Getting started with EZ-PD™ PMG1-B1 MCU using ModusToolbox™

About this document

Scope and purpose

This application note introduces you to the capabilities of the EZ-PD™ PMG1-B1 (Power Delivery Microcontroller Gen1) MCU with USB-C Power Delivery (PD) and helps you to get started with the EVAL_PMG1_B1_DRP Kit project in Eclipse IDE for ModusToolbox™ software.

Intended audience

This document is intended for designers who want to sink or source power through the USB-C PD port in applications such as cordless power tool chargers, wireless speakers, and portable electronics.

Table of contents

About this document **1**
Table of contents **2**
1 Introduction **3**
1.1 PMG1-B1 MCU general description 3
1.2 Features 3
1.3 PMG1-B1 MCU 5
2 Development ecosystem **6**
2.1 PMG1 MCU resources 6
2.2 Firmware/application development using ModusToolbox™ software 6
2.2.1 PMG1 MCU software resources 8
2.2.1.1 Configurators 8
2.3 ModusToolbox™ software help 9
2.4 Support for other IDEs 9
2.5 Programming and debugging 9
2.6 PMG1 MCU kits 9
3 Application development on the EVAL_PMG1_B1_DRP kit using ModusToolbox™ **10**
3.1 PMG1-B1 kit applications 10
3.2 Using the kit code example 10
3.3 Programming and debugging ModusToolbox™ software 14
3.3.1 Programming the PMG1-B1 device 16
3.3.1.1 Load the programming file 16
3.3.1.2 Connect the device 17
3.3.1.3 Program the device 18
3.3.2 Debugging using ModusToolbox™ software 19
3.4 Configuring the PDOs, PD, and buck-boost parameters 20
4 Summary **25**
References **26**
Acronyms and abbreviations **27**
Glossary **28**
Revision history **29**
Disclaimer **30**

1 Introduction

1.1 PMG1-B1 MCU general description

EZ-PD™ PMG1-B1 is a highly integrated single-port USB Type-C Power Delivery (PD) solution with integrated buck-boost controllers, and is from the portfolio of EZ-PD™ PMG1 high-voltage USB-C PD microcontrollers. These MCUs include the Arm® Cortex® CPU core, USB-C PD controller, and configurable integrated analog and digital peripherals. It has integrated gate drivers for VBUS NFET on the consumer path for sink application. It also includes hardware-controlled protection features for voltage and current on the USB-C connector and the battery terminal.

PMG1-B1 is targeted embedded systems that power from a high-voltage USB-C port and need an MCU to implement the product features. These typically include battery-powered applications that are powered by USB-C PD such as cordless power tool chargers, wireless speakers, and portable electronics.

The EVAL_PMG1_B1_DRP kit is an evaluation platform for the PMG1-B1 USB-C Power Delivery (PD) microcontroller (MCU) with an integrated buck-boost battery charge controller.

1.2 Features

Table 1 Features of PMG1-B1 MCU

Subsystem	Item	PMG1-B1
CPU and memory subsystem	Core	Arm® Cortex®-M0
	Max freq (MHz)	48
	Flash (KB)	128
	SRAM (KB)	16
Power Delivery	Power Delivery ports	1
	Role	DRP
	MOSFET gate drivers	4 x NFET
	Fault protection	VBUS, OVP, UVP, and OCP, SCP, VBUS to CC short protection, OTP
USB	Integrated Full Speed USB 2.0 device with Billboard Class support	No
Buck boost controller	Switching frequency Control range: Input Output	150 kHz to 600 kHz 5.5 V to 24 V 3.3 V to 21.5 V
VBUS_Ctrl	VBUS_CTRL pin strength to turn On/Off FET	To turn off the external NFET, the gate driver drives VBUS_CTRL to 0 V. To turn on the external NFET, drive the gate (connected to VBUS_CTRL) to VBUS_C + 8 V. Absolute ratings on this pin are –0.5 V to 32 V. Gate strength is 150 nA to 10 µA

Introduction

Subsystem	Item	PMG1-B1
HS and LS gate driver strength	Pullup-rising edge	
	HS1 & HS2	HSDR PMOS/pull-up driver strength configuration 0x1 = 20.1 Ω (Slow) 0x4= 9.7 Ω (Normal) 0x7 = 3.3 Ω (Fast)
	LS1 & LS2	LSDR PMOS/Pull-up driver strength configuration 0x1 = 17.9 Ω (Slow) 0x4 = 8.2 Ω (Normal) 0x7 = 2.9 Ω (Fast)
HS and LS gate driver strength	Pulldown-falling edge	
	HS1 & HS2	HSDR NMOS/pull-down driver strength configuration 0x1 = 20.4 Ω (Slow) 0x4 = 10.1 Ω (Normal) 0x7 = 3.4 Ω (Fast)
	LS1 & LS2	LSDR NMOS/pull-down driver strength configuration 0x1 = 20.3 Ω (Slow) 0x4 = 9.3 Ω (Normal) 0x7 = 3.1 Ω (Fast)
Voltage range	Supply (V)	VSYS (2.75 V to 5.5 V)
		VIN (4 V to 24 V)
	I/O (V)	1.71 V to 5.5 V
Digital	SCB (configurable as I2C/UART/SPI)	3
	TCPWM block (configurable as timer, counter, or pulse width modulator)	8
Analog	ADC	2x 8-bit SAR 1x 12-bit SAR
	On-chip temperature sensor	Yes
GPIO	Max # of I/O	21 (19 + 2 OVT)
Charging standards	Charging source	BC 1.2, Apple Charging 2.4 A
	Charging sink	BC 1.2, Apple Charging 2.4 A
ESD protection	ESD protection	Yes (ESD_HBM: 2 kV and ESD_CDM: 500 V)
Packages	Package options	48-pin QFN (6 × 6 mm, 0.4-mm pitch)

1.3 PMG1-B1 MCU

EZ-PD™ PMG1-B1 MCU is a member of the PMG1 MCU family. It includes a 48 MHz Arm® Cortex®-M0 processor with 128 KB flash, a complete Type-C USB PD transceiver with all termination resistors Rp, Rd, and 21 GPIOs. The dead battery Rd termination resistor is available in MPN CYPM1116-48LQXI. It is available in a 48-pin QFN package.

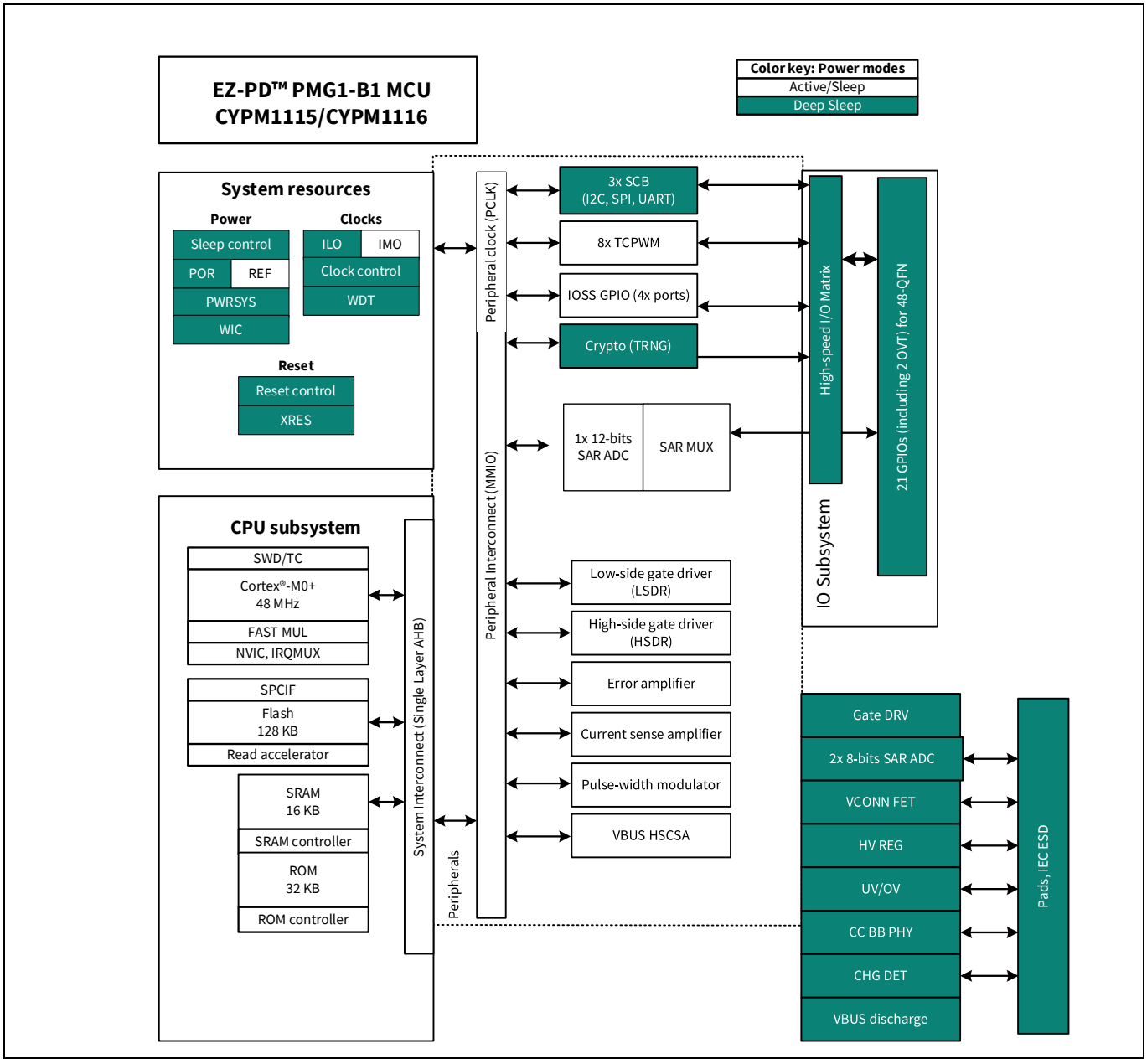


Figure 1 PMG1-B1 MCU block diagram

2 Development ecosystem

2.1 PMG1 MCU resources

The PMG1 MCU family has a rich set of documentation, development tools, and online resources to assist you during the development process. The following is an abbreviated list of resources for the PMG1 MCU. Visit the PMG1 MCU webpage to find out more.

- PMG1 MCU webpage
- Datasheets provide all the information needed to select and use a particular device, including functional description and electrical specifications.
- Application notes and code examples cover a broad range of topics, from basic to advanced level.
- Reference manuals provide detailed descriptions of the architecture and registers in each device family.
- Prototyping kits are available for evaluation, design, and development of different applications using PMG1 MCUs.
- Technical support: PMG1 MCU community forum, knowledge base articles.

2.2 Firmware/application development using ModusToolbox™ software

The ModusToolbox™ development platform used for firmware/application development with the PMG1 MCU. This latest generation toolset includes the Eclipse IDE and therefore is supported across Windows, Linux, and macOS platforms. The ModusToolbox™ software includes configurators, low-level drivers, middleware libraries, as well as other packages that enable you to create the PMG1 MCU applications. Using the configurators, you can set the configuration of different blocks in the device and generate code that can be used in firmware development.

The Eclipse IDE for ModusToolbox™ is integrated with the quick launchers for tools and design configurators in the Quick Panel. ModusToolbox™ also supports third-party IDEs, including Visual Studio Code, Arm® MDK (µVision), and IAR Embedded Workbench.

A high-level view of the tools/resources included in the ModusToolbox™ software is shown in Figure 2.
For more details on the ModusToolbox™, see the ModusToolbox™ software user guide.

