the Library Manager.

1.3.1.2 Indirect dependencies

For indirect dependencies, an *mtb* file for each library is automatically created in the *libs* directory. The version of each dependent library is also captured in the file *locking_commit.log* in the *deps* directory.

Once all the *mtb* files are in the application (both direct and indirect), the libraries they point to are pulled in from the specified Git repos and stored in the specified location (i.e. *mtb_shared* for shared libraries and *libs* for local libraries). Finally, a file called *mtb.mk* is created in the application's *libs* directory. That file is what the build system uses to find all the libraries required by the application.

Since the libraries are all pulled in using make getlibs, you don't typically need to check them in to a revision control system - they can be recreated at any time by re-running make getlibs. This includes both shared libraries (in *mtb_shared*) and local libraries (in *libs*) - they all get pulled from GitHub when you run make getlibs.

You also don't need to check in the *mtb* files for indirect references and the *mtb.mk* file which are stored in the *libs* directory. In fact, the default *.gitignore* file in our code examples excludes the entire *libs* directory since you should not need to check in any files from that directory – they can be recreated at any time.



1.3.2 Including libraries are not specified in a manifest

If you a have your own custom libraries, you can create and include your own custom manifest files so that your libraries will show up in the Library Manager just like Infineon libraries. That method also allows you to specify the dependencies in the manifest so that they are automatically added as indirect dependencies when the dependent library is added to the application. Custom manifests are covered in the ModusToolbox[™] tools package user guide.

1.3.2.1 Manually adding a library

If you do not want to create a manifest file for your custom libraries, then you must manually include the library and its dependencies in each application that uses the library. You can include the library several different ways, such as:

- git clone the version of the library that you want (or use some other revision control system) into the application's directory
- unzip an archive file of the library into the application's directory
- recursive copy the library into the application's directory
- modify the SEARCH variable in the application's Makefile to point to the library
- add an *mtb* file to the *deps* directory for the library and run make getlibs (requires that the library is in a Git repo)

For the first four methods, the library can go anywhere in the application's directory tree. Remember that if you put it in the *libs* directory it will not be checked into source control by default, so you will normally want put manually added libraries somewhere else.

If you use the *mtb* file (i.e., 5th) method, there are two things to be aware of: the library will still not show up in the Library Manager; and latest locking will not be performed on that library so you should use a fixed version in the *mtb* file unless you want the dynamic update behavior.

1.3.2.2 Manually adding dependencies

You can easily add dependencies using the Library Manager tool (for dependencies that are in a manifest file such as Infineon libraries). For dependencies that are not in a manifest file, you can use the same manual methods that are used for including the library itself.

1.3.3 Specifying dependencies for custom BSPs

Custom BSPs use an alternate method to specify dependencies. The advantage is that the specification of the dependencies is self-contained within the custom BSP itself.

This method is accomplished by having files with the extension *mtbx* for each dependency in the *deps* directory inside the custom BSP. An *mtbx* file has the exact same content at an *mtb* file with a different file extension.

The process of getting dependencies from *mtbx* files only searches in the target BSP for the application, so if your application contains multiple custom BSPs, only the dependencies for the active target (as specified in the TARGET variable in the *Makefile* or on the command line) will be included in the application.

The *mtbx* files are automatically created for you, so you typically don't need to do anything with them unless you want to add or remove dependencies from your custom BSP. You can use the BSP Assistant to do this; for details, refer to the BSP Assistant user guide.



During make getlibs, *mtbx* files are treated just like indirect dependencies that are specified in a manifest file. That is, the process will copy each *mtbx* file to the application's *libs* directory and will lock the version in the *locking_commit.log* file in the *deps* directory. The libraries are then pulled down according to the information contained in the *mtb* files.



Launch the Library Manager

2 Launch the Library Manager

There are numerous ways to launch the Library Manager, and that depends on how you use the various tools included with ModusToolbox™ software.

2.1 make command

As described in the <u>ModusToolbox[™] tools package user guide</u> build system chapter, you can run numerous make commands in the application directory, such as launching the Library Manager. After you have created a ModusToolbox[™] application, navigate to the application directory and type the following command in the appropriate bash terminal window:

make library-manager

This command opens the Library Manager GUI for the specific application in which you are working.

2.2 VS Code and Eclipse

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

- <u>VS Code for ModusToolbox[™] user guide</u>
- Eclipse IDE for ModusToolbox[™] user guide

2.3 Stand-alone GUI mode

You can launch the Library Manager in stand-alone mode by running its executable as applicable for your operating system (for example, double-click or select it using the Windows **Start** menu). By default, it is installed here:

<install_dir>/ModusToolbox/tools_<version>/library-manager

When run in stand-alone mode, the Library Manager opens with the target directory set as <user-home> or as the directory selected from a previous stand-alone session.

Note: When launching from the executable, you can add the --verbose [0-3] option to specify more or less messaging in the console. The default level is 1.



Launch the Library Manager

If you haven't opened the Library Manager tool previously, it starts at your home directory and looks for an application Makefile, and then it opens the first application it finds. If the tool does not find an application, it doesn't display any BSPs or libraries.

<u>S</u> ettings <u>H</u> elp					
Application Directory: C:/Users/ftcj/					Browse
Enter filter text		₽₽	Properties	README.md Description	
Name	Update Available	Remove	Property	Value	
			Version		~
			Туре		~
			Location		
			Source		
			*.mtb file loca	tion	
Add BSP Add Library					
ERROR:the directory 'C:/Users/ftcj' does not contain the file 'Makefile'					^
Press "Retry" to try again.					
					×
					Retry Close

Click the **Browse...** button next to the **Application Directory** field and navigate to the appropriate directory that contains the desired application.

