USB PD-integrated 2 to 5-cell battery charger using EZ-PD™ PMG1-B1 MCU

About this document

Scope and purpose

This application note explains in detail how to implement a USB Power Delivery (PD) integrated 2 to 5-cell battery charger system using EZ-PD™ PMG1-B1 MCU.

EZ-PD™ PMG1-B1 is a high-voltage USB-C PD microcontroller with integrated buck-boost battery charger. The MCU includes Arm® Cortex® CPU core, USB-C PD controller, buck-boost controller, high-voltage protections, gate drivers, and configurable integrated analog and digital peripherals.

EZ-PD™ PMG1-B1 MCU offers a highly integrated solution with reduced BOM cost for battery-powered applications and is targeted for applications that are powered by USB-C PD such as cordless power tool chargers, wireless speakers, and portable electronics.

The firmware discussed in this application note is available as a code example with the ModusToolbox™ software development kit (SDK).

Intended audience

This document is intended for design and application engineers who are familiar with Li-ion battery and want to integrate 2 to 5-cell battery-charging with USB PD and battery monitoring functions using EZ-PD™ PMG1-B1 MCU.

Attention: ***To test with Li-ion battery cells, all safety cautions regarding using and handling Li-ion battery cells must be applied. Do not use old battery cells. New and identical battery cells must be used to minimize any risk.***

Table of contents

1 EZ-PD™ PMG1 MCU introduction

EZ-PD™ PMG1 from Infineon is a family of high-voltage microcontrollers with USB-C Power Delivery (PD). These chips include an Arm® Cortex®-M0/M0+ CPU and USB-C PD controller, buck-boost controller, high-voltage protections, gate drivers along with analog and digital peripherals.

EZ-PD™ PMG1 is targeted at any embedded system that provides to or consumes power from a high-voltage USB-C PD port and leverages the MCU to provide additional control capability.

EZ-PD™ PMG1-B1 MCU is a highly integrated single-port USB Type-C PD solution with integrated buck-boost controller for battery charger applications. It complies with the latest USB Type-C and PD specifications. EZ-PD™ PMG1-B1 MCU has integrated gate drivers for VBUS NFET on the consumer path for sink application. It also includes hardware-controlled protection features on the VBUS.

The EZ-PD™ PMG1 MCU family is fully compliant with the USB PD and Type-C standards. Table 1 compares the features of different MCUs in the EZ-PD™ PMG1 family.

Table 1 Comparison of features of different EZ-PD™ PMG1 family MCUs

Subsystem or range	Item	PMG1-S0	PMG1-S1	PMG1-S2	PMG1-S3	PMG1-B1
CPU and memory subsystem	Core	Arm® Cortex®-M0	Arm® Cortex®-M0	Arm® Cortex®-M0	Arm® Cortex®-M0+	Arm® Cortex®-M0
	Maximum frequency (MHz)	48	48	48	48	48
	Flash (KB)	64	128	128	256	128
	SRAM (KB)	8	12	8	32	16
Power Delivery	PD ports	1	1	1	1 port for 48-QFN package 2 ports for 97-BGA package	1
	Role	Sink	DRP	DRP	DRP	DRP
	MOSFET gate drivers	2× PFET	2× PFET	2× NFET	Flexible 2× NFET	2× NFET
	Fault protections	VBUS OVP and UVP	VBUS OVP, UVP, and OCP SCP and RCP (for source configuration only)	VBUS OVP, UVP, and OCP	VBUS OVP, UVP, and OCP SCP and RCP (for source configuration only)	VBUS, OVP, UVP, and OCP, SCP, VBUS to CC short protection, OTP
USB 2.0 device	Integrated Full-Speed USB 2.0 device with Billboard class support	No	No	Yes	Yes	No

Subsystem or range	Item	PMG1-S0	PMG1-S1	PMG1-S2	PMG1-S3	PMG1-B1
Buck-boost controller	Switching frequency Control range: Input and output	No	No	No	No	150 kHz to 600 kHz Input: 4 V to 24 V Output: 3.3 V to 21.5 V
	HS and LS gate driver Strength: Pull-up rising edge	No	No	No	No	HSDR PMOS/pull-up driver strength configuration 0 x 1 = 20.1 Ω (slow) 0 x 4= 9.7 Ω (normal) 0 x 7 = 3.3 Ω (fast) LSDR PMOS/pull-up driver strength configuration 0 x 1 = 17.9 Ω (slow) 0 x 4 = 8.2 Ω (normal) 0 x 7 = 2.9 Ω (fast)
	HS and LS gate driver Strength: Pull-down falling edge	No	No	No	No	HSDR NMOS/pull-down driver strength configuration 0 x 1 = 20.4 Ω (slow) 0x4 = 10.1 Ω (normal) 0x7 = 3.4 Ω (fast) LSDR NMOS/pull-down driver strength configuration 0 x 1 = 20.3 Ω (slow) 0 x 4 = 9.3 Ω (normal) 0 x 7 = 3.1 Ω (fast)
Voltage range	Supply (V)	VDDD (2.7 to 5.5) VBUS (4 to 21.5)	VSYS (2.75 to 5.5) VBUS (4 to 21.5)	VSYS (2.7 to 5.5) VBUS (4 to 21.5)	VSYS (2.8 to 5.5) VBUS (4 to 30)	VSYS (2.75 to 5.5) VIN (4 to 24)
	I/O (V)	1.71 to 5.5	1.71 to 5.5	1.71 to 5.5	1.71 to 5.5	1.71 to 5.5

Subsystem or range	Item	PMG1-S0	PMG1-S1	PMG1-S2	PMG1-S3	PMG1-B1
Digital	SCB (configurable as I^2C/UART/SPI)	2	4	4	7 for 48-QFN package (out of which 5 can be configured as SPI and UART) 8 for 97-BGA package	3
	TCPWM block (configurable as timer, counter or pulse width modulator)	4	2	4	7 for 48-QFN package 8 for 97-BGA package	8
	Hardware authentication block (Crypto)	No	No	Yes (AES-128/192/256, SHA1, SHA2-224, SHA2-256, PRNG, CRC)	Yes (AES-128, SHA2-256, TRNG, vector unit)	No
Analog	ADC	2× 8-bit SAR	1× 8-bit SAR	2× 8-bit SAR	2× 8-bit SAR 1× 12-bit SAR	2× 8-bit SAR 1× 12-bit SAR
	On-chip temperature sensor	Yes	Yes	Yes	Yes	Yes
Direct memory access (DMA)	DMA	No	No	No	Yes	No
GPIO	Maximum number of I/Os	12 (10+2 OVT)	17 (15+2 OVT)	20 (18+2 OVT)	26 (24+2 OVT) for 48-QFN 50 (48+2 OVT) for 97-BGA	21 (19+2 OVT)
Charging standard	Charging source	–	BC 1.2, AC	BC 1.2, AC	BC 1.2, AC, AFC, and Quick Charge 2.0/3.0	BC 1.2, AC, AFC, and Quick Charge 2.0/3.0
	Charging sink	BC 1.2, Apple charging (AC)	BC 1.2, AC	BC 1.2, AC	BC 1.2, AC	BC 1.2, AC
ESD protection	ESD protection	Yes (up to ±8 kV contact discharge, up to ±15 kV air discharge, human body model, and charged device model)	Yes (human body model and charged device model)	Yes (up to ±8 kV contact discharge, up to ±15 kV air discharge, human body model and charged device model)	Yes (human body model and charged device model)	Yes (ESD_HBM: 2 kV and ESD_CDM: 500 V)

Subsystem or range	Item	PMG1-S0	PMG1-S1	PMG1-S2	PMG1-S3	PMG1-B1
Packages	Package options	24-QFN (4 × 4 mm, 0.5 mm pitch)	40-QFN (6 × 6 mm, 0.5 mm pitch) 42-CSP (2.63 × 3.18 mm, 0.4 mm pitch)	40-QFN (6 × 6 mm, 0.5 mm pitch)	48-QFN (6 × 6 mm, 0.5 mm pitch) 97-BGA (6 × 6 mm, 0.5 mm, and 0.65 mm pitch)	48-pin QFN (6 × 6 mm, 0.4-mm pitch)

2 Introduction to USB-C PD-integrated battery charging system

In recent years, the USB-C connector has become widely popular and has dominated many applications such as mobile phones, laptops, tablets, PCs, and other consumer electronic devices. The USB Type-C and Power Delivery specification made it possible for USB-C connectors to provide up to 100 W (20 V at 5 A) over the connector. With the latest USB Power Delivery specification 3.1, USB-C power adapters can now provide up to 240 W (48 V at 5 A) on the Type-C connector. The standardization of USB-C connectors has enabled many PC, notebook, and mobile phone vendors to move away from their proprietary chargers and use USB-C power adapters as primary chargers for their devices. This has resulted in reduced electronic waste and the elimination of inbox chargers.

The USB-C revolution has led many other consumer applications, such as smart speakers, power tools, cameras, media players, and smart home appliances, to begin adopt USB-C as their primary connector for charging and data communication. The USB-C port on battery-operated devices, such as smart speakers, can be a dual-role port (DRP) device. This allows the speaker to charge its battery when connected to a USB-C charger and can also charge a phone when connected to it.

2.1 Traditional battery charging system

In the past, power tool chargers are typically shipped with their own proprietary AC/DC charger adapters. These charger connectors are often incompatible with other adapters, making it impossible to share or reuse chargers. Additionally, shipping proprietary AC/DC chargers also increased the total cost of the product. The battery chargers for power tools typically included several components. Figure 1 shows a traditional battery charger system block diagram.

- Buck-boost/battery charging IC to enable the charging of battery packs. These ICs typically supports various battery charging modes such as trickle charging, pre-charging, constant current, and constant voltage mode, among others. The battery charging IC also supports safety timers for protection against prolonged battery charging.
- Battery management MCU responsible for monitoring and cutting off charging in case the primary battery charging IC failed. Additionally, sensors are included to monitor the system and battery temperature.
- Finally, battery protection and individual cell-monitoring circuits are also included to ensure that each individual cell voltage did not exceed safe limits.

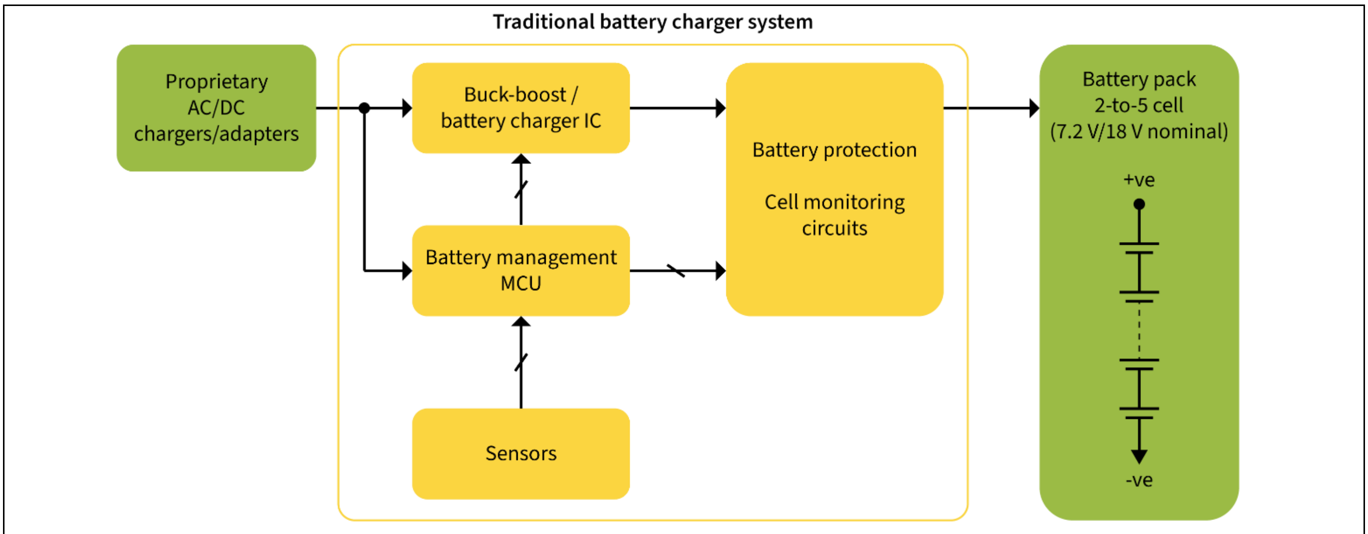

Figure 1 Traditional battery charger system

2.2 USB PD-integrated battery charger system

By replacing proprietary connectors in battery-powered applications with USB-C connectors, the device can be charged with any USB-C adapter. In the battery charger system shown in Figure 2, the USB-C PD controller is responsible for negotiating the USB Type-C and PD contract with the USB-C power adapters and delivering the power to the battery charger IC for charging the battery pack. In dual-role power applications, the USB-C controller can also take battery power and make it available on the USB-C connector to charge your phones and laptops.

EZ-PD™ PMG1-B1 MCU from Infineon offers an integrated single chip solution for USB PD-integrated 2 to 5-cell battery charging system with battery protection and cell monitoring. Figure 2 shows a USB-C PD powered system using EZ-PD™ PMG1-B1 for battery charging and the MCU for power and system management.

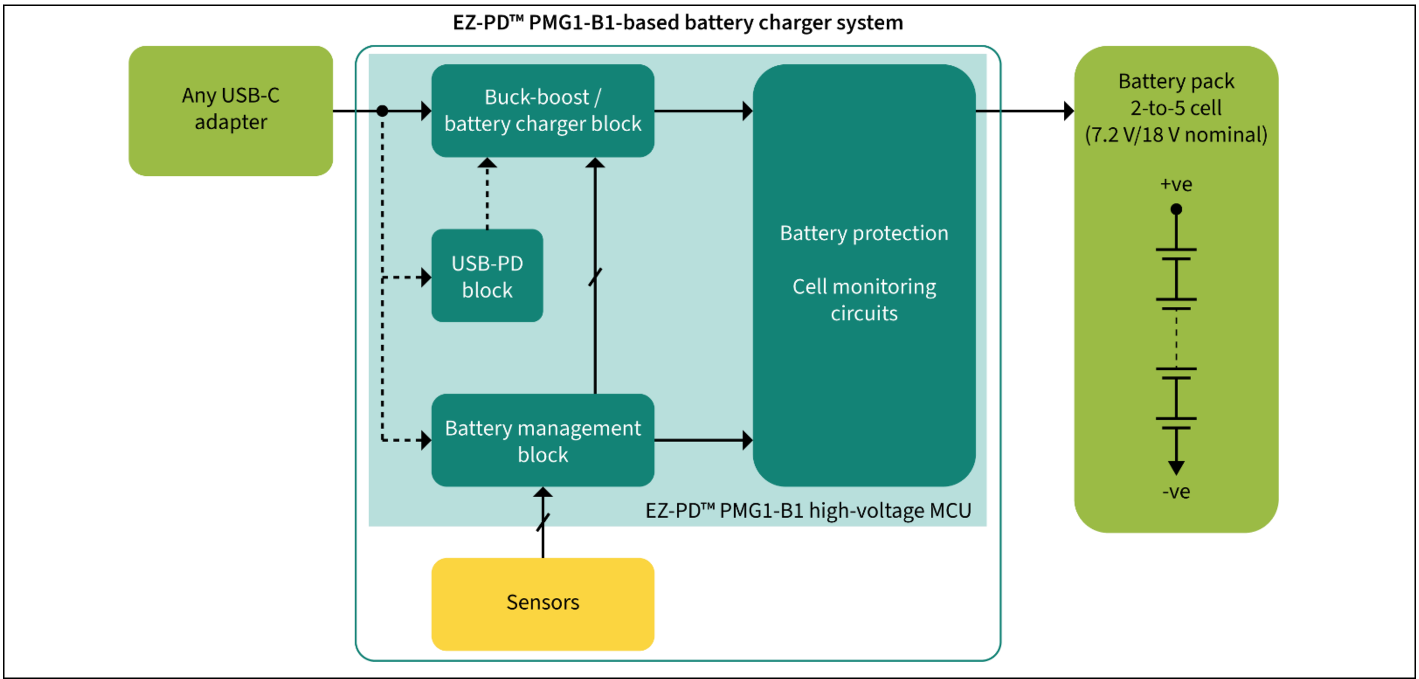

Figure 2 EZ-PD™ PMG1-B1-based USB-C battery charger system

3 EZ-PD™ PMG-B1 device as controller for USB PD-integrated battery charging system

EZ-PD™ PMG1-B1 is a highly integrated single-port USB Type-C 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.

EZ-PD™ PMG1-B1 is targeted embedded systems that power from a high-voltage USB-C port and need an MCU to implement the product features.

3.1 Feature integration

This section lists the key features of the EZ-PD™ PMG1-B1 device, which are critical for the implementation of the USB PD integrated 2 to 5-cell lithium-ion (Li-ion) battery charging solution discussed in this document.

- 32-bit Arm® Cortex®-M0 CPU
- 16 KB RAM, 32 KB ROM, and 128 KB flash memory
- Type-C and USB PD blocks with integrated NFET gate drivers
- Integrated buck boost battery charge controller
- Programmable switching frequency (150 to 600 kHz) in the integrated buck-boost controller
- Programmable analog block with 8-bit SAR ADC and 12-bit SAR ADC
- Programmable digital blocks such as SCB and TCPWM
- Up to 21 programmable GPIO pins (on 48-QFN package)
- Wide input voltage range (4 V to 24 V with 40 V tolerance)

3.2 EZ-PD™ PMG1-B1-based USB PD-integrated battery charger system

EZ-PD™ PMG1-B1 MCU provides a range of features that enable efficient monitoring, charging, and protection of batteries. It includes a USBPD block and a built-in buck-boost block, which allows the MCU to negotiate up to 100 W power delivery contract over the USB-C connector with the USB-C power adapter. This makes the power easily accessible for charging the battery pack.

Additionally, EZ-PD™ PMG1-B1 MCU offers several battery charging modes, such as constant current, constant voltage, pre-charge, and trickle charge modes, making it easy to implement a 2-cell to 5-cell battery charging system.

EZ-PD™ PMG1-B1 MCU also supports high-voltage analog peripherals such as current sense amplifier (CSA), undervoltage overvoltage (UVOV) blocks, 12-bit SAR ADC and 8-bit SAR ADC, which enable the monitoring of critical battery parameters such as total battery voltage, individual battery cell voltages, total battery charging current, battery pack temperature, and battery charger system temperature, etc. This enables EZ-PD™ PMG1-B1 MCU to integrate battery protection circuitry and ensuring safe and efficient charging of batteries. The MCU's timer block adds another layer of protection by enabling the implementation of battery charging safety timers. These timers prevent the battery from overcharging because of abnormal conditions, therefore, increasing the battery's life.

EZ-PD™ PMG1-B1 MCU has a high-voltage regulator that allows direct power from the VBUS supply on the USB-C connector, eliminating the need for an external regulator and reducing system cost. Furthermore, a standby/low-power regulator is available to power the device from the battery pack when the USB-C VBUS supply is not available. The low-power consumption of the standby regulator and the deepsleep support of the EZ-PD™ PMG1-B1 MCU ensure that the device does not drain the battery pack, therefore, extending its shelf life.

EZ-PD™ PMG1-B1 also provides the functionality of a traditional MCU through software programmability. This feature enables developers to use the firmware to configure an array of battery charging parameters such as battery cell count, charging thresholds, cell monitoring, battery cell thresholds, temperature monitoring thresholds, battery protection features, charging algorithms, etc. Additionally, EZ-PD™ PMG1-B1 MCU includes peripherals such as I2C and UART, which allow EZ-PD™ PMG1-B1 MCU to communicate the battery status to an external embedded host controller and allows to implement custom features as well.

Figure 3 shows EZ-PD™ PMG1-B1 MCU hardware blocks the for battery charging system. In summary, EZ-PD™ PMG1-B1 MCU provides efficient battery charging and protection with USB-C support with various features such as USBPD block, integrated buck-boost block, timer block, high voltage regulator, standby/low power regulator, and low power consumption.

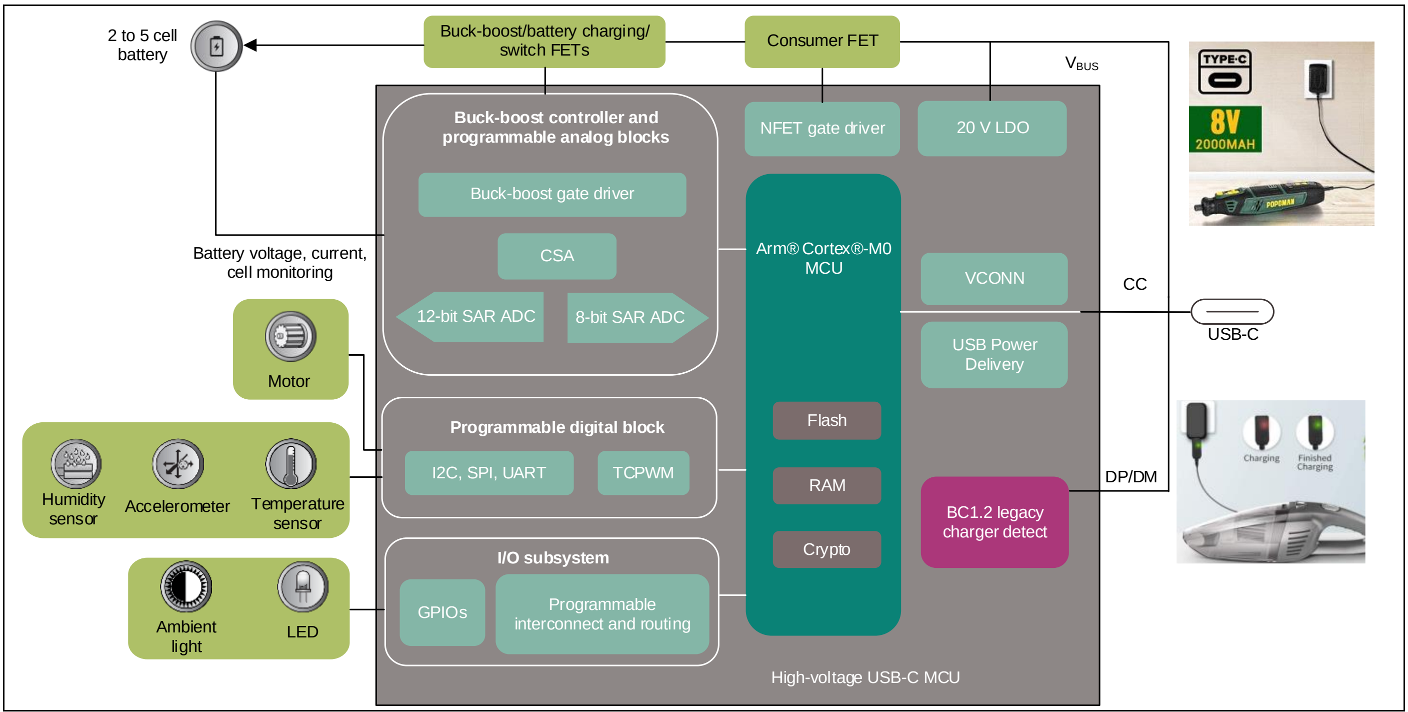

Figure 3 **EZ-PD™ PMG1-B1 MCU hardware blocks for battery charging system**

4 EZ-PD™ PMG1-B1 battery charger application development

4.1 EZ-PD™ PMG1-B1 MCU resources

The EZ-PD™ 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 EZ-PD™ PMG1 high voltage MCUs. Visit the EZ-PD™ PMG1 MCU webpage to find out more.

