ModusToolbox™ run-time software reference guide

About this document

Scope and purpose

This document helps you understand what is included in the ModusToolbox™ run-time software.

Document conventions

Convention	Explanation
Bold	Emphasizes heading levels, column headings, menus and sub-menus
Italics	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

Reference documents

Refer to the following documents for more information as needed:

- ModusToolbox™ tools package user guide
- Project Creator user guide
- Library Manager user guide
- https://github.com/Infineon

Table of contents

About this document . . . 1

Table of contents . . . 2

1 Overview . . . 3

2 Getting started . . . 4

3 Run-time software descriptions and documentation . . . 5

3.1 Device support libraries . . . 6

3.2 Middleware . . . 8

3.3 Connectivity middleware tools . . . 17

4 Library dependencies . . . 18

4.1 Library manager . . . 18

4.2 Adding libraries . . . 19

5 Library configuration files . . . 20

5.1 Wi-Fi core configuration . . . 20

5.2 MQTT . . . 21

5.3 OTA . . . 21

6 Working with examples . . . 22

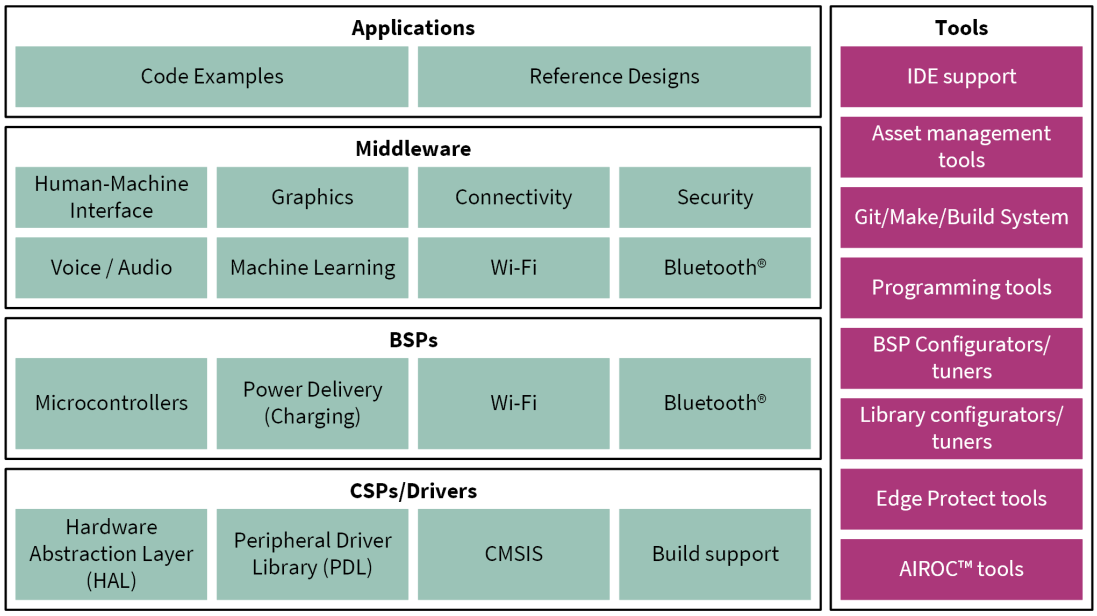
7 Adding and configuring low power . . . 23

8 Adding and configuring Bluetooth® . . . 24

Disclaimer . . . 25

1 Overview

The overall ModusToolbox™ ecosystem provides many types of software to help you develop applications for Infineon devices. This software includes GUIs, command-line programs, applications, middleware, and third-party software that you can use in just about any combination you need. The following shows a high-level diagram of the ecosystem.



One part of the ModusToolbox™ ecosystem is run-time software that helps you rapidly develop applications using Infineon BSPs.

This software is based on the industry standard lwIP TCP/IP stack and Mbed TLS network security. They provide core functionality including connectivity, security, firmware upgrade support, and application layer protocols like MQTT for applications that do not use commercial cloud management systems such as Arm Pelion or Amazon AWS IoT Core.

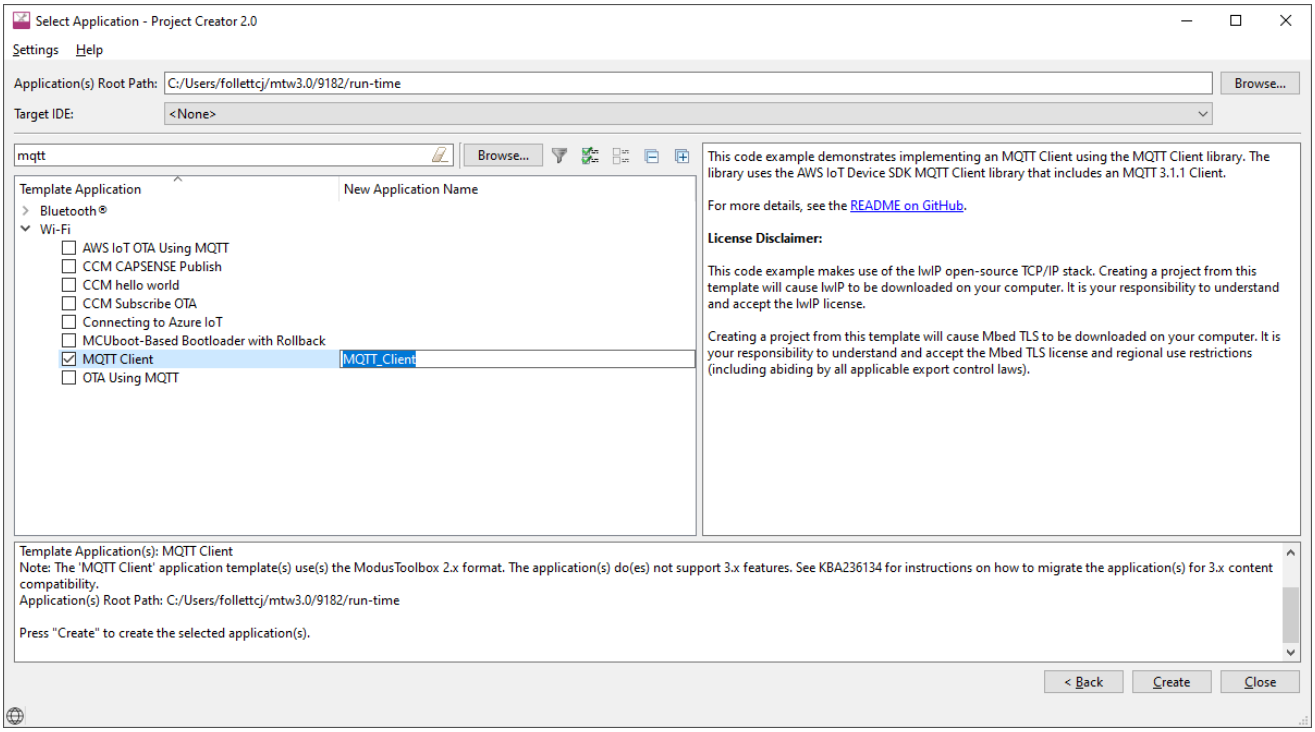
This software is intended for customers who have their own cloud device management backend, whether hosted on AWS, Google, Microsoft Azure, or another cloud infrastructure. They enable development with custom or alternative third-party cloud management approaches with a fully open, customizable, and extensible source code distribution. You can modify or extend them to match your needs.

The run-time software provides features such as the Wi-Fi Connection Manager, a Secure Socket layer, support for application layer cloud protocols, Bluetooth® Low Energy (LE) functionality, and Low Power Assist (LPA). The software currently supports TCP, MQTT, UDP, and HTTP/HTTPS client and server application layer protocols.