Note: The next time you open the Library Manager in stand-alone mode, it will open with the most recently selected application.

2.4 Local Content mode

If the Library Manager cannot connect to the internet when launching, it displays messages/errors in the console that it cannot access the online manifest file. If you intend to work without Internet, enable **Local Content** mode from the **Settings** menu. Refer to the <u>Local Content Storage CLI user guide</u> for details about creating local content.

If you did not intend to work without Internet, check your Internet settings, adjust your proxy settings (from the **Settings** menu), and then click the **Retry** button to re-read library information.

2.5 Non-GUI command line interface (CLI)

You can run a non-graphical interface from the command line. However, there are only a few reasons to do this in practice. The primary use case would be part of an overall build script for the entire application. For information about command line options, run the <code>library-manager-cli</code> executable using the <code>-h</code> option.

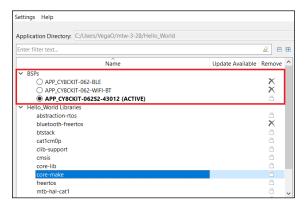


3 Working with BSPs

This section covers the common tasks involved when working with BSPs for your application, including:

- Select Active BSP
- <u>Rename BSP</u>
- Add BSPs
- <u>Remove BSP</u>

The Library Manager displays the BSP(s) included with the application at the top of the list above the libraries, by default.



3.1 Select Active BSP

The active BSP for the application displays in bold text with "(ACTIVE)" next to it. An application can only have one active BSP. If your application has more than one BSP included, select one of the other BSPs to be active by clicking the radio option next to it. You can also choose to right-click on it and select **Set as Active BSP**. Then, click the **Update** button.

`	✓ BSPs	-
	APP_CY8CKIT-062-BLE	×
	APP_CY8CKIT-062-WIFI-BT	×
	APP_CY8CKIT-062S2-43012 (ACTIVE)	
	APP_CY8CPROTO-062-4343W	×
`	✓ Hello_World Libraries	* *

When you click the **Update** button, the system displays progress in the message console and updates your application's Makefile "TARGET=" field to specify the new BSP as active.



3.2 Add BSPs

When you add a BSP to an application, you have several options, including adding a BSP from a template, browsing to one on disk, and creating a new BSP from a device part number. To begin the process, click the **Add BSP** button located below the list of BSPs and libraries.

	Name	Update Available	Remove
~	BSPs		
	○ APP_CY8CKIT-062-BLE		X
	APP_CY8CKIT-062-WIFI-BT (ACTIVE)		
	APP_CY8CKIT-062S2-43012		X
	APP_CY8CPROTO-062-4343W		- X
~	Hello_World Libraries		
	cat1cm0p		8
	cmsis		8
	core-lib		X
	core-make		8
	mtb-hal-cat1		8
	mtb-pdl-cat1		8
	recipe-make-cat1a		- X
	retarget-io		- X
	udb-sdio-whd		8
	Add BSP Add Library		

This opens the Add BSP dialog. There are different steps to add, select, or create a BSP.

3.2.1 Add BSP from template

On the Add BSP dialog, select the desired BSP from the list, change the **New BSP name**, if desired, and click **OK**.