- EZ-PD™ PMG1-B1 MCU webpage.
- EZ-PD™ PMG1-B1 datasheet provides all the information needed to select and use the device, including functional description and electrical specifications.
- EVAL_PMG1_B1_DRP Evaluation Kit is available for evaluation, design, and development of different applications using EZ-PD™ PMG1-B1 MCUs.
- Reference manuals provide detailed descriptions of the architecture and registers in each device family.
- Application notes and code examples cover a broad range of topics, from basic to advanced level. See the application note – Getting started with EZ-PD™ PMG1-B1 MCU using ModusToolbox™ for more details on information such as:
  - Firmware/application development using ModusToolbox™ software
  - ModusToolbox™ software help
  - Programming and debugging
  - Application development on the EVAL_PMG1_B1_DRP kit using ModusToolbox™
- Technical support: EZ-PD™ PMG1 MCU community forum, knowledge base articles.

4.2 EVAL_PMG1_B1_DRP kit

The EVAL_PMG1_B1_DRP is an evaluation kit for EZ-PD™ PMG1-B1 USB PD MCU with integrated buck-boost battery charger. EVAL_PMG1_B1_DRP can be used to sink or source power through the USB-C PD port in applications such as the cordless power tool charger, wireless speakers, and portable electronics. The kit can be used to sink up to 100 W and source up to 27 W.

The kit can also be used to charge 2 to 5-cell batteries and the battery charging algorithm is implemented as part of EZ-PD™ PMG1-B1 SDK in ModusToolbox™ software.

- EVAL_PMG1_B1_DRP kit guide provides detailed information about the hardware capabilities of the kit and how-to setup the kit for various demos such as charging a 5-cell battery with and without battery cell monitoring enabled.
- Hardware design guidelines and design calculator for EVAL_PMG1-B1_DRP_KIT provides comprehensive and easy-to-follow guidelines for the power stage design calculator for the peak current mode controlled (PCMC) and continuous conduction mode (CCM) power tool sink battery charger for 2-5 cell batteries based on the EZ-PD™ PMG1-B1 MCU.

Figure 4 shows the front view of the EVAL_PMG1_B1_DRP kit with critical components.

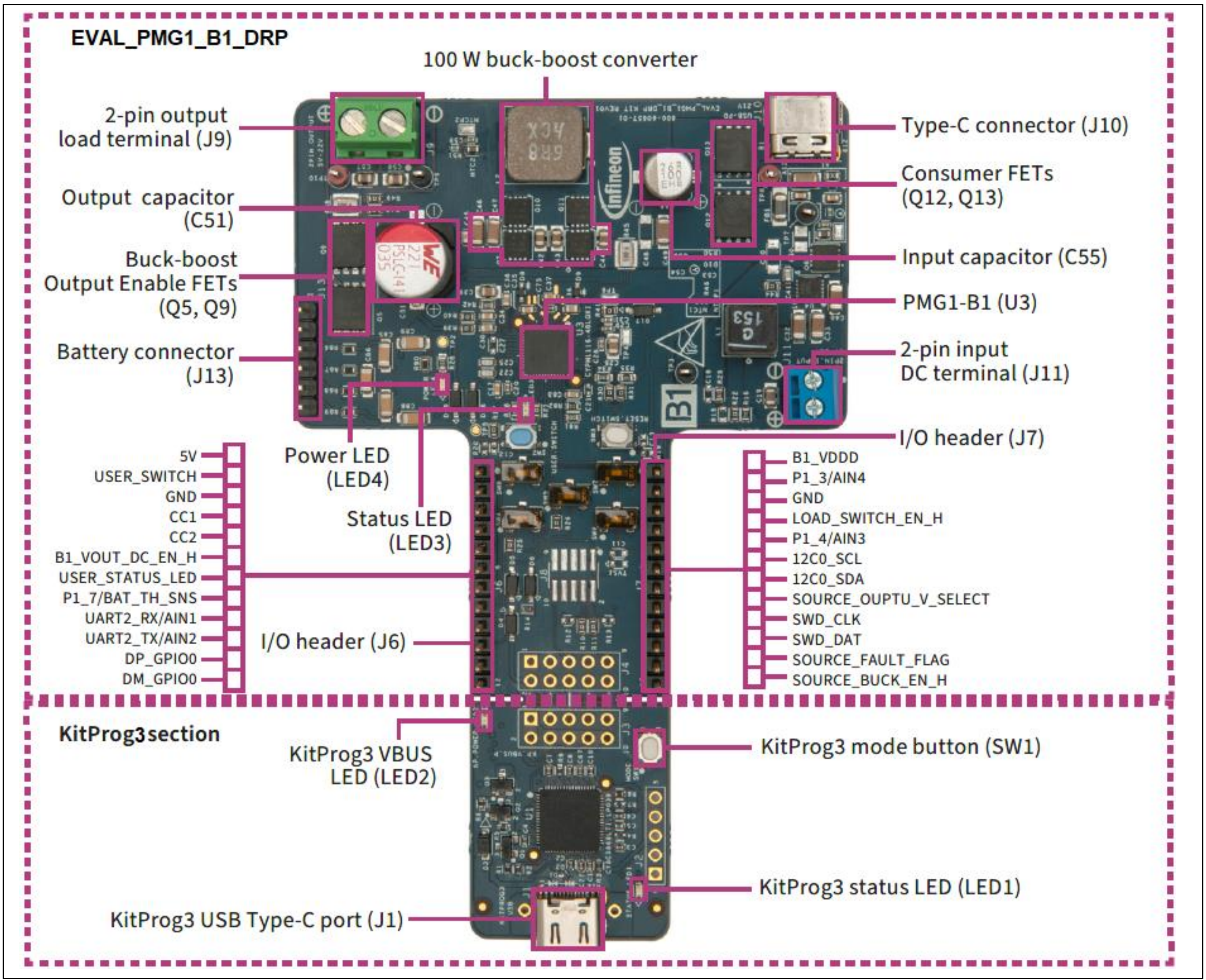

Figure 4 **EVAL_PMG1_B1_DRP board (top view)**

4.3 Creating EZ-PD™ PMG1-B1 Li-ion battery charger application using ModusToolbox™

This code example is developed with the support of ModusToolbox™ software development environment. The ModusToolbox™ package provides many integrated configuration tools along with Eclipse IDE which enables an easy development of embedded applications. The following sections discuss the process of creating and programming this code example with EVAL_PMG1_B1_DRP kit.

4.3.1 Creating the USB-C PD Li-ion battery charger application

Install ModusToolbox™ package on your PC and click on Eclipse IDE for ModusToolbox™ from the start menu (Windows) to open the IDE.

1. In the Eclipse IDE for ModusToolbox™, provide the appropriate directory path and folder name to create a workspace for the project.
2. In the Quick Panel, click **New Application** to launch the Project Creator tool as shown in Figure 5.

3. In PMG1 BSPs, select **EVAL_PMG1_B1_DRP** board support package (BSP) and click **Next** as shown in the following figure.

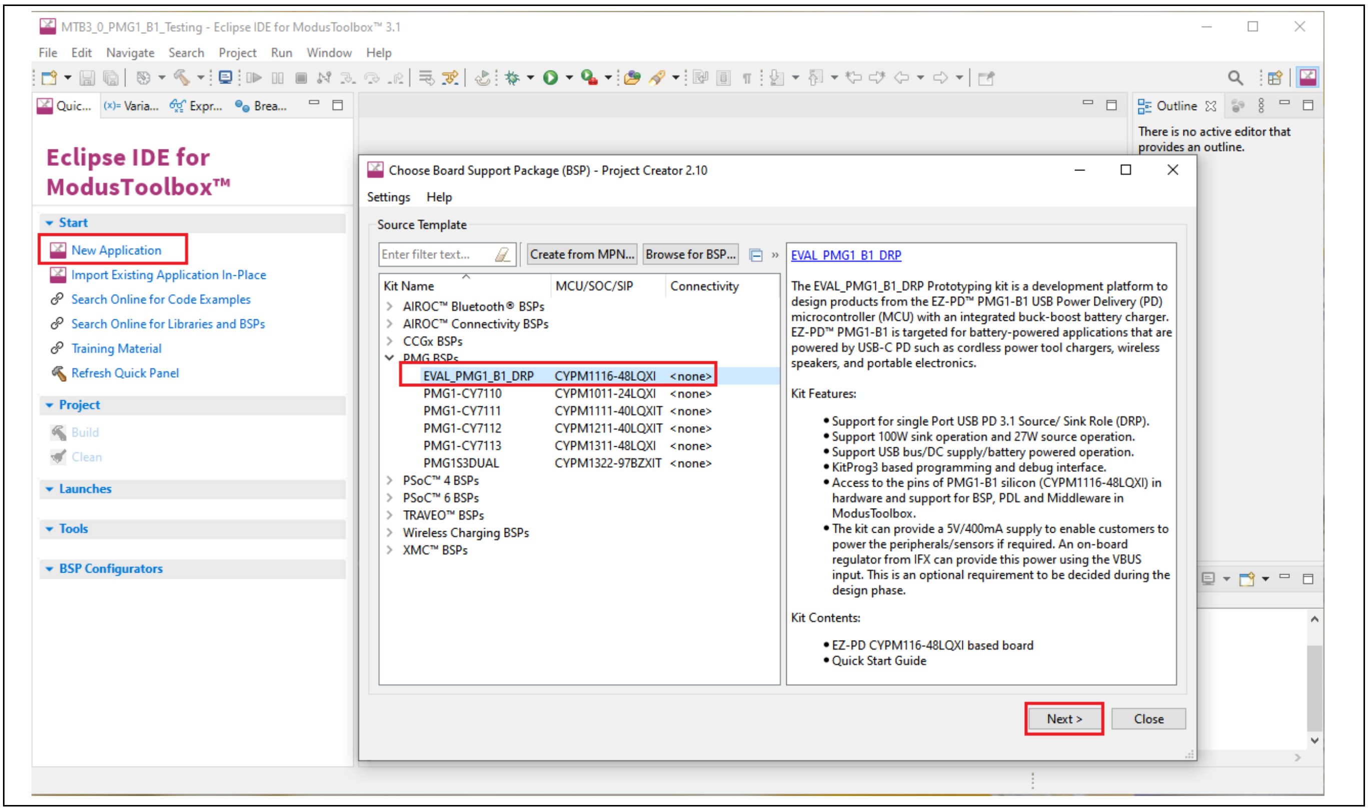

Figure 5 **Creating a new application on ModusToolbox™**

4. Select **USBPD Li-ion Battery Charger** and click **Create** as shown in the following figure to launch the application.

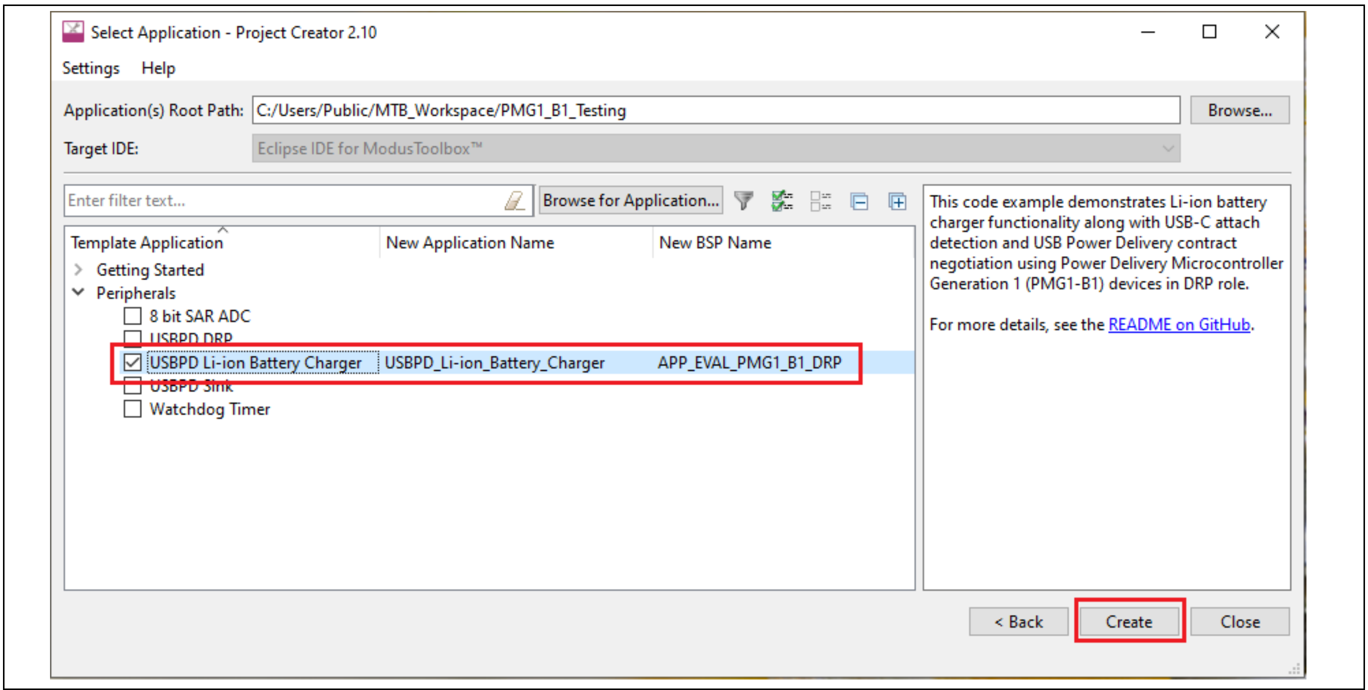

Figure 6 **Creating a new application on ModusToolbox™**

5. The application will be created in the new workspace and all the related files and libraries are listed under the **Project Explorer** as shown in the following figure.

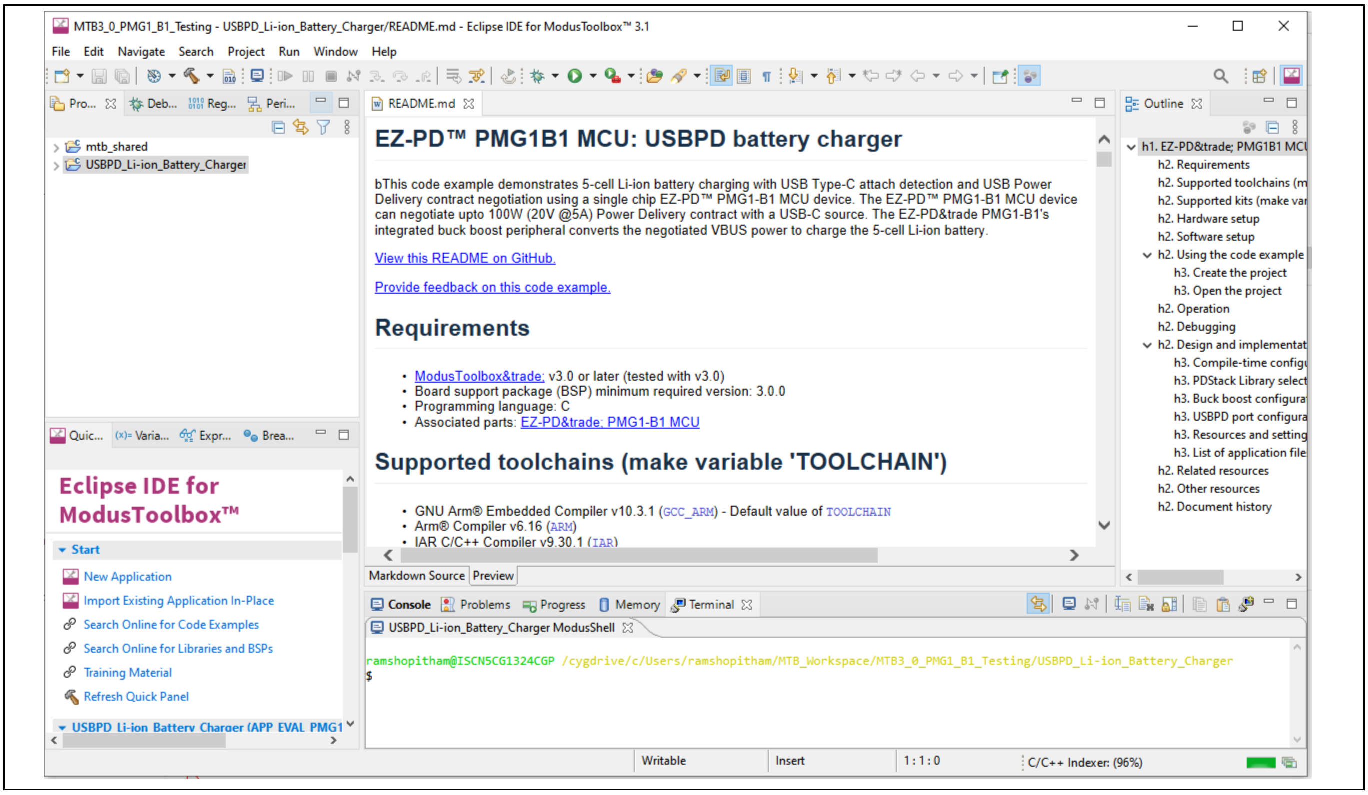

Figure 7 **Creating the new application in the workspace**

4.3.2 Opening the Device Configurator

The Device Configurator enable and configure device peripherals, such as ADC, USB PD, buck-boost, clocks, and pins, as well as standard MCU peripherals that do not require their own configurator tool. This configurator generates the initialization code used in the battery charger application.

The template folder contains target files and has the *design.modus* file.

1. Click the *design.modus* file to open the design configuration as defined by the BSP.
2. Click the **Device Configurator** under the **BSP Configurators (APP_EVAL_PMG1_B1_DRP**) in the **Quick Panel** to view and modify the design configuration. See Figure 8.

You can also double-click to open the other *design.modus* files under the template section to open the respective configurators or click the corresponding links in the **Quick Panel**.

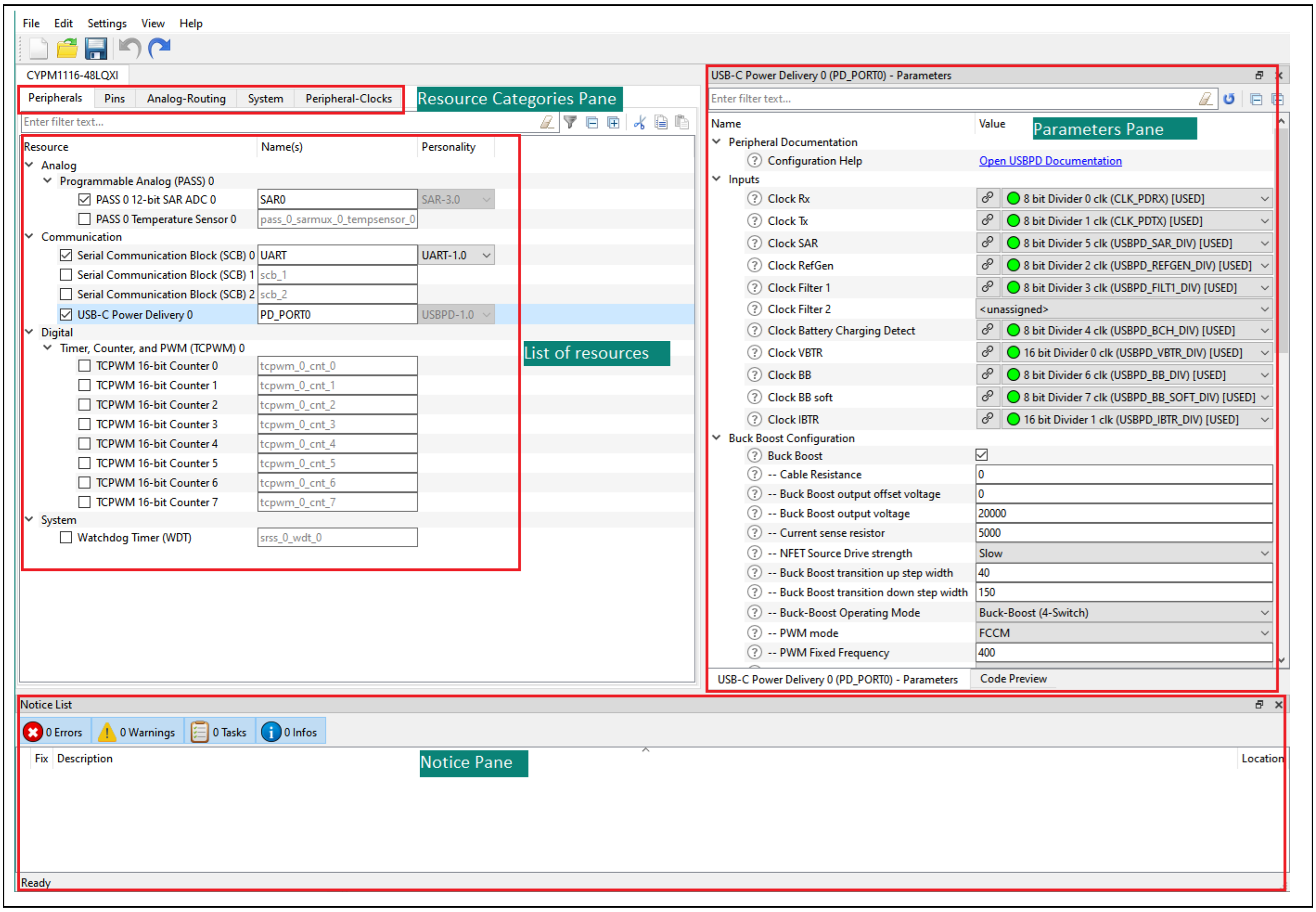

Figure 8 **design.modus overview**

3. From the list of resources in the resources categories tab in **Device Configurator**, choose from the different resources available in the device such as peripherals, pins, and clocks.