Development ecosystem

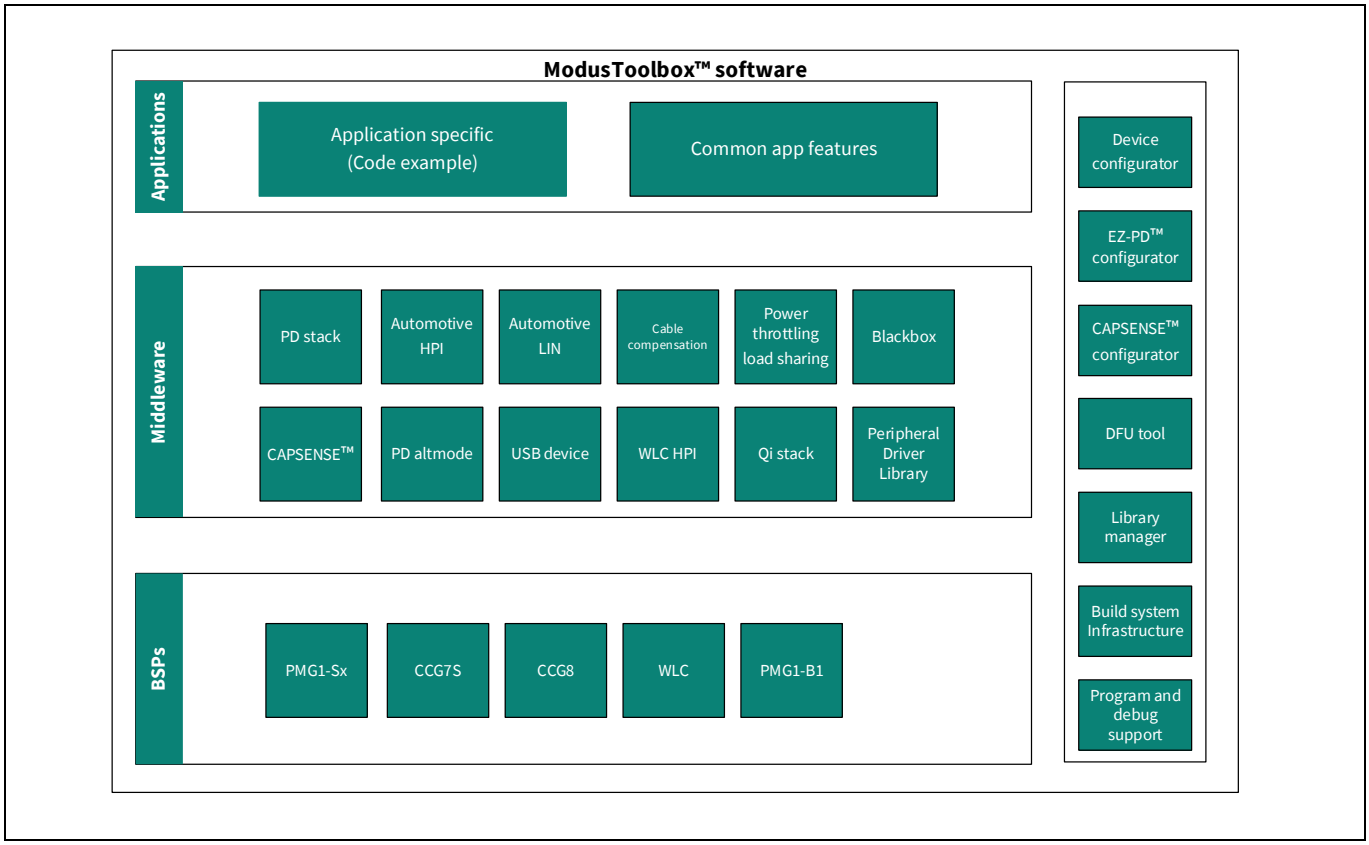


Figure 2 ModusToolbox™ software

The following the low-level resources are available for the PMG1 MCU.

1. **Board support packages (BSP):** A BSP is the layer of firmware containing board-specific drivers and other functions. The board support package is a set of libraries that provide APIs to initialize the board and provide access to board-level peripherals. It includes low-level resources such as Peripheral Driver Library (PDL) for PMG1 MCU and has macros for board peripherals. Custom BSPs can be created to enable support for end-application boards. See the **Board support packages** section in the ModusToolbox™ software user guide for more information.
2. **Peripheral Driver Library (PDL):** The PDL integrates device header files, start-up code, and peripheral drivers into a single package. The drivers abstract the hardware functions into a set of easy-to-use APIs. This reduces the need to understand register usage and bit structures, thus easing software development for the extensive set of peripherals in the PMG1 MCU series. Configure the driver for the application, and then use API calls to initialize and use the peripheral. For a PMG1 MCU solution, the on-chip blocks including USB PD and SCBs are controlled by the CAT2 PDL.
3. **Middleware:** The middleware is a set of firmware modules that provide specific capabilities to an application. PD stack is the middleware available for the PMG1 MCU, which contains the Type-C and USB PD state machine implementations, the PD protocol, and PD Policy Manager blocks.

All low-level resources are delivered as libraries via GitHub repositories.

Figure 3 shows the blocks of ModusToolbox™ software that are relevant for the PMG1 MCU.

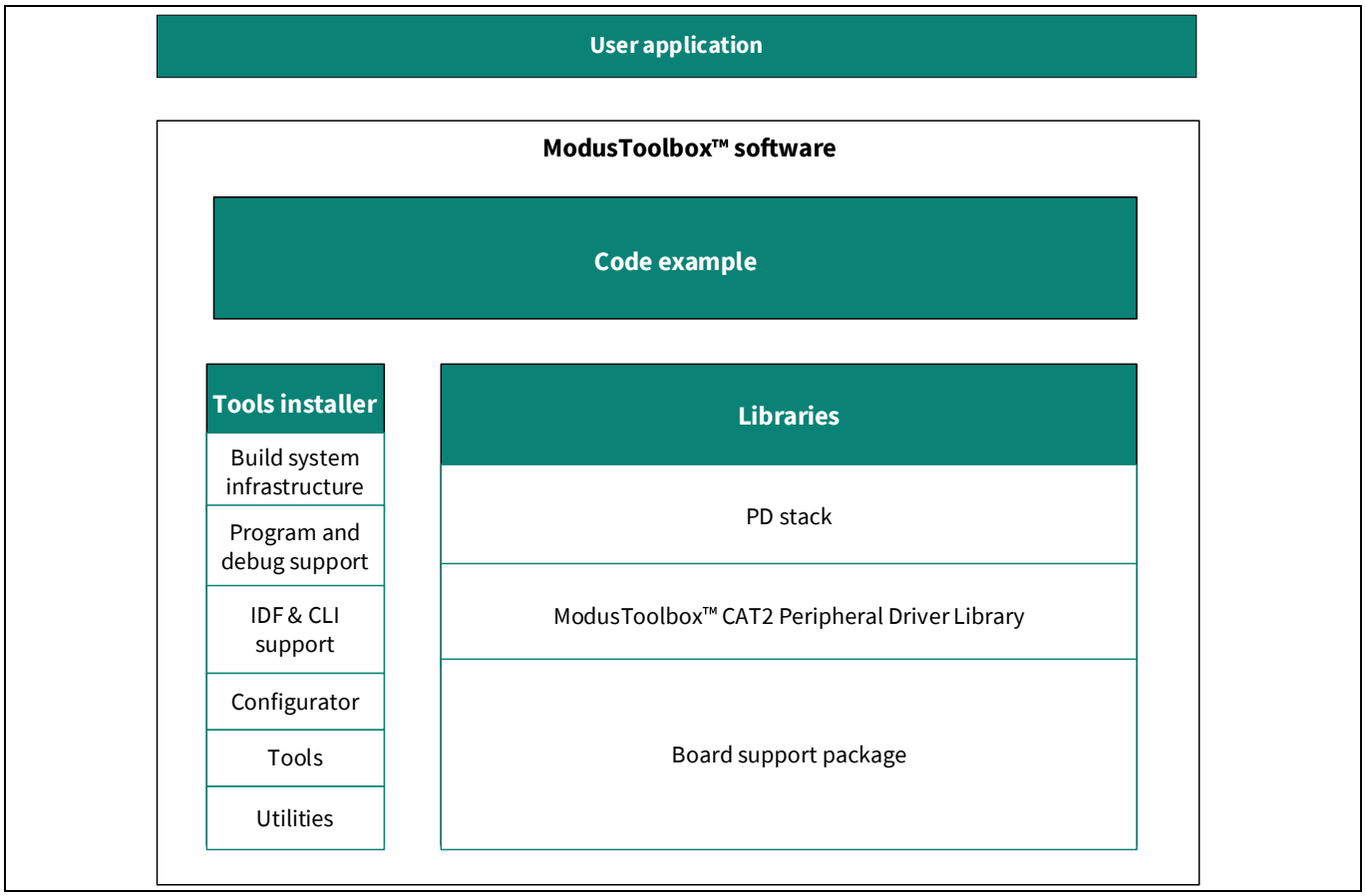


Figure 3 ModusToolbox™ software for PMG1 MCU

2.2.1 PMG1 MCU software resources

The PMG1 MCU software resources include configurators, drivers, libraries, and middleware, as well as Makefiles and scripts to get you started with developing firmware with PMG1 MCU.

2.2.1.1 Configurators

ModusToolbox™ software provides graphical applications called Configurators that make it easier to configure a hardware block or middleware. For example, instead of having to search through all the documentation to configure a serial communication block as a UART with a desired configuration, open the Configurator and set the baud rate, parity, and stop bits. Upon saving the hardware configuration, the tool generates the C code to initialize the hardware with the desired configuration.

Configurators are independent of each other, but they can be used together to provide flexible configuration options. They can be used standalone, with other tools, or within a complete IDE. Configurators are used for:

- Setting options and generating code to configure drivers
- Setting up connections such as pins and clocks for a peripheral
- Setting options and generating code to configure middleware

For PMG1 MCU applications, the following configurators are available:

- **Device Configurator:** Set up the system (platform) functions, as well as the basic peripherals (for example, UART, Timer, PWM).
- **EZ-PD™ Configurator:** EZ-PD™ Configurator provides a user-friendly tool for selecting the features of the PD stack middleware and configuring parameters to generate the required code.

These configurators create their own files (for example, *design.mtbezpd* for EZ-PD™ Configurator). The configurator file (*design.modus*) is usually provided with the BSP and *design.mtbezpd* is provided as part of the code example. When an application is created based on a BSP, the files are copied into the application. You can also create custom Device Configurator files for an application and override the ones provided by the BSP.

2.3 ModusToolbox™ software help

Visit the ModusToolbox™ software home page to download and install the latest version of ModusToolbox™ software. Launch Eclipse IDE for ModusToolbox™ software and navigate to the following items for ModusToolbox™ Help.

Choose **Help** > **ModusToolbox™ General Documentation** or **Help** > **Eclipse IDE for ModusToolbox™ Documentation**

2.4 Support for other IDEs

You can develop firmware for PMG1 MCUs using your preferred IDE such as IAR Embedded Workbench or Visual Studio Code in addition to the Eclipse IDE.

See the “Exporting to IDEs” section in the ModusToolbox™ software user guide for more details.

2.5 Programming and debugging

The Eclipse IDE of ModusToolbox™ software supports KitProg3 and uses the Open on-chip debugger (OpenOCD) protocol for debugging PMG1 MCU applications. It also supports GNU debugger (GDB) debugging using industry-standard probes like the Segger J-Link.

All PMG1 MCU kits have a KitProg3 onboard programmer/debugger. It supports Cortex® Microcontroller Software Interface Standard - Debug Access Port (CMSIS-DAP). See the KitProg3 user guide for details.

For more information on programming/debugging firmware on PMG1 devices with ModusToolbox™ software, see the “Program and Debug Support” section in the ModusToolbox™ software user guide.

2.6 PMG1 MCU kits

Table 2 BSPs for PMG1 MCU Kits

PMG1 MCU	PMG1 MCU Kit	BSP
CYPM1116-48LQXI	EVAL_PMG1_B1_DRP Kit	EVAL_PMG1_B1_DRP
CYPM1011-24LQXI	CY7110 EZ-PD™ PMG1-S0 Prototyping Kit	PMG1-CY7110
CYPM1111-40LQXI	CY7111 EZ-PD™ PMG1-S1 Prototyping Kit	PMG1-CY7111
CYPM1211-40LQXI	CY7112 EZ-PD™ PMG1-S2 Prototyping Kit	PMG1-CY7112
CYPM1311-48LQXI	CY7113 EZ-PD™ PMG1-S3 Prototyping Kit	PMG1-CY7113
CYPM1322-97BZXI	NA	CYPM1322-97BZXI

3 Application development on the EVAL_PMG1_B1_DRP kit using ModusToolbox™

3.1 PMG1-B1 kit applications

The following PMG1-B1 Kit applications are available in ModusToolbox™ to exercise the kit functionality and as a reference for new application development.