 > AROC[™] Bluetooth BSPs > AROC[™] Bluetooth BSPs > AROC[™] Connectivity BSPs > PMG BSS > PSoC[™] 6 BSPs > PSoC[™] 6 BSPs < < < < < < <<<<<<<<	Enter filter text		<i>I</i>	Create from MPN	Browse for BSP	₽₽	CY8CKIT-062S2-43012
CV8CEVAL-06252-LAI-43439M2 CV8C624A82I-S2D44 Sterling-LWB+ (CVW43439KUBG) 42.0 release Support of up to 2MB Flash and 1MB SRAM CV8CEVAL-06252-LAI-4373M2 CV8C624A82I-S2D44 Sterling-LWB5+ (CVW4373FUBGT) 42.0 release Support of up to 2MB Flash and 1MB SRAM CV8CEVAL-06252-MUR-3439M2 CV8C624A82I-S2D44 LBEE5KL10N (CVW4337SUBGT) 41.0 release Support of up to 2MB Flash and 1MB SRAM CV8CEVAL-06252-MUR-4373M2 CV8C624A82I-S2D44 LBEE5FXL2B (CVW4373UBGT) 41.0 release Support of up to 2MB Flash and 1MB SRAM CV8CEVAL-06252-MUR-4373M2 CV8C624A82I-S2D44 LBEE5FXL2B (CVW4373UBGT) 41.0 release Support of up to 2MB Flash and 1MB SRAM CV8CKIT-062-8LE CV8C64482I-S2D44 LBEE5FXL1DX (CVW4337UBGT) 41.0 release Support of up to 2MB Flash and 1MB SRAM CV8CKIT-062-WIR-48T CV8C64482I-S2D44 LBEE5FXL1DX (CVW4373UBGT) 41.0 release Support of up to 2MB Flash and 1MB SRAM CV8CKIT-0648052-4343W CV8C6247B2I-D53 <none> 42.0 release Support of up to 2MB Flash and 1MB SRAM CV8CKIT-0648052-4343W CV8C6247B2I-D54 LBEE5KL1DX (CVW430172C0WKWBG) 42.0 release Support of up to 2MB Flash and 1MB SRAM CV8CKIT-0648052-4343W CV80644782I-S2D44 LBEE5KL1DX (CVW430172C0WKWBG)<!--</th--><th> > AIROC[™] Bluetooth® BSPs > AIROC[™] Connectivity BSPs > PMG BSPs > PSoC[™] 4 BSPs ~ PSoC[™] 6 BSPs </th><th></th><th>-</th><th></th><th></th><th>^</th><th>Murata 1LV Module (CYW43012 Wi-Fi + Bluetooth Combo Chip), industry-leadir CAPSENSE[™] for touch buttons and slider, on-board debugger/programmer with KitProg3, microSD card interface, 512-Mb Quad-SPI NOR flash, PDM-PCM</th></none>	 > AIROC[™] Bluetooth® BSPs > AIROC[™] Connectivity BSPs > PMG BSPs > PSoC[™] 4 BSPs ~ PSoC[™] 6 BSPs 		-			^	Murata 1LV Module (CYW43012 Wi-Fi + Bluetooth Combo Chip), industry-leadir CAPSENSE [™] for touch buttons and slider, on-board debugger/programmer with KitProg3, microSD card interface, 512-Mb Quad-SPI NOR flash, PDM-PCM
CY8CKIT-062-WIFI-BT CY8C6247B2I-D54 LBEE5KL1DX (CYW4343WKUBG) 4.20 release regrammable digital blocks. CY8CKIT-062-WIFI-BT CY8C6247B2I-D54 LBEE5KL1DX (CYW4343WKUBG) 4.20 release KIt Contents: CY8CKIT-0648052-4343W CY80644A82I-S2D44 LBEE5KL1DX (CYW4343WKUBG) 4.20 release CY8CKIT-0648052-4343W CY80644A82I-S2D44 LBEE5KL1DX (CYW4343WKUBG) 4.20 release CY8CKIT-0648052-4343W CY80644A82I-S2D44 LBEE5KL1DX (CYW4343WKUBG) 4.20 release CY8CKIT-062-WIFI-BT CY80644A82I-S2D44 LBEE5KL1DX (CYW4343WKUBG) 4.20 release CY8CROTO-062-3434W CY80644A82I-S2D44 LBEE5KL1DX (CYW4343WKUBG) 4.20 release CY8CROTO-062-3434W CY8062451_S2D44 LBEE5KL1DX (CYW4343WKUBG) 4.20 release CY8CROTO-062-3434W CY8C62451_CP-S3D72 LBEE5KL1DX (CYW4343WKUBG) 4.20 release VIW jumper wires (4 inches each) Two jumper wires (5 inches each) CY8CPROTO-063-81E CY8LF-416045-02 (CY806447B2I-BLDS) CY8LF-416045-02 (CY806447B2I-BLDS) 4.20 release VIW jumper wires (5 inches each) VIW jumper wires (5 inches each) <td>CY8CEVAL-06252-LAI-43439M2 CY8CEVAL-06252-LAI-4373M2 CY8CEVAL-06252-MUR-43439M1 CY8CEVAL-06252-MUR-4373EM. CY8CEVAL-06252-MUR-4373M2 CY8CEVAL-06252-MUR-4373M2 CY8CKIT-062-BLE</td> <td>CY8C624A8Z1-S2D44 CY8C624A8Z1-S2D44 2 CY8C624A8Z1-S2D44 2 CY8C624A8Z1-S2D44 CY8C624A8Z1-S2D44 CY8C624A8Z1-S2D44 CY8C624A8Z1-S2D44</td> <td>Sterling-LWB+ (CYW Sterling-LWB5+ (CY LBEE5KL1YN (CYW4 LBEE5PK2AE (CYW4 LBEE5PK2BC (CYW4 <none></none></td> <td>W4373EUBGT) 3439KUBG) 373EUBGT) 373IUBGT)</td> <td>4.2.0 release 4.2.0 release 4.2.0 release 4.1.0 release 4.1.0 release 4.2.0 release 4.2.0 release</td> <td>> > > > ></td> <td> Support of up to 2MB Flash and 1MB SRAM Dedicated SDHC to interface with WICED wireless devices. Delivers dual-cores, with a 150-MHz Arm © Cortex®-M4 as the primary application processor and a 100-MHz Arm © Cortex®-M0+ as the secondary processor for low-power operations. Supports Full-Speed USB, capacitive-sensing with CAPSENSE, a PDM-PCM digital microphone interface, a Quad-SPI interface, 13 serial </td>	CY8CEVAL-06252-LAI-43439M2 CY8CEVAL-06252-LAI-4373M2 CY8CEVAL-06252-MUR-43439M1 CY8CEVAL-06252-MUR-4373EM. CY8CEVAL-06252-MUR-4373M2 CY8CEVAL-06252-MUR-4373M2 CY8CKIT-062-BLE	CY8C624A8Z1-S2D44 CY8C624A8Z1-S2D44 2 CY8C624A8Z1-S2D44 2 CY8C624A8Z1-S2D44 CY8C624A8Z1-S2D44 CY8C624A8Z1-S2D44 CY8C624A8Z1-S2D44	Sterling-LWB+ (CYW Sterling-LWB5+ (CY LBEE5KL1YN (CYW4 LBEE5PK2AE (CYW4 LBEE5PK2BC (CYW4 <none></none>	W4373EUBGT) 3439KUBG) 373EUBGT) 373IUBGT)	4.2.0 release 4.2.0 release 4.2.0 release 4.1.0 release 4.1.0 release 4.2.0 release 4.2.0 release	> > > > >	 Support of up to 2MB Flash and 1MB SRAM Dedicated SDHC to interface with WICED wireless devices. Delivers dual-cores, with a 150-MHz Arm © Cortex®-M4 as the primary application processor and a 100-MHz Arm © Cortex®-M0+ as the secondary processor for low-power operations. Supports Full-Speed USB, capacitive-sensing with CAPSENSE, a PDM-PCM digital microphone interface, a Quad-SPI interface, 13 serial
	CY8CKIT-062-V4 CY8CKIT-062-V41F-BT CY8CKIT-0648052-4343W CY8CFR0T0-062-4343W CY8CFR0T0-0622-4343W CY8CFR0T0-06253-4343W CY8CFR0T0-06253-4343W CY8CFR0T0-0635-BLE CY8CFR0T0-0648051-BLE	CY8C6244LQ1-S4D92 CY8C62478Z1-D54 CY8C644A8Z1-S2D44 CY8C644A8Z1-S2D44 CY8C624A8Z1-S2D44 CY8C624A8Z1-S2D44 CY8C6244B21-S2D44 CY8C6245LQ1-S3D72 CY8LE-41607-02 (CY8C63478Z1-BLD53) CYBLE-41607-02 (CY8C64478Z1-BLD53)	<none> LBEE5KL1DX (CYW4 LBEE5KL1DX (CYW4 LBEE5KL1DX (CYW4 LBEE5KL1DX (CYW4 LBEE5KL1DX (CYW4 LBEE5KL1DX (CYW4 CYBLE-416045-02 (C CYBLE-416070-02 (C</none>	1343WKUBG) 1343WKUBG) 1343WKUBG) 1343WKUBG) 13439KUBG) 1343WKUBG) 1343WKUBG) 12466347BZI-BLD53)	4.2.0 release 4.2.0 release		programmable digital blocks. Kit Contents: • PSoC [™] 6S2 Wi-Fi BT Pioneer Board • USB Type-A to Micro-B cable • Quick Start Guide • Four jumper wires (4 inches each)