2 Getting started

The easiest way to get started is with an example. We provide many code examples that allow you to experiment with various features. You can get the examples by creating a ModusToolbox™ application, or by downloading them from the GitHub website:

- **Creating a ModusToolbox™ application**: Inside the ModusToolbox™ Project Creator tool, look for template applications under various categories such as Bluetooth® or Wi-Fi to find one for your needs. You can also search for a term like "mqtt" or other terms as needed and limit the number of examples that display. Refer to the Project Creator user g uide for more details.

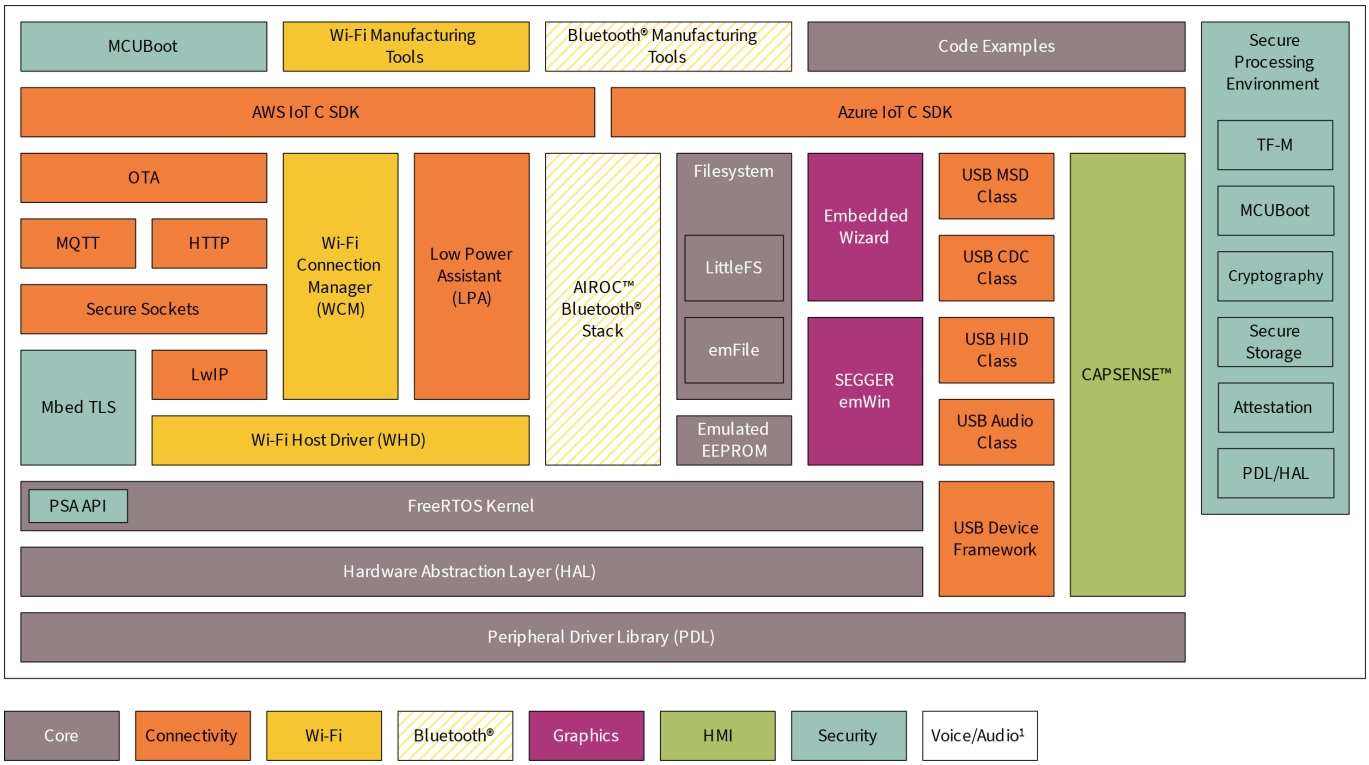


- **Downloading a code example**: Download examples directly from the GitHub website: https://github.com/Infineon/Code-Examples-for-ModusToolbox-Software

3 Run-time software descriptions and documentation

The ModusToolbox™ run-time software works together to help you easily get your IoT device connected to the cloud. Some of the libraries were written by us, while others use industry standard open source libraries. As you will see later in this document, these can be pulled into a ModusToolbox™ application using the Library Manager tool.

These libraries fit with various libraries as shown in the following diagram. See Library dependencies later in this document for how of these libraries are related.



[1] Voice/Audio is part of Machine Learning enablement; see https://www.infineon.com/cms/en/design-support/tools/sdk/modustoolbox-software/modustoolbox-machine-learning/ for more details.

All libraries are available as GitHub repositories. These "repos" contain source files, readme files, and documentation such as an API reference. When you include a library in your ModusToolbox™ application, the repository is downloaded into a shared directory next to the application directory. See the ModusToolbox™ tools package user guide for more details about an application's structure.

The following subsections provide brief descriptions for the main libraries, as well as links to the repository where you can find more information about them.

3.1 Device support libraries

The following tables provide links to the libraries and documentation for various devices, based on the organization in the Library Manager tool.

3.1.1 PSOC™ Edge

PSOC™ Edge device support libraries provide all necessary device specific components to be used during software development for this family of devices:

Library	Details	Docs
mtb-dsl-pse8x6	Provides the components required to develop software for the PSOC™ Edge E8x2, E8x3, E8x5, and E8x6 MCUs (6 MB total RAM: 5 MB System SRAM + 1 MB SRAM). It is a required dependency for every application targeting these devices and is added automatically when a project is created for a supported board in ModusToolbox™.	API reference
mtb-dsl-pse8xxgo	These packages include: • PSOC™ Edge MCU Peripheral Driver Library (PDL) - provides, device header files, start-up code, and peripheral drivers as a set of low-level APIs to configure, initialize, and use various supported peripheral drivers • Hardware Abstraction Layer (HAL) - provides high-level API wrappers around lower-level device specific APIs(such as PSOC™ MCU PDL) • Device Utilities (device-utils) - handles the pre-defined System Power Management (SysPm) callback implementations for peripherals which require special handling when the MCU enters or exits DeepSleep • make - provides build recipe make files and scripts for building and programming device applications	API reference
mtb-dsl-pse8xxgp		API reference

3.1.2 PSOC™ Control

Library	Details	Docs
mtb-dsl-psc3m6	Provides all necessary device specific components to be used during software development for the PSC3M6 family of devices.	API reference
mtb-dsl-psc1m3	Provides all necessary device specific components to be used during software development for the PSC1M3 family of devices.	API reference
retarget-io-psc1	A utility library to retarget the standard input/output (STDIO) messages to a UART port on PSC1M3 devices.	API reference
mtb-emeeprom-psc1	Emulates an EEPROM device in the PSC1M2/PSC1M3 device flash memory. The EEPROM middleware operates on top of the Flash driver included in the PSC1M3 Peripheral Driver Library.	API Reference
mtb-hal-psc3	Hardware Abstraction Layer for PSOC™ Control.	API reference
mtb-pdl-cat1	The Peripheral Driver Library (PDL) integrates device header files, startup code, and low-level peripheral drivers into a single package.	API reference

Library	Details	Docs
recipe-make-cat1b	This repo provides the build recipe make files and scripts for building and programming PSOC™ Control C3 applications.	Release notes
syspm-callbacks-psc3	All prebuilt callback implementations will provide a function signature which conforms to the Cy_SysPmCallback function pointer signature. This enables them to be directly registered without the application needing to write a wrapper function to adapt the signature.	N/A