Table 3 Example projects in ModusToolbox™ software

Example project	Description
USB PD sink only role	The buck-boost converter is in the sink path and converts VBUS to the required output voltage. The default voltage is 20 V.
USB PD DRP role	Buck-boost is used in the sink path, and an external buck load switch in the source path sourcing 5 V and 9 V at 3 A.
Battery charging in USB PD sink only role	Battery charging code configurable to 2-5 cell battery

A detailed explanation of the code examples is available in the ModusToolbox™ software. For the hardware setup of the EVAL_PMG1-B1_DRP Kit, refer to the kit's user guide on the PMG1-B1 webpage.

3.2 Using the kit code example

1. Open Eclipse IDE for ModusToolbox™.
2. Navigate to **Quick Panel** and click **New Application** under **Start.**

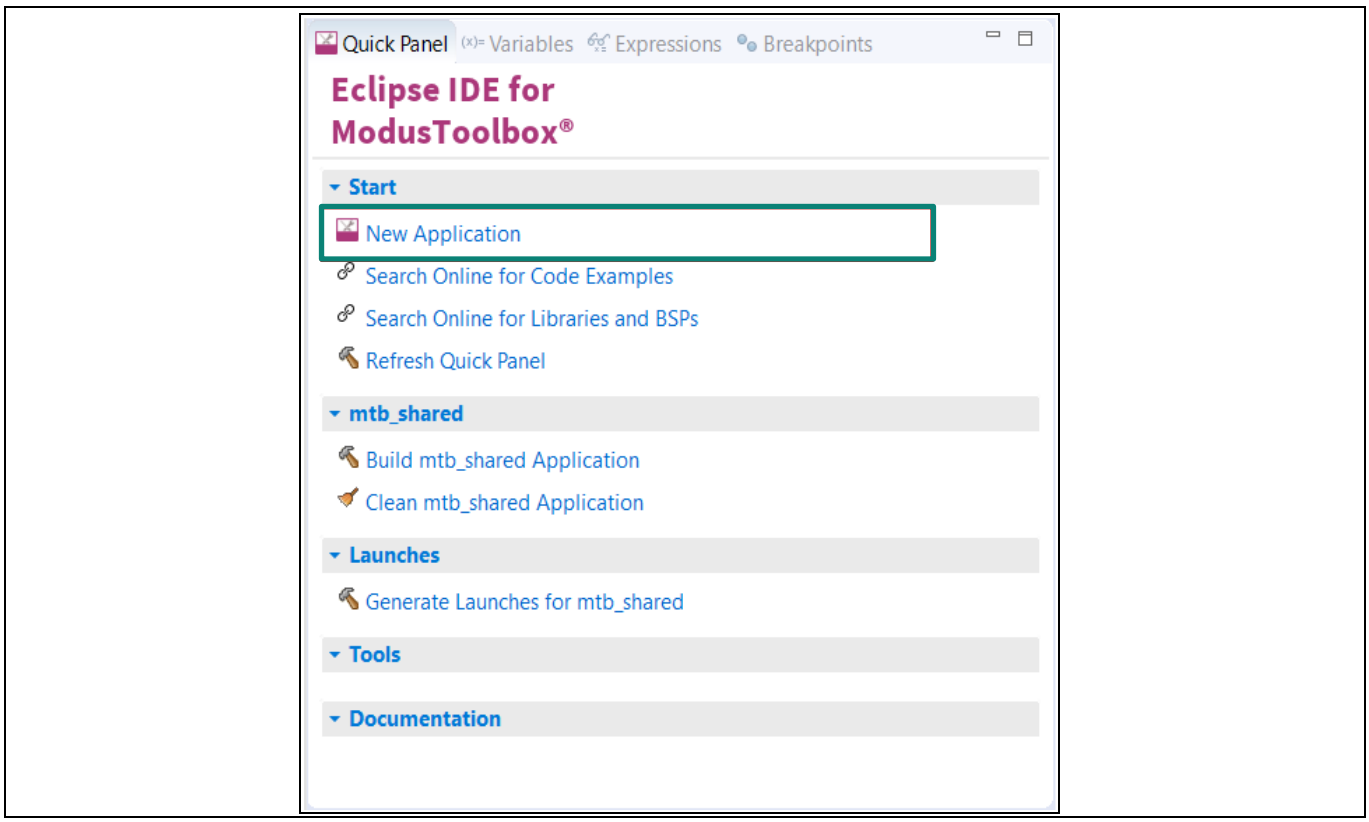


Figure 4 Creating a new application

3. In the Project Creator window, choose the board support package (BSP) from the list. Click **Next**.

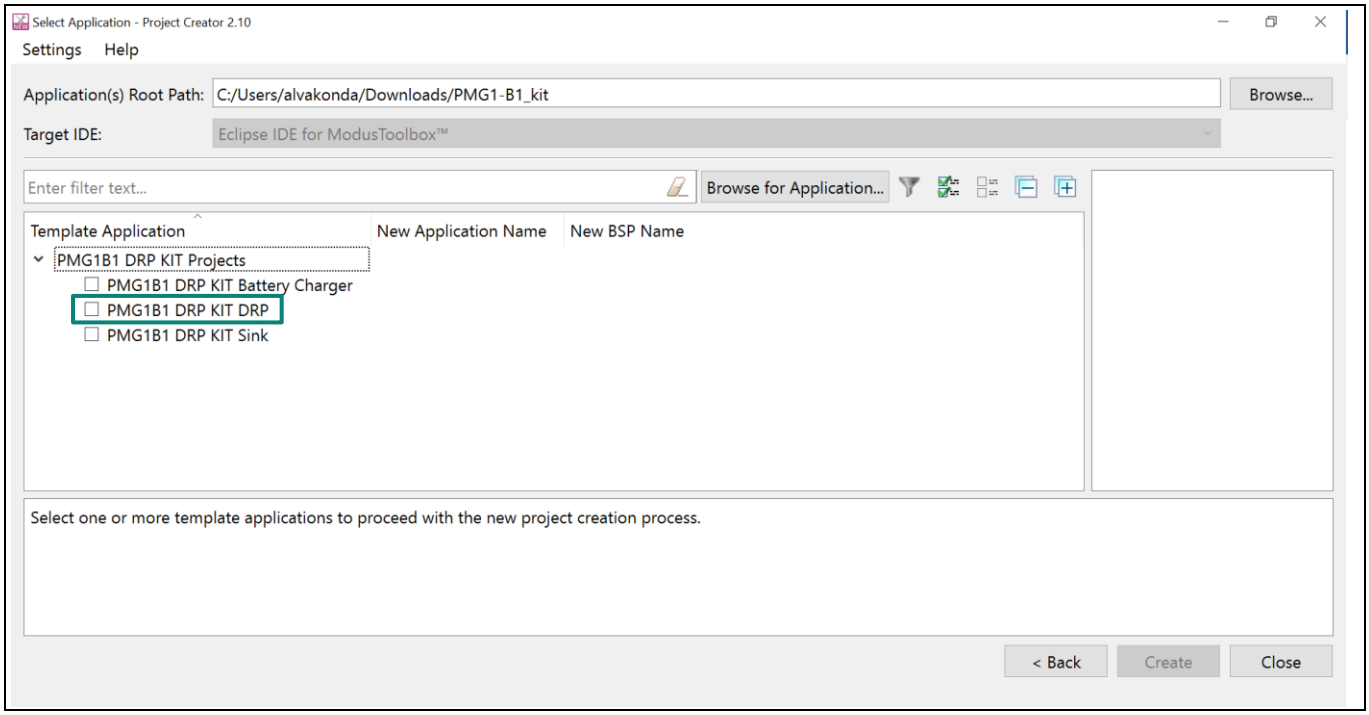


Figure 5 Selecting the BSP

4. Select the desired code example. Enter the **New Application Name** if you want to change it from the default. Click **Create**.

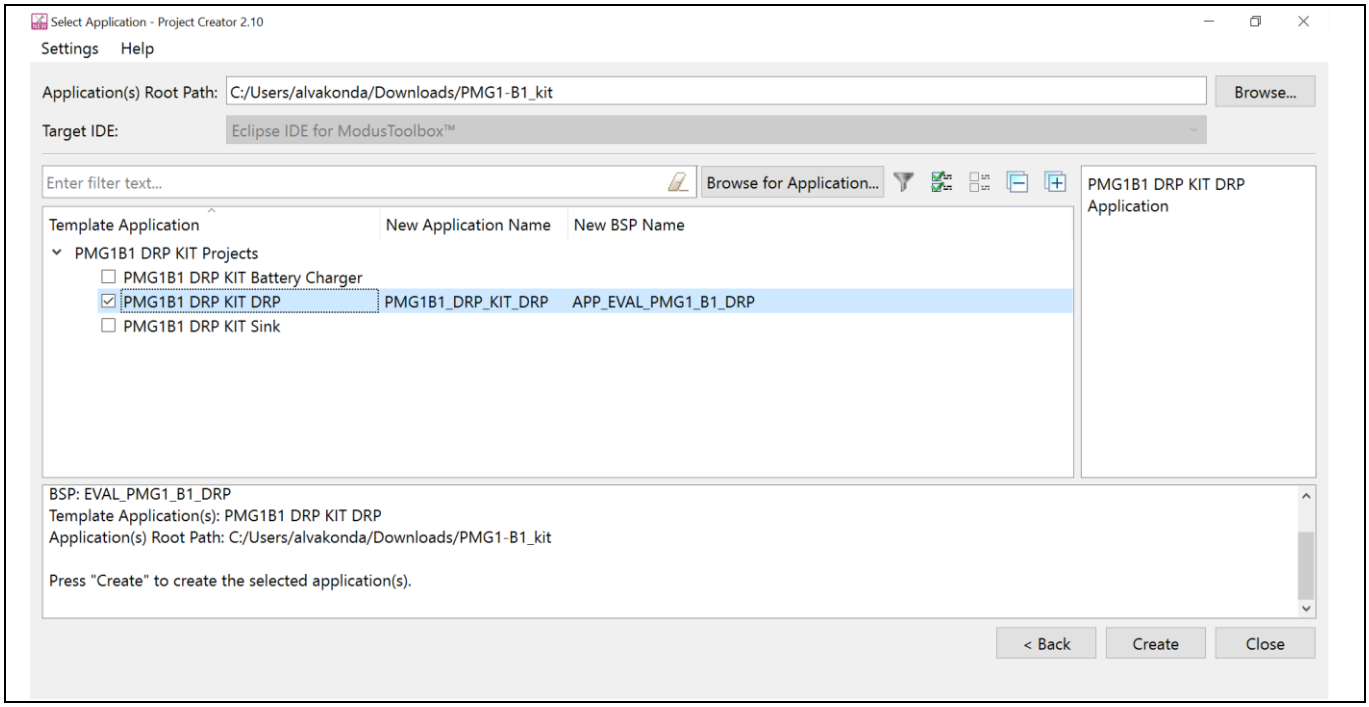


Figure 6 Selecting the application

ModusToolbox™ software downloads the resources required from GitHub for creating the project and automatically closes the project creator window when the project creation is completed.