The added BSP will appear on the list of BSPs on the main Library Manager window.



3.2.2 Create BSP from MPN

On the Add BSP dialog, click the **Create from MPN** button. This command opens the BSP Assistant tool to create a BSP based on device part numbers. Refer to the <u>BSP Assistant user guide</u> for more details about how to create a BSP from MPN. When you close the BSP Assistant, the new BSP will show up in the Add BSP dialog under **Import**.

Enter filter text	Create from M	PN Browse for BSP E This is an imported BSP, located at C:/Users/follettcj/
Kit Name	MCU/SOC/SIP	Connectivity AppData/Local/Temp/BSPfromMPN-zGyAwM/TARGET_My- MPN-BSP.
> AIROC [™] Connectivity	BSPs	
✓ Import Mv-MPN-BSP		
> PMG BSPs		
> PSoC [™] 4 BSPs		
✓ PSoC™ 6 BSPs		>
estination		
arent directory:	C:/Users/follettcj/mtw3.1/10177/hw/Hello_W	orld/bsps

Change the **New BSP name**, if desired, and click **OK**. The new BSP will appear on the list of BSPs on the main Library Manager window.

3.2.3 Browse for existing BSP

On the Add BSP dialog, click the **Browse for BSP** button. Then on the Select Folder dialog, navigate to and select the desired BSP and click **Select Folder**.

Enter filter text	Create from N	IPN Browse	for BSP 📄 🗭		
Kit Name	MCU/SOC/SIP Connectivity Versio	on			
 > AIROC[™] Bluetooth [®] BSPs > AIROC[™] Connectivity BSPs > PMG BSPs > PSoC[™] 4 BSPs 	Select Folder $\leftrightarrow \rightarrow \checkmark \bigstar \heartsuit \ll User$	s > Public >	Example > V Č) D Search Example	
 > PSoC™ 6 BSPs > TRAVEO™ BSPs > Wireless Charging BSPs 	Organize 👻 New folder				
> XMC [™] BSPs	💿 My Desktop	^	Name	Date modified	Ту
	My Documents		TARGET_myXMCBSP-3-31	3/31/2023 4:14 PM	Fil
	wy Documents		ARGEL_My-new-XMC-BSP	3/31/2023 3:03 PM	Fi
	💿 My Sync		TARGET_AnotherBSP	3/31/2023 2:51 PM	Fi
	This PC		TARGET_MyBSP-XMC	3/31/2023 2:47 PM	Fi
	3D Objects		TARGET_MyXMCBSP	3/31/2023 2:47 PM	Fi
			TARGET_MyXMC-BSP	3/31/2023 2:42 PM	Fi
	Desktop		TARGET_3-24-23	3/27/2023 7:15 AM	Fi
			TARGET_MyP4SMAX	3/13/2023 9:05 AM	Fi
	Downloads	~	TARGET_My-1.10-BSP	1/18/2023 2:45 PM	Fi
	h Music				
	Folder:	TARGET_myXN	ACB2P-3-31		
	_			Select Folder Car	icel
estination		_			_
arent directory:					
New BSP name: TARGET_					
-					
				OK	Cance



The added BSP will show up in the Add BSP dialog under Import.