3.1.3 PSOC™ 6, AIROC™ CYW20829, TRAVEO™ II, XMC7000

Library	Details	Docs
cat1cm0p	Arm® Cortex®-M0+ prebuilt images enabling flash write services and Bluetooth® Low Energy event handling.	Release notes
mtb-hal-cat1	The Hardware Abstraction Layer package provides a set of APIs to initialize, configure, and use the resources using our defined Hardware Abstraction Layer.	API reference
mtb-hal-t2gc2d6m	For newer generations of BSPs, the Hardware Abstraction Layer provides a generic, device agnostic interface that can be used across multiple product families. The HAL is designed for portability, not as a general-purpose user library.	API reference
mtb-hal-xmc72		API reference
mtb-pdl-cat1	The Peripheral Driver Library (PDL) integrates device header files, startup code, and low-level peripheral drivers into a single package.	API reference
recipe-make-cat1a	This repo provides the build recipe make files and scripts for building and programming PSOC™ 6 applications.	Release notes
recipe-make-cat1b	This repo provides the build recipe make files and scripts for building and programming AIROC™ CYW20829 applications.	Release notes
recipe-make-cat1c	This repo provides the build recipe make files and scripts for building and programming XMC7000 and TRAVEO™ II applications.	Release notes
syspm-callbacks-t2gc2d6m	All prebuilt callback implementations will provide a function signature which conforms to the Cy_SysPmCallback function pointer signature. This enables them to be directly registered without the application needing to write a wrapper function to adapt the signature.	N/A
syspm-callbacks-xmc72		N/A

3.1.4 PSOC™ 4, PMG

Library	Details	Docs
mtb-hal-cat2	The Hardware Abstraction Layer package provides a set of APIs to initialize, configure, and use the resources using the defined Hardware Abstraction Layer.	API reference
mtb-pdl-cat2	The Peripheral Driver Library (PDL) integrates device header files, startup code, and low-level peripheral drivers into a single package.	API reference

Library	Details	Docs
recipe-make-cat2	This repo provides the build recipe make files and scripts for building and programming PSOC™ 4 and PMG applications.	Release notes
lin	Implements LIN 2.2 Slave node for the PSoC 4 devices	API reference

3.1.5 XMC1000, XMC4000

Library	Details	Docs
mtb-xmclib-cat3	The XMC™ peripheral library consists of low-level drivers and CMSIS start-up code for the XMC™ product family peripherals.	API reference
recipe-make-cat3	This repo provides the build recipe make files and scripts for building and programming XMC™ applications.	Release notes

3.1.6 AIROC™ CYW43907, CYW54907

Library	Details	Docs
mtb-hal-cat4	Hardware Abstraction Layer package, provides a set of APIs to initialize, configure, and use the CYW43907/CYW54907 resources using the defined Hardware Abstraction Layer.	API reference
recipe-make-cat4	This repo provides the build recipe make files and scripts for building and programming CYW43907/CYW54907 applications.	Release notes

3.1.7 AIROC™ CYW5551x

Library	Details	Docs
mtb-hal-cat5	Hardware Abstraction Layer package, provides a set of APIs to initialize, configure, and use the CYW5551x resources using the defined Hardware Abstraction Layer.	API reference
mtb-pdl-cat5	The PDL simplifies software development for CYW5551x family of devices. The PDL integrates device header files, startup code, and peripheral drivers into a single package.	API reference
recipe-make-cat5	This repo provides the build recipe make files and scripts for building and programming CYW55513/CYW55913 applications.	Release notes

3.2 Middleware

This document provides links and descriptions for the various categories of middleware that you might use in a ModusToolbox™ application.

You do not need to use these libraries directly. When you create a ModusToolbox™ application, the system downloads all the libraries your application needs to take full advantage of the various features of our devices. A ModusToolbox™ application can use different libraries based on the Active BSP. If you wish to add various libraries to your application, use the Library Manager tool.

- Drivers and core middleware
- ModusToolbox™ for connectivity
- ModusToolbox™ for Bluetooth®
- ModusToolbox™ for Wi-Fi and ethernet

- ModusToolbox™ for Machine Learning
- ModusToolbox™ for power conversion
- ModusToolbox™ for sensing
- ModusToolbox™ for Security
- ModusToolbox™ for Graphics

3.2.1 Drivers and core middleware

Library	Details	Docs
core-lib	Header files that declare basic types and utilities (such as result types or ASSERT) that can be used by multiple BSPs.	API reference
cmsis	Used by application or middleware to link CMSIS Core headers.	CMSIS documentation
freertos	FreeRTOS kernel, distributed as standard C source files with configuration header files.	FreeRTOS webpage
freertos-posix	The Portable Operating System Interface (POSIX) is a family of standards specified by the IEEE Computer Society for maintaining compatibility between operating systems. freertos-posix implements a small subset of the POSIX threading API.	POSIX API Reference
command console	Provides a framework to add command console support to the application (or) product use cases.	API reference
csdadc	Enables the ADC functionality of the CAPSENSE™ Sigma-Delta (CSD) hardware block. Useful for devices that do not include other ADC/IDAC options.	API reference
csdidac	The same, for IDAC functionality.	API reference
emeeprom	The Emulated EEPROM library provides an API to manage an emulated EEPROM in flash. It has support for wear leveling and restoring corrupted data from a redundant copy.	API reference
emfile	A FAT16/32 filesystem for embedded systems supporting SPI NOR flash and SD card.	User guide
emusb -host	CPU-independent USB Host stack.	API reference
emusb -device	Enables easy integration of USB functionality into an embedded system.	API reference
usbdev	The USB Device library provides a full-speed USB 2.0 Chapter 9 specification compliant device framework.	API reference
littlefs	A little fail-safe filesystem designed for microcontrollers.	Design document
mtb-littlefs	Provides a set of block device drivers for use with the littlefs file system.	API reference

Library	Details	Docs
abstraction- rtos	Abstraction-RTOS has been created by Infineon to allow the flexibility of Infineon created software to run within any RTOS. It may be recommended by Developers providing libraries to use the abstraction RTOS; however, it is not recommended to use this as the main RTOS interface. It is recommended that developers should use the APIs of the chosen RTOS (FreeRTOS/Azure RTOS/ Zephyr RTOS).	API reference
clib -support	The CLib FreeRTOS support library provides the necessary hooks to make C library functions such as malloc and free thread safe.	API reference
retarget- io	Provides a board-independent API to retarget text input/output to a serial UART on a kit.	API reference
serial-flash	Provides a board-independent API to use the serial flash on a kit.	API reference
mtb-stl	The ModusToolbox™ Safety Test Library provides functional safety APIs to implement overall safety of a system that depends on automatic protection suitable for using in industrial environments and home appliances.	Release notes
mtb-ipc	The IPC driver allows communication between multiple CPUs or between multiple tasks operating in different domains within a single CPU.	API reference
async-transfer	This provides an implementation of data transfer functions in which the calling application initiates the data transfer on the desired communication peripheral and then the data transfer happens in the background without the application involvement.	API reference
device-db	Provides device information for project-creator, configurators, and other ModusToolbox™ tools.	Release notes