Note: *Ensure that your computer is connected to the internet to access the GitHub repository.*

5. Click Create and wait for the Project Creator to automatically close once the project is successfully created.

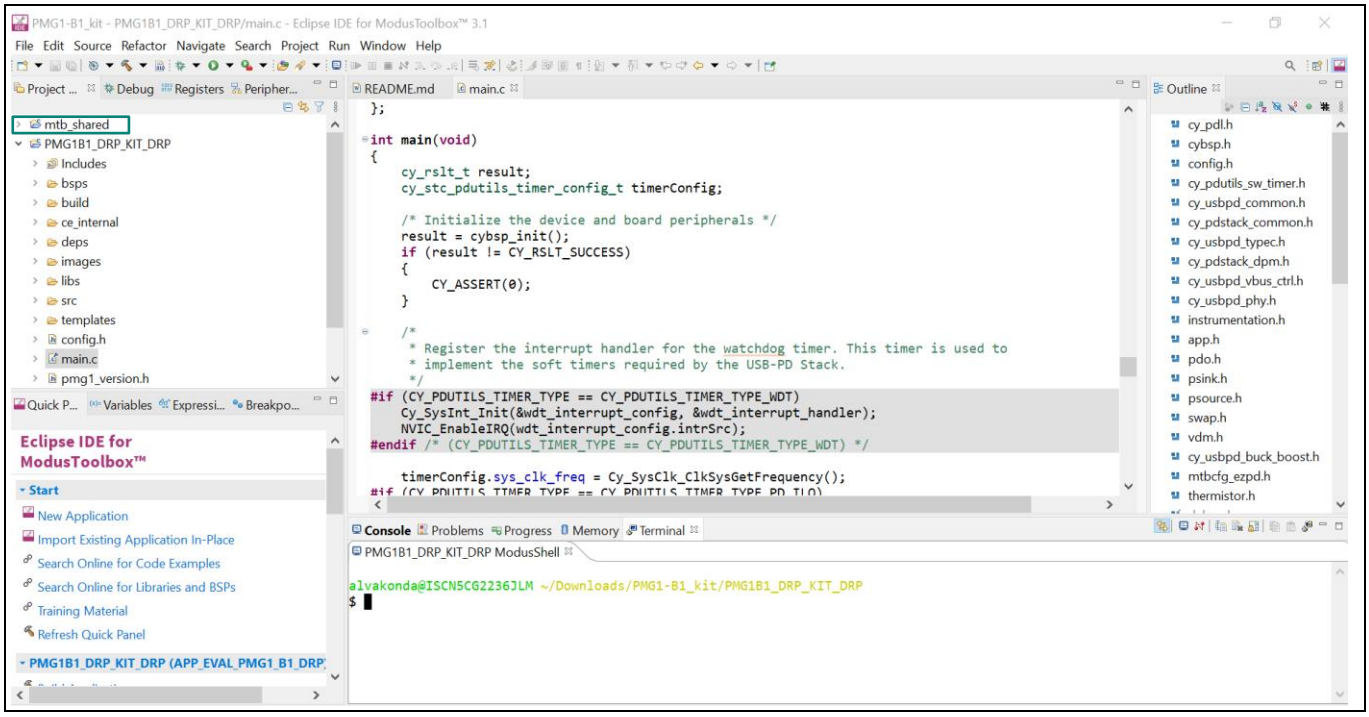


Figure 7 Application source files

By default, when creating a new application or adding a library to an existing application and specifying it as shared, all libraries are placed in an *mtb_shared* directory adjacent to the application directory.

The *mtb_shared* folder shown in Figure 7 is shared between different applications that use the same versions of BSP/library.

See the ModusToolbox™ software user guide to learn more about the procedure to create new or modify existing applications, to configure, and build applications.

6. Building the application

   Select the application project in the Project Explorer window and click on the **Build <name> Application** shortcut under the <name> group in the Quick Panel. It selects the Debug build configuration and compiles/links all projects that constitute the application.

The console view lists the results of the build operation:

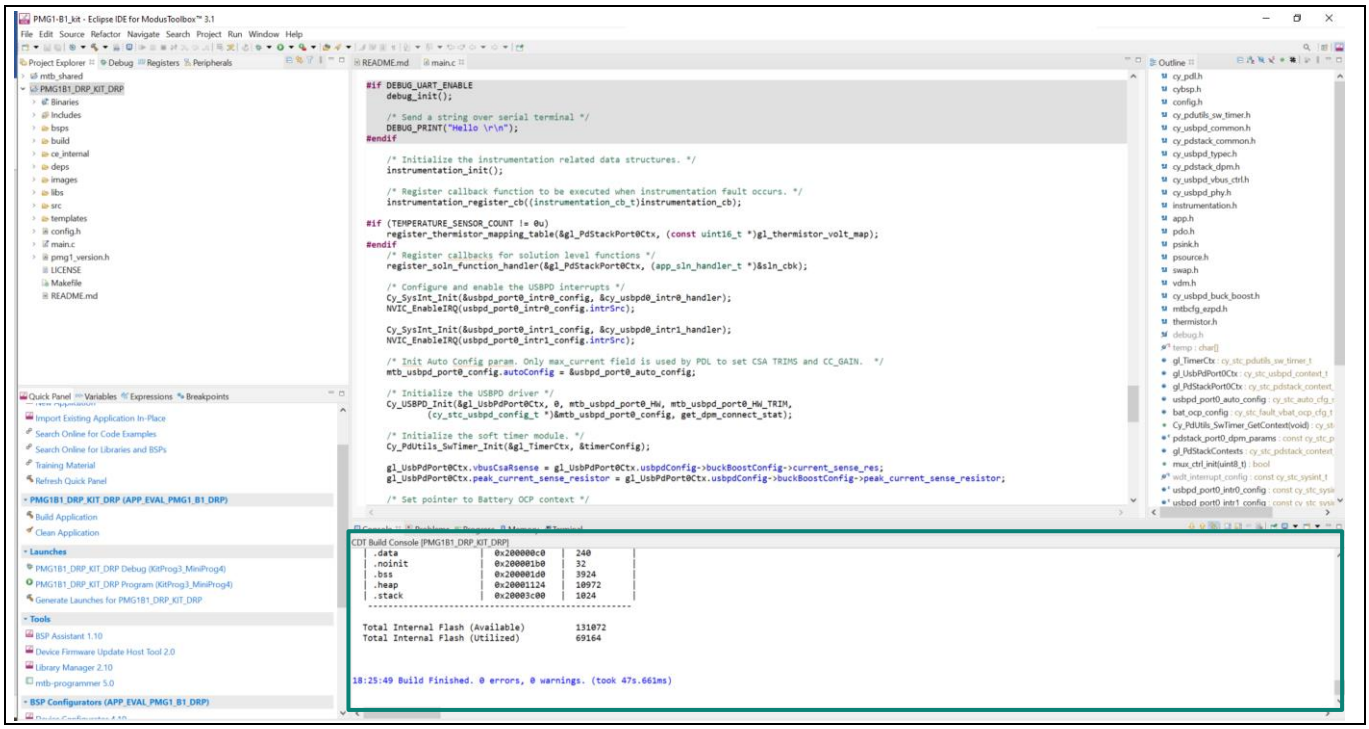


Figure 8 Build the application

If you encounter errors, try the following options:

- Revisit earlier steps to ensure that you accomplished all the required tasks.
- Check the **Problems** tab to find more details about the errors and to solve them.
- From Quick Panel, click 'Clean <application name>' and try to build the project again.

*Note: You can also use the command-line interface (CLI) to build the application. See the Using the Command-Line section in the ModusToolbox™ software user guide. This document can also be accessed in **ModusToolbox™ software** via the help menu > **ModusToolbox™ General Documentation**.*

3.3 Programming and debugging ModusToolbox™ software

KitProg3 is a PSoC™ 5LP MCU-based onboard programming and debugging solution integrated with the EZ-PD™ PMG1 MCU Prototyping Kits. You can program and debug the kit without using a separate programming and debugging module by doing the following:

Connect the kit to the computer using the programming connecter as shown in Figure 9.

The power cycle mode of KitProg3 is recommended for acquiring the EVAL_PMG1_B1_DRP kit for programming/debugging. In this mode, the device is acquired by toggling the target (EZ-PD™ PMG1-B1 MCU) power in a sequence.

The KitProg3 programmer chooses the acquisition mode depending on the target MCU and the prototyping kit.

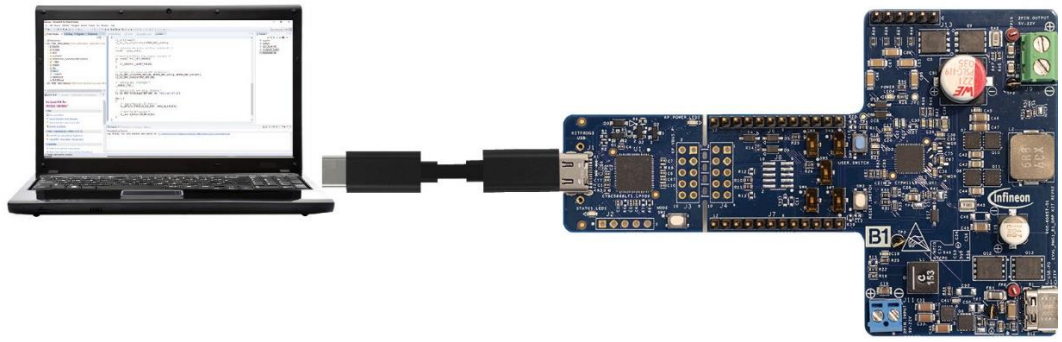


Figure 9 Programming through the KitProg3 interface

Do the following to program the prototyping kit using the KitProg3 interface from the ModusToolbox™ software:

1. Open Eclipse IDE for ModusToolbox™ on the host PC.
2. Connect the kit to the host PC through the KitProg3 USB Type-C port (J1).
3. Ensure that the LED1 and LED2 glow in amber color.
   LED2 (KitProg3 Power LED) indicates that the KitProg3 module and target MCU are powered.
   LED1 (status LED) indicates the programming mode and status, and is ON when KitProg3 is powered.
4. In Project Explorer, select the correct project as the currently active project as the follows:

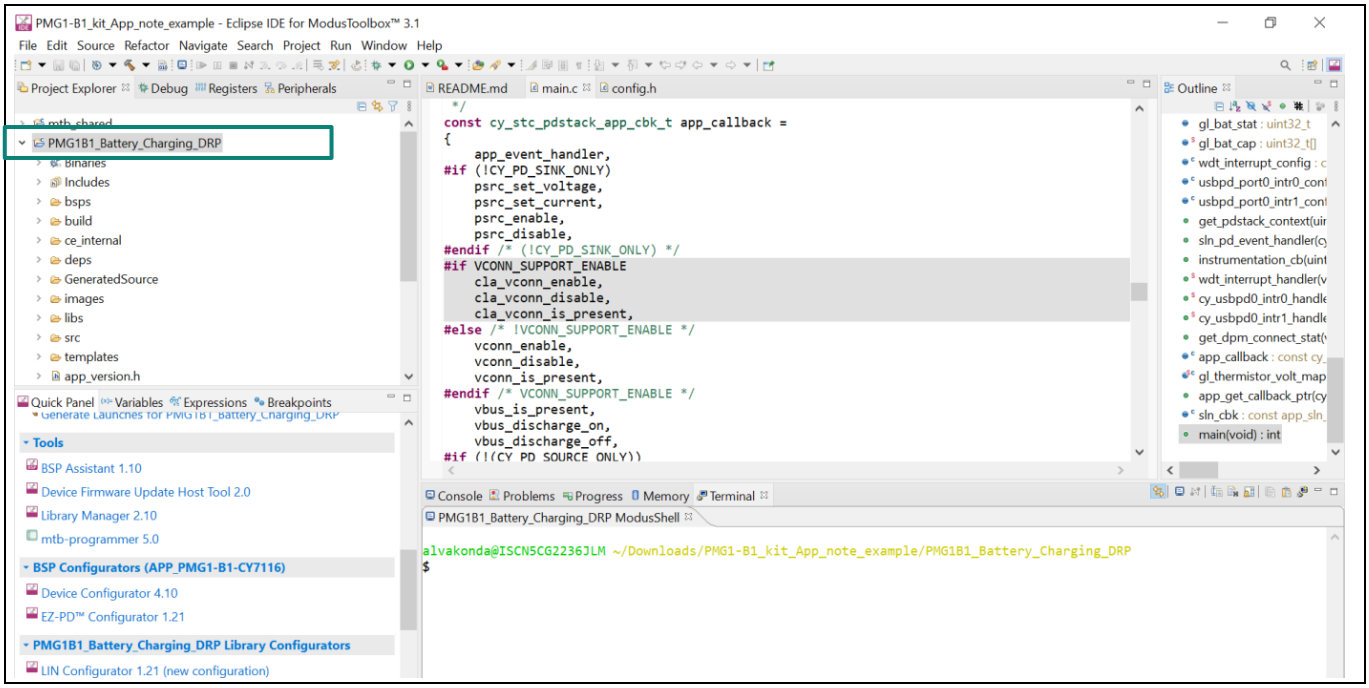


Figure 10 Eclipse IDE for ModusToolbox™ software

5. On the Quick Panel tab, click <Application name> Program (KitProg3_MiniProg4) from the Launches section.

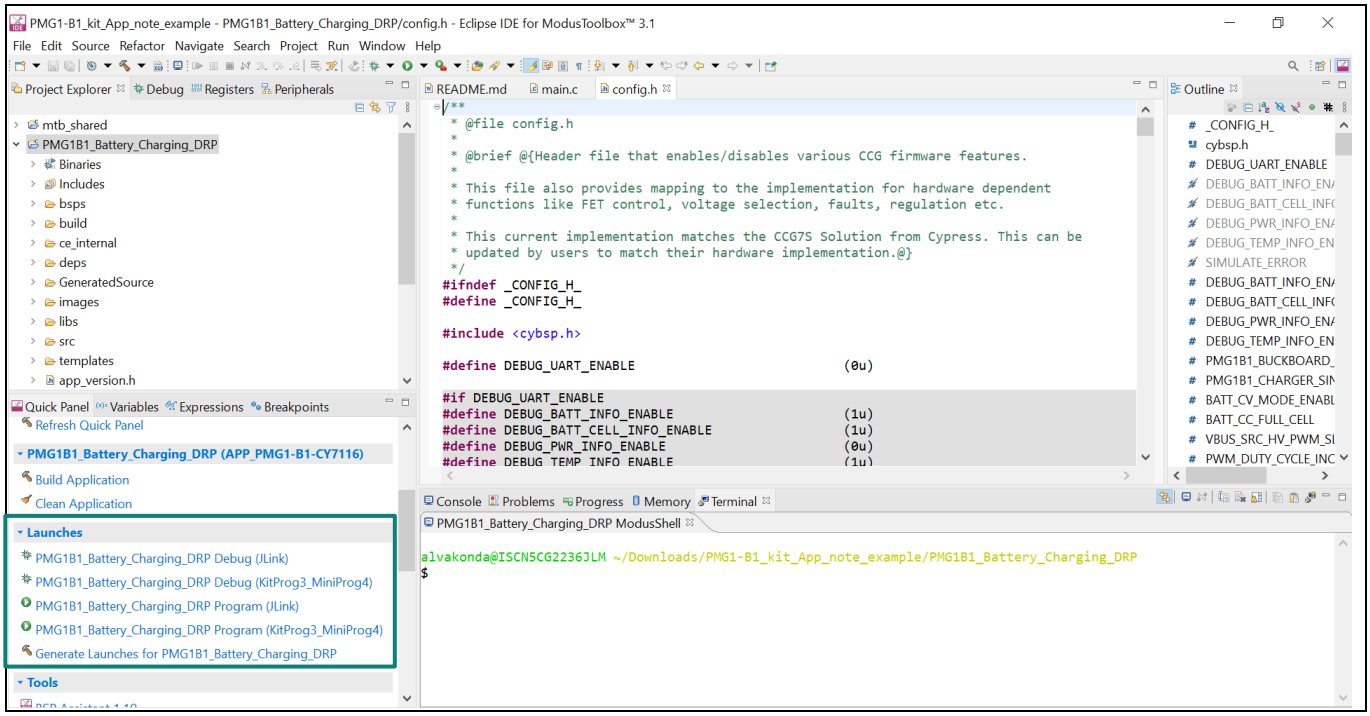


Figure 11 Quick Panel and programming options

The Console tab displays the progress of the build/program. You can check the Problems tab for any errors and warnings. The success message is displayed on the Console tab if the programming is successfully completed.

3.3.1 Programming the PMG1-B1 device

Download and install the ModusToolbox™ Programming Package and do the following to program the PMG1-B1 MCU.

3.3.1.1 Load the programming file

1. Connect the device to the host computer. Select the device name in the **Probe/Kit** drop-down.

The ModusToolbox™ Programming Package displays information under **Probe Settings** (if the **Settings** section is viewable).

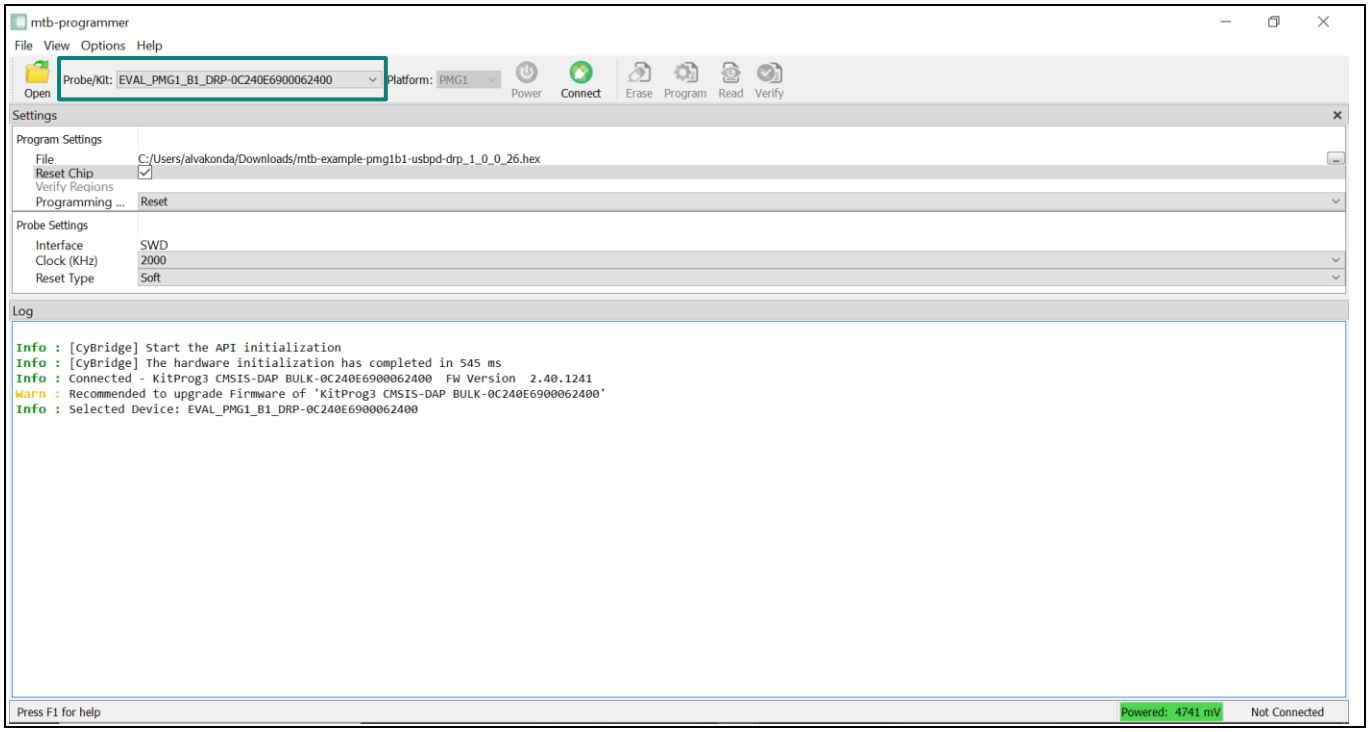


Figure 12 Probe/Kit setting

2. Click **Open**.
3. On the **Open Programming File** dialog box, navigate to the HEX file location, select it, and click **Open**.

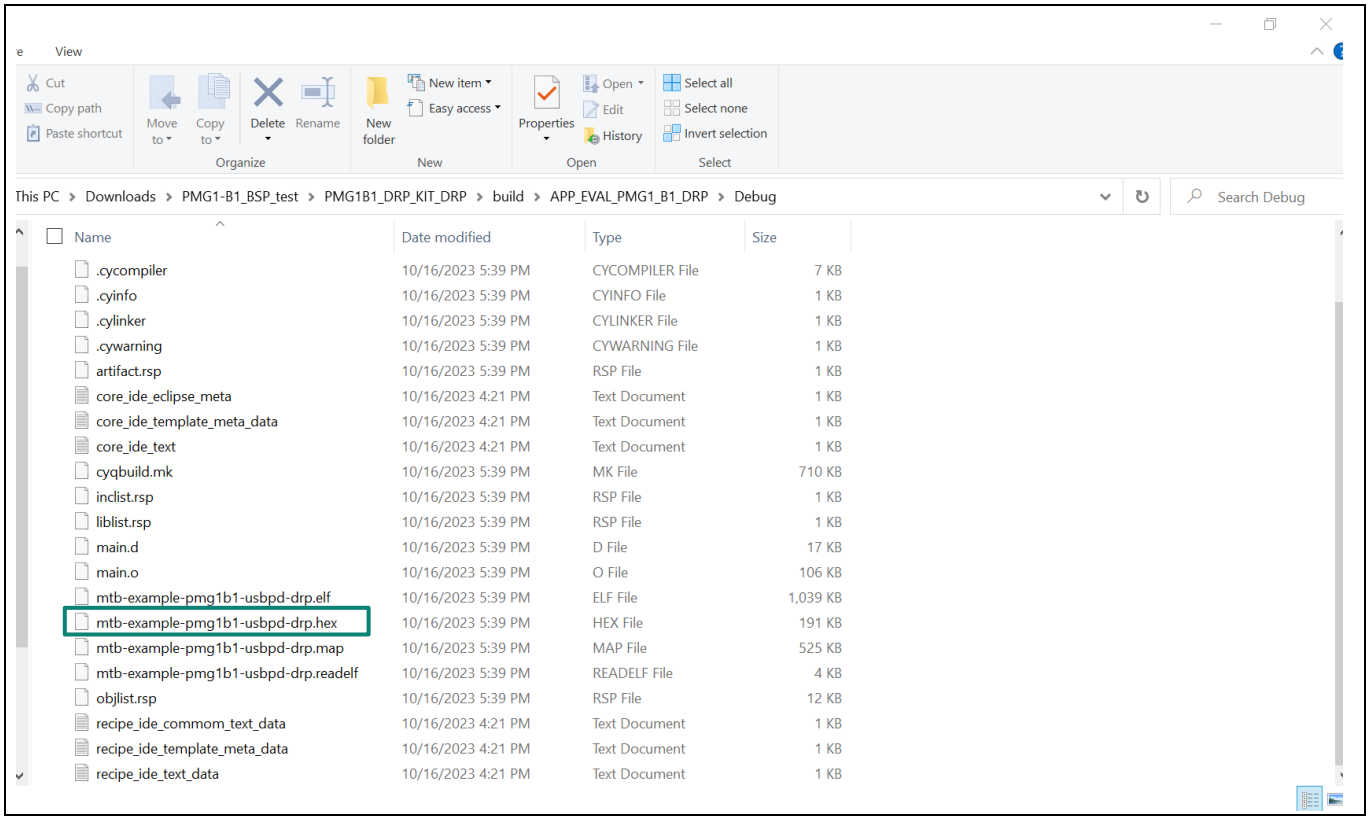


Figure 13 Select the HEX file

3.3.1.2 Connect the device

1. If the device is not powered, the status message “Not Powered” is displayed in the status bar. Click **Power** to power up the device and then.
2. Click **Connect**.
3. ModusToolbox™ Programming Package communicates with the device and displays various messages in the Log window. Then, a message in the status bar indicates that it is connected as shown in Figure 14.