   You can choose how a resource behaves by choosing a personality for the resource. For example, a Serial Communication Block (SCB) resource can have an EZI2C, I2C, SPI, or UART personalities. The Alias is the 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).

4. In the **Parameters** window, enter the configuration parameters for each enabled resource and the selected personality.

   The code **Preview** pane shows the configuration code generated per the configuration parameters selected. This code is populated in the *cycfg_* file in the *GeneratedSource* folder.

   The **Notices** pane displays the errors, warnings, and information messages arising out of the configuration.

See Section 5.2 on how to configure EZ-PD™ PMG1-B1 peripherals using the Device Configurator for the USB PD Li-ion battery charger application.

4.3.3 USB PD Li-ion battery charger firmware flow

1. Examine the code in the *main.c* file of the application. For the firmware flowchart, see Figure 10.
2. Resource initialization is performed by the CM0 CPU. It configures the system clocks, pins, clock to peripheral connections, 12-bit SAR ADC, USB-C PD, buck boost, and other platform resources required to implement the USB-C PD Li ion battery charger application.

3. See Section 6 for in depth details on the EZ-PD™ PMG1-B1 firmware implementation for USB PD-integrated Li-ion battery charger application.

Note: *Note that the application code uses BSP/middleware functions to execute the intended functionality.*

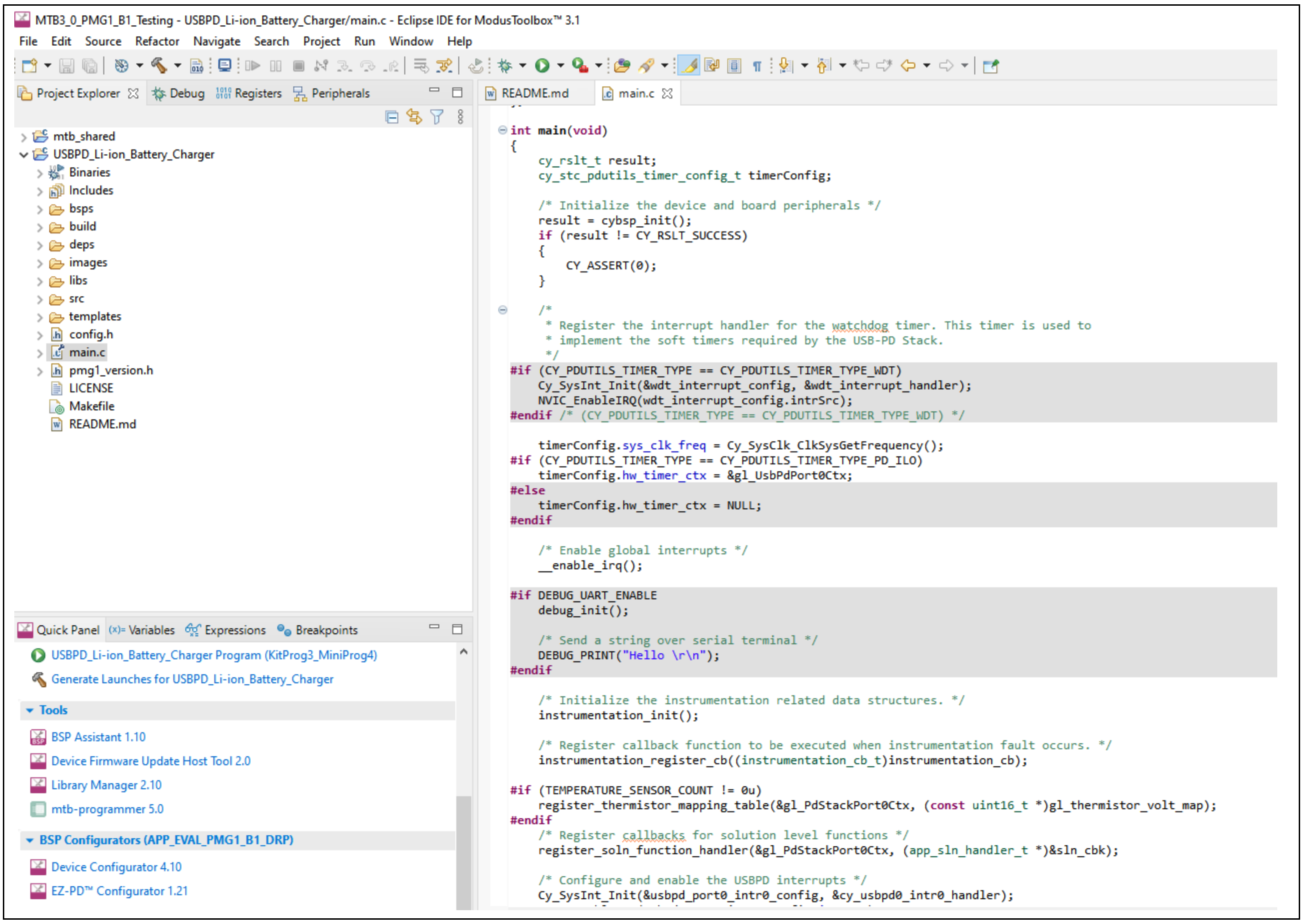

Figure 9 main.c file

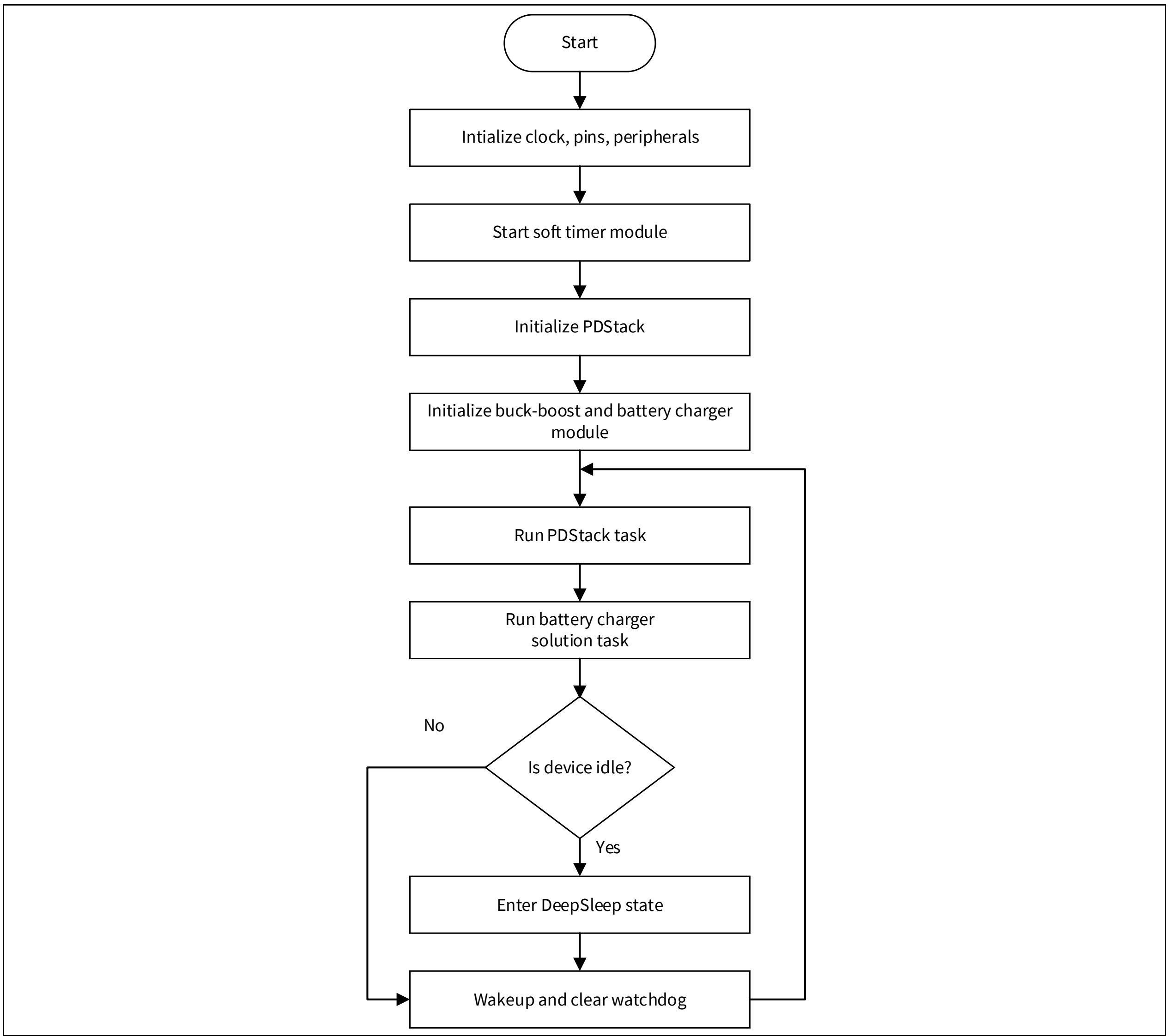

Figure 10 Firmware flowchart

4.3.4 Building and programming the application

Building of the application involves generating the source files corresponding to the configurator design files, compiling, and linking the C code to generate the programmable hex file. Follow these steps to build the application:

1. Select the application by clicking on the project file by the **Project Explorer**.
2. Navigate to the Quick Panel and click on **Build Application** to start the build process, see Figure 11.

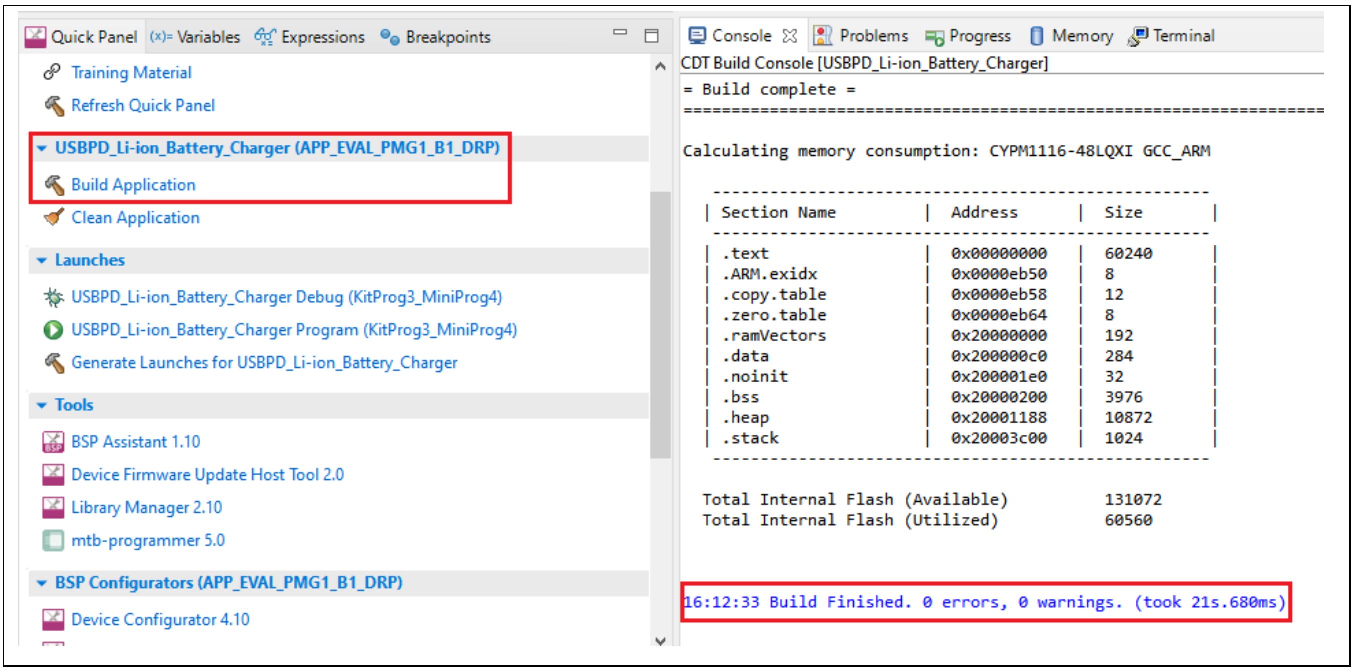

Figure 11 **Building the application**

3. Follow the updates on the **Console** window to note the final result after the build is complete. Ensure that there are no error/warnings listed in the console.
4. After the build is complete, the resultant programmable hex file will be generated in the following path: **<Application-name> > build > APP_EVAL_PMG1_B1_DRP > Debug**.
5. To program the target EZ-PD™ PMG1-B1 MCU, connect a USB Type-C cable to KitProg3 USB Type-C port (J1) on the prototyping kit.
6. Navigate to the Quick Panel and click on **<Application_name> Program (KitProg3_MiniProg4)** link under Launches to start the programming. Follow the console window to note the result of programming as shown in the following figure.

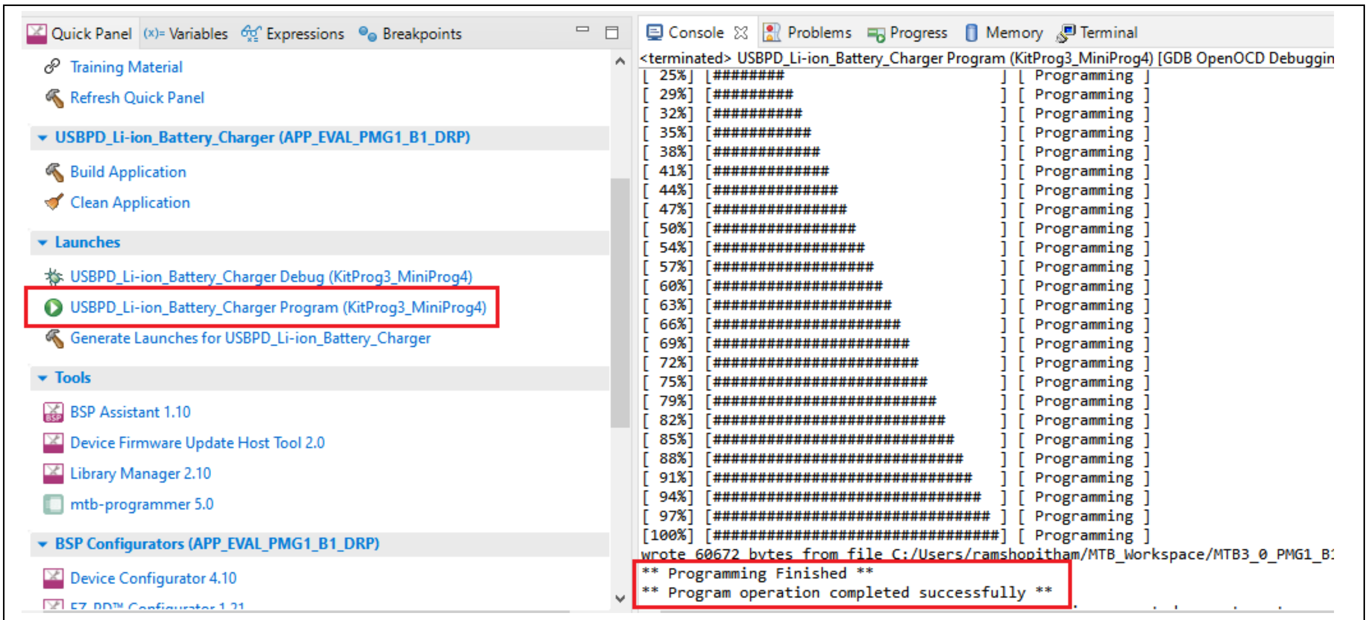

Figure 12 **Programming the application to the target EVAL_PMG1_B1_DRP kit**

5 Hardware design for USB PD-integrated battery charger system using EZ-PD™ PMG1-B1 MCU

5.1 Hardware architecture for 5-cell Li-ion battery pack charger solution

Figure 13 shows the hardware architecture of a typical power tool battery charger application using the EZ-PD™ PMG1-B1 devices.

In this application, the EZ-PD™ PMG1-B1 MCU negotiates the required power with the connected USB-C power adapter and uses the integrated buck-boost controller to convert the input Type-C VBUS supply to the required output voltage and current and charges the battery.

EZ-PD™ PMG1-B1 MCU measures the battery voltage and lowers the buck-boost output voltage if the measured battery voltage is lower than the user-configured threshold. It monitors the various temperatures using external NTC thermistors. EZ-PD™ PMG1-B1 MCU throttles the output power based on temperature and/or shuts off the power under critical conditions. When no load is connected to the USB Type-C port, EZ-PD™ PMG1-B1 remains in standby mode without switching on the buck-boost controller.

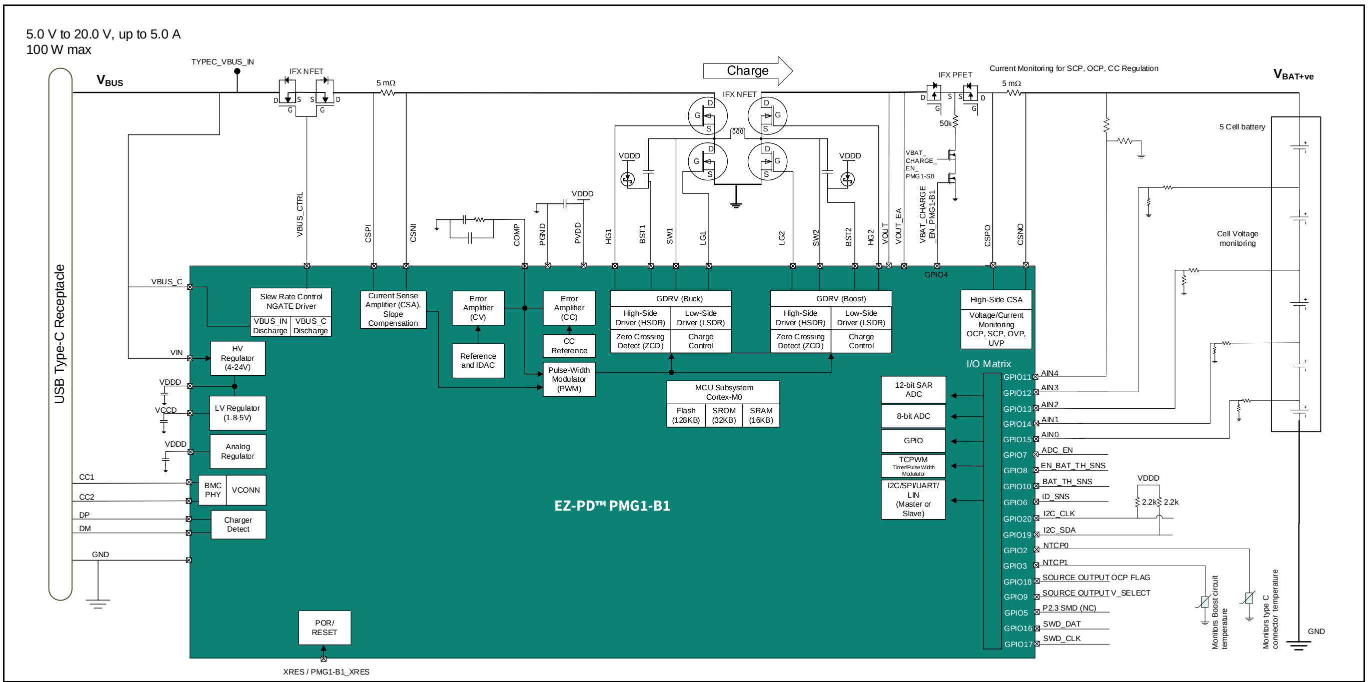

Figure 13 Hardware architecture of a 5-cell battery charger system using EZ-PD™ PMG1-B1 MCU

5.2 Peripheral configuration

ModusToolbox™ offers easy configuration of all the peripherals using its Device Configurator tool. See Section 4.3.2 for details on how to launch the Device Configurator tool.

This section discusses in detail about the configuration's settings for various EZ-PD™ PMG1-B1 MCU peripherals used in the USB PD Li-ion battery charger firmware implementation. Table 2 lists the various EZ-PD™ PMG1-B1 MCU hardware peripheral blocks used for the development of this solution, with their purpose. Figure 8 provides the USBPD Li-ion Battery Charger peripheral design configuration.

Table 2 List of EZ-PD™ PMG1-B1 peripherals used

Peripherals	Uses
PASS 0 12-bit SAR ADC 0	Reads analog signals corresponding to total battery voltage, individual cell voltage, battery temperature, etc.
USB-C Power Delivery 0– Power Delivery configuration	Configures PD communication and contract negotiation with a USB PD source. Enables fault protection and legacy charging configurations on the Type-C port.
USB-C Power Delivery 0– Buck-Boost configuration	Configures integrated buck-boost parameters such as buck-boost output voltage, current sense resistors, transition step up width, PWM modes, PWM frequency, PWM dithering type, etc. Also allows configuring HS1, HS2, LS1, LS2 gate pull-up/down drive strength.

5.2.1 12-bit SAR ADC

EZ-PD™ PMG1-B1 MCU has one 12-bit successive approximation register analog-to-digital converter (SAR ADC). This SAR ADC subsystem consists of:

- A 12-bit SAR converter (SAR ADC)
- An embedded reference block (SARREF)
- A mux (SARMUX) at the inputs of the converter
- A sequence controller (SARSEQ) that enables multi-channel acquisition in a round robin fashion, without CPU intervention, to maximize scan rates

The 12-bit SAR ADC block can configure up to 8 analog channels that are automatically scanned with the results placed in individual result registers. EZ-PD™ PMG1-B1 battery charger application uses the 12-bit SAR ADC to measure the total battery pack voltage, individual battery cell voltages, and the battery temperature measurement. The battery pack and individual cell voltages need to be scaled down to the I/O voltage range for measurement. See Section 5.4.1 for more detail on the battery cell voltage and temperature monitoring circuit.

Table 3 lists the various parameter settings for the 12-bit SAR ADC that are required for firmware implementation, based on the Device Configurator tool of ModusToolbox™.