3.2.2 ModusToolbox™ for connectivity

Library	Details	Docs
aws-iot-device-sdk-embedded-C	Collection of C source files under the MIT open source license that can be used in embedded applications to securely connect IoT devices to AWS IoT Core.	API reference
aws-iot-device-sdk-port	Contains the port layer implementation for the MQTT and HTTP Client libraries to work with the AWS-IoT-Device-SDK-Embedded-C library on platforms with network connectivity.	API Reference
azure-c-sdk-port	Implements the port layer for the Azure SDK for Embedded C to work on platforms with network connectivity.	API reference
azure-sdk-for-c	Allows small embedded (IoT) devices to communicate with Azure services.	README
http-client	Provides the HTTP Client implementation that can work on platforms with Wi-Fi or Ethernet connectivity.	API reference
http-server	Provides communication functions for an HTTP server.	API Reference

Library	Details	Docs
lpa	The Low Power Assistant (LPA) is a library and associated settings in the ModusToolbox™ Device Configurator that allow you to configure a Host and WLAN (Wi-Fi / BT Radio) device for optimized low-power operation.	API reference
mqtt	This library includes the open source AWS IoT device SDK embedded C library plus some glue to ensure seamless MQTT cloud connectivity.	API reference
secure-sockets	The Secure Sockets library eases application development by exposing a socket like interface for both secured and non-secured socket communication.	API reference
connectivity-utilities	General purpose middleware connectivity utilities, for instance a linked_list or a json_parser.	API reference
ota-update	Provides support for Over-The-Air update of the application code running on various devices.	API reference
ota-bootloader-abstraction	The OTA Bootloader Abstraction library has implementation for bootloader specific storage interface APIs for handling OTA upgrade images.	API reference
netxduo	Advanced, industrial-grade TCP/IP network stack designed specifically for deeply embedded real-time and IoT applications.	Overview
netxduo-network-interface-integration	This library is an integration layer that links the NetXDuo network stack with the underlying Wi-Fi host driver (WHD).	API reference
memfault -firmware- sdk	An SDK responsible for working with Memfault device monitoring, debugging and OTA management platform.	External website
golioth -firmware- sdk	A software development kit for connecting embedded devices to the Golioth IoT Cloud.	External website
lwIP	Lightweight open-source TCP/IP stack.	External website
virtual-connectivity-manager	Virtual-Connectivity-Manager (VCM) is a library that enables connectivity libraries to add multi-core support through virtualization. Virtualization allows the connectivity stack running on one core to be accessed from another core using Inter Process Communication (IPC).	API reference
buffer-pool-manager	This library provides APIs to create pool of fixed size buffers, receive the available buffer and free it back to the pool after the use.	API reference

3.2.3 ModusToolbox™ for Wi-Fi and ethernet

Library	Details	Docs
wifi -mw-core	The Wi-Fi Middleware Core library bundles the core libraries that any Wi-Fi application needs. This has been deprecated.	API reference
wifi-host-driver	The Wi-Fi Host Driver (WHD) is an independent, embedded driver that provides a set of APIs to interact with our WLAN chips.	API reference

Library	Details	Docs
wifi-connection-manager	The Wi-Fi Connection Manager (WCM) provides easy to use APIs to establish and monitor Wi-Fi connections on our devices that support Wi-Fi connectivity.	API reference
wifi-core- freertos-lwip-mbedtls	This repo comprises core components needed for Wi-Fi connectivity support. The library bundles FreeRTOS, lwIP TCP/IP stack, mbed TLS for security, Wi-Fi host driver (WHD), wifi connection manager (WCM), secure sockets, connectivity utilities and configuration files.	API reference
wifi-resources	This repo contains clm and nvram files for the supported Wi-Fi boards from associated chip set.	API reference
whd-bsp-integration	This library provides some convenience functions for connecting the Wi-Fi Host Driver (WHD) library to a BSP that includes a WLAN chip.	API reference
smartcoex	Provides an API to configure the coex parameters for WLAN and Bluetooth® Low Energy on platforms with Wi-Fi & Bluetooth® combo chip.	API reference
enterprise-security	This library implements a collection of the most commonly used Extensible Authentication Protocols (EAP) used in enterprise Wi-Fi networks.	API reference
wpa3-external-supplicant	The WPA3 External Supplicant supports WPA3 SAE authentication using HnP (Hunting and Pecking Method) using RFC https://datatracker.ietf.org/doc/html/rfc7664 and H2E (Hash to Element Method) using RFC https://datatracker.ietf.org/doc/html/draft-irtf-cfrg-hash-to-curve-10 and following 802.11 spec 2016.	See links to the left.
ethernet-connection-manager	Ethernet Connection Manager (ECM) is a library which helps application developers to manage ethernet connectivity for XMC7200 devices.	API Reference
ethernet-core-freertos-lwip-mbedtls	This repo comprises core components needed for ethernet connectivity support for XMC7200 devices	API reference
ethernet-phy-driver	Provides the Ethernet PHY-related interface APIs as required by Ethernet connection manager library to enable the completion of Ethernet-based applications on supported platforms.	API reference
lwip-network-interface-integration	This library is an integration layer that links the lwIP network stack with the underlying Wi-Fi host driver (WHD) and Ethernet driver. This library interacts with FreeRTOS, lwIP TCP/IP stack, Wi-Fi host driver (WHD), and Ethernet driver. It contains the associated code to bind these components together.	API Reference
nonrtos-lwip-mbedtls	Contains core components needed for non-rtos connectivity support. The library bundles lwIP TCP/IP stack, mbed TLS for security, connectivity utilities and configuration files.	API reference
rtlabs-uphy-lib	Partner library for with support of RT-Labs Industrial Communication stack on XMC7000.	RT-Labs u-phy website

3.2.4 ModusToolbox™ for Bluetooth®

Library	Details	Docs
ble-mesh	Provides Application Programming Interfaces (APIs) for application developers to use and create Mesh Applications.	API reference
btstack	BTSTACK is our Bluetooth® Host Protocol Stack implementation. The stack is optimized to work on our Bluetooth® controllers.	API reference
btstack-integration	Platform adaptation layer (porting layer) between AIROC™ BT Stack and Abstraction Layers (CYHAL and CYOSAL) for different hardware platforms.	README
bless	This library is specific to devices like CY8C6347BZI-BLD53, with onboard Bluetooth® and no separate connectivity device. Not recommended for new designs.	API reference
bt-fw-qtl-cyw55512	The bt-fw-qtl-cyw55512 middleware library provides firmware patches for the CYW55512 AIROC™ Wi-Fi and Bluetooth® combo devices	N/A

For information about BTSDK, see https://Infineon.github.io/btsdk-docs/BT-SDK/index.html.

3.2.5 ModusToolbox™ for motor control

Library	Details	Docs
motor-ctrl-lib	Motor Control Library represents integration of the implemented cross platform motor control algorithms with the MCU-family oriented Motor Control Driver Interface (MCDI).	API reference

3.2.6 ModusToolbox™ for power conversion

Library	Details	Docs
mtb-pwrconv	Power conversion middleware.	API reference
mtb-pfc-lib	Provides a universal PFC controller to create various PFC power conversion applications.	Firmware Reference Manual (pdf)
mtb-mw-pctrl	Provides the optimized and reusable functional blocks for building power topology applications in ModusToolbox™.	API reference
mtb-pmbus	PMBus (Power Management Bus) is an open standard digital power management protocol that enables communication with power conversion and management devices.	API reference
mtb-xmc-ecat	Consists of all the elements necessary to start the EtherCAT slave development with the Beckhoff Slave Stack Code (SSC).	API reference