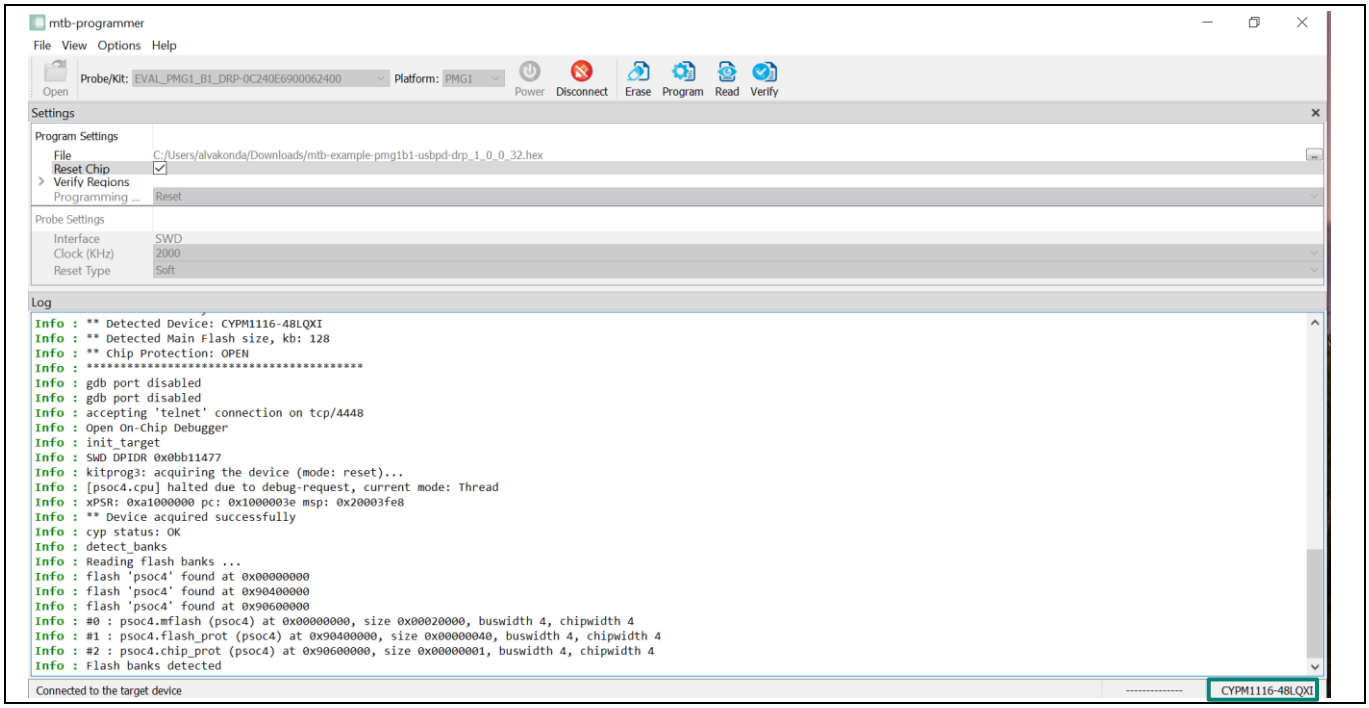


Figure 14 Connect

3.3.1.3 Program the device

Click **Program**. The programmer downloads the program file on to the device and displays the messages in **Log**.

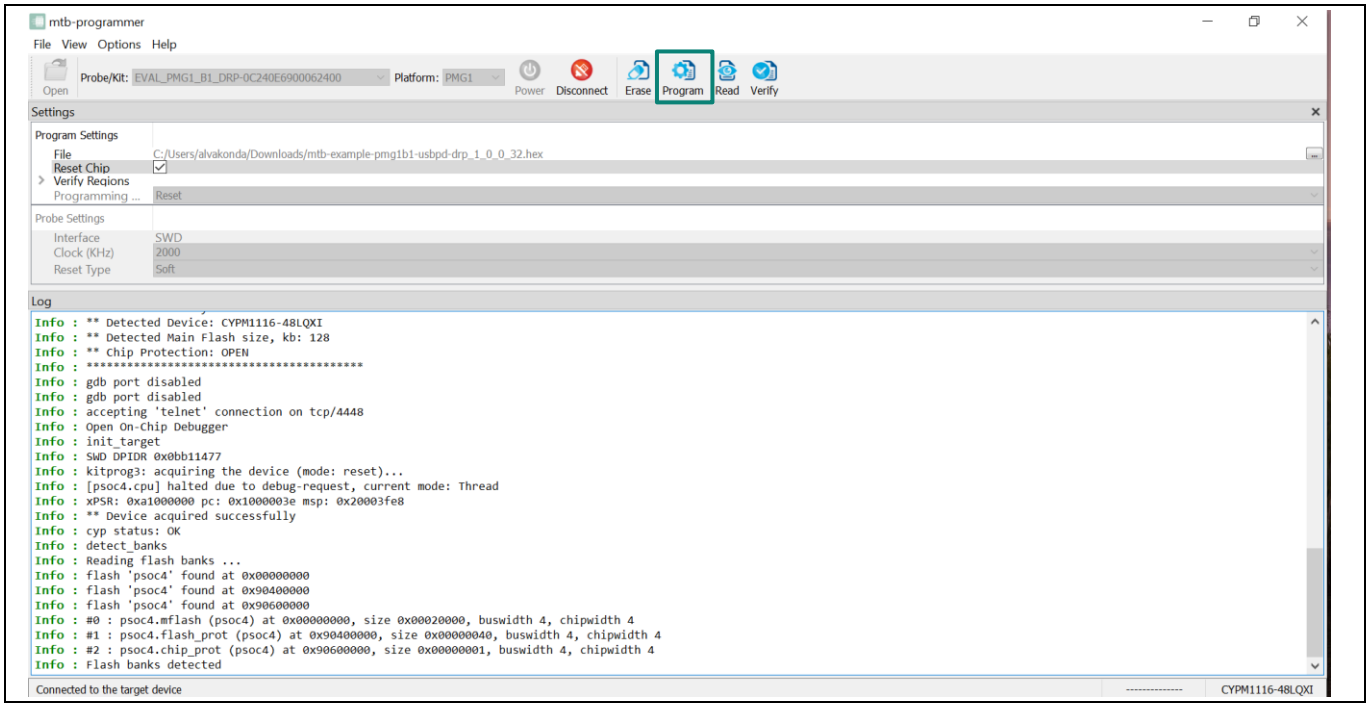


Figure 15 Program

3.3.2 Debugging using ModusToolbox™ software

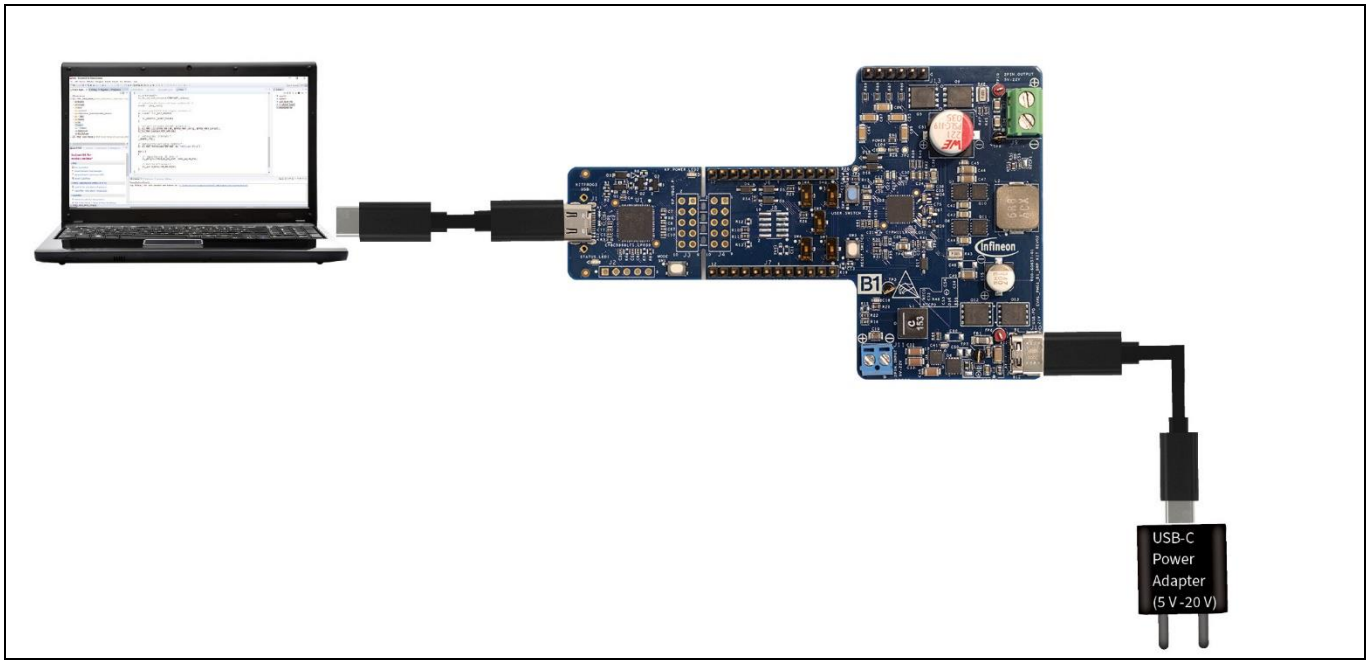


Figure 16 Debugging using EVAL_PMG1_B1_DRP Kit

Do the following to debug an application on the EVAL_PMG1_B1_DRP Kit over the KitProg3 interface using the Eclipse IDE for ModusToolbox™ software:

1. Open the Eclipse IDE for ModusToolbox™ on the host PC and select the PMG1 project. For details, see the Eclipse IDE for ModusToolbox™ user guide.
2. On the **Quick Panel**, click **<Application name> Debug (KitProg3_MiniProg4)** from the **Launches** section.

The IDE will switch to debugging mode and will halt at the first line of the main function. This indicates that the application is ready for debugging.

1. Select **Run** > **Debug configurations**, as shown in Figure 16.
2. Under GDB OpenOCD Debugging, select **<Application name> Attach (KitProg3_MiniProg4)** and click on **Debug**.

This will start a debugging session, attaching to a running target without programming or reset. For more information on debug configurations, see the ModusToolbox™ user guide.

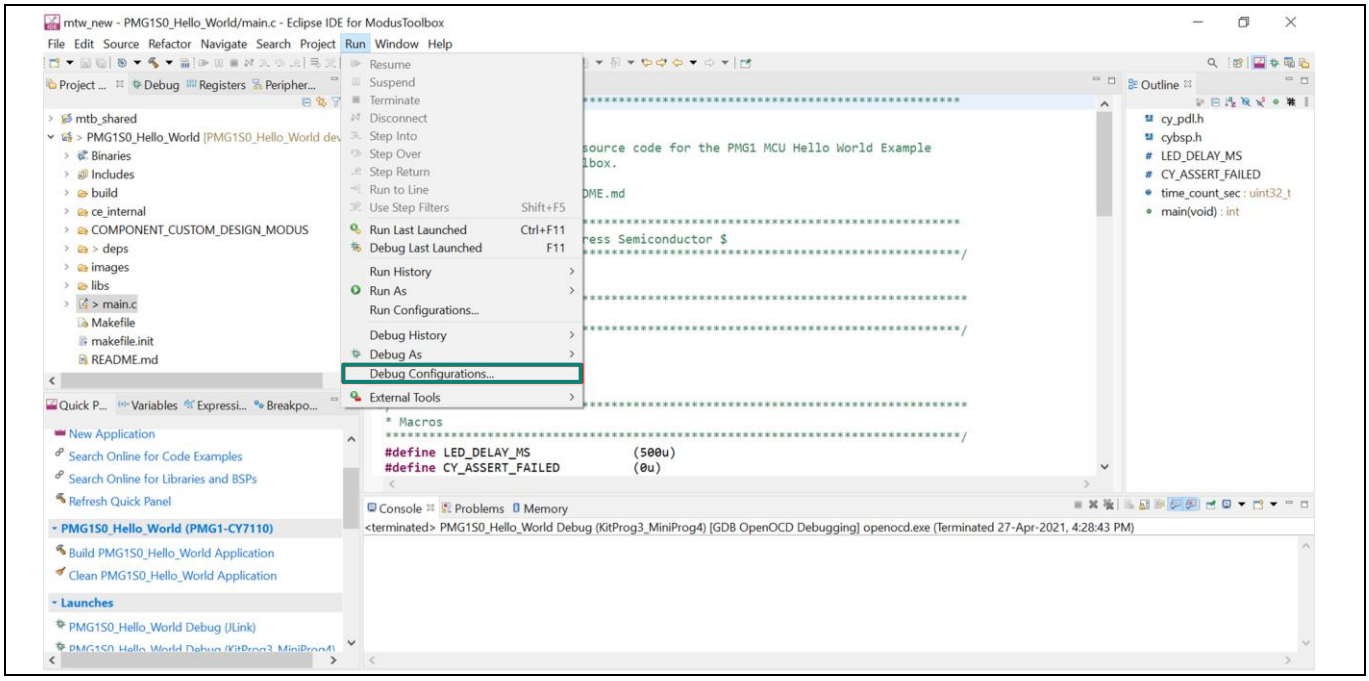


Figure 17 Selecting debug configurations from the Run menu

3.4 Configuring the PDOs, PD, and buck-boost parameters

The EZ-PD™ Configurator tool, integrated with the Eclipse IDE for ModusToolbox™ software, provides a tool for selecting and configuring the parameters of the PD stack. Do the following to configure PDOs and PD parameters using the EZ-PD™ Configurator Tool.