Table 3 Configuration of PASS 0 12-bit SAR ADC 0

Parameter	Value
Reference voltage (Vref)	1.2 V from internal bandgap reference
Vref bypass capacitor	Not used up to clock frequency of 1.6 MHz
Number of channels	6
Input mode	Single-ended
Targeted scan rate	500 sps
Clock frequency	1.548 MHz
Vneg/Vminus value for single-ended mode	Vref
Single-ended result format	Unsigned
Number of samples for averaging	16
Resolution	12-bit

Parameter	Value
Minimum acquisition time for channels	668 ns
Channel 0 connection	VBAT total battery voltage measurement
Channel 1 connection	Individual battery cell voltage measurement – Cell 4
Channel 2 connection	Individual battery cell voltage measurement – Cell 3
Channel 3 connection	Individual battery cell voltage measurement – Cell 2
Channel 4 connection	Individual battery cell voltage measurement – Cell 1
Channel 5 connection	Battery temperature sensor voltage measurement

Figure 14, Figure 15, and Figure 16 show the configuration of the 12-bit SAR ADC using the Device Configurator tool of ModusToolbox™ that is required for the firmware implementation.

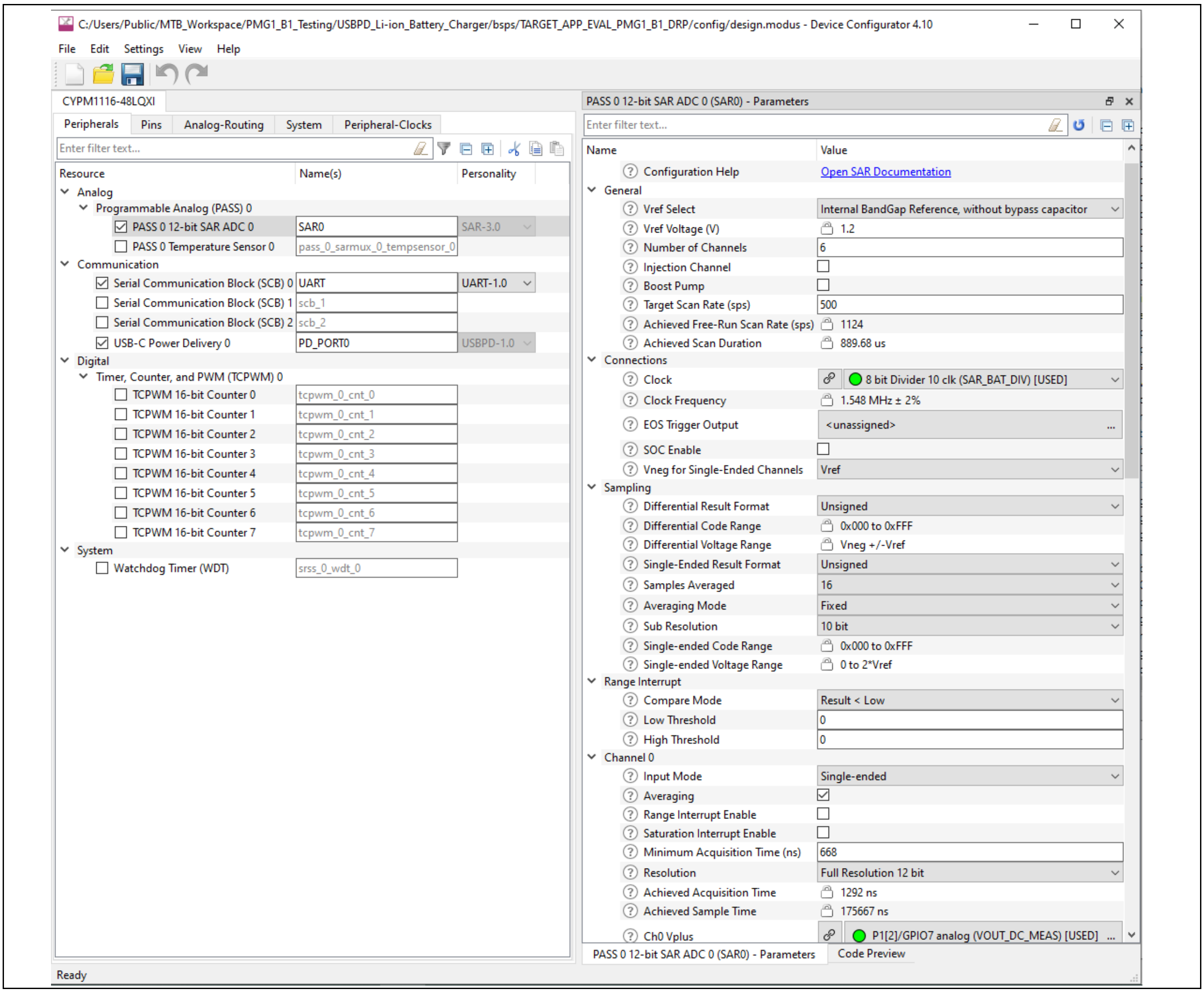

Figure 14 Part 1: Configuration of PASS 0 12-bit SAR ADC 0 in Device Configurator

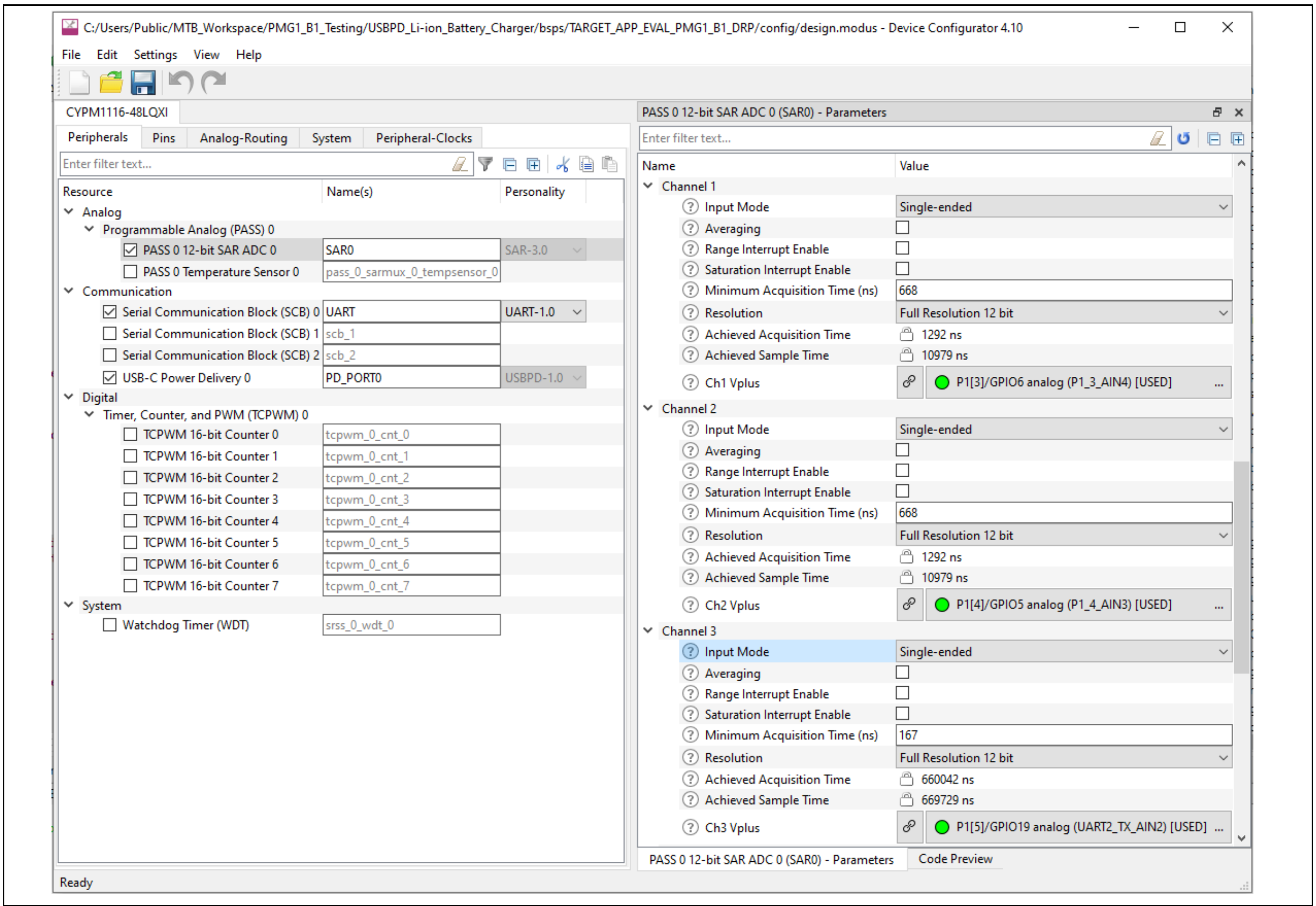

Figure 15 Part 2: Configuration of PASS 0 12-bit SAR ADC 0 in Device Configurator

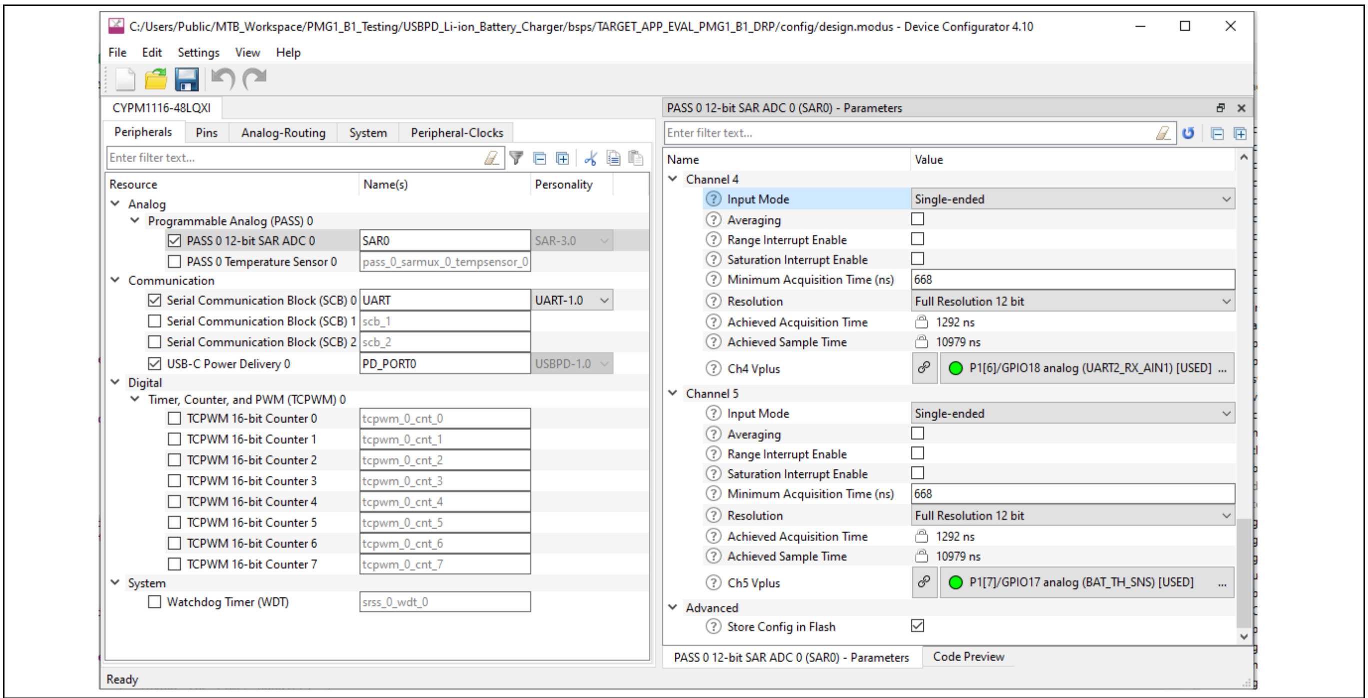

Figure 16 Part 3: Configuration of PASS 0 12-bit SAR ADC 0 in Device Configurator

5.3 USB-C Power Delivery (USBPD) block

5.3.1 USB Power Delivery configuration

The USBPD block of EZ-PD™ PMG1-B1 MCU handles the PD communications through CC lines, PD contract negotiations, and driving the external FETs in the power route. It contains a USB PD biphase mark code (BMC) transceiver, PD PHY, integrated resistor terminations, 8-bit SAR ADCs, integrated VCONN FETs, NFET gate drivers, and load switches for implementing overvoltage (OV), undervoltage (UV), overcurrent (OC), short-circuit (SC), and reverse-current (RC) protection features at device level.

In addition to using the 12-bit SAR ADC, the 8-bit SAR ADC in the USBPD block can be used for additional voltage measurement such as any additional onboard temperature sensors etc. The legacy charging configuration in the USBPD block also uses to enable legacy charging protocols such as BC 1.2, Apple Charging 2.4A, etc. in the sink mode. Table 4 shows the general parameter settings for the USBPD block as used in this solution example.

Note: This document does not discuss the external hardware design for USB PD sink operation. See the hardware design guidelines and design calculator for EVAL_PMG1_B1_DRP Kit – EZ-PD™ PMG1-B1 MCU Prototyping Kit for more details on designing the USB PD power path.

Table 4 Configuration of USB-C Power Delivery 0 block

Parameter	Value
Rx clock frequency	12 MHz
Tx clock frequency	600 kHz
SAR ADC clock frequency	1 MHz
Fault filter 1 clock frequency	500 kHz
VBUS OV protection	Enabled, detect through comparator, hardware-controlled FET
VBUS UV protection	Enabled, detect through comparator, hardware-controlled FET

Figure 17 shows the clock inputs, fault protection, and legacy charging configurations of the USB-C Power Delivery block using the Device Configurator tool of ModusToolbox™ that is required for the firmware implementation.

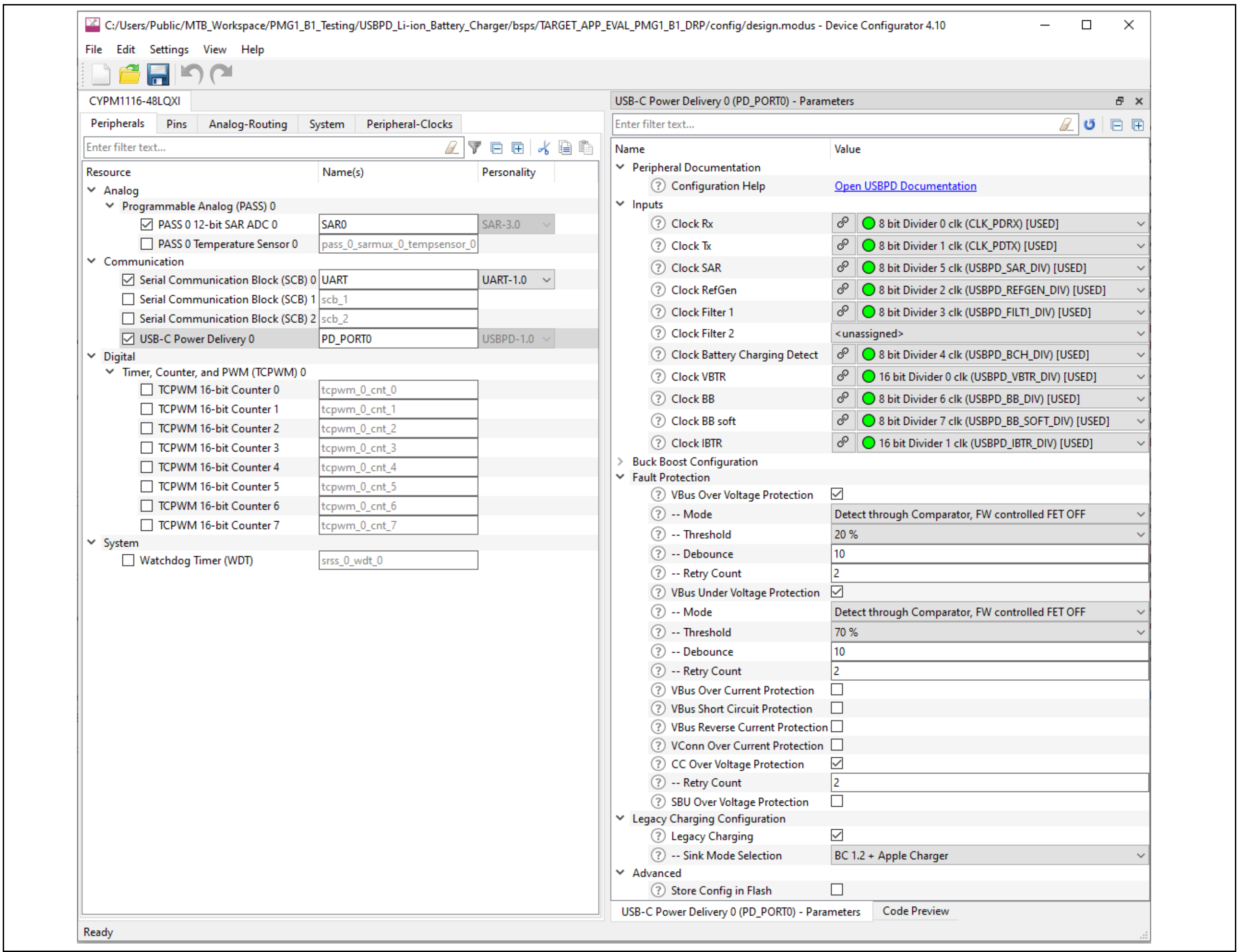

Figure 17 Configuration of USB-C Power Delivery 0 in Device Configurator

5.3.1.1 Integrated buck-boost configuration

EZ-PD™ PMG1-B1 MCU uses the integrated buck-boost subsystem to convert the negotiated USB-C VBUS voltage to the required output voltage and current to charge the Li-ion battery. The buck-boost subsystem can operate in buck-only, boost-only, or buck-boost mode. The buck-boost subsystem in the EZ-PD™ PMG1-B1 device have integrated the following key functional blocks:

- High-side (cycle-by-cycle) CSA
- High-side and low-side gate driver
- Pulse-width modulator (PWM)
- Error amplifier (EA)

The high-side CSA is used for peak current sensing through an external resistor placed in series with the buck control FET. The high-side and low-side gate drivers are used for controlling the buck-boost's switching FETs function. The gate driver block includes zero-crossing detection (ZCD) to implement discontinuous conduction mode (DCM) with diode emulation. The error amplifier (EA) handles the output voltage and current regulation. The programmable EA circuit can be configured to achieve the required VOUT voltage output. The pulse-width modulator (PWM) generates the control signal for the gate drivers driving the external FET. The PWM pulse width, minimum/maximum period, frequency and pulse skip levels, etc. can be configured.

Table 5 lists the buck-boost parameter settings required for the battery charger firmware implementation. These parameters are configured using the Device Configurator tool of ModusToolbox™.

Table 5 Configuration of buck-boost parameters

Parameter	Value
VBTR clock frequency	2 kHz
BB clock frequency	24 MHz
IBTR clock frequency	2 kHz
Buck-boost output voltage	20000 mV
Current sense resistor	5000 µΩ
NFET source drive strength	Slow
Buck-boost transition up step width	40 V/µs
Buck-boost transition down step width	150 V/µs
Buck boost operating mode	Buck_boost (4-switch)
PWM mode	FCCM
PWM fixed frequency	400 kHz
PWM dithering type	Triangular
PWM dithering frequency range	5
Peak current sense resistor	5000 µΩ
Peak current limit	10000 mA

Figure 18 shows the configuration of the buck-boost block using the Device Configurator tool of ModusToolbox™ that is required for the firmware implementation.

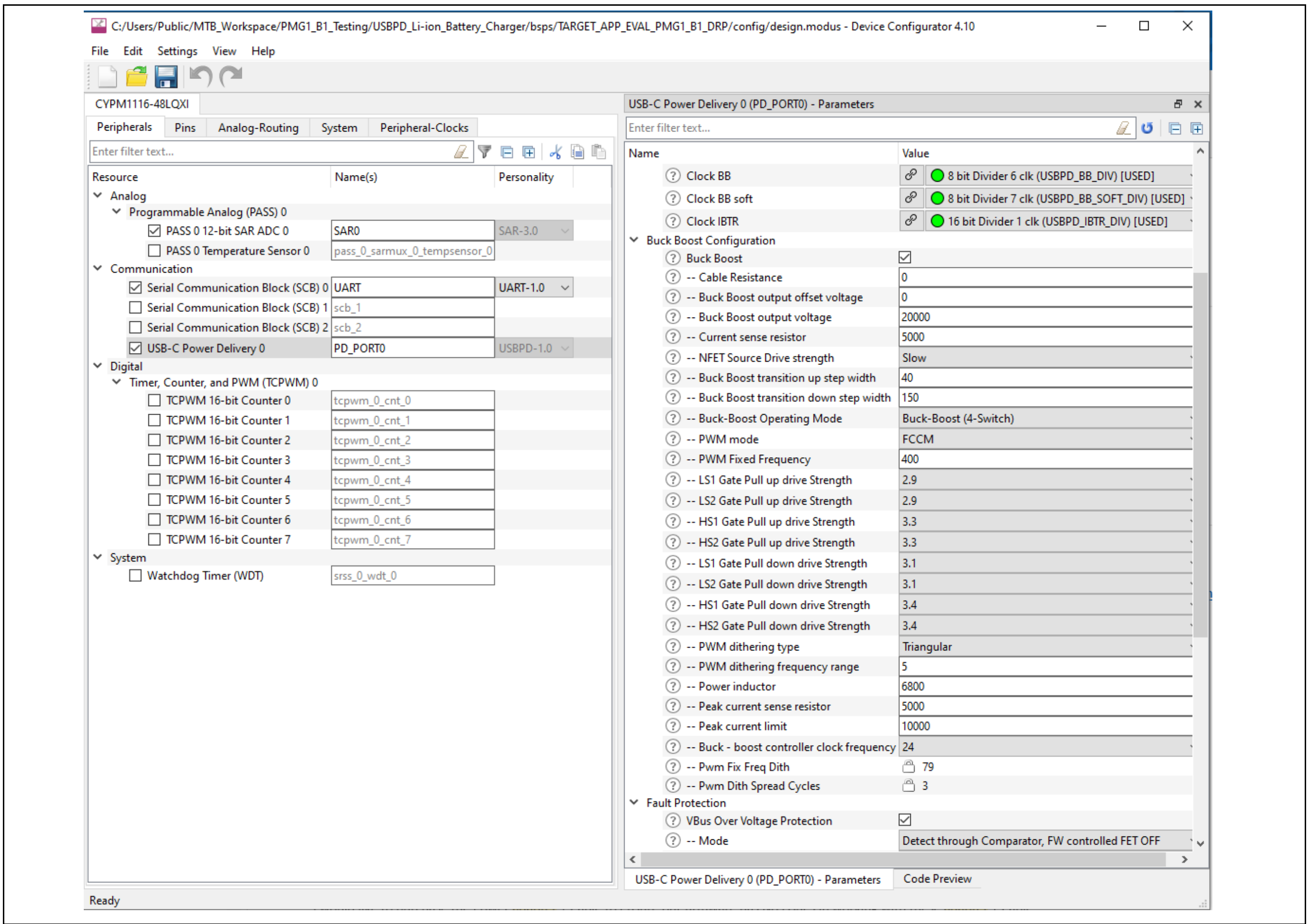

Figure 18 Configuration of buck-boost in Device Configurator

5.4 External hardware design

EZ-PD™ PMG1-B1 devices have integrated high-voltage pins such as VOUT, VBUS_C to monitor the buck-boost output voltage, Type-C connector VBUS voltage, etc. In addition, EZ-PD™ PMG1-B1 devices also have 21 GPIOs including the I2C and SWD pins that can be used as GPIOs. These GPIOs can be configured to implement various battery monitoring features such as individual battery cell monitoring, battery temperature monitoring, board temperature monitoring, etc. The I2C communication block along with the I2C GPIO pins can be configured to communicate the battery status to an external controller. Implementing some of these features requires an additional external hardware design.