3.2.7 ModusToolbox™ for Graphics

Library	Details	Docs
emwin	SEGGER embedded graphic library and graphical user interface (GUI) framework designed to provide processor- and display controller-independent GUI for any application that needs a graphical display.	Overview
display-amoled-co5300	This is the driver library for CO5300 display controller driving 1.43-inch 466x466 MIPI DSI AMOLED display connected to the PSOC™ Edge E84 Evaluation Kit.	API reference
display-dsi-waveshare-4-3-lcd	This driver library is designed to support a Waveshare 4.3 inch DSI Capacitive Touch Display on PSOC™ Edge E8x kits.	API reference
display-dsi-waveshare-7-0-lcd-c	This is the driver library for Waveshare 7-inch Raspberry Pi DSI LCD C Display interfaced to PSOC™ Edge E84 Evaluation Kit.	API reference
display-tft-ek79007ad3	This driver library is designed to support 10.1 inch MIPI DSI TFT LCD, model WF101JTYAHMNB0.	API reference
display-tft-st7735s	This library provides functions to support a 0.96-inch TFT LCD driven by an ST7735S controller. This is the display used on the SHIELD_XENSIV_A shield.	API reference
touch-ctp-ft3268	This library provides functions to support Capacitive Touch Panel (CTP) on 1.43 inch AMOLED circular (CO5300) wearable display driven by FT3268 controller. This 1.43 inch MIPI DSI AMOLED display is used with PSOC™ EDGE E84 Evaluation kit.	API reference
touch-ctp-ft5406	This library provides basic set of functions to support Capacitive Touch Panel (CTP) of Waveshare 4.3 inch DSI display driven by FT5406 CTP controller. This Waveshare 800 x 480 pixel 4.3 inch video mode display panel is used with PSOC™ Edge E8x kits.	API reference
touch-ctp-ft6146-m00	This library provides functions to support Capacitive Touch Panel (CTP) of 1.43 inch AMOLED circular wearable display driven by FT6146-M00 controller. This 1.43 inch 466 x 466 pixels MIPI DSI AMOLED display is used with PSOC™ Edge E84 Evaluation Kit.	API reference
touch-ctp-gt911	This is the driver library for the GT911 capacitive sensing touch panel.	API reference
touch-ctp-ili2511	This library provides functions to support Capacitive Touch Panel (CTP) of 10.1 inch TFT LCD WF101JTYAHMNB0 driven by ILI2511 controller. This 1024 x 600 pixel display panel is used with PSOC™ Edge E8 Evaluation Kit.	API reference

3.2.8 ModusToolbox™ for Security

Library	Details	Docs
trusted-firmware-m	Provides secure world software for Arm Cortex-M processors.	Introduction
dfu	The Device Firmware Update (DFU) library provides an API for updating firmware images.	API reference
Arm Mbed TLS	A library to include cryptographic and SSL/TLS capabilities in an embedded application.	README

Library	Details	Docs
mbedtls-pal	ModusToolbox™ porting layer for upstream Mbed TLS 4.x (https://github.com/ARMmbed/mbedtls).	MBEDTLS README
cy-mbedtls-acceleration	We provide a library that extends MbedTLS to enable hardware-accelerated encryption on various MCUs.	API reference
MCUBoot	Secure bootloader for 32-bits microcontrollers.	README
ifx-mcuboot-pse84	The IFX MCUboot PSE84 middleware is an Infineon-maintained ModusToolbox™ library based on the MCUboot project, specifically adapted for PSOC™ Edge PSE84 microcontrollers. This middleware provides the core components and APIs needed to build secure bootloader applications for PSE84 devices.	Code example
mtb-srf	Secure request framework.	API reference

3.2.9 ModusToolbox™ for Machine Learning

Library	Details	Docs
ml-inference	A set of pre-compiled libraries which provide easy to use API's to run ML workloads on embedded platforms.	API reference
ml-middleware	Helper functions to simplify integration of ML models.	API reference
ml-tflite-micro	A pre-configured TensorFlow tflite-micro runtime library for Infineon devices.	TensorFlow Lite documentation
ml-executorch	The ml-executorch asset provides the ExecuTorch runtime integration for Infineon PSOC™ Edge microcontrollers. It packages the embedded inference libraries, headers, and source files required to run PyTorch ExecuTorch models on target devices.	ExecuTorch README

3.2.10 ModusToolbox™ for Voice

Library	Details	Docs
cyberon-dspotter-lib-psoc6-cm0p	Highly-accurate and lightweight wake word engine. It enables building always-listening voice-enabled applications for the CM0p Infineon devices.	External website
cyberon-dspotter-lib-psoc6-cm4	Highly-accurate and lightweight wake word engine. It enables building always-listening voice-enabled applications for the CM4 Infineon devices	README

3.2.11 ModusToolbox™ for sensing

Library	Details	Docs
capsense	Capacitive sensing can be used in a variety of applications and products where conventional mechanical buttons can be replaced with sleek human interfaces to transform the way users interact with electronic systems.	API reference

Library	Details	Docs
capsense-touchpad-host	Portable middleware for controlling an Infineon PSOC™ 4000T CAPSENSE™ touchpad from a bare-metal or RTOS-based host MCU.	API reference
linux-i2c-touchpad	This simplifies the integration of I2C-based touchpad devices into Linux systems. It provides a robust and efficient implementation for communicating with touchpad hardware over the I2C bus.	Reference guide
sensor-humidity-sht4x	This library provides functions for interfacing with SHT40 digital humidity and temperature sensor from Sensirion on PSOC™ Edge device.	API reference
sensor-motion-bmi270	This library provides functions for interfacing with the BMI270 I2C 16-bit inertial measurement unit (IMU) with 3-axis accelerometer and 3-axis gyroscope used on the PSOC™ Edge E84 Evaluation Kit.	API reference
sensor-orientation-bmm350	This library provides functions for interfacing with the BMM350 I3C 3-axis magnetometer used on the PSOC™ Edge E84 Evaluation Kit and PSOC™ Edge E84 AI Kit.	API reference

3.2.12 ModusToolbox™ for Audio

Library	Details	Docs
audio-codec-tlv320dac3100	This is the driver library supporting a Texas Instruments TLV320DAC3100 low-power stereo audio DAC with audio processing on PSOC™ Edge E84 Evaluation Kit (`KIT_PSE84_EVAL_EPC2`, `KIT_PSE84_EVAL_EPC4`).	API reference
audio-front-end	This is an agnostic audio processing technology for more accurate voice command recognition (far-field or close field) by removing interfering sounds captured from the microphone.	API reference
audio-sw-codecs	This implements an abstraction layer to a audio software codec. It abstracts multiple audio codecs that help applications to have an unified interface across the various software codec.	API reference
audio-voice-core	This consists of core audio algorithm implementation. It contains core algorithm implementation for speech-onset-detection, low-power wake word detection, and audio front-end.	README
speech-onset-detection	This is used to detect the beginning of a spoken word or utterance. For the target application, this is the first stage of a low power multi-stage wake-up phrase solution.	API reference
staged-voice-control	Staged voice control middleware provides easy to use API on core1 to feed the audio data, get notified about current stage in the audio pipeline. On core2, it provides simple to use API to get the data from core1 and sending the notification about stage change to core1.	API reference
voice-assistant	The DEEPCRAFT voice assistant is a Edge AI software product used to create and deploy voice assistants (Wake Word Detection + Natural Language Understanding) into an embedded environment.	Code example

3.3 Connectivity middleware tools

This section list and describes various tools used for testing and bring-up.

3.3.1 Wi-Fi Bluetooth® tester

This application integrates the Command Console Library including Wi-Fi iPerf and Bluetooth® LE functionality. You can use this application to characterize the Wi-Fi/ Bluetooth® LE functionality and performance.