Enter filter text		Create f	from MPN Browse for BSP	₽₽	This is an imported BSP, located at C:/Users/Public/Examp
Kit Name > AIROC [™] Bluetooth® BSP > AIROC [™] Connectivity BS [™] Import [™] Import [™] MXMCBSP-3-31 > PMC MSPs > PSoC [™] 4 BSPs > PSoC [™] 4 BSPs > TRAVEO [™] BSPs > Wireless Charging BSPs > XMC [™] BSPs	s	Connectivity V	/ersion		TARGET_myXMCBSP-3-31.
estination arent directory: C;	/Users/follettcj/mt	w3.1/10177/hw/H	Hello_World/bsps		
lew BSP name: TARGET_ m	XMCBSP-3-31				

Change the **New BSP name**, if desired, and click **OK**. The added BSP will appear on the list of BSPs on the main Libbrary Manager window.

3.2.4 Finalize Add BSP process

After adding a BSP you can choose to <u>set it as the active BSP</u>. You can also choose to <u>remove</u> the non-active BSP and/or <u>rename</u> either BSP.

Regardless of whether you added an existing or new BSP, and whether you made other changes, the final step is to click the **Update** button on the main Library Manager window to commit the changes. This copies the BSP to the application and then it becomes owned by the application. This also updates the application's *Makefile*.

3.3 Rename BSP

There are a few ways to rename a BSP, as follows:

3.3.1 Existing BSP

For a BSP already shown in the main Library Manager list, right-click and select **Rename BSP** or double-click on the BSP. Then in the **Properties** tab **Value** field for the Name, type the new name for the BSP.

Properties REA	ADME.md	
Property	Value	
Name	APP_CY8CKIT-062S2-43012	
Version	4.2.0.31135	\sim
Туре	App Owned	\sim
Location	C:/Users/VegaO/mtw-3-28/Hello_World/bsps/	
Source		
*.mtb file location		

You can also double-click the BSP name shown in the **Value** field to rename it.

To commit the change, click **Update**.



3.3.2 Added BSP

You can also rename a BSP when adding one. On the Add BSP dialog, the added BSP appears under **Import**. When you select the added BSP, you can type a new name for it in the **New BSP name** field.

Kit Name MCU/SOC/SIP Connectivity Version > AIROC [™] Bluetooth © BSPs SPs SPs > AIROC [™] Connectivity BSPs Import Import myXMCBSP-3-31 PMG BSPs SPsoC [™] 4 BSPs > PSoC [™] 4 BSPs SPsoC [™] 4 BSPs > PSoC [™] 6 BSPs SPsoC [™] 4 BSPs > TRAVEO [™] BSPs SPsoC [™] 4 BSPs > Wireless Charging BSPs SMC [™] BSPs > XMC [™] BSPs SMC [™] BSPs	Enter filter text	Create from MPN Browse f	BSP E This is an imported BSP, located at C:/Users/Public/Example/ TARGET_myXMCBSP-3-31.	
Parent directory: C:/Users/follettcj/mtw3.1/10177/hw/Hello_World/bsps	Kit Name > AIROC [™] Bluetoot > AIROC [™] Concort impXMCBSP-3 > PMG BSPs > PSoC [™] 4 BSPs > PSoC [™] 4 BSPs > TRAVEO [™] BSPs > Wireless Charging	h® BSPs ivity BSPs -31		
	Destination			
	Parent directory:	C:/Users/follettcj/mtw3.1/10177/hw/He	p_World/bsps	
New BSP name: TARGET_ myXMCBSP-3-31	New BSP name: TARG	ET_ myXMCBSP-3-31		

3.4 Remove BSP

To remove a BSP, click the **Delete** symbol under **Remove** for the appropriate BSP.

	Name	Update Available	Remove
~	BSPs		
	APP_CY8CKIT-062-BLE		- X
	APP_CY8CKIT-062-WIFI-BT (ACTIVE)		<u>a</u>
	APP_CY8CKIT-062S2-43012		X
	APP_CY8CPROTO-062-4343W		X
~	Hello_World Libraries		

Note: The **Lock** symbol for the active BSP means you cannot remove the active BSP from the application.

A dialog displays a message asking you to confirm the removal.

The selected BSP will be deleted from into a ZIP file.	n your application	n and archived
	Cancel	Delete

Click **Delete** to remove the BSP from the list of BSPs for the application. Then, click the **Update** button in the main window to commit the change.



4 Working with libraries

This section covers various ways to update BSPs and libraries, including:

- Add library (single-core application)
- Add library (multi-core application)
- <u>Update indirect dependency libraries</u>
- <u>Change library version</u>
- Share/unshare libraries

The Library Manager displays the libraries included with the application below the BSP(s), by default.

Name	Update Available	Remove
✓ BSPs		
APP_CY8CKIT-062S2-43012 (ACTIVE)		A
 Hello_World Libraries 		
cat1cm0p		<u> </u>
cmsis		-
core-lib		<u> </u>
core-make		-
mtb-hal-cat1		00000
mtb-pdl-cat1		-
recipe-make-cat1a		<u> </u>
retarget-io		- X
Add BSP Add Library		

For a multi-core application, there are two sets of libraries; one for each core project.

When a library has a newer version available than the one currently selected, a "newer version" symbol displays under **Update Available**, indicating that a newer major or minor version is available.

✓ app_cm0p Libraries		
core-lib		<u>a</u>
core-make	*	X
device-db		Ä
mtb-hal-cat1		X
mtb-pdl-cat1		ā
psoc6cm0p		<u>a</u>
recipe-make-cat1a		<u>a</u>

When you select a library, various information displays on the right side of the Library Manager See <u>Tabs</u> for more information.