1. Open Eclipse IDE for ModusToolbox™ software on the host PC and select the **PMG1** project.
2. Open EZ-PD™ Configurator 1.21 from the **BSP Configurators** menu in the Quick Panel.

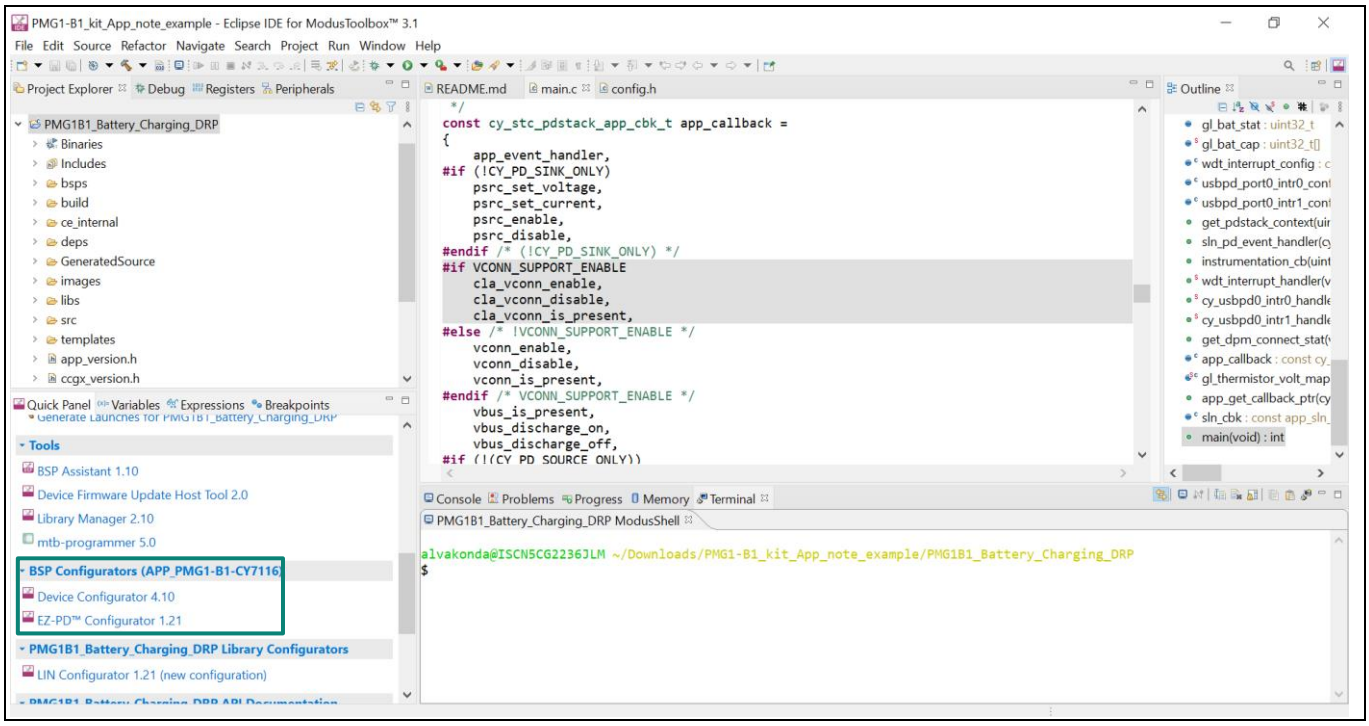


Figure 18 Launching the EZ-PD™ Configurator tool

3. On the EZ-PD™ Configurator tool window, click on the category name to expand.

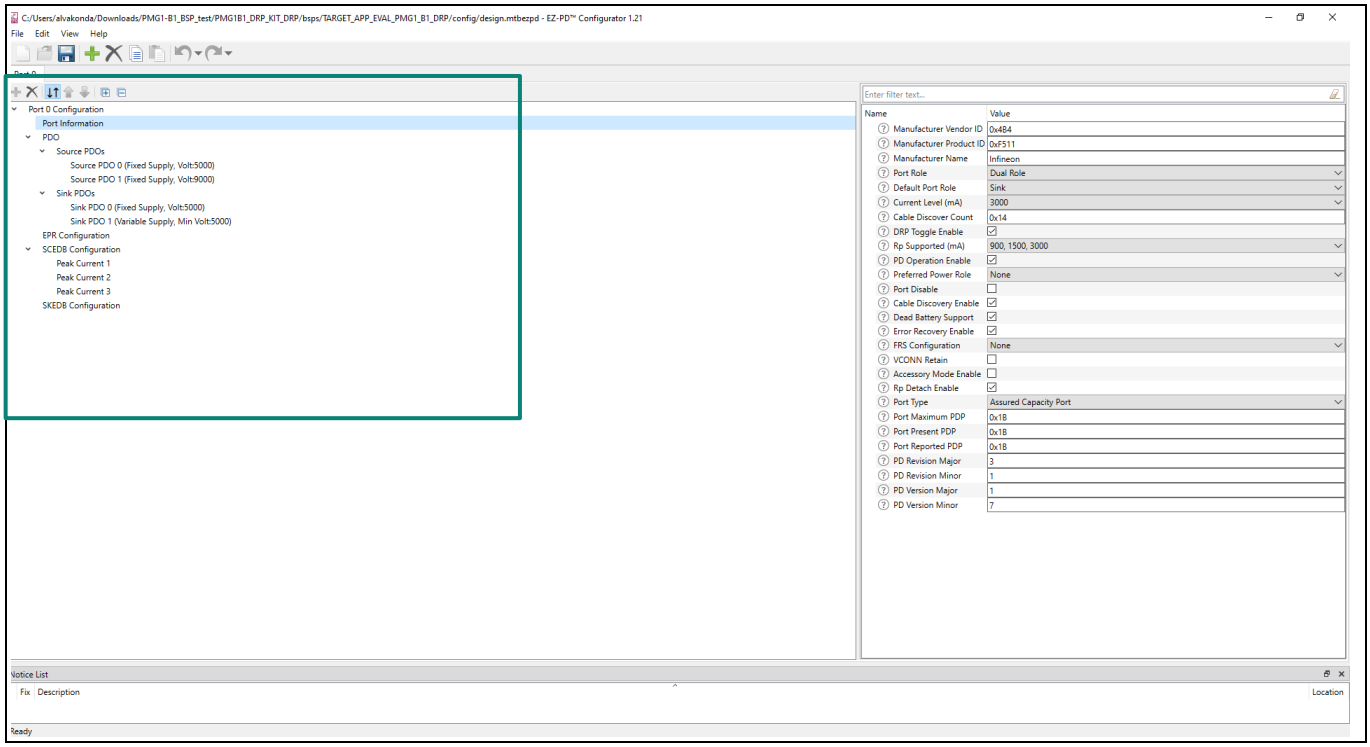


Figure 19 EZ-PD™ Configurator and list of categories

You can configure the port Information, source PDO, sink PDO, SCEDB configuration, and SKEDB configuration parameter categories in the tool. For more details, see the ModusToolbox™ EZ-PD™ Configurator guide.

*Note: To open the ModusToolbox™ software EZ-PD™ Configurator guide, press **F1** or click **Help** > **View Help**.*

4. Select Sink PDO in the parameter category to configure parameters related to the sink PDO.

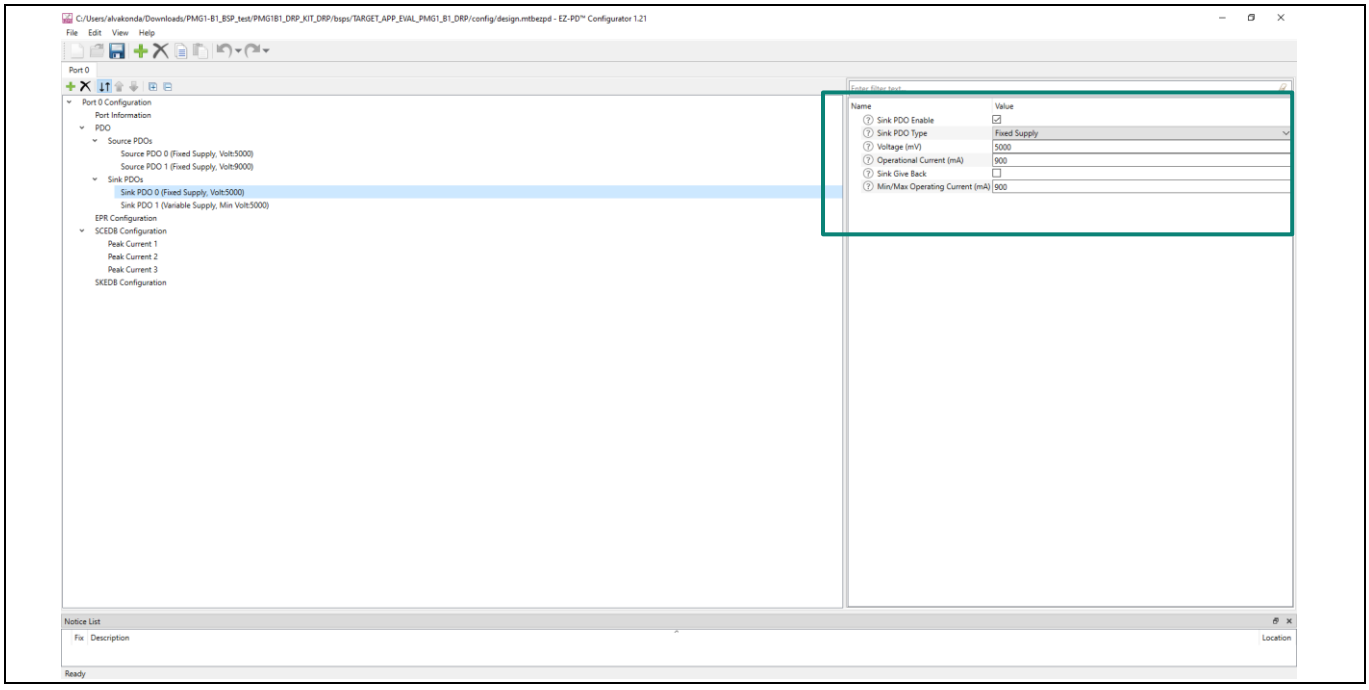


Figure 20 Sink PDO parameters

- Sink PDO count: Select the number of sink PDOs (the maximum is 7).
- Sink PDO <PDO number> enable: Enables a specific PDO.
- Sink PDO <PDO number>: Configures the PDO value.
- Sink PDO min/max current: Configures the min/max PDO current.

Note: See section 6.4.1.3 of the USB Power Delivery specification 3.1 to understand how to configure PDO value and min/max current.

5. Click **Save** to save the configuration and close the EZ-PD™ Configurator tool. Recompile the project in Eclipse IDE of ModusToolbox™.

   The Device Configurator tool in ModusToolbox™ helps you to configure hardware blocks or middleware libraries. The configurator is available as a GUI and command-line interface (CLI) tool. If you use the Eclipse IDE provided with ModusToolbox™, you can access the Device Configurator tool from the IDE's Quick Panel.
6. Click on **Device Configurator** in Quick Panel to view and modify the design configuration. The configuration files are in the *COMPONENT_BSP_DESIGN_MODUS* folder. However, note that if you upgrade the BSP library to a newer version, the manual edits done to the *design.x* files are lost.