See the following for more details:

- https://github.com/Infineon/mtb-tester-psoc-edge-enterprise-security
- https://github.com/Infineon/mtb-tester-psoc-edge-wifi-bluetooth
- https://github.com/Infineon/mtb-tester-wifi-bluetooth

3.3.2 WLAN manufacturing test application

The WLAN Manufacturing Test Application is used to validate the WLAN firmware and radio performance of various Wi-Fi chips.

See https://github.com/Infineon/mtb-wifi-mfg-tester for more details.

- **WLAN Manufacturing Test Middleware**

  The WLAN Manufacturing Test Middleware application is used to validate the WLAN firmware and radio performance of Wi-Fi devices.

  See https://github.com/ Infineon /wifi-mfg-test for more details.

3.3.3 Wi-Fi cert tester tool

This tool is used for Wi-Fi Certification of 11n, PMF, WPA3 and 11AC. The Wi-Fi cert tester tool uses the command console asset to initialize and invokes wifi-cert middleware init function.

- **Wi-Fi Cert Middleware for All SDKs**

  The Wi-Fi Cert middleware provides an easy way to test WFA cert test such as 11n, PMF, WPA3 and 11AC across all SDKs providing a common interface for all SDKs.

  See https://github.com/ Infineon /wifi-cert for more details.

3.3.4 Bluetooth® MFG tester

The Bluetooth® Manufacturing Test Application is used to validate the Bluetooth® Firmware and RF performance of Bluetooth® BR/EDR/LE devices.

https://github.com/Infineon/mtb-anycloud-bluetooth-mfg-tester

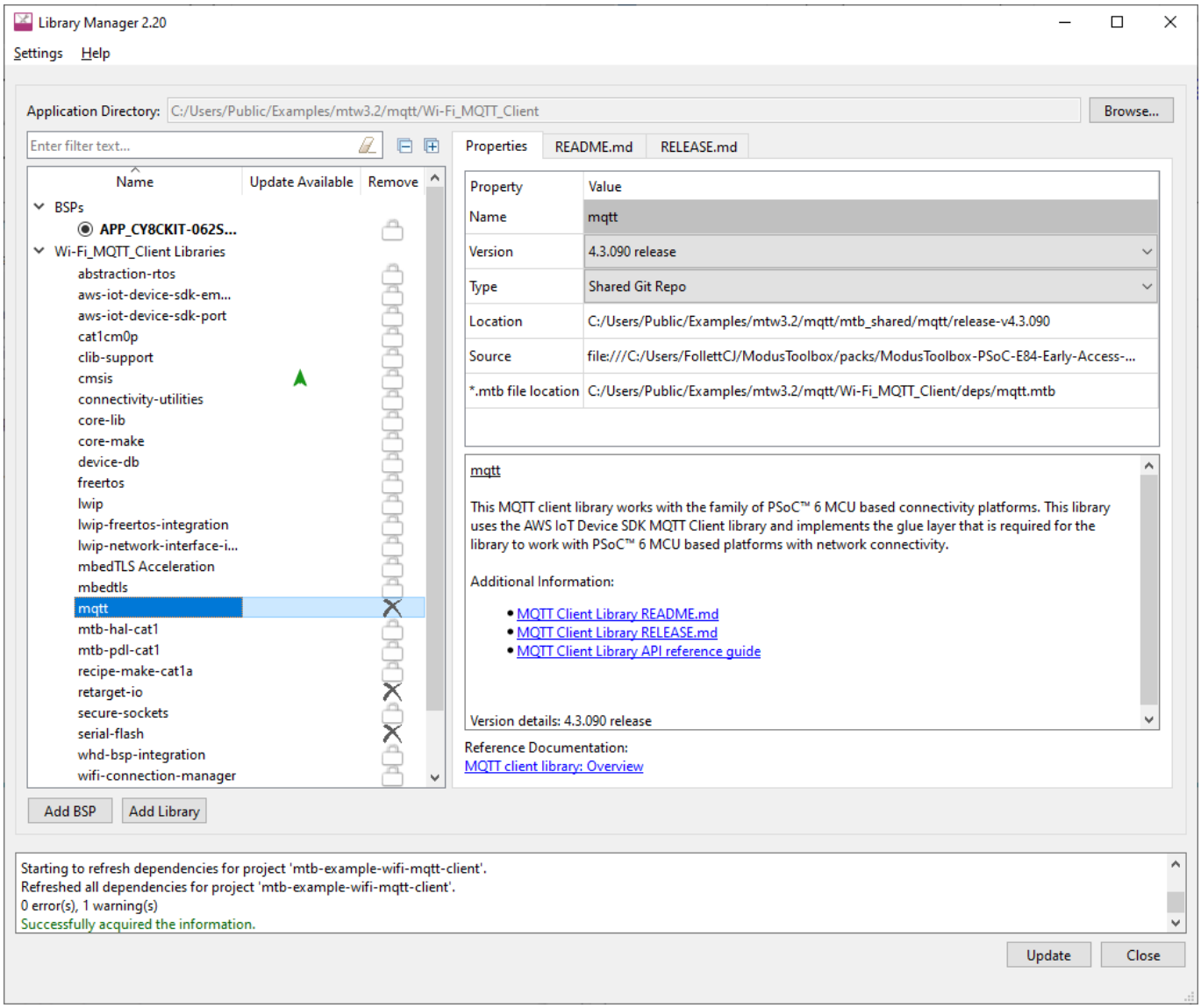
4 Library dependencies

When you include certain libraries, there are dependencies on other libraries to ensure everything works correctly. As an example, the MQTT Client code example includes four .mtb files directly. After the application has been created and processed, it contains numerous different libraries. See Adding libraries later in this section for more details.

Using the wifi -core- freertos - lwip-mbedtls library ensures you always have the essential Wi-Fi and networking libraries, plus good default configurations, in every cloud connected application you create.

4.1 Library manager

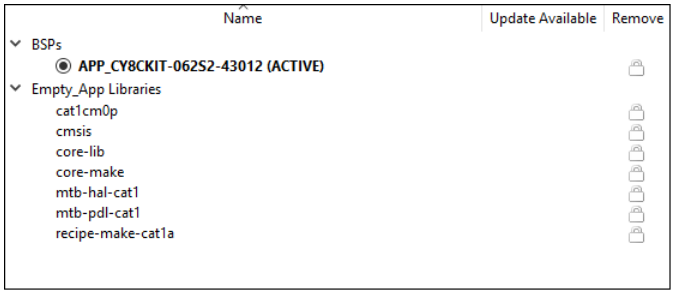
The ModusToolbox™ Library Manager tool allows you to add and remove board support packages (BSPs) and libraries, as well as select specific versions of BSPs and libraries. Refer to the Library Manager user guide for more details. As you can see in the following image, the MQTT Client code example application already includes several libraries.

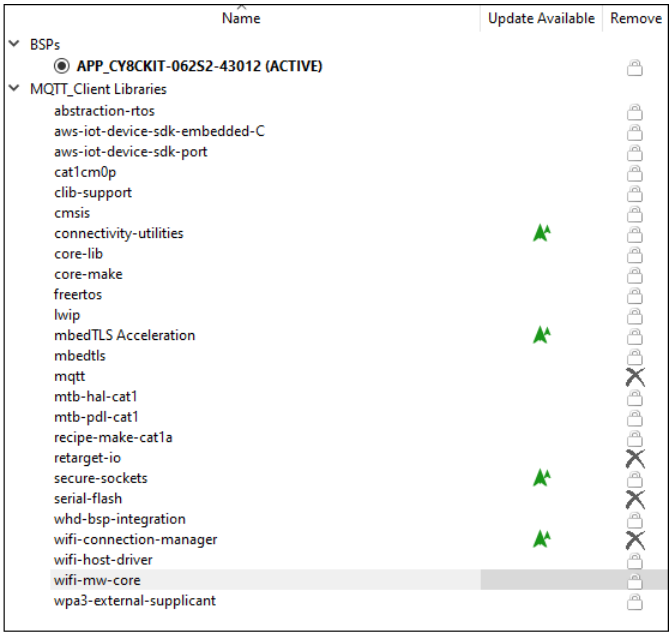


4.2 Adding libraries

As noted earlier, when the ModusToolbox™ build system encounters a .mtb file, it adds that library of code to the application. That library may contain additional .mtb files for dependent libraries. The ModusToolbox™ build system parses the .mtb files recursively so that all dependent libraries are added automatically. You do not need to know the dependencies.