Note: *The individual battery cell and board temperature monitoring are optional features. If the design or the battery pack already includes an onboard battery monitoring system (BMS) to monitor and balance individual battery cells and monitor the board temperature, then skip the hardware implementation of individual cell monitoring and board temperature monitoring. In such cases, these GPIOs can be repurposed for other MCU features, such as communicating with the BMS IC or BMS MCU using communication interfaces like UART or I2C.*

This section describes an external hardware design to implement battery monitoring features such as cell monitoring and battery temperature monitoring.

5.4.1 Battery pack and cell monitoring

In a typical 5-cell Li-ion battery charger design, the maximum voltage measured at positive node of cell 1(C1), cell 2 (C2), cell 3 (C3), cell 4 (C4), and cell 5 (VABT /VOUT_DC) will be 4.2 V, 8.4 V, 12.6 V, 16.8 V, and 21 V respectively. In a 5-cell battery pack, the cell 5 voltage will also be the total battery voltage measured. The battery pack and individual cell voltages need to be scaled down to the EZ-PD™ PMG1-B1 12-bit SAR ADC voltage range for measurement. As described in Section 5.2.1, the 12-bit SAR ADC's single-ended voltage range is from 0 to 2 × Vref (2 × 1.2 V). The individual cell voltages need to be scaled down using an external resistor divider network. The voltage divider network is different for each cell as they are selected to step down the cell voltage to be within the ADC input range (0 to 2.4 V). Each individual battery cell voltages are calculated by subtracting its positive node voltage from the previous cell's positive node voltage using the EZ-PD™ PMG1-B1 MCU firmware. The following figure shows the voltage divider network. Figure 19 also gives an example of individual voltage measured at each cell when total battery voltage is 20 V. In a real system, cell voltages may not always be balanced as shown in the following figure.

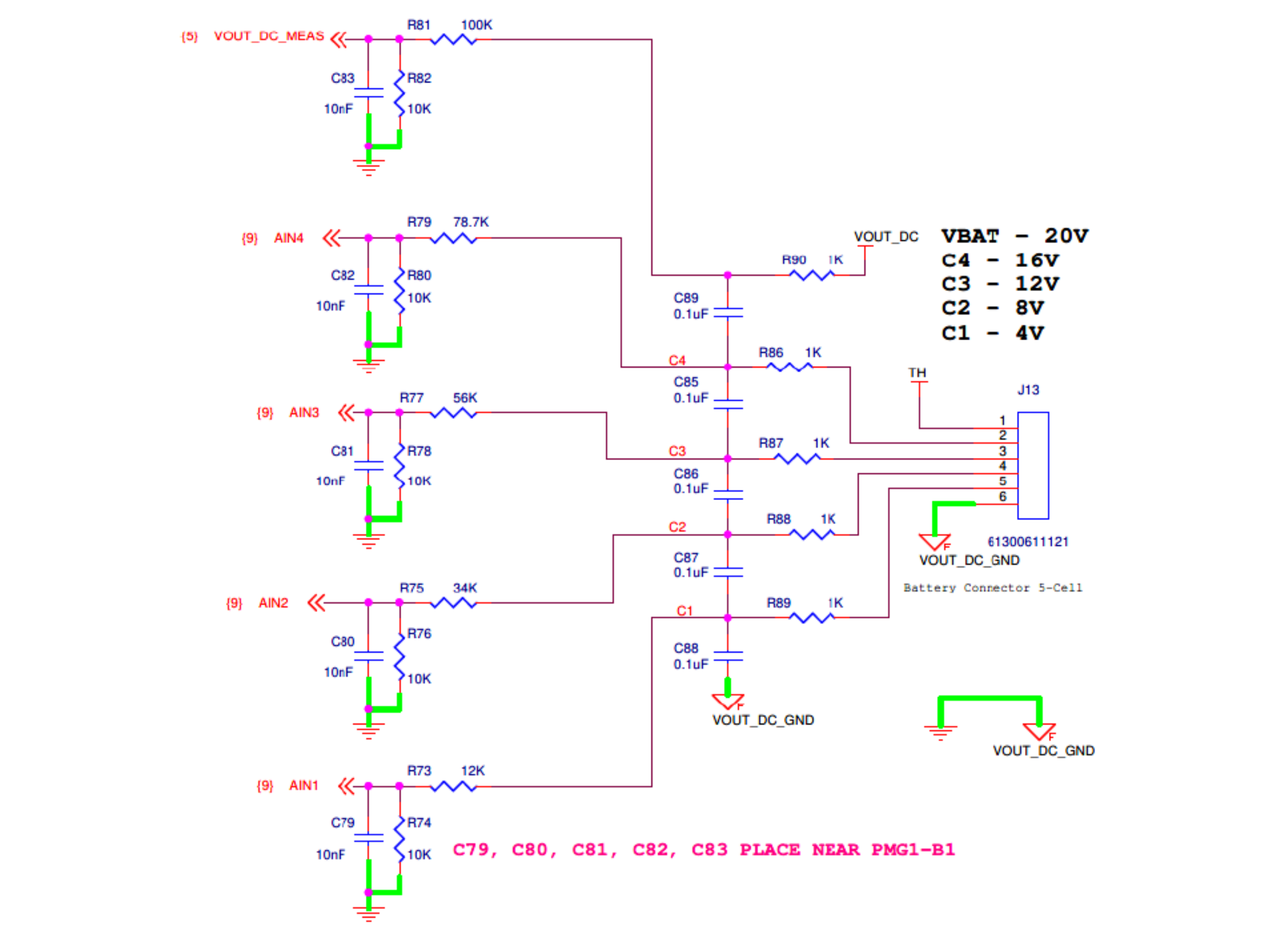

Figure 19 EZ-PD™ PMG1-B1 battery cell voltage monitoring circuit

Having a resistor divider connected to each individual cell and total battery voltage can slowly discharge the battery pack in standby mode when the USB-C adapter is not connected to the EZ-PD™ PMG1 B1 device. An additional external NFET as shown in Figure 20 can be added to enable the resistor divider connection only during the cell voltage measurement. The resistor divider can be disabled after the ADC measurement is complete. This NFET can be controlled using EZ-PD™ PMG1-B1 GPIO.

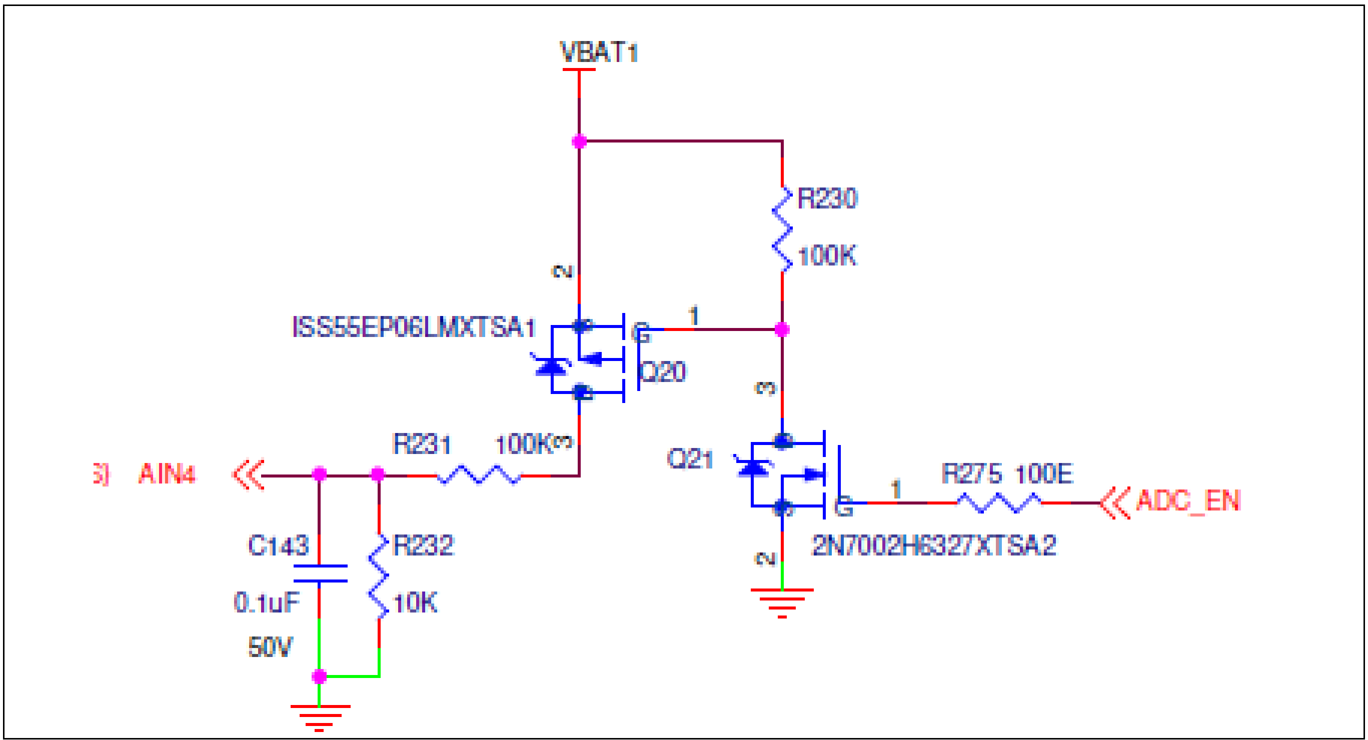

Figure 20 **Battery cell voltage monitoring with external FET to enable/disable external resistor divider**

5.4.2 Battery temperature and board temperature monitoring

In addition to the individual cell monitoring feature, EZ-PD™ PMG1-B1 MCU can also support battery temperature and board temperature monitoring. Typically, Li-ion battery cells charge reliably if the battery pack temperature is between 5°C to 45°C. Charging the Li-ion battery back outside this temperature band affect the longevity of the battery pack. EZ-PD™ PMG1-B1 using its GPIO and 12-bit or 8-bit SAR ADC can measure the battery pack temperature and the board temperature. The measured data can be used by EZ-PD™ PMG1-B1 MCU for effective thermal management and improving the longevity of the battery pack. EZ-PD™ PMG1-B1 can enable the charging of the Li-ion battery pack only if the measured battery pack temperature falls within the configured safe operating limit. If the measured voltage is outside the configured band, EZ-PD™ PMG1-B1 MCU can throttle the charge power or cut-off the battery charging.

To conserve battery pack power during standby mode, an additional NFET can be placed in series with the NTC circuit and this NFET can be controlled using EZ-PD™ PMG1-B1 GPIO as shown in Figure 21. Section 6.4.4 describes in detail how the EZ-PD™ PMG1-B1 battery charger firmware needs to be modified with an appropriate battery pack-specific temperature sensor-related code to accurately measure the battery temperature.

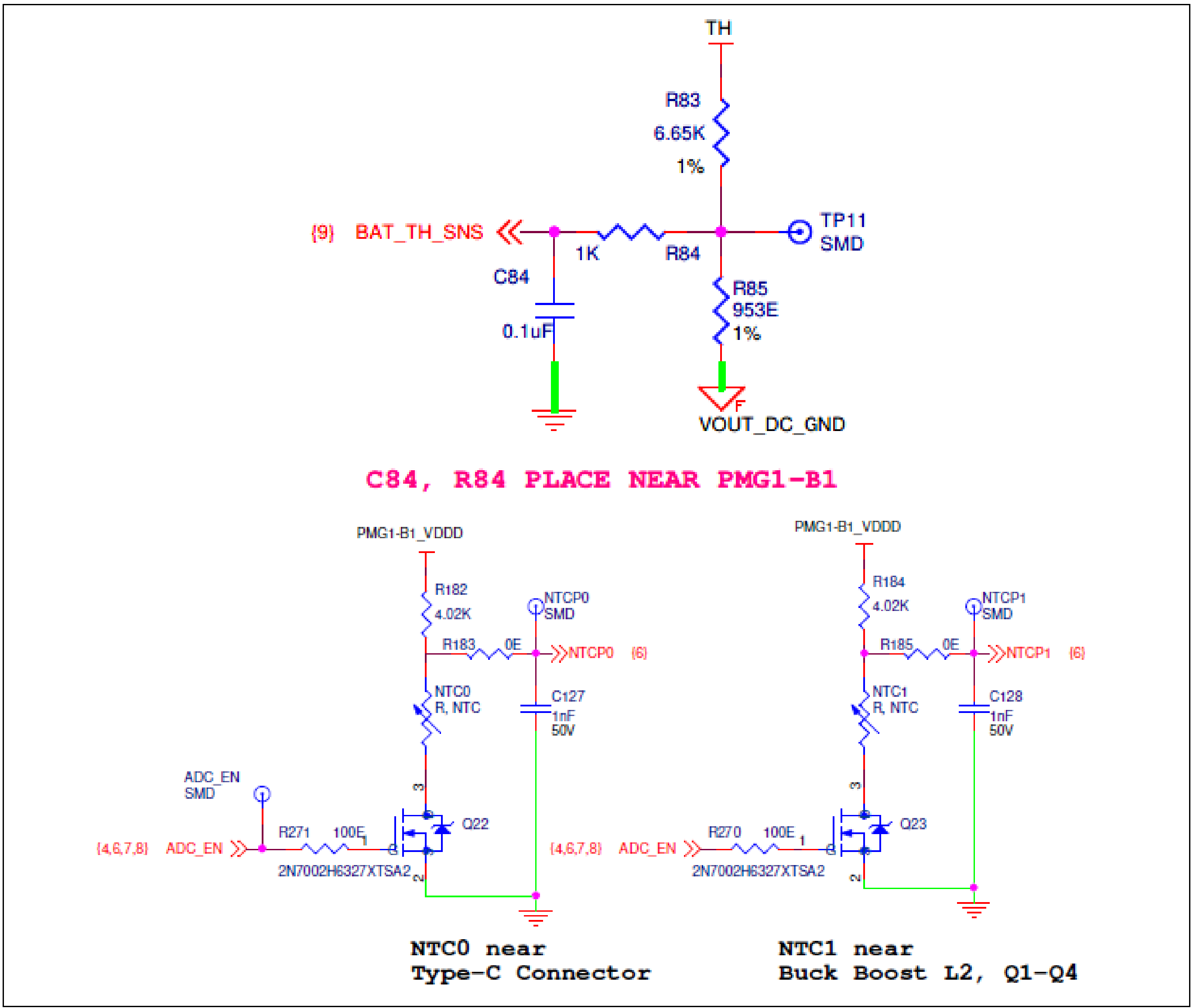

Figure 21 NTC-based board temperature and battery temperature monitoring circuit

6 Firmware overview of EZ-PD™ PMG1-B1 MCU Li-ion battery charger application

This section describes the firmware implementation of USB PD-integrated 2 to 5-cell Li-ion battery charging and battery monitoring system using the EZ-PD™ PMG1-B1 MCU with the ModusToolbox™ software development environment. Configuration of various EZ-PD™ PMG1-B1 MCU peripherals is easily achieved using the various configurator tools provided in the ModusToolbox™ package.

6.1 USB Power Delivery operation

EZ-PD™ PMG1-B1 MCU with the integrated USBPD and buck-boost blocks enable a one-chip solution for USB PD-integrated Li-ion battery charging and battery monitoring applications. The USBPD block of the EZ-PD™ PMG1-B1 MCU implements the USB Type-C and Power Delivery Specification 3.1. The USBPD block enables the EZ-PD™ PMG1-B1 to negotiates up to 100 W (20 V at 5 A) standard power range from the connected USB-C power adapters. The integrated buck-boost block then uses this negotiated power to charge the Li-ion battery pack.

In DRP application, the battery pack voltage can be fed through the integrated buck-boost to provide power over the USB-C connector and charge devices such as phones and laptops. Section 7 describes in detail the implantation of bidirectional DRP battery charger design with external FETs. Optionally, an external buck regulator can be used to provide power from the battery in source mode to charge devices such as phones.

6.1.1 Supported sink PDO

Table 6 PDO specification of the firmware

Sink PDO type	Voltage	Operational current	Min operating current	Power
Fixed supply PDO	5 V	0.9 A	0.9 A	4.5 W (sink minimum PDP)
Variable supply PDO	5 V to 21 V	0.9 A	0.9 A	100 W (sink maximum PDP)

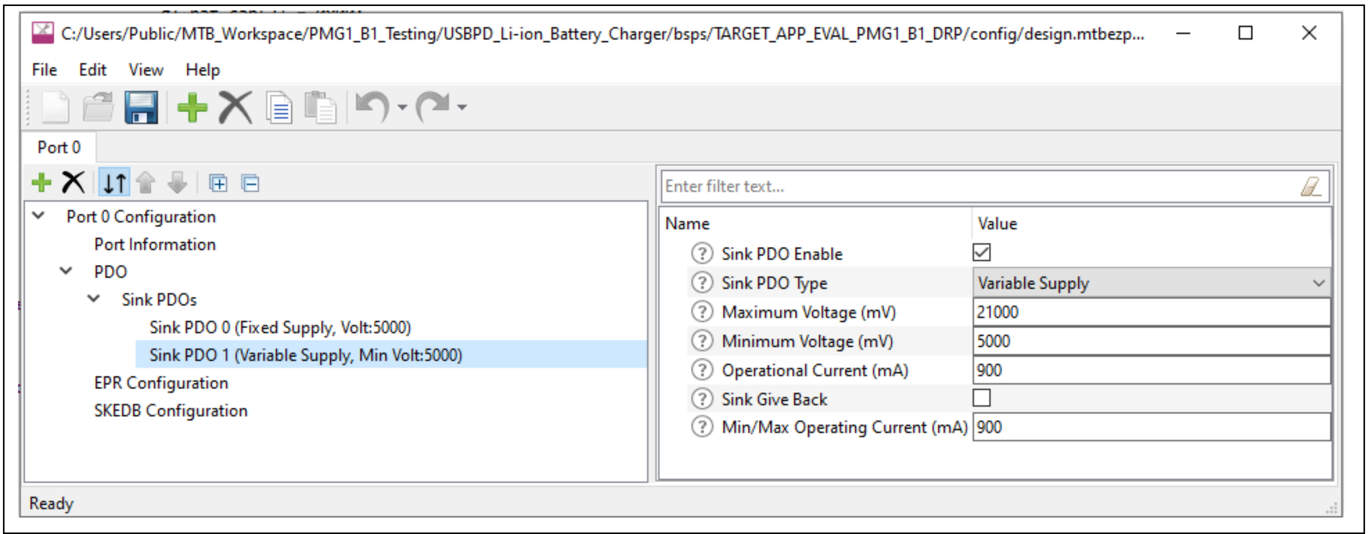

Figure 22 PDO configuration using EZ-PD™ Configurator tool of ModusToolbox™

6.1.2 Legacy charging support

EZ-PD™ PMG1-B1 MCU supports legacy battery charger block that enables emulation and detection (source and sink) of various legacy charging protocols such as USB BC.1.2, Apple charging, etc. This allows the EZ-PD™ PMG1-B1 devices to charge the Li-ion battery pack even when connected to legacy USB 2.0 Type-A port using a USB-C to USB-A cable. EZ-PD™ PMG1-B1 MCU battery charger firmware by default provides support for USB BC1.2 and legacy Apple charging protocol. This feature can be enabled or disabled using the Project Makefile as shown in the following figure.

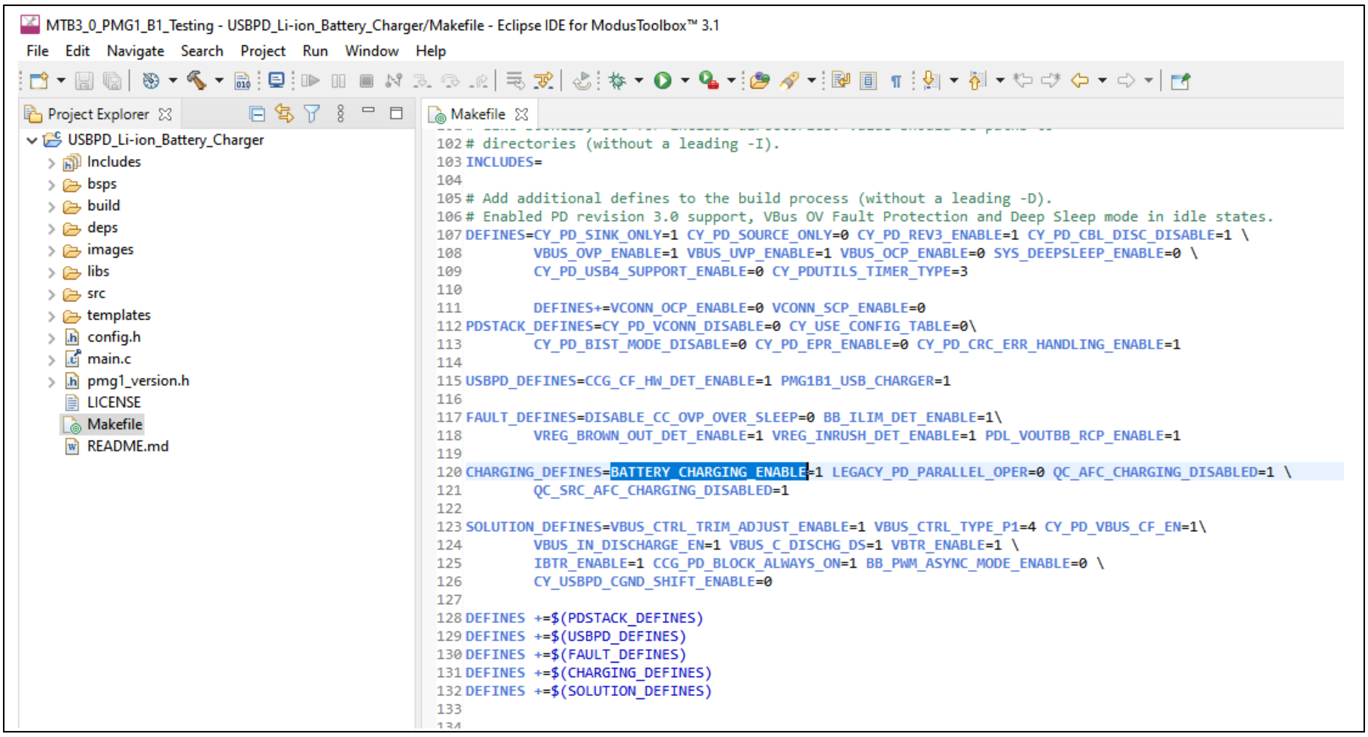

Figure 23 Enable/disable legacy charging using Li-ion battery charger project Makefile

6.2 Configuring battery cell count (2-5 cell) for battery charger application

EZ-PD™ PMG1-B1 MCU's programmable buck-boost subsystem can be configured to output any voltage from 3.3 V to 21.5 V. Therefore, EZ-PD™ PMG1-B1 MCU firmware can be configured to support 2 to 5-cell battery charging design.