Properties	READ	OME.md	RELEASE.md			
Property		Value			^	
Name	Name		cat1			
Version		3.9.0 relea	ise	~		
Туре		Shared Gi	t Repo	~		
Location		C:/Users/	Public/Examples/mtw3.2/2-12/mtb_shared/mtb-pdl-cat1/release-v3.9.0		~	
The Periphe into a single Version deta	<u>mtb-pdl-cat1</u> The Peripheral Driver Library (PDL) integrates device header files, startup code, and low-level peripheral drive into a single package. Version details: 3.9.0 release		ivers	;		
MTB CAT1 Pe Overview	Reference Documentation: MTB CATI Peripheral driver library: Main Page Overview System View Description					



4.1 Add library (single-core application)

Click the Add Library button.

V Hello_World Libraries	
cat1cm0p	<u> </u>
cmsis	<u> </u>
core-lib	<u> </u>
core-make	<u> </u>
mtb-hal-cat1	a
mtb-pdl-cat1	<u></u>
recipe-make-cat1a	<u> </u>
retarget-io	- X
Add BSP Add Library	

On the Add Library dialog, click one or more check boxes for libraries to add, and if needed, specify the **Version**. Then click **OK**.

Note:

The **Target Project** option for a single-core application has only one choice and it is selected by default.

Enter filter text		7 🖻 🕀	capsense
Name Bluetooth® Core Middleware aws-iot-device-sdk-embedded-C aws-iot-device-sdk-port azure-c-sdk-port azure-c-sdk-port cure-sdk-forc ble-mesh Capasense csdadc csdidac dfu emceprom emfile emwin	Version 5.0.0 release v4. beta release 2.2.2 release 1.2.1 release 1.1.0 release 2.10 release 2.10 release 2.0.0 release 2.0.0 release 2.0.0 release 1.0.0 release Latest 2.X release Latest 3.X release Latest 4.X release Latest 4.X release Latest 4.X release Latest 4.X release Latest 4.X release	×	Capacitive sensing solution (CAPSENSE [®]) enables a variety of applications such as touch, gesture and proximity detection for home appliances, automotive, IoT, industrial applications, and more. A CAPSENSE [®] upports multiple interfaces (widgets) using both Self-Capacitance (CSD) and Mutual-Capacitance (CSX) sensing methods. Additional Information: • <u>CAPSENSE[®] Middleware Library README.md</u> • <u>CAPSENSE[®] Middleware Library README.md</u> • <u>CAPSENSE[®] Middleware Library API reference guide</u> Version details: 4.0.0 release

Back on the Library Manager, click the **Update** button to commit the changes.

4.2 Add library (multi-core application)

Click the Add Library button.

~	proj_cm0p Libraries	
	cat1cm0p	8
	cmsis	8
	core-lib	8
	core-make	8
	mtb-hal-cat1	8
	mtb-pdl-cat1	0000
	recipe-make-cat1a	8
~	proj_cm4 Libraries	
	cat1cm0p	-
	cmsis	0000
	core-lib	9
	core-make	<u>a</u>
	mtb-hal-cat1	8
	mtb-pdl-cat1	<u>a</u>
	recipe-make-cat1a	8
	Add BSP Add Library	



Select the appropriate core project from the Target Project pull-down menu.

Target Project: F Enter filter text	roj_cm0p roj_cm4	-
Name > Bluetooth® > Core > Middleware > Peripheral > Utilities > Wi-Fi	Version	

Click one or more check boxes for libraries to add, and if needed, specify the **Version**. Then click **OK**.

Enter filter text	V	ĒĒ	capsense
Name > Bluetooth® Core Middleware Middleware Middleware wis-iot-device-sdk-embedded-C aws-iot-device-sdk-port azure-c-sdk-port azure-sdk-for-c ble-mesh cogasense coddac cdidac dfu emceprom emfile emwin emfile emwin	Version 5.0.0 release v4_beta release 2.2.2 release 1.2.1 release 1.1.0 release 2.0.0 release 2.0.0 release 2.0.0 release 3.0.0 release 2.0.0 release Latest 2.X release Latest 2.X release Latest 2.X release Latest 3.X release Latest 3.X release Latest 2.X release Latest 3.X release Lates	×	Capacitive sensing solution (CAPSENSE ^w) enables a variety of applications such as touch, gesture and proximity detection for home appliances, automotive, IoT, industrial applications, and more. A CAPSENSE ^w supports multiple interfaces (widgets) using both Self-Capacitance (CSD) and Mutual-Capacitance (CSX) sensing methods. Additional Information:

4.3 Update indirect dependency libraries

Some libraries included in your application show a lock symbol by default. This symbol indicates that a library is an <u>indirect dependency</u>, and it is required by the BSP and/or another library.

~	Hello_World Libraries	
	capsense	A
	core-lib	
	core-make	ā
	mtb-hal-cat1	
	mtb-pdl-cat1	8
	psoc6cm0p	8
	recipe-make-cat1a	<u>a</u>
	retarget-io	X
	udb-sdio-whd	<u>a</u>

These libraries cannot be removed using the Library Manager; however, you can change whether or not it is shared and select a different version.

When you make a change to an Indirect Dependency library, the lock symbol changes to a **Delete** symbol. Unlike BSPs, this symbol indicates that after you click **Update**, the library is now a <u>direct dependency</u>, and it is your responsibility to manage it either as a local library and/or with the selected version.

 Hello_World Libraries 		
capsense		<u>a</u>
core-lib		
core-make	A	
mtb-hal-cat1		<u>a</u>
mtb-pdl-cat1		<u> </u>
psoc6cm0p		<u>a</u>
recipe-make-cat1a		<u>a</u>
retarget-io		×
udb-sdio-whd		à



To change the library back to an Indirect Dependency, click the **Delete** symbol. The icon reverts back to the lock symbol, and the settings revert back to the defaults. You must click the **Update** button to commit the changes.