The following images show the libraries included as part of an Empty PSOC™ 6 application before and after adding libraries.





When adding only a few libraries to the application, several other libraries are automatically included as well. The added libraries have their own dependency libraries, which in turn may have dependency libraries they depend upon. The same paradigm applies to various libraries that have dependencies. If you add a library that requires other libraries, they will be added automatically.

Dependencies are not necessarily bi-directional. While some libraries have dependencies, other libraries don't -- even if they are a dependency of another library.

5 Library configuration files

Some libraries provide configuration header file templates, such as the *FreeRTOSConfig.h* file. When adding a library to an application, copy the configuration file to the top-level application directory where you can edit it to customize the library. Even though there may be multiple files with the same name in an application, the ModusToolbox™ build system automatically picks the one at the top-level.

If you want to put the application-specific configuration files in a different location, you must specify the path to that directory (relative to the application's root directory) in the application's *Makefile* `INCLUDE` variable. The build system includes files in that path before it searches through the application's hierarchy.

The following are some examples of configuration files that you may need:

5.1 Wi-Fi core configuration

The Wi-Fi middleware core library has been deprecated. Use the porting guide (https://github.com/Infineon/lwip-network-interface-integration/blob/master/porting_guide.md) to migrate the application to use the lwIP network interface integration library.

The following are template files to use for configuration:

- *FreeRTOSConfig.h*: Settings for FreeRTOS.

  Use this file as a template for FreeRTOS configuration instead of the one from the FreeRTOS library. It has some modifications specific to use with other ModusToolbox™ run-time software.
- *mbedtls_user_config.h*: Settings for Mbed TLS.

  In addition to copying this file, you must also configure the macro MBEDTLS_USER_CONFIG_FILE to specify the file's location and add the macro to the list of DEFINES in the application's Makefile. For example, if you put the file in the top level, you would include this in the Makefile:

  ```
  DEFINES+=MBEDTLS_USER_CONFIG_FILE='"mbedtls_user_config.h"'
  ```

  Note that many code examples use this format instead:

  ```
  MBEDTLSFLAGS = MBEDTLS_USER_CONFIG_FILE='"mbedtls_user_config.h"'

  DEFINES+=$(MBEDTLSFLAGS)
  ```

- *lwipopts.h*: Settings for lwIP. Applications may choose to modify this file in order to optimize memory consumption based on the Wi-Fi characteristics of the application.

5.1.1 Optimizing smaller memory devices

Depending on your application or device size, you may need to reduce flash and RAM usage. The configuration files included with the lwIP and MbedTLS libraries provide various parameters to enable or disable features and optimize your application's size. For example, the MBEDTLS_SSL_SRV_C parameter enables code when the device is expected to function as a SSL/TLS server. If you don't need this feature, you can disable it to save flash:

```
#undef MBEDTLS_SSL_SRV_C
```

5.2 MQTT

The template file is in the mqtt/include subdirectory. See also the Quick Start in the library's README.md file (https://github.com/Infineon/mqtt).

- *core_mqtt_config.h*: Settings for MQTT.

5.3 OTA

The template file is in the ota-update/configs subdirectory. See also the library's README.md file (https://github.com/Infineon/ota-update).

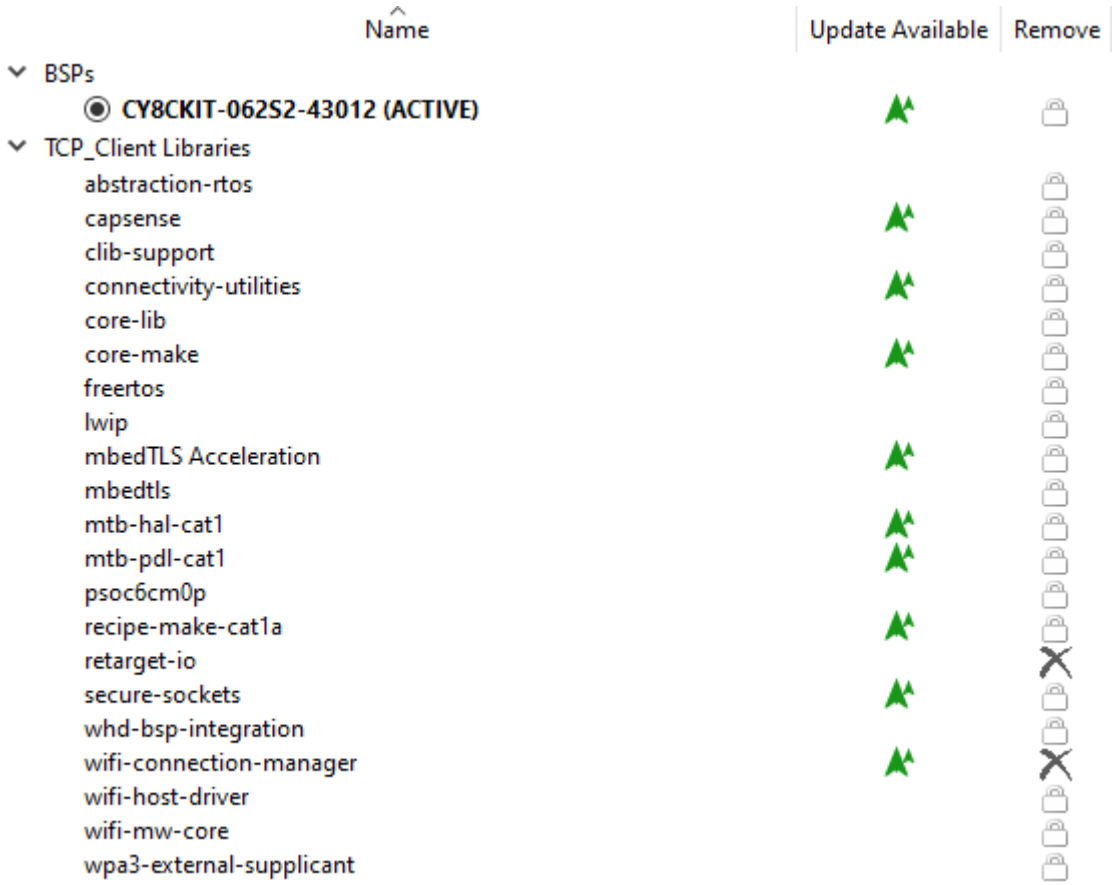
- *cy_ota_config.h*: Settings for OTA.

6 Working with examples

As described previously, the best way to learn is to download some examples and work through them. The examples include README.md files that guide you through the process to create and configure the application.

If you're already familiar with ModusToolbox™ software, then create the TCP Client example using your normal process. If you're new to ModusToolbox™ software, refer to the quick start guide as needed.

After creating the application, open the Library Manager and notice that several of the libraries discussed in this guide are already present.