When the input VBUS voltage is less than the output battery voltage (VBUS << VBAT), EZ-PD™ PMG1-B1 device operates in the boost region. When the input VBUS voltage is greater than the battery voltage, EZ-PD™ PMG1-B1 device operates in the buck region. When VBUS is slightly higher than VBAT or VBUS is slightly lower than VBAT, the device operates in the buck-boost region.

6.2.1 Modifying battery cell count

Modify the `TOTAL_BATTERY_CELL_COUNT` macro as shown in Figure 24 to update the battery cell count. The macro is located in the *config.h* file. EZ-PD™ PMG1-B1 MCU battery charger firmware measures the battery voltage and automatically updates the buck-boost output voltage to charge the 2-5 cell batteries based on the cell count. The default cell count in the firmware is 5-cell.

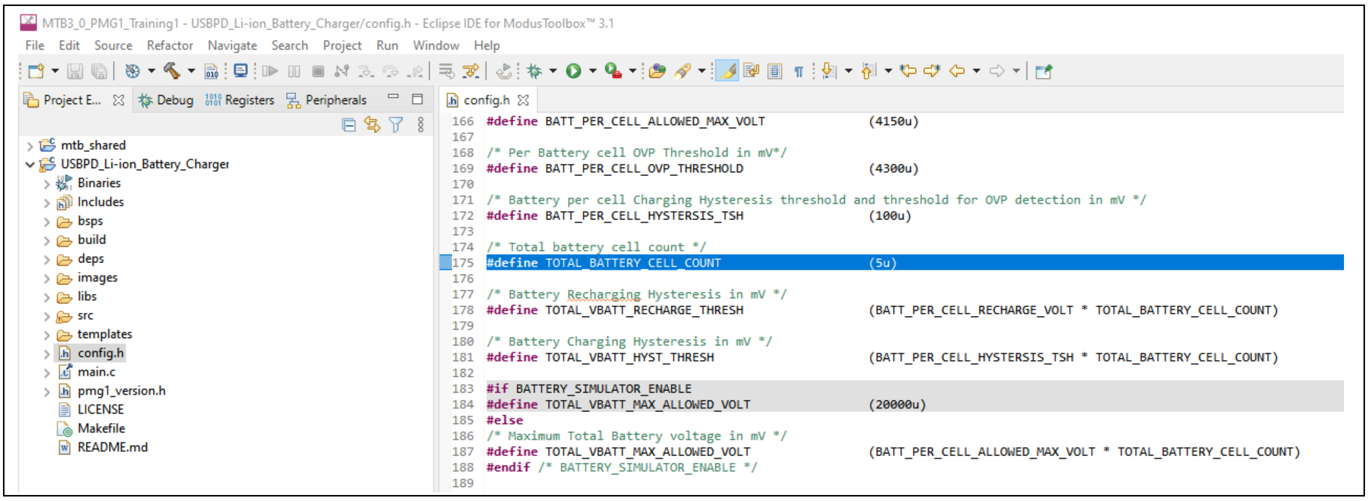

Figure 24 **Modifying battery cell count in EZ-PD™ PMG1-B1 firmware using ModusToolbox™**

6.2.2 Modifying battery charge current

The programmability of the current regulation error amplifier in the EZ-PD™ PMG1-B1 buck-boost subsystem along with the configurability of the current sense resistor allow the buck-boost output current to be configured from 150 mA to 6.5 A in EZ-PD™ PMG1-B1 MCU firmware using the ModusToolbox™ ecosystem.

Modify the `VBAT_INPUT_CURR_MAX_SETTING` macro as shown in Figure 25 to update the maximum battery charging current. The maximum battery current is expressed in a multiple of 10 mA. The default maximum current setting in EZ-PD™ PMG1-B1 MCU firmware is 200 u and it indicates a maximum current of 2 A. This value can be set from 15 u to 650 u and this corresponds to 150 mA to 6.5 A.

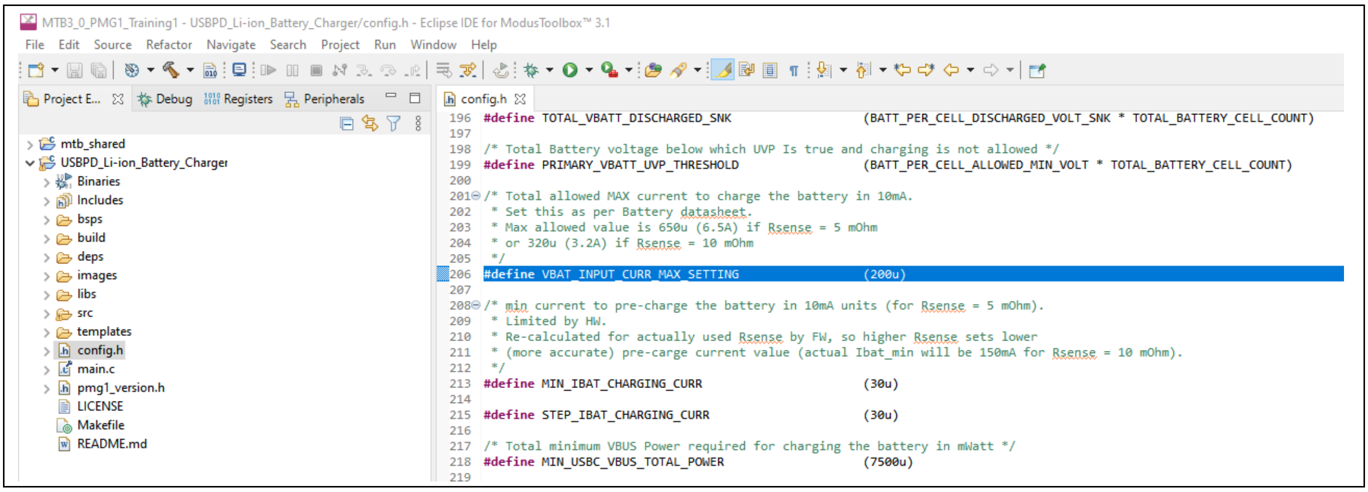

Figure 25 **Modifying battery cell count in EZ-PD™ PMG1-B1 firmware using ModusToolbox™**

6.3 Smart battery charging algorithm

To ensure longevity and maximum capacity utilization of battery, there are various charging methods such as trickle charge, constant current (CC), constant voltage (CV), and top-off charge are used to charge the battery based on the state of charge (SoC) and temperature of the battery. In CC mode, the charging current is constant and in CV mode, the output voltage to the battery pack made constant.

EZ-PD™ PMG1-B1 MCU also supports a smart charging algorithm in which the MCU calculates the input power available from the USB-C adapters, then periodically measures the battery output voltage and dynamically updates the battery charging current. This enables maximum utilization of the available input USB-C power.

The Code Listing 1 of code in the *solution.c* file demonstrates how EZ-PD™ PMG1-B1 MCU firmware dynamically updates the buck-boost output voltage and current based on the input power available from the connected USB-C adapter and the measured battery voltage.

Code Listing 1 Dynamic calculation of buck-boost output voltage based on measured battery voltage and input supply

```
/* Update the Buck boost output voltage and current based on the USB-C Input
power and current battery voltage */
void calc_buck_boost_out_pwr_settings (cy_stc_pdstack_context_t*
ptrPdStackContext)
{
    cy_stc_battery_charging_context_t* ptrBatteryChargingContext =
get_battery_charging_context(ptrPdStackContext->port);
    cy_stc_battery_status_t* batt_stat = &(ptrBatteryChargingContext-
>batteryStatus);
    uint16_t calc_batt_ip_volt = TOTAL_VBATT_MAX_ALLOWED_VOLT;
    uint32_t calc_output_power;
    uint32_t calc_input_power;

#if BATTERY_CHARGING_ENABLE
    if(bc_get_status(ptrPdStackContext)->connected == true)
    {
        calc_output_power = (uint32_t) ((CY_PD_VSAFE_5V *
bc_get_status(ptrPdStackContext)->cur_amp)
                                        *
INPUT_OUTPUT_EFFICIENCY_REDUCE_PERCENTAGE) / 100;
    }
    else
#endif
    {
        calc_output_power = (uint32_t)((ptrPdStackContext-
>dpmStat.contract.minVolt * ptrPdStackContext->dpmStat.contract.curPwr)
                                        *
INPUT_OUTPUT_EFFICIENCY_REDUCE_PERCENTAGE) / 100;
    }

    /* Source 5% voltage tolerance */
    calc_output_power = (calc_output_power * VBUS_TOLERANCE_PERCENTAGE) / 100;
```

```
    /* Manufacture efficiency */
    calc_output_power = (calc_output_power *
MANUFACTURE_EFFICIENCY_PERCENTAGE) / 100;

#if ENABLE_ALL_BATT_MONITORING
    calc_batt_ip_volt = batt_stat->curr_batt_volt + TOTAL_VBATT_HYST_THRESH;

    if(calc_batt_ip_volt > TOTAL_VBATT_MAX_ALLOWED_VOLT)
    {
        calc_batt_ip_volt = TOTAL_VBATT_MAX_ALLOWED_VOLT;
    }
#else
    calc_batt_ip_volt = TOTAL_VBATT_MAX_ALLOWED_VOLT;
#endif

    batt_stat->cur_bb_pwr = calc_output_power;
    batt_stat->cur_bb_vout = calc_batt_ip_volt;

    /* Start with min current at first */
    batt_stat->cur_bb_iout = update_current_limit (ptrPdStackContext,
MIN_IBAT_CHARGING_CURR);

#if BATTERY_CHARGING_ENABLE
    if(bc_get_status(ptrPdStackContext)->connected == true)
    {
        calc_input_power = (CY_PD_VSAFE_5V * bc_get_status(ptrPdStackContext)-
>cur_amp) / 100;
    }
    else
#endif
    {
        calc_input_power = (ptrPdStackContext->dpmStat.contract.minVolt *
ptrPdStackContext->dpmStat.contract.curPwr) / 100;
    }

    if(calc_input_power >= MIN_USBC_VBUS_TOTAL_POWER)
    {
        DEBUG_PRINT("\n Input Power sufficient");
        batt_stat->is_vbus_pwr_sufficient = true;
    }
```

```
    else
    {
        DEBUG_PRINT("\n Input Power NOT sufficient");
        batt_stat->is_vbus_pwr_sufficient = false;
    }

    DEBUG_PRINT("\r\n BAT VA UPDATE ");
    DEBUG_PRINT_VAR("\n BB Out Curr:: %d mA\n", (batt_stat->cur_bb_iout *
10));
    DEBUG_PRINT_VAR("\n BB Out Volt:: %d mV\n", batt_stat->cur_bb_vout);
    DEBUG_PRINT_VAR("\n BB Out Pwr:: %ld mW\n", batt_stat->cur_bb_pwr);
}
```

6.3.1 Multi-stage constant current mode charging

In constant current (CC) mode, a fixed current is used to charge the battery continuously. A high charging current charges the battery faster. However, it significantly affects the life span of the battery. While a low charging current can provide high-capacity utilization, it takes a longer time to charge the battery and is inconvenient for the users. Therefore, an optimum charging strategy based on the battery capacity must be chosen to increase the charge capacity and decrease the charging periods without increasing the battery temperature or compromising the battery life.

Implementing a multi-stage constant current battery charging algorithm provides better utilization of the battery capacity. After the battery voltage reaches the specified cut-off voltage, the charging current can be reduced to 40%, 20%, and 6% (these % current steps can be customized) of the rated current respectively when the terminal voltage reaches the specified cut-off voltage. Figure 26 shows an EZ-PD™ PMG1-B1 MCU-based multi-stage battery charging profile of a 5-cell Li-ion battery with support for the smart charging algorithm. Observe the battery current is periodically updated as the measured battery voltage increases in the following figure.

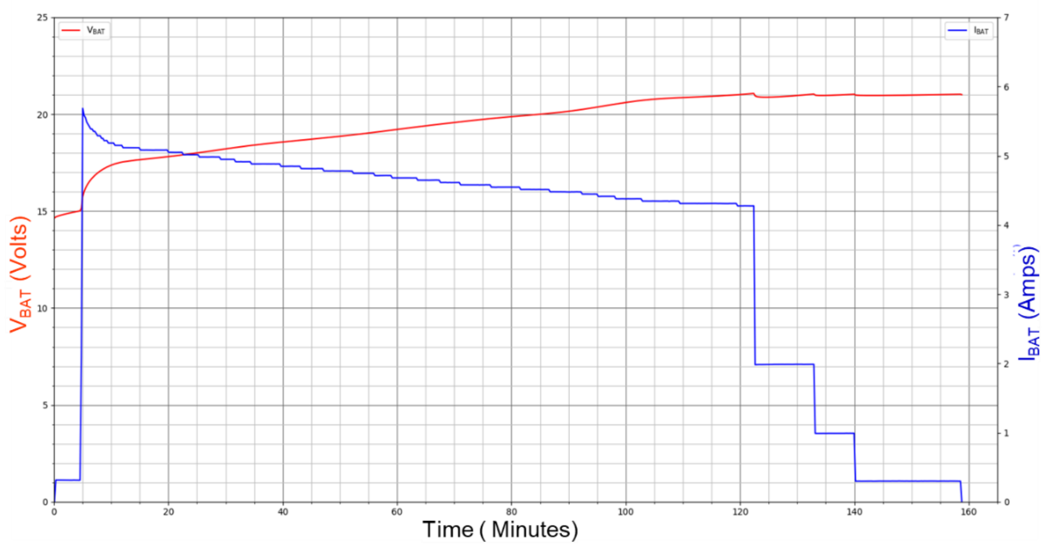

Figure 26 **Multistage constant current battery charging profile for a 5-cell 10 Ah Li-ion battery**

6.3.2 Mixed constant current and constant voltage mode charging

In this method, the charging starts with constant current (CC) mode and when the battery terminal voltage reaches the maximum safe threshold value, the charging mode transfers to the constant voltage (CV) charging method. The charging process is complete when the charging current goes below the cut-off threshold. The constant current accelerates the charging time and the constant voltage mode influences the capacity utilization of the battery.

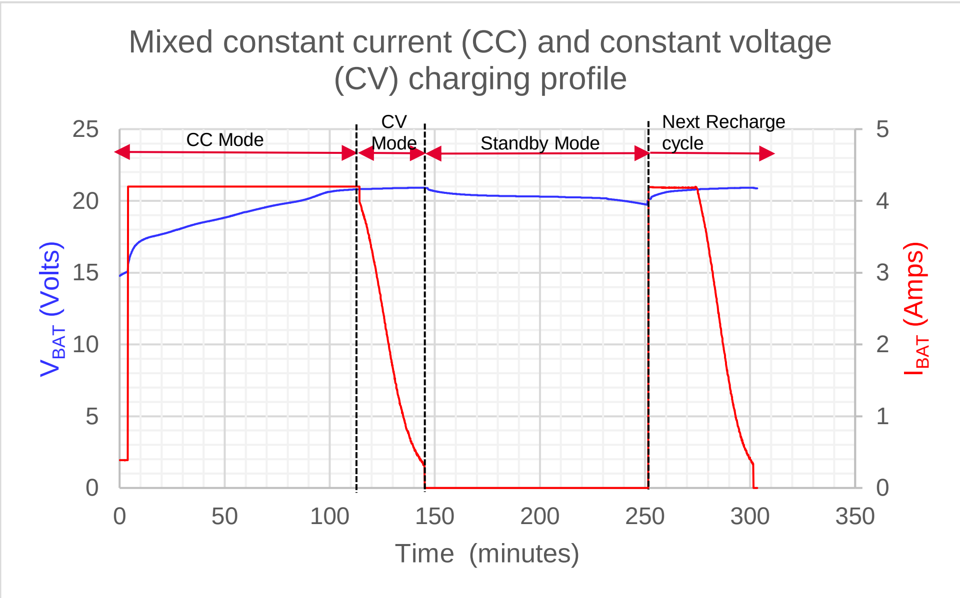

Figure 27 **Constant current (CC) and constant voltage (CV) modes charging profile for a 5-cell 10 Ah Li-ion battery**

Mixed CC-CV charging mode provides longer battery life and greater safety. The constant current mode prevents overcurrent charging and constant voltage charging prevent overvoltage. If the battery voltage goes below a specified threshold, then the charging can be resumed automatically (Figure 27).

The default EZ-PD™ PMG1-B1 MCU firmware supports the mixed constant current and constant voltage mode charging. The firmware can be modified to support the multi-stage constant current mode or other algorithms preferred by the manufacturer of the battery charging device.

6.3.3 Pre-charging/trickle charging support

Typically, the operating voltage range of a Li-ion battery are 3.0 V to 4.2 V. In a 5-cell design, the total battery pack voltage varies from 15.0 V to 20.5 V. However, if the battery pack is deeply discharged, the pack voltage will be less than 15.0 V. When battery packs are discharged below 3 V/cell, EZ-PD™ PMG1-B1 MCU pre-charges the battery pack at a low-rate until the individual cell voltages and the total battery pack voltage reaches the required limit. For example, in a 5-cell design this is typically set to 3 V per cell and 15 V for total battery pack respectively.

Use the `TRICKLE_CHARGE_ENABLE` macro to enable this trickle charging support in the EZ-PD™ PMG-B1 MCU firmware. By default, this feature is disabled and it can be enabled by writing '1u' to the macro located in the *config.h* file.

```
/* Enable trickle charge */
#define TRICKLE_CHARGE_ENABLE                              (0u)
```

EZ-PD™ PMG1-B1 MCU firmware allows you to configure the trickle charging current using the `MIN_IBAT_CHARGING_CURR` macro. The current is set in multiple of 10 mA. The default trickle charging current is set to 300 mA. For a 5 mΩ Rsense, 300 mA is the recommended trickle charging current. For a lower trickle charging current of 150 mA, the Rsense needs to be 10 mΩ.

```
/* min current to pre-charge the battery in 10mA units (for Rsense = 5 mOhm).
 * Limited by HW.
 * Re-calculated for actually used Rsense by FW, so higher Rsense sets lower
 * (more accurate) pre-charge current value (actual Ibat_min will be 150mA for Rsense = 10 mOhm).
 */
#define MIN_IBAT_CHARGING_CURR                             (30u)
```

EZ-PD™ PMG1-B1 MCU firmware can set the minimum and maximum cell voltage and total battery pack voltage for trickle charging. If the battery pack voltage falls below the minimum threshold, EZ-PD™ PMG1-B1 MCU will not initiate charging. However, if the voltage is within the trickle charging thresholds, EZ-PD™ PMG1-B1 buck-boost will continue to trickle charge the battery until the pack voltage reaches the maximum limit. After the maximum voltage is reached, EZ-PD™ PMG1-B1 MCU will switch from trickle charging to constant current charging and charge the battery pack with the maximum current defined in Section 6.2.2.

The `PRIMARY_VBATT_UVP_THRESHOLD` macro sets the total battery voltage that must be maintained for charging to occur. In a 5-cell setup, with a minimum recommended voltage of 2.0 V per cell, EZ-PD™ PMG1-B1 MCU will only initiate charging if the measured battery pack voltage is more than 10 V. This will activate the trickle charging mode.

```
/* Min allowed battery cell voltage per cell in mV */
#define BATT_PER_CELL_ALLOWED_MIN_VOLT                     (2000u)


/* Total Battery voltage below which UVP Is true and charging is not allowed */
#define PRIMARY_VBATT_UVP_THRESHOLD
(BATT_PER_CELL_ALLOWED_MIN_VOLT * TOTAL_BATTERY_CELL_COUNT)
```

Likewise, the `TOTAL_VBATT_DISCHARGED_SNK` macro specifies the maximum voltage for trickle charging. For instance, in a 5-cell configuration with 3.0 V per cell, the value will be 15.0 V based on the macro definition.

```
/* Completely discharged battery cell voltage per cell in mV/SNK */
#define BATT_PER_CELL_DISCHARGED_VOLT_SNK                  (3000u)
/* Total voltage of completely discharged Battery/SNK role */
#define TOTAL_VBATT_DISCHARGED_SNK
(BATT_PER_CELL_DISCHARGED_VOLT_SNK * TOTAL_BATTERY_CELL_COUNT)
```

6.4 Battery monitoring algorithm overview

EZ-PD™ PMG1-B1 MCU implements the battery charging algorithm and monitors various battery parameters such as total battery voltage, battery charge current, battery temperature, individual cell voltages, board temperature, etc., to ensure that the battery packs are charged safely. EZ-PD™ PMG1-B1 MCU has built-in analog peripherals such as a 12-bit SAR ADC, 8-bit SAR ADC, CSA, UVOV block, etc., to implement these various battery monitoring features.

See Section 5.2.1 on how to configure the 12-bit SAR ADC for battery and cell voltage monitoring using the ModusToolbox™ Device Configurator tool. EZ-PD™ PMG1-B1 MCU battery charger firmware allows to enable/disable and modify the various battery monitoring thresholds parameters easily. This section describes in detail how to enable/disable and configure the battery monitoring parameters.

6.4.1 Battery pack voltage monitoring

Excessive charging of Li-ion battery pack can result in thermal runaway and other safety-related issues. So, sensing battery pack voltage is highly critical in a battery charging application. EZ-PD™ PMG1-B1 MCU uses its GPIO and the 12-bit SAR ADC to monitor the total battery pack voltage. The minimum and maximum battery pack voltage threshold between which EZ-PD™ PMG1-B1 MCU charges the battery pack can be modified using the firmware.