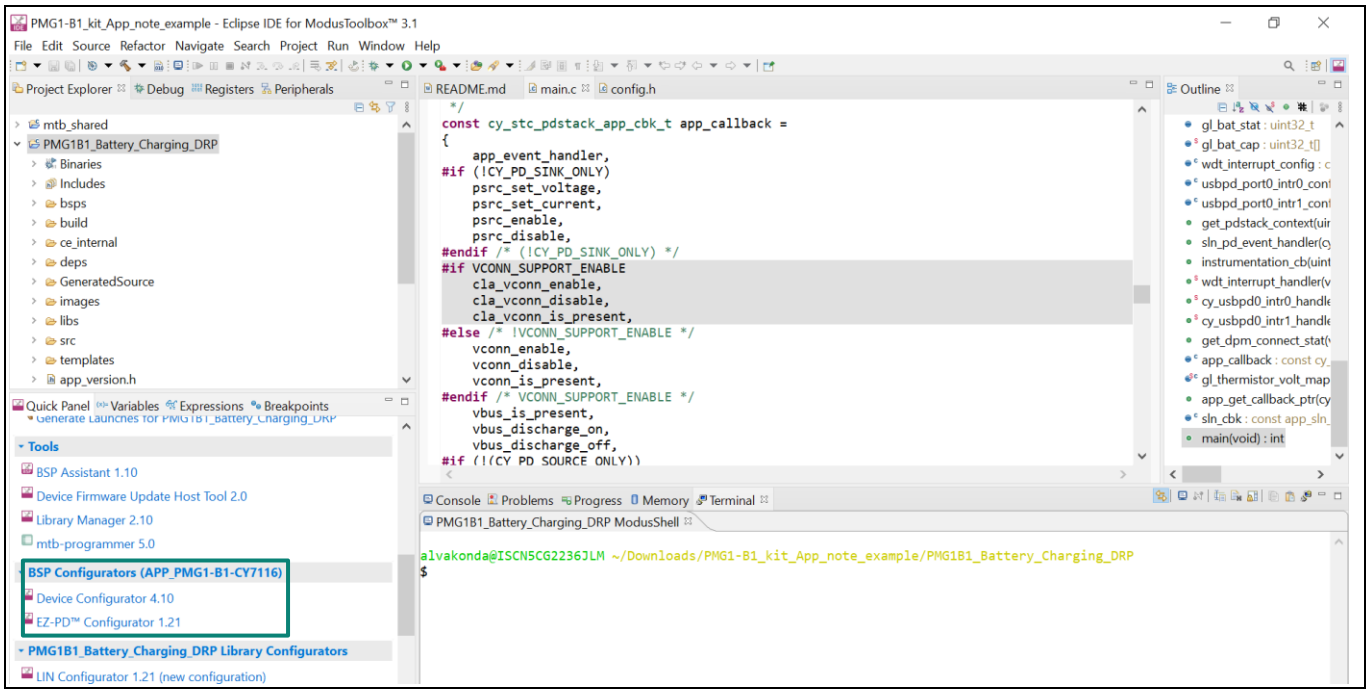


Figure 21 Open Device configurator

7. The **Device Configurator** window has a **Resources Categories** pane, where you can choose between different resources available in the device such as peripherals, pins, and clocks from the **List of Resources**:

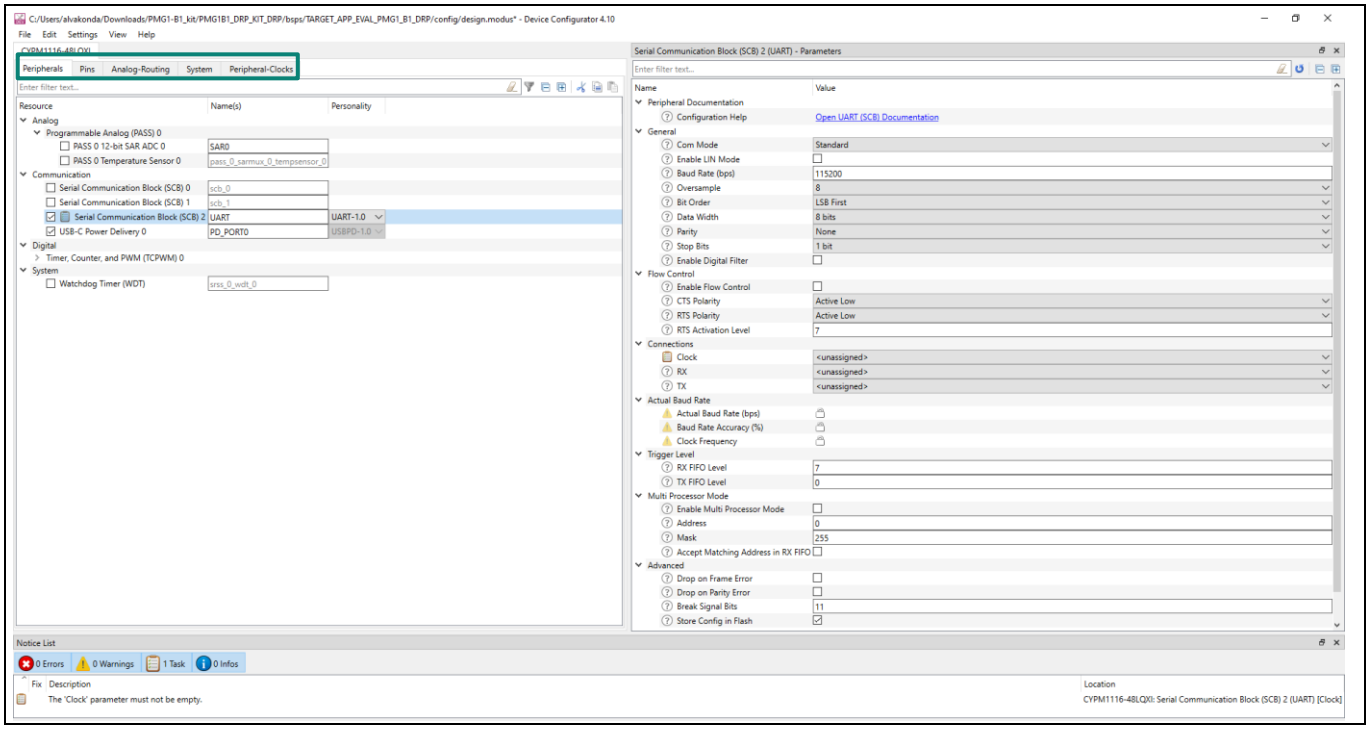


Figure 22 Device Configurator

You can choose how a resource behaves by choosing a Personality for the resource. For example, a Serial Communication Block (SCB) resource can have a EZI2C, I2C, SPI, or UART personalities. The “Name” is your name for the resource, which is used in firmware development. One or more names can be specified by using a comma to separate them (with no spaces).

8. Select **USB-C Power Delivery** in the peripherals category to configure parameters related to the buck-boost.

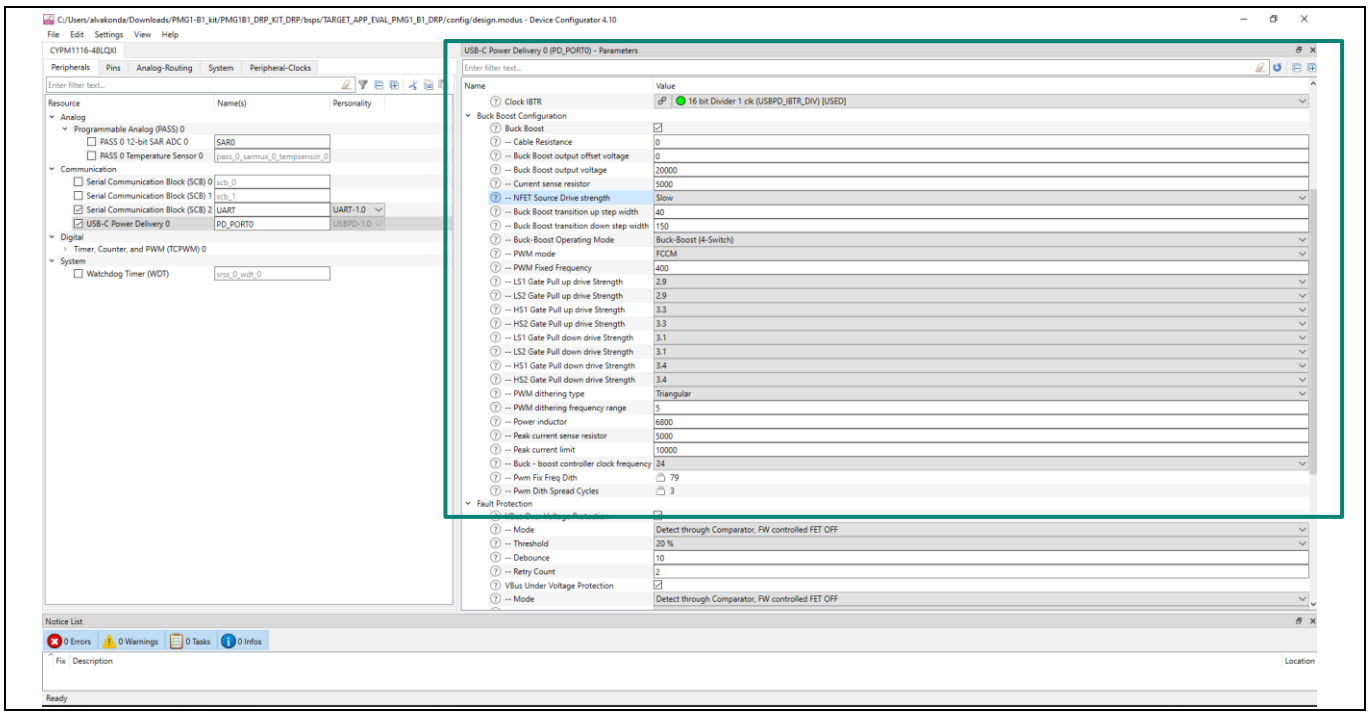


Figure 23 Buck-boost parameters

After updating the configuration using the Device Configurator, save and compile. Now, program PMG1-B1 as described in the **Programming PMG1-B1** device section.

4 Summary

This application note explored the PMG1-B1 MCU device architecture and the associated development tools. PMG1-B1 MCU is a programmable embedded system-on-chip with configurable analog and digital peripheral functions.

References

For a complete and updated list of PMG1 MCU code examples, visit Infineon's GitHub repository.
For more PMG1 MCU-related documents, visit the PMG1 MCU product web page.

Acronyms and abbreviations

Table 4 Acronyms and abbreviations

Acronym	Expansion
AC	Apple Charging
AES	advanced encryption standard
AFC	adaptive fast charging
BC	battery charging
CDM	charged device model
CDM	charge device model
CRC	cyclic redundancy check
ESD	electro-static discharge
HBM	human body model
HSDR	high side driver
LSDR	low side driver
OCP	overcurrent protection
OVP	overvoltage protection
OVT	overvoltage-tolerant
PRNG	pseudo-random number generation
RCP	reverse current protection. Supported in source configuration only.
SAR	successive approximation register
SCP	short circuit protection. Supported in source configuration only.
SHA	secure hash algorithm
TRNG	true random number generation
UVP	undervoltage protection

Glossary

This section lists the most commonly used terms that you might encounter while working with the PMG1 MCU family of devices in ModusToolbox™ software.

Board support package (BSP): A BSP is the layer of firmware containing board-specific drivers and other functions. The board support package is a set of libraries that provide firmware APIs to initialize the board and provide access to board level peripherals.

KitProg: The KitProg is an onboard programmer/debugger with USB-I2C and USB-UART bridge functionality. The KitProg is integrated onto most PMG1 MCU kits.

MiniProg3/4: Programming hardware for development that is used to program PMG1 MCU devices on your custom board or PMG1 MCU kits that do not support a built-in programmer.

Personality: A personality expresses the configurability of a resource for a functionality. For example, the SCB resource can be configured to be an UART, SPI or I2C personality.

Middleware: Middleware is a set of firmware modules that provide specific capabilities to an application. This provides high-level software interfaces to device features (for example, PD stack).

ModusToolbox™: An Eclipse-based embedded design platform for developers that provides a single, coherent, and familiar design experience combining the driver libraries, middleware, lowest power, and most flexible MCUs with best-in-class sensing.

Peripheral Driver Library: The peripheral driver library (PDL) simplifies software development for the PMG1 MCU architecture. The PDL reduces the need to understand register usage and bit structures, thus easing software development for the extensive set of peripherals available.

Revision history

Document revision	Date	Description of changes
**	2023-12-20	Initial release.

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

Edition 2023-12-20

Published by

Infineon Technologies AG

81726 Munich, Germany

© 2023 Infineon Technologies AG. All Rights Reserved.

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

002-38945 Rev **

Important notice

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

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

Warnings

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

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