4.4 Change library version

Libraries have a specific version, by default. You can change a library version by selecting the pull-down menu next to **Version** and choosing another version. See <u>Version</u> for a detailed description.

Property	Value	_
Name	mtb-pdl-cat1	
Version	3.9.0 release	~
Туре	Latest 3.X release	^
21-	Latest 2.X release	
Location	3.9.0 release	
	3.8.0 release	
mtb-pdl-cat1	3.7.0 release	
	3.6.0 release	
The Peripheral Driv	3.4.0 release	5
into a single packa	3.3.1 release	
	3.3.0 release	
Version details: 3.9.	3.2.0 release	~

Click the **Update** button to commit the changes.

When you change versions for a shared ibrary, the tool creates a new subdirectory for the added version to ensure that the existing version is available for other applications. However, when you change a version for a local library, the system will not create a new local subdirectory. Instead, the version of the library will just be updated with the appropriate tag.

4.5 Share/unshare libraries

When you create applications, libraries can be shared or local depending on the application and manifest. Shared libraries are included in a *mtb_shared* directory, which is adjacent to your application directory. Libraries that are local (aka, not shared) are included in the *libs* subdirectory inside your application's directory.

Properties	README.md RELEASE.md		
Property	Value	^	
Name	mtb-pdl-cat1		
Version	3.9.0 release	\sim	
Туре	Shared Git Repo 🗸 🗸		
Location	Local Git Repo		
200000	Shared Git Repo	×	
mtb-pdl-ca	Shared Git Repo		

If you select **Type** "Local Git Repo" under the **Properties** tab, a new library will be created in the *libs* subdirectory, and it will not be removed from the *mtb_shared* library. This is because the system assumes a shared library is being used by another application. However, if you select "Shared Git repo" for a library already included as local in the application and click **Update**, that library will be removed from the *libs* subdirectory and will instead be populated in the *mtb_shared* directory.

Click the **Update** button to commit the change(s). The message console displays the progress and indicates when changes are complete.



5 GUI description

The Library Manager contains menus, controls, tabs and buttons used to manage BSPs and libraries.

nter filter text	<u>/</u> E E	Properties RE	ADME.md RELEASE.md	
Name PSPs O APP_CY8CKIT-06252 Hello_World Libraries cat1cm0p cmsis core-lib core-make mtb-hal-cat1 mtb-pdl-cat1 recipe-make-cat1a retarget-io	Update Available Remove	Property Name Version Type Location Source *.mtb file locatio	Value APP_CY8CKIT-06252-43012 4.2.0.33410 App Owned C:/Users/Public/Examples/mtw3.2/2-12/Hello_World/bsps/ n	~
	pr project 'mtb-example-hal-hello ject 'mtb-example-hal-hello-worl			

5.1 Menus

The Library Manager has two menus, as follows:

- Settings -
 - **Local Content**: Check box to toggle between LCS mode on and off, and read the local copy of the manifest file installed from the LCS Manager. See <u>Local Content mode</u> for more details.
 - **Proxy Settings**: Opens a dialog to specify direct or manual proxy settings.

Oirect			
O Manual:	http(s)://hostname:port		
	ОК	Cancel	

Note: Save any changes before attempting to update proxy settings. A dialog will warn you that changes could be lost.

- Help
 - View Help: Opens this document.
 - **About**: Displays tool version information, with a links to open Infineon.com.

5.2 Application directory

This is the location of the application's top-level directory, which contains one or more ModusToolbox[™] Makefile projects. Use the **Browse...** button to select a different directory, if needed.



5.3 BSP/library controls

The left side of the Library Manager contains various controls for managing BSPs and libraries:

- Filter text Field used to show only the BSPs/libraries that match the text entered.
- **Collapse All** Click to collapse all item trees.
- **Expand All** Click to expand all item trees.
- **Remove Library** This button allows you to remove libraries from the application.
- Add BSP This button allows you to add and import BSPs to the application.
- Add Library This button allows you to add libraries to the application.

5.4 BSP context menus

If you right-click on a BSP, different commands are available. See "Working with BSPs" for details.



- Set as Active BSP This button allows you to switch between multiple BSPs, choosing which BSP is active and non-active.
- **Rename BSP** This button allows you to specify the name for a BSP created as an in-app BSP.
- **Remove BSP** This button allows you to delete a non-active BSP.

5.5 Tabs

The right side of the Library Manager contains the following tabs for a BSP or library: **Properties**, **README.md**, **and RELEASE.md**. Each tab may contain links to reference documentation that you can follow to find more specific information about your selected BSP or library.

5.5.1 Properties tab

The **Properties** tab contains several fields of information about the selected BSP or library.

5.5.1.1 Name

The **Name** field contains the BSP or library name. For BSPs, this field can be changed. See <u>Rename BSP</u> for more information on how to use the feature.

5.5.1.2 Version

The **Version** field doesn't apply to version 3.x BSPs, which are owned by the application. If this field is populated, it just displays information from the JSON file.

For all libraries and 2.x BSPs, you can select a dynamic "Latest X.Y release" or a fixed "X.Y.Z release" version. These represent tags for versions of libraries and version 2.x BSPs in GitHub repos:

• **Dynamic** – If you select a "Latest X.Y release " version, then it is dynamic. A ModusToolbox[™] v2.x application will download and use the appropriate version specified in the manifest and attempt to resolve any potential conflicts with different versions. Using a "Latest" version means the item will update to the latest, backward-compatible version whenever you click the Library Manager **Update** button.