The `BATT_PER_CELL_ALLOWED_MAX_VOLT` macro allows to configure the maximum voltage allowed per cell in the battery pack. The firmware uses this value along with the cell count to calculate the maximum allowed battery pack voltage as shown in the following code snippet from the *config.h* file. EZ-PD™ PMG1-B1 MCU initiates charging only if the measured battery pack voltage is below `TOTAL_VBATT_MAX_ALLOWED_VOLT` and it continues to charge the battery pack until the battery pack voltage reaches the `TOTAL_VBATT_MAX_ALLOWED_VOLT`. The charging is cut off after the battery pack reaches this threshold.

```
/* Total battery cell count */
#define TOTAL_BATTERY_CELL_COUNT                    (5u)

/* Maximum allowed battery cell voltage per cell in mV */
#define BATT_PER_CELL_ALLOWED_MAX_VOLT              (4150u)
/* Maximum Total Battery voltage in mV */
#define TOTAL_VBATT_MAX_ALLOWED_VOLT
(BATT_PER_CELL_ALLOWED_MAX_VOLT * TOTAL_BATTERY_CELL_COUNT)
```

The `ENABLE_ALL_BATT_MONITORING` macro allows to configure the enabling and disabling of the total battery voltage monitoring code. If this MACRO is set to 0u, the firmware uses the maximum allowed battery voltage `TOTAL_VBATT_MAX_ALLOWED_VOLT` to compute the battery charging current instead of the measured battery voltage as described in Section 6.3. Therefore, this results in reduced charge current.

Note: It is recommended not to disable this MACRO.

```
/* This macro enables /disables all the battery monitoring code */
#define ENABLE_ALL_BATT_MONITORING                 (1u)
```

See Section 5.4.1 about the hardware design required for the battery pack voltage monitoring. The following code function defined in the *solution_task.c* implements the battery pack voltage monitoring task.

```
void run_battery_volt_monitor_task (cy_stc_battery_charging_context_t
*ptrBatteryChargingContext)
```

6.4.2 Individual battery cell voltage monitoring

EZ-PD™ PMG1-B1 MCU can use its 12-bit SAR ADC and GPIOs to accurately monitor individual battery cell voltages in addition to the total battery pack voltage. The real-time measurement of battery cell voltages during the battery charging process ensures that no individual Li-ion battery cell in the battery pack is charged more than 4.15 V and therefore, ensure the battery safety by preventing thermal runaway of any individual cells in the battery pack.

The individual battery cell monitoring feature requires the battery charger manufacturer to connect the individual cells in the battery pack to EZ-PD™ PMG1-B1 GPIOs. See Section 5.4.1 for the required hardware connection. By default, EZ-PD™ PMG1-B1 MCU battery charger firmware has this feature disabled and it can be enabled by setting the `CELL_MONITORING_DISABLE` macro to '0u'.

```
/* Disable individual cell voltage calculation
 * and protection checks by default.
 * Set this to 0 if cell voltage monitoring is need
 * and appropriate cable is connected.
 */
#define CELL_MONITORING_DISABLE                     (1u)
```

The various individual battery cell thresholds are configured by updating the following macros.

```
/* Min allowed battery cell voltage per cell in mV */
#define BATT_PER_CELL_ALLOWED_MIN_VOLT              (2000u)

/* Completely discharged battery cell voltage per cell in mV/SNK */
#define BATT_PER_CELL_DISCHARGED_VOLT_SNK           (3000u)

/* Completely discharged battery cell voltage per cell in mV/SRC */
#define BATT_PER_CELL_DISCHARGED_VOLT_SRC           (3100u)

/* Battery cell Recharge voltage per cell in mV */
#define BATT_PER_CELL_RECHARGE_VOLT                 (4000u)

/* Maximum allowed battery cell voltage per cell in mV */
#define BATT_PER_CELL_ALLOWED_MAX_VOLT              (4150u)
```

The individual battery cell monitoring feature is integrated along with the battery pack voltage monitoring function. See the Code Listing 2 code function defined in the *solution_task.c* file for more details on the battery cell voltage monitoring implementing in the EZ-PD™ PMG1-B1 MCU battery charger firmware.

Code Listing 2 Battery pack voltage and individual cell voltage monitoring using EZ-PD™ PMG1-B1 MCU firmware

```
void run_battery_volt_monitor_task (cy_stc_battery_charging_context_t
*ptrBatteryChargingContext)
{
..
#if (CELL_MONITORING_DISABLE == 0)
    /* Monitor each cell voltage and if any cell voltage is higher than
4.2V disabled Charging */
    for(cell_num = 0; cell_num < TOTAL_BATTERY_CELL_COUNT; cell_num++ )
    {
        switch(cell_num)
        {
            case 0: curr_cell_volt = adcResult->cell1;
#if PRINT_CV
            cell[0]=curr_cell_volt;
#endif /* PRINT_CV */
                break;
            case 1: curr_cell_volt = adcResult->cell2 - adcResult->cell1;
#if PRINT_CV
            cell[1]=curr_cell_volt;
#endif /* PRINT_CV */
                break;
            case 2: curr_cell_volt = adcResult->cell3 - adcResult->cell2;
#if PRINT_CV
            cell[2]=curr_cell_volt;
#endif /* PRINT_CV */

#if SIMULATE_ERROR_SNK
            if(gl_sln_batt_chg_state == BATT_CHG_CHARGE_LOOP)
            {
                if(Cy_PdUtils_SwTimer_IsRunning(PdStackContext-
>ptrTimerContext, BATT_TIMER_ID) == false)
                {
                    curr_cell_volt = SIMULATE_CELL_VOLT;
                }
            }
#endif
                break;
```

Code Listing 2 **Battery pack voltage and individual cell voltage monitoring using EZ-PD™ PMG1-B1 MCU firmware**

```
            case 3: curr_cell_volt = adcResult->cell4 - adcResult->cell3;
#if PRINT_CV
            cell[3]=curr_cell_volt;
#endif /* PRINT_CV */
                break;
            case 4: curr_cell_volt = adcResult->vbat - adcResult->cell4;
#if PRINT_CV
            cell[4]=curr_cell_volt;
#endif /* PRINT_CV */
                break;
            default:
                break;
        }
#if DEBUG_UART_ENABLE && DEBUG_BATT_INFO_ENABLE &&
DEBUG_BATT_CELL_INFO_ENABLE
        sprintf(temp, "\n Cell %d voltage is: %i mV",cell_num,
curr_cell_volt);
        debug_print( temp);
#endif

        if(curr_cell_volt > BATTERY_REMOVAL_THRESHOLD)
        {
            battery_removed = false;
        }

        /* Deny recharging if at least one cell exceeds threshold */
        if(curr_cell_volt > BATT_PER_CELL_RECHARGE_VOLT)
        {
            is_cell_recharge = false;
        }

        /* check if separate cell is in full charge in CV mode */
        if(curr_cell_volt > BATT_PER_CELL_ALLOWED_MAX_VOLT)
        {
            /* Cell voltage is more than 4.2V , so battery cell is fully
charged */
            is_cell_chrg_full = true;
        }
```

Code Listing 2 Battery pack voltage and individual cell voltage monitoring using EZ-PD™ PMG1-B1 MCU firmware

```
        if(curr_cell_volt >= BATT_PER_CELL_OVP_THRESHOLD)
        {
            /* Cell voltage is more than 4.3V . Battery is in OVP */
            is_batt_in_ovp = true;
        }

        /* SNK case */
        if(curr_cell_volt < BATT_PER_CELL_ALLOWED_MIN_VOLT)
        {
            /* Cell voltage is less 3V , so cell is in UVP */
            is_batt_in_uvp = true;
        }
    }
#endif /* (CELL_MONITORING_DISABLE == 0) */
..
}
```

6.4.3 Battery current monitoring

EZ-PD™ PMG1-B1 MCU using its built-in CSA and an external 5 mΩ series Rsense continuously monitor the battery charge current and ensures the battery pack is charged safely within the limits recommended by the vendors. EZ-PD™ PMG1-B1 MCU battery charger firmware allows to enable/disable the battery current monitoring as well as specify the battery charge current threshold using the following macros.

Code Listing 3 Configuring battery charge current thresholds using EZ-PD™ PMG1-B1 MCU firmware

```
/* This macro enables /disables the battery current monitoring code */
#define ENABLE_BATT_CURR_MONITORING                 (1u)
/* Total allowed MAX current to charge the battery in 10mA.
 * Set this as per Battery datasheet.
 * Max allowed value is 650u (6.5A) if Rsense = 5 mOhm
 * or 320u (3.2A) if Rsense = 10 mOhm
 */
#define VBAT_INPUT_CURR_MAX_SETTING                 (200u)
/* min current to pre-charge the battery in 10mA units (for Rsense = 5
mOhm).
 * Limited by HW.
```

Code Listing 3 Configuring battery charge current thresholds using EZ-PD™ PMG1-B1 MCU firmware

```
 * Re-calculated for actually used Rsense by FW, so higher Rsense sets
lower
 * (more accurate) pre-charge current value (actual Ibat_min will be 150mA
for Rsense = 10 mOhm).
 */
#define MIN_IBAT_CHARGING_CURR                      (30u)

#define STEP_IBAT_CHARGING_CURR                     (30u)
```

By default, the maximum charge current macro is set to '200u' corresponding to 2 A charge current. EZ-PD™ PMG1-B1 MCU allows to set maximum charge current up to 6.5 A for a 5 mΩ Rsense. With 10 mΩ Rsense, the maximum charge current recommended is 3.2 A for accurate current measurement.

Upon connecting to USB-C charger, EZ-PD™ PMG1-B1 MCU starts charging the battery pack with the current specified using the `MIN_IBAT_CHARGING_CURR` macro and it increases the current gradually in steps specified using the `STEP_IBAT_CHARGING_CURR` macro until the battery charging current reach the current specified using the `VBAT_INPUT_CURR_MAX_SETTING` macro. For example, in the default code, the firmware starts charging the battery pack with 300 mA of current. It will increase the current in steps of 300 mA until the maximum current of 2 A is reached and will then continue to charge the battery at 2 A until the battery pack voltage reaches the `TOTAL_VBATT_MAX_ALLOWED_VOLT`.

If a USB-C adapter with lower output capability is connected, EZ-PD™ PMG1-B1 MCU then uses the smart battery charging algorithm described in Section 6.3 to dynamically calculate the maximum battery charging current and sets the buck-boost output accordingly. EZ-PD™ PMG1-B1 MCU using its CSA block and series resistor continues to monitor the battery current and ensures the measured charging current is within the specified limit. In the event the battery charge current exceeds the specified limit, EZ-PD™ PMG1-B1 buck-boost will cut-off charging the battery pack. Code Listing 4 code defined in the *solution_task.c* file implements the battery pack current monitoring task.

Code Listing 4 Battery current monitoring task

```
void run_battery_curr_monitor_task (cy_stc_battery_charging_context_t
*ptrBatteryChargingContext)
{
    cy_stc_battery_status_t* batt_stat = &(ptrBatteryChargingContext-
>batteryStatus);
    cy_stc_pdstack_context_t * PdStackContext = ptrBatteryChargingContext-
>ptrPdStack;
    uint32_t batt_cur_ocp_limit = batt_stat->cur_bb_iout;

    batt_stat->curr_batt_curr = Cy_USBPD_Hal_MeasureCur(PdStackContext-
>ptrUsbPdContext);
    DEBUG_PRINT_VAR("\n Measured IBAT: %i mA", batt_stat->curr_batt_curr *
10);
```

Code Listing 4 Battery current monitoring task

```
    batt_cur_ocp_limit = PRIMARY_IBATT_OCP_CURR_THRESHOLD;

#if DEBUG_BATT_INFO_ENABLE
    DEBUG_PRINT_VAR("\n BatOCP set to: %lu mA", (batt_cur_ocp_limit * 10));
#endif

#if BATT_PRI_OCP_ENABLE
    if(batt_stat->curr_batt_curr > batt_cur_ocp_limit)
    {
        batt_stat->batt_ocp_fault_active = true;
        DEBUG_PRINT("\n Battery OCP is active");
    }
    else
    {
        batt_stat->batt_ocp_fault_active = false;
    }
#endif /* BATT_PRI_OCP_ENABLE */
}
```

6.4.4 Battery temperature monitoring

At lower temperatures, the internal resistances of the battery increase and the battery capacity decrease. At colder temperatures, it is recommended to charge the battery using reduced current to prevent the batteries from getting damaged. Similarly, charging a hot battery at full current can result in the battery catching fire. So, measuring the battery temperature and modulating or terminating the battery charging process is critical for safety and for increased life of the battery.

EZ-PD™ PMG1-B1 MCU uses its 12-bit SAR ADC can measure the battery temperature based on the battery pack vendor provided information. Because the battery pack temperature sensor varies design to design, this feature is disabled by default using the `ENABLE_BATT_TEMP_MONITORING` macro in the *config.h* file.

```
/* Enable battery temperature monitoring */
#define ENABLE_BATT_TEMP_MONITORING                    (0u)
```

EZ-PD™ PMG1-B1 MCU battery charger code example provides a sample thermistor-based temperature measurement as an example demonstrating the battery temperature measurement. To enable battery temperature monitoring, set the `ENABLE_BATT_TEMP_MONITORING` to '1u' in the *config.h* file and update the following section of the code located in the *solution_tasks.c* file with an appropriate battery pack temperature sensor-related code to accurately measure the battery temperature.

Code Listing 5 Battery temperature measurement function

```
/* Thermistor resistance array */
#define TEMP_MAP_MIN                     (-25)
#define TEMP_MAP_MAX                     (70)
#define TEMP_MAP_RESOLUTION_BATT         (5)
#define TEMP_MAP_TABLE_COUNT             (20)
const uint32_t gl_res_temp_map[TEMP_MAP_TABLE_COUNT] =
{
    87717, /* -25C */
    68424, /* -20C */
    53752, /* -15C */
    42558, /* -10C */
    33963, /* -5C  */
    27305, /*  0C  */
    22097, /*  5C  */
    17985, /*  10C */
    14716, /*  15C */
    12101, /*  20C */
    10000, /*  25C */
    8305,  /*  30C */
    6932,  /*  35C */
    5815,  /*  40C */
    4902,  /*  45C */
    4152,  /*  50C */
    3533,  /*  55C */
    3019,  /*  60C */
    2589,  /*  65C */
    2229   /*  70C */
};

/* Sum of resistors according to schematic 1k+6,65k */
#define R240     (7650u)
#define R242     (953u)

int8_t get_bat_temperature(uint16_t battery_voltage, uint16_t therm_volt)
{
    /* calculate thermistor resistance */
    uint32_t therm_res = (uint32_t)(R242 * battery_voltage) / therm_volt -
R242 - R240;
```

Code Listing 5 Battery temperature measurement function

```
    /* calculate temperature */
    if(therm_res > gl_res_temp_map[0])
    {
        return TEMP_MAP_MIN;
    }
    if(therm_res < gl_res_temp_map[TEMP_MAP_TABLE_COUNT - 1])
    {
        return TEMP_MAP_MAX;
    }
    for(uint8_t i = 1; i < TEMP_MAP_TABLE_COUNT; i++)
    {
        if(therm_res >= gl_res_temp_map[i])
        {
            uint32_t resolution = (gl_res_temp_map[i - 1] -
gl_res_temp_map[i]);
            uint32_t diff = (therm_res - gl_res_temp_map[i]) *
TEMP_MAP_RESOLUTION_BATT;
            return (TEMP_MAP_MIN + TEMP_MAP_RESOLUTION_BATT * i - diff /
resolution);
        }
    }
    return TEMP_MAP_MAX;
}
```

In addition to implementing the typical CC and CV charging loop, EZ-PD™ PMG1-B1 MCU battery charging algorithm incorporates throttling of the battery charging currents based on measured battery temperatures. Figure 28 shows how the EZ-PD™ PMG1-B1 MCU battery charger firmware modulates or terminates the battery charge current based on the measured battery temperature.

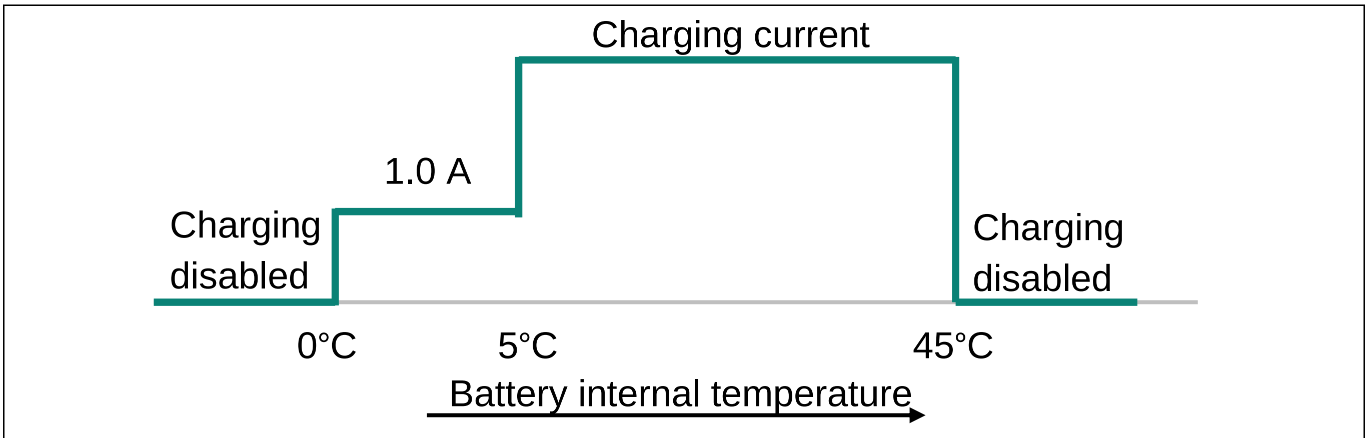

Figure 28 Battery charge current control based on battery temperature variation

EZ-PD™ PMG1-B1 MCU firmware allows the firmware developer to modify these temperature thresholds based on which EZ-PD™ PMG1-B1 MCU battery charging algorithm will modulate/terminate the battery charging current. Code Listing 6 shows the battery temperature threshold settings used for modulating or terminating the battery charging current. The temperature thresholds are measured in Celsius. The firmware also supports a 5°C hysteresis to provide a stable charging current.

Code Listing 6 Battery temperature-based charge current modulation/termination threshold

```
#define PRIMARY_BATT_OTP_THRESHOLD                    (50)
#define PRIMARY_BATT_ROOM_THRESHOLD                   (20)
#define PRIMARY_BATT_COLD_THRESHOLD                   (10)
#define PRIMARY_BATT_HYSTERESIS                       (5)
```

6.4.5 Board temperature monitoring

In addition to the battery pack temperature monitoring, EZ-PD™ PMG1-B1 MCU also supports the board temperature monitoring feature. The EVAL_PMG1_B1_DRP kit has two onboard temperature sensors, one located near the boost FET and another near the Type-C connector. EZ-PD™ PMG1-B1 MCU uses the 8-bit SAR ADC to measure these sensor data and therefore, monitor the board temperature and ensures that the board does not exceed any given temperature limit. Similar to the battery temperature based current throttling, EZ-PD™ PMG1-B1 MCU firmware also throttles the charge current based on the board temperature measurement. This feature is enabled by default in the firmware. Use the macros described in Code Listing 7 to enable/disable this feature and also update the temperature thresholds for board temperature measurement. These macros are defined in the *config.h* file of the EZ-PD™ PMG1-B1 battery charger code example.

Code Listing 7 Board temperature-based charge current modulation/termination threshold

```
/* Enable board temperature monitoring */
#define ENABLE_NTCP0_TEMP_MONITORING                  (1u)
#define ENABLE_NTCP1_TEMP_MONITORING                  (1u)

#define PRIMARY_NTCP0_OTP_THRESHOLD                   (85)
#define PRIMARY_NTCP0_HOT_THRESHOLD                   (70)
#define PRIMARY_NTCP0_HYSTERESIS                      (10)

#define PRIMARY_NTCP1_OTP_THRESHOLD                   (100)
#define PRIMARY_NTCP1_HOT_THRESHOLD                   (85)
#define PRIMARY_NTCP1_HYSTERESIS                      (10)
```

6.5 Battery protection overview

EZ-PD™ PMG1-B1 MCU offers hardware-based detection and protection against battery overvoltage, undervoltage, overcurrent, etc. This section explains how to enable and disable these hardware protections using EZ-PD™ PMG1-B1 MCU battery charger MCU firmware.

6.5.1 Battery overvoltage and undervoltage protections

EZ-PD™ PMG1-B1 module offers two UVOV blocks that can be used to monitor the battery UVOV and Type-C VBUS UVOV. Besides using the 12-bit SAR ADC to monitor the battery voltage, EZ-PD™ PMG1-B1 MCU also utilizes the UVOV block to monitor the VOUT_EA pin during the charging process. This block provides hardware-based protection against battery overvoltage and undervoltage conditions, and immediately cuts-off charging upon detection of battery overvoltage (OV). In a DRP design, EZ-PD™ PMG1-B1 MCU can also cut-off source mode and prevent the battery from discharging if a battery undervoltage (UV) is detected.

EZ-PD™ PMG1-B1 MCU battery firmware uses the macros listed below to enable/disable battery UVP and OVP features.

```
#define BAT_HW_UVP_ENABLE                                   (1u)
#define BAT_HW_OVP_ENABLE                                   (1u)
```

EZ-PD™ PMG1-B1 MCU firmware also allows to configure the overvoltage and undervoltage detection thresholds.

Code Listing 8 Configuration of overvoltage and undervoltage detection thresholds

```
/* Min allowed battery cell voltage per cell in mV */
#define BATT_PER_CELL_ALLOWED_MIN_VOLT              (2000u)

/* Maximum Battery voltage for OVP Threshold in mV.
 * Added 200mV to prevent false triggering when switching CC->CV.
 */
#define PRIMARY_VBATT_OVP_THRESHOLD
((BATT_PER_CELL_OVP_THRESHOLD * TOTAL_BATTERY_CELL_COUNT) + 200u)

/* Per Battery cell OVP Threshold in mV*/
#define BATT_PER_CELL_OVP_THRESHOLD                 (4300u)
/* Maximum Battery voltage for OVP Threshold in mV.
 * Added 200mV to prevent false triggering when switching CC->CV.
 */
#define PRIMARY_VBATT_OVP_THRESHOLD
((BATT_PER_CELL_OVP_THRESHOLD * TOTAL_BATTERY_CELL_COUNT) + 200u)
```

The battery charger application firmware enables the overvoltage and undervoltage protection block on the VOUT_EA pin before it turns on the battery charger FET and starts charging the battery pack. The Code Listing 9 shows the EZ-PD™ PMG1-B1 MCU firmware enabling the UVP and OVP before turning on the battery charging FETs.

Code Listing 9 EZ-PD™ PMG1-B1 MCU enabling battery UV and OV protections

```
case BATT_CHGE_BB_OUT_EN:

#if BAT_HW_UVP_ENABLE
        Cy_USBPD_Fault_Vbat_UvpEnable(ptrPdStackContext->ptrUsbPdContext,
PRIMARY_VBATT_UVP_THRESHOLD, BAT_UVOV_FILTER, soln_vbat_uvp_cbk,
CCG_SRC_FET);
#endif /* BAT_HW_UVP_ENABLE */
#if BAT_HW_OVP_ENABLE
        Cy_USBPD_Fault_Vbat_OvpEnable(ptrPdStackContext->ptrUsbPdContext,
PRIMARY_VBATT_OVP_THRESHOLD, BAT_UVOV_FILTER, soln_vbat_ovp_cbk,
CCG_SRC_FET);
#endif /* BAT_HW_OVP_ENABLE */

        Cy_GPIO_Write(B1_VOUT_DC_EN_H_PORT, B1_VOUT_DC_EN_H_PIN, 1u);
        batt_stat->cur_batt_charging_status = true;
        DEBUG_PRINT("\n BAT Switch ON ..\r\n");
        DEBUG_PRINT_VAR("\n >N_CNTis %i \n",batt_stat ->
curr_chrg_cycle_num);
        break;
```

6.5.2 Battery overcurrent protection

Like the UVOV block, the CSA block in the EZ-PD™ PMG1-B1 MCU also provides hardware-based protection against overcurrent during battery charging. The voltage drop across the series Rsense resistor is monitored to sense the current magnitude on the battery VOUT path and compared against a programmable threshold. If an OCP occurs, EZ-PD™ PMG1-B1 MCU can be set up to deactivate the buck-boost controller and interrupt the battery charging process.

EZ-PD™ PMG1-B1 MCU battery charger firmware has the battery overcurrent protection enabled by default using the `BAT_HW_OCP_ENABLE` macro. The default OCP threshold is set to 7.5 A.

```
#define BAT_HW_OCP_ENABLE                              (1u)
/* OCP Threshold in percentage for Battery Charging Current  */
#define PRIMARY_IBATT_OCP_CURR_THRESHOLD               (750u)
```

In addition, EZ-PD™ PMG1-B1 MCU battery charger firmware uses the smart battery charging algorithm described in Section 6.3 to decide the battery charge current. The battery charge current is calculated based on the measured battery voltage and available input power from the USB-C adapter. EZ-PD™ PMG1-B1 MCU firmware uses this calculated battery charge current to set the overcurrent protection threshold. See the following section of code described in Code Listing 10 to understand how EZ-PD™ PMG1-B1 MCU firmware set the overcurrent protection threshold. This section of code is located in the *solution.c* file.

Code Listing 10 EZ-PD™ PMG1-B1 MCU enabling battery overcurrent protection

```
static void sln_ibtr_cb(void * callbackCtx, bool value)
{
    (void)value;
#if BAT_HW_OCP_ENABLE
    cy_stc_usbpd_context_t *ptrUsbPdContext = (cy_stc_usbpd_context_t
*)callbackCtx;
    cy_stc_pdstack_context_t * ptrPdStackContext =
(cy_stc_pdstack_context_t *)ptrUsbPdContext->pdStackContext;
    cy_stc_battery_charging_context_t* ptrBatteryChargingContext =
get_battery_charging_context(ptrPdStackContext->port);
    cy_stc_battery_status_t* batt_stat = &(ptrBatteryChargingContext-
>batteryStatus);

    /* Set OCP level to 1A for low output current, because of high current
monitor error when I < 1A */
    if(batt_stat->cur_bb_iout <= CY_USBPD_I_1A)
    {
        Cy_USBPD_Fault_Vbat_OcpEnable(ptrUsbPdContext, CY_USBPD_I_1A,
soln_vbat_ocp_cbk);
    }
    else
    {
        Cy_USBPD_Fault_Vbat_OcpEnable(ptrUsbPdContext, batt_stat-
>cur_bb_iout, soln_vbat_ocp_cbk);
    }
#else
    (void)callbackCtx;
#endif /* BAT_HW_OCP_ENABLE */
}
```

6.5.3 Battery charging safety timers

EZ-PD™ PMG1-B1 MCU battery charger firmware comes with safety timers that prevent an extended charging cycle in case of abnormal battery conditions. EZ-PD™ PMG1-B1 MCU firmware offers the timers described in the following sections.

6.5.3.1 Pre-charge timer

EZ-PD™ PMG1-B1 MCU firmware utilizes the pre-charge timer to monitor the time taken to pre-charge the battery pack when the battery cell voltage reading is less than the `TOTAL_VBATT_DISCHARGED_SNK` threshold (e.g., for a 5-cell battery pack, this threshold is 3 V × 5 = 15 V). You can adjust this timer or disable it using the EZ-PD™ PMG1-B1 MCU firmware, which sets the default pre-charge time to 90 minutes (1.5 hours) and enables this timer by default. If the battery pack voltage does not reach the `TOTAL_VBATT_DISCHARGED_SNK` threshold before the timer expires, EZ-PD™ PMG1-B1 MCU firmware will classify the battery as abnormal and stop charging the pack.

To enable/disable the pre-charge timer and change the timer value, update the following macros:

```
/* Enable pre-charge timer */
#define PRE_CHARGE_TIMER_ENABLE                     (1u)
/* Pre-charge timeout in minutes. */
#define PRE_CHARGE_TIMER_VALUE                      (90u)
```

6.5.3.2 Normal charge timer

If the measured battery pack voltage is more than `TOTAL_VBATT_DISCHARGED_SNK` threshold (e.g., for a 5-cell battery pack, this threshold is 3 V × 5 = 15 V), EZ-PD™ PMG1-B1 MCU firmware uses the normal charge timer to monitor the time it takes to charge the battery. This timer monitors the time that it takes for the battery charger to finish both CC and CV loop as part of the charging process. By default, this timer is enabled and the time is set to 360 minutes (6 hours). If the battery pack voltage does not reach the `TOTAL_VBATT_MAX_ALLOWED_VOLT` threshold (e.g., for a 5-cell battery pack, this threshold is 4.15 V × 5 = 20.75 V) before the timer expires, EZ-PD™ PMG1-B1 MCU firmware will classify the battery as abnormal and stop charging the pack.

To enable/disable normal charge timer and change the timer value, update the following macros:

```
/* Enable normal charge timer CC + CV */
#define NORMAL_CHARGE_TIMER_ENABLE                  (1u)
/* Normal charge timeout in minutes. */
#define NORMAL_CHARGE_TIMER_VALUE                   (6 * 60u)
```

6.5.4 Watchdog (WDT) protection timer

EZ-PD™ PMG1-B1 MCU offers a hardware-based watchdog timer to enhance the reliability and safety of the battery charging design. The WDT runs from the LFCLK (32 kHz), generated by the ILO. EZ-PD™ PMG1-B1 MCU battery charger firmware has the WDT enabled by default and the firmware periodically services the WDT timer to avoid the timer lapse and generating a device reset. The WDT timer ensures that the application firmware is not stuck on any faulty code during the firmware development and ensure safe charging of the battery pack.

EZ-PD™ PMG1-B1 MCU battery charger firmware allows with options to control the watchdog-related feature. This includes enabling or disabling the watchdog and setting up the device's response when the watchdog timer expires. Upon the lapse of WDT timer, the chip can either reset or go through hard faults. See the following code macro to make these WDT-related configurations.

```
/* Enable watchdog hardware reset for CPU lock-up recovery */
#define WATCHDOG_HARDWARE_RESET_ENABLE              (1u)

/* Disable device reset on error (watchdog expiry or hard fault). */
#define RESET_ON_ERROR_ENABLE                       (1u)
```

In addition, EZ-PD™ PMG1-B1 MCU firmware also allows to select the WDT reset period in milliseconds (ms). The default WDT expiry period is 750 ms.

```
/*
 * Watchdog reset period in ms. This should be set to a value greater than
 * 500 ms to avoid significant increase in power consumption.
 */
#define WATCHDOG_RESET_PERIOD_MS                    (750u)
```

6.6 DeepSleep feature

EZ-PD™ PMG1-B1 MCU offers a standby regulator that can power the EZ-PD™ PMG1-B1 devices from the battery pack. When the USB-C adapter is not attached to the Type-C port and the input supply to the IC is from the CSNO pin, the standby regulator is turned on and it provides 3 V to EZ-PD™ PMG1-B1 VDDD.

In addition to the standby regulator, EZ-PD™ PMG1-B1 MCU offers a deep sleep power mode. In this mode, the CPU, most peripherals (including buck-boost block, USBPD block), and the internal main oscillator (IMO) clock are disabled. The internal low-speed oscillator (ILO) clock remains active and can be used to clock the watchdog timer (WDT), which can also be used as a sleep timer to wake the system from deepsleep mode. In deepsleep mode, EZ-PD™ PMG1-B1 MCU consumers about 80 µA current from the VDDD.

Enabling this deepsleep mode ensures that EZ-PD™ PMG1-B1 devices draw less power from the battery connected and there reduces the battery discharge and increases the shelf life. To enable/disable the deepsleep mode, set the `SYS_DEEPSLEEP_ENABL`E macro defined in the Makefile to ‘1’.

```
# Add additional defines to the build process (without a leading -D).
# Enabled PD revision 3.0 support, VBus OV Fault Protection and Deep Sleep
mode in idle states.
DEFINES=CY_PD_SINK_ONLY=1 CY_PD_SOURCE_ONLY=0 CY_PD_REV3_ENABLE=1
CY_PD_CBL_DISC_DISABLE=1 \
        VBUS_OVP_ENABLE=1 VBUS_UVP_ENABLE=1 VBUS_OCP_ENABLE=0
SYS_DEEPSLEEP_ENABLE=0 \
        CY_PD_USB4_SUPPORT_ENABLE=0 CY_PDUTILS_TIMER_TYPE=3
```

Upon enabling the deepsleep feature, EZ-PD™ PMG1-B1 MCU battery charging firmware can enter into deepsleep mode after running the USB PD task, application task, and timer task as shown in Code Listing 11.

Code Listing 11 EZ-PD™ PMG1-B1 MCU DeepSleep application task

```
for (;;)
    {
        /* Handle the device policy tasks for each PD port. */
        Cy_PdStack_Dpm_Task(&gl_PdStackPort0Ctx);

        /* Perform any application level tasks. */
        app_task(&gl_PdStackPort0Ctx);

        /* Perform tasks associated with instrumentation. */
        instrumentation_task();
```

Code Listing 11 EZ-PD™ PMG1-B1 MCU DeepSleep application task

```
#if SYS_DEEPSLEEP_ENABLE
       if(gl_PdStackPort0Ctx.dpmConfig.connect
                || (battery_measure_sar_is_active() == false)
                )
#endif /* SYS_DEEPSLEEP_ENABLE */
       {
           soln_task(&gl_PdStackPort0Ctx);
       }

#if SYS_DEEPSLEEP_ENABLE
        /* If possible, enter sleep mode for power saving. */
       if((gl_PdStackPort0Ctx.dpmConfig.connect == false)
                && (battery_measure_sar_is_active() == false)
                )
       {
           system_sleep(&gl_PdStackPort0Ctx,NULL);
       }
#endif /* SYS_DEEPSLEEP_ENABLE */
#if (CY_PDUTILS_TIMER_TYPE != CY_PDUTILS_TIMER_TYPE_WDT)
        /* Clears the WatchDog to prevent device reset */
        Cy_WDT_ClearWatchdog();
#endif /* (CY_PDUTILS_TIMER_TYPE != CY_PDUTILS_TIMER_TYPE_WDT) */
    }
```

7 Bidirectional DRP battery charger design with external FETs

EZ-PD™ PMG1-B1 MCU supports charging and discharging the Li-ion battery pack up to 100 W. The bidirectional DRP buck-boost design requires an additional two pairs of PFET and EZ-PD™ PMG1-B1 GPIO to control the FET and enable the DRP support.

See Figure 29 for the block diagram of the bidirectional DRP and the details are as follows:

- The yellow and red path together shows EZ-PD™ PMG1-B1 MCU charging the Li-ion battery pack using the integrated buck-boost block from the power available on the USB-C connector connected to a power adapter.
- The blue path shows EZ-PD™ PMG1-B1 MCU using the battery pack and integrated buck-boost to make power from the battery available on the USB-C connector to charge external devices such as phones.
- When the PMG1-B1 is connected to a USB-C adapter, it triggers the closure of the NFET and VBAT_FET (red) switches. This facilitates battery charging through the integrated buck-boost system. During this phase, the S0 and S1 switches are open. Conversely, when EZ-PD™ PMG1-B1 MCU is connected to a sink-only device like a phone, it closes the S0 and S1 switches and opens the NFET and VBAT_FET switches. In this case, EZ-PD™ PMG1-B1 device uses the integrated buck-boost to convert the battery pack voltage to the necessary voltage negotiated on the USB-C VBUS path. The sink devices are charged this way.

For the full EZ-PD™ PMG1-B1 bidirectional DRP battery charger reference schematics and the code examples, contact Infineon Support.

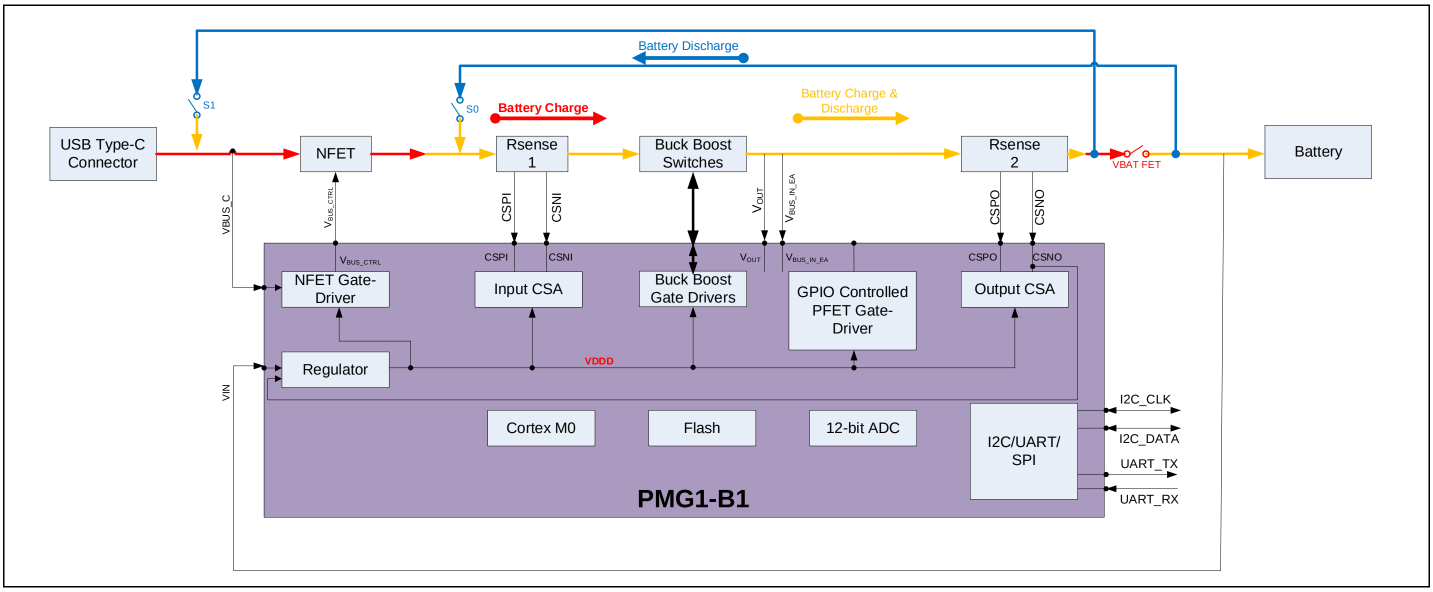

Figure 29 Block diagram of bidirectional DRP design with external FETs using EZ-PD™ PMG1-B1 MCU

7.1 System power supply design

EZ-PD™ PMG1-B1's high voltage regulator gets inputs from the VIN supply pin. In a USB-C sink-only battery charger design, the USB-C VBUS is connected to the VIN pin (Pin 48) to power the chip from Type-C VBUS. When the USB-C adapter is unconnected, EZ-PD™ PMG1-B1 MCU power from the standby regulator which receives it input from the CSNO pin (Pin 16). The integrated buck-boost block requires the high voltage regulator to be ON for regular buck boost operation. Therefore, in a bidirectional DRP design, the VIN pin needs to be powered from both battery and Type-C VBUS to enable integrated buck-boost and provide voltage on the Type-C VBUS in source mode. The following figure shows EZ-PD™ PMG1-B1 MCU powered from battery or Type-C VBUS using the OR'ed diode circuitry.

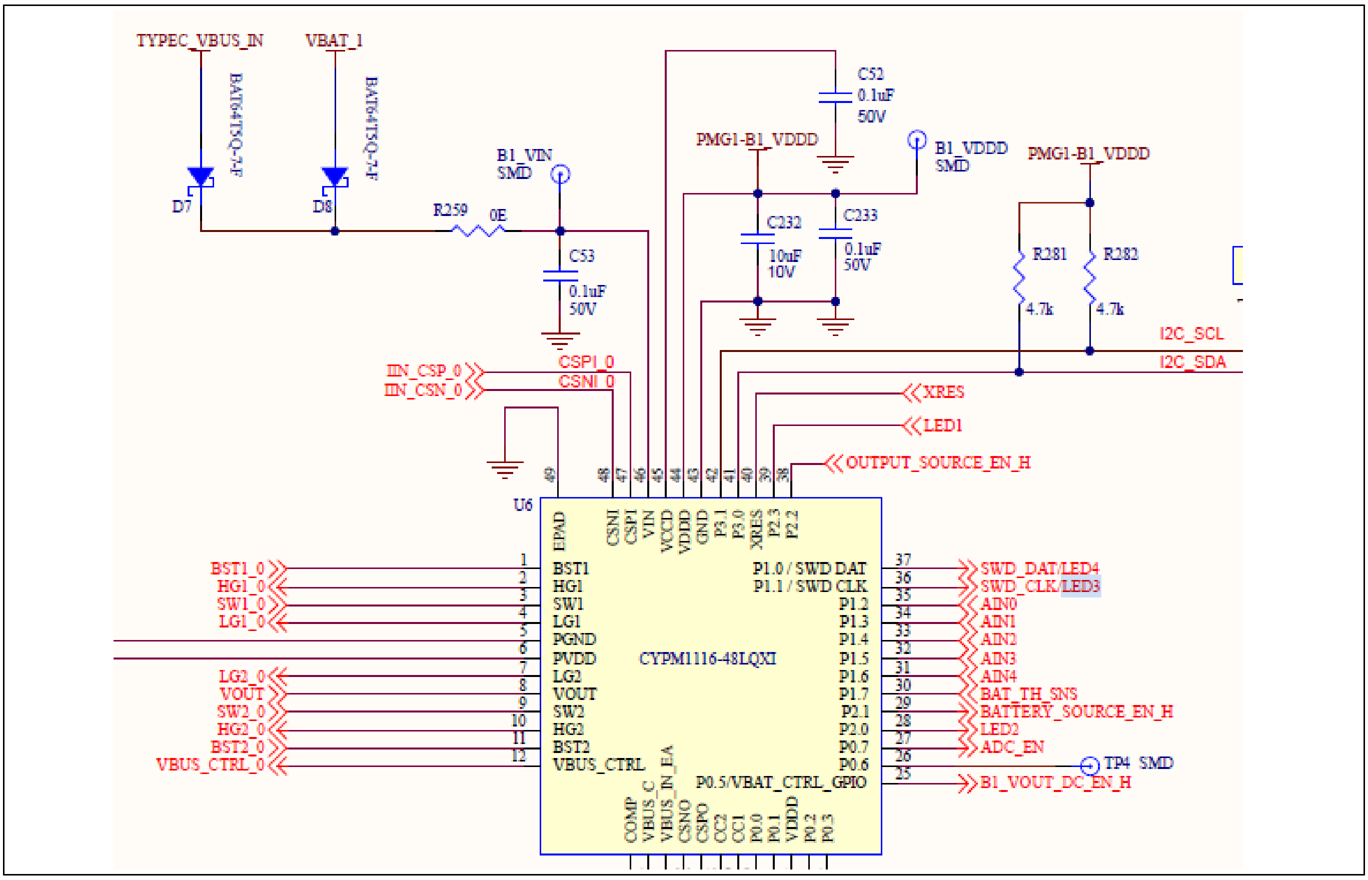

Figure 30 **EZ-PD™ PMG1-B1 VIN pin powered from Type-C VBUS and VBAT using diode based OR-ing circuit**

7.2 Battery charging FET and Type-C VBUS provide FET design

EZ-PD™ PMG1-B1 MCU as described in Section 7, require an additional two pairs of PFETs and two additional GPIOs to control these FET and enable the bidirectional battery charger design with DRP support. See Figure 31 for the charge and discharge FET control block diagram of the bidirectional DRP battery charger design.

In the USB-C sink-only mode, EZ-PD™ PMG1-B1 MCU uses the NFETs Q5, Q6 to connect the Type-C VBUS to the input of the EZ-PD™ PMG1-B1 buck-boost block and uses the PFET Q28 and Q8 to connect the buck-boost output to the battery pack and charge the battery pack. The NFET Q5 and Q6 gates are tied together and connected to EZ-PD™ PMG1-B1's dedicated NFET gate driver pin while the PFET Q28 and Q8 gate is controlled by EZ-PD™ PMG1-B1 GPIO.

In the USB-C source mode, EZ-PD™ PMG1-B1 MCU controls the PFET, Q7, and Q9 to connect the battery pack voltage to the input of the EZ-PD™ PMG1-B1 buck-boost block and uses the PFET Q11 and Q26 to connect the output of the buck boost to the Type-C VBUS. When a sink-only device such as phones or speakers are connected, EZ-PD™ PMG1-B1 MCU negotiates the required VBUS voltage over the USB-C CC line and configures the buck-boost block to output the required VBUS voltage to charge the sink-only device such as phones from the connected battery pack. The red path shows the battery charge path and the blue path shows the battery discharge path in the following figure.