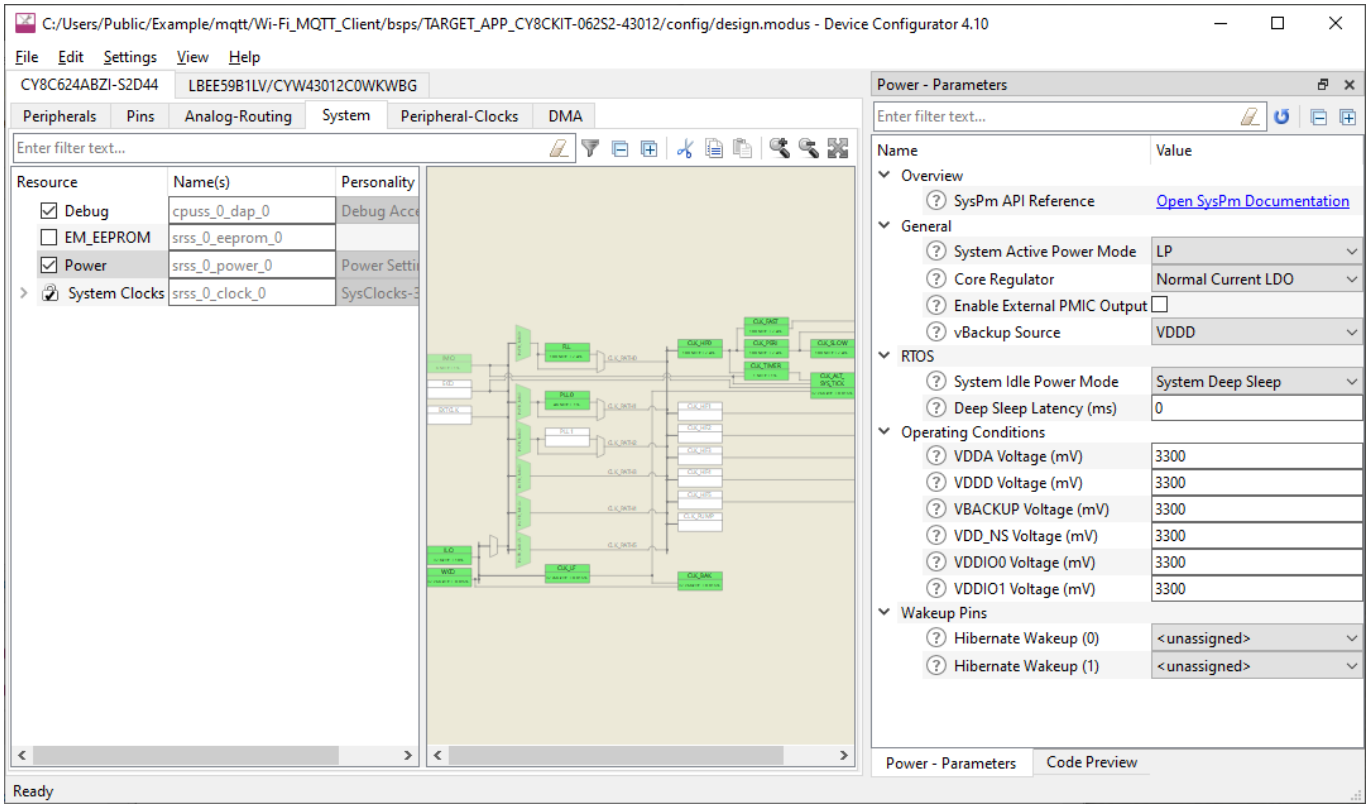


Then, create another application from an example, such as WLAN Low Power. Notice for this application that the LPA library is included.

Experiment with a few examples, and notice the differences in the libraries included, as well as various configuration options set for them. Obviously, review the code as well.

7 Adding and configuring low power

The WLAN Low Power example is one of several that demonstrate the low power features in ModusToolbox™ software. Among the low power features are the power settings accessible in the Device Configurator.



There are settings for the PSOC™ 6 MCU and the connectivity combo device. For more details about how to configure low power, refer to the LPA Guide here:

https://infineon.github.io/lpa/api_reference_manual/html/index.html

8 Adding and configuring Bluetooth®

As you have probably already discovered, there are several Bluetooth® examples as well. For example, the Wi-Fi Onboarding Using Bluetooth® LE shows how to use Bluetooth® on the combo device to help connect the Wi-Fi device to an access point. It also shows how to enable low-power modes on both the Wi-Fi and Bluetooth® devices. For more details, refer to the example's *README.md* file.

Trademarks

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

Edition 2026-10-16

Published by

Infineon Technologies AG

81726 Munich, Germany

© 2026 Infineon Technologies AG

All Rights Reserved.

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

Email: erratum@infineon.com

Document reference
IFX-yol1756416687278

Important notice

Products which may also include samples and may be comprised of hardware or software or both ("Product(s)") are sold or provided and delivered by Infineon Technologies AG and its affiliates ("Infineon") subject to the terms and conditions of the frame supply contract or other written agreement(s) executed by a customer and Infineon or, in the absence of the foregoing, the applicable Sales Conditions of Infineon. General terms and conditions of a customer or deviations from applicable Sales Conditions of Infineon shall only be binding for Infineon if and to the extent Infineon has given its express written consent.

For the avoidance of doubt, Infineon disclaims all warranties of non-infringement of third-party rights and implied warranties such as warranties of fitness for a specific use/purpose or merchantability. Infineon shall not be responsible for any information with respect to samples, the application or customer's specific use of any Product or for any examples or typical values given in this document.

If this document is marked as 'Preliminary', 'Target', 'Draft' or 'Proposal', Infineon reserves the right to change all information given in this document at any time without notice. Subject to the development and release of a Product for series supply by Infineon, the technical specifications of the Product are set forth in the relevant datasheet provided by Infineon without such marking.

The data contained in this document is exclusively intended for technically qualified and skilled customer representatives. It is the responsibility of the customer to evaluate the suitability of the Product for the intended application and the customer's specific use and to verify all relevant technical data contained in this document in the intended application and the customer's specific use. The customer is responsible for properly designing, programming, and testing the functionality and safety of the intended application, as well as complying with any legal requirements related to its use.

Unless otherwise explicitly approved by Infineon, Products may not be used in any application where a failure of the Products or any consequences of the use thereof can reasonably be expected to result in personal injury.

However, the foregoing shall not prevent the customer from using any Product in such fields of use that Infineon has explicitly designed and sold it for, provided that the overall responsibility for the application lies with the customer. Infineon expressly reserves the right to use its content for commercial text and data mining (TDM) according to applicable laws, e.g. Section 44b of the German Copyright Act (UrhG). If the Product includes security features:

Because no computing device can be absolutely secure, and despite security measures implemented in the Product, Infineon does not guarantee that the Product will be free from intrusion, data theft or loss, or other breaches ("Security Breaches"), and Infineon shall have no liability arising out of any Security Breaches.

If this document includes or references software:

The software is owned by Infineon under the intellectual property laws and treaties of the United States, Germany, and other countries worldwide. All rights reserved. Therefore, you may use the software only as provided in the software license agreement accompanying the software.

If no software license agreement applies, Infineon hereby grants you a personal, non-exclusive, non-transferable license (without the right to sublicense) under its intellectual property rights in the software (a) for software provided in source code form, to modify and reproduce the software solely for use with Infineon hardware products, only internally within your organization, and (b) to distribute the software in binary code form externally to end users, solely for use on Infineon hardware products. Any other use, reproduction, modification, translation, or compilation of the software is prohibited.

For further information on the Product, technology, delivery terms and conditions, and prices, please contact your nearest Infineon office or visit www.infineon.com.