• **Fixed** – By selecting a specific "release-vX.Y.Z" version, you assign a fixed version of a specific, official release of the item that does not change, unless you manually change it.

When you create a new version 3.x application from an Infineon code example, the application converts libraries to use the "X.Y.Z release" version, by default. This ensures that they will not be updated automatically, unless you change the version to "Latest X.Y release."

When you open the Library Manager for an application and make any kind of change, the change summary displays in the console. If a "latest-vX.Y" tag has a newer version, the console displays a warning about the "latest-vX.Y" tag. It includes a hyperlink to open a dialog that explains how the "latest-vX.Y" tag auto-update works.

If you click **Update**, all items with the "latest-vX.Y" tag will be moved to the newer version, even if you didn't make changes to them.

5.5.1.3 Type

The **Type** field doesn't apply to version 3.x BSPs. All 3.x BSPs are owned by the application.

For all libraries and 2.x BSPs, this field contains a pull-down menu to specify if the selected library is shared or local. See <u>Specifying dependencies</u> for decriptions of shared and local.

5.5.1.4 Location

This shows the current location on disk for the selected BSP or library.

5.5.1.5 Source

This field doesn't apply to version 3.x BSPs. It shows the source location (usually on GitHub) for the library or version 2.x BSP.

5.5.1.6 MTB file location

This field doesn't apply to version 3.x BSPs. It shows the location on disk for library and version 2.x BSP .mtb files.

5.5.1.7 Description

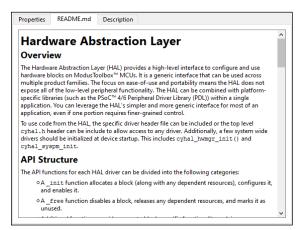
For most libraries, there is a brief description included below the table.

Properties	README.md	RELEASE.md		
Property	Value			^
Name	recipe-ma	ike-cat1a		
Version	2.1.2 relea	se	~	
Туре	Shared Gi	t Repo	~	~
applications	ake build recipe p	rovides the make files and scripts for building and programming PSoC™	6	



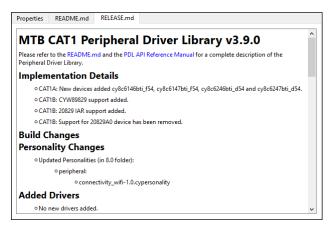
5.5.2 README.md tab

This tab dispays the *README.md* file for the selected BSP or library, if available.



5.5.3 RELEASE.md tab

This tab dispays the *RELEASE.md* file for the selected BSP or library, if available.



5.6 Buttons

The Library Manager contains the following buttons to perform the described actions:

- **Update** Use this button to update your project with changes made in the Library Manager. This action runs the make getlibs command and you can use it at any time, even if you made no changes. You might do this for a project that has no libraries yet, or to update the libraries specified as "Latest" to get any updates. This button becomes bolded when there are pending changes requiring an update.
- Close Use this button to close the Library Manager.
- **Retry** This button displays if the message console indicates that the tool cannot access the manifest file. Use the **Retry** button after adjusting your internet and/or proxy settings to check if the tool can access the manifest file.

5.7 Message console

The area below the BSPs and Libraries displays various messages, such as when you select/deselect an item, click a button, or select Local Content mode.



Tool change description

6 Tool change description

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

Version	Change descriptions		
1.0	New tool.		
	Added Settings and Help menus.		
1.1	Moved link about version changes to the message console, when it is applicable.		
	Added icon to indicate online/offline status.		
	Removed Summary dialog; summary is shown in the console.		
1.2.0	Added Retry button.		
	Changed Apply button to Update.		
	Added Close button.		
	Tool can be launched from the Windows Start menu.		
	Updated versioning to support patch releases.		
	Updated for the MTB flow.		
	Added toolbar commands to show/hide items, as well as collapse and expand the trees.		
	Added indications for newer versions of BSPs and libraries.		
1.30	For BSPs and libraries that are not selected, the displayed version was changed from Latest #.X release to the actual most recent X.Y.Z release.		
1.40	Updated the handling on .mtbx files.		
2.0	Entire GUI updated to support ModusToolbox™ version 3.0.		
	Added rename BSP feature.		
2 10	Added Create from MPN feature.		
2.10	Changed Offline mode to Local Content mode.		
	Removed online status icon.		
2.20	Back-end changes and bug fixes.		
2.20	Updated the Properties tab to include descriptions.		



Tool change description

Revision history

Revision	Date	Description
**	2019-10-16	New document.
*A	2019-10-17	Added a warning about removing all Cypress BSPs/libraries when there is no custom BSP.
*В	2020-03-26	Updated to version 1.1.
*C	2020-09-01	Updated to version 1.2.0.
*D	2020-10-07	Added details for BTSDK 2.8.
*E	2021-03-25	Updated to version 1.30.
*F	2021-09-22	Updated to version 1.40.
*G	2022-09-14	Updated entire document to reflect version 2.0.
*Н	2023-05-09	Updated document to version 2.10.
*	2024-02-12	Updated document to version 2.20.

Trademarks

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

Edition 2024-02-12 Published by Infineon Technologies AG

81726 Munich, Germany

© 2024 Infineon Technologies AG. All Rights Reserved.

Do you have a question about this document?

Email: erratum@infineon.com

Document reference 002-28736 Rev. *I

IMPORTANT NOTICE

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics ("Beschaffenheitsgarantie").

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, 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.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

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. For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (www.infineon.com).

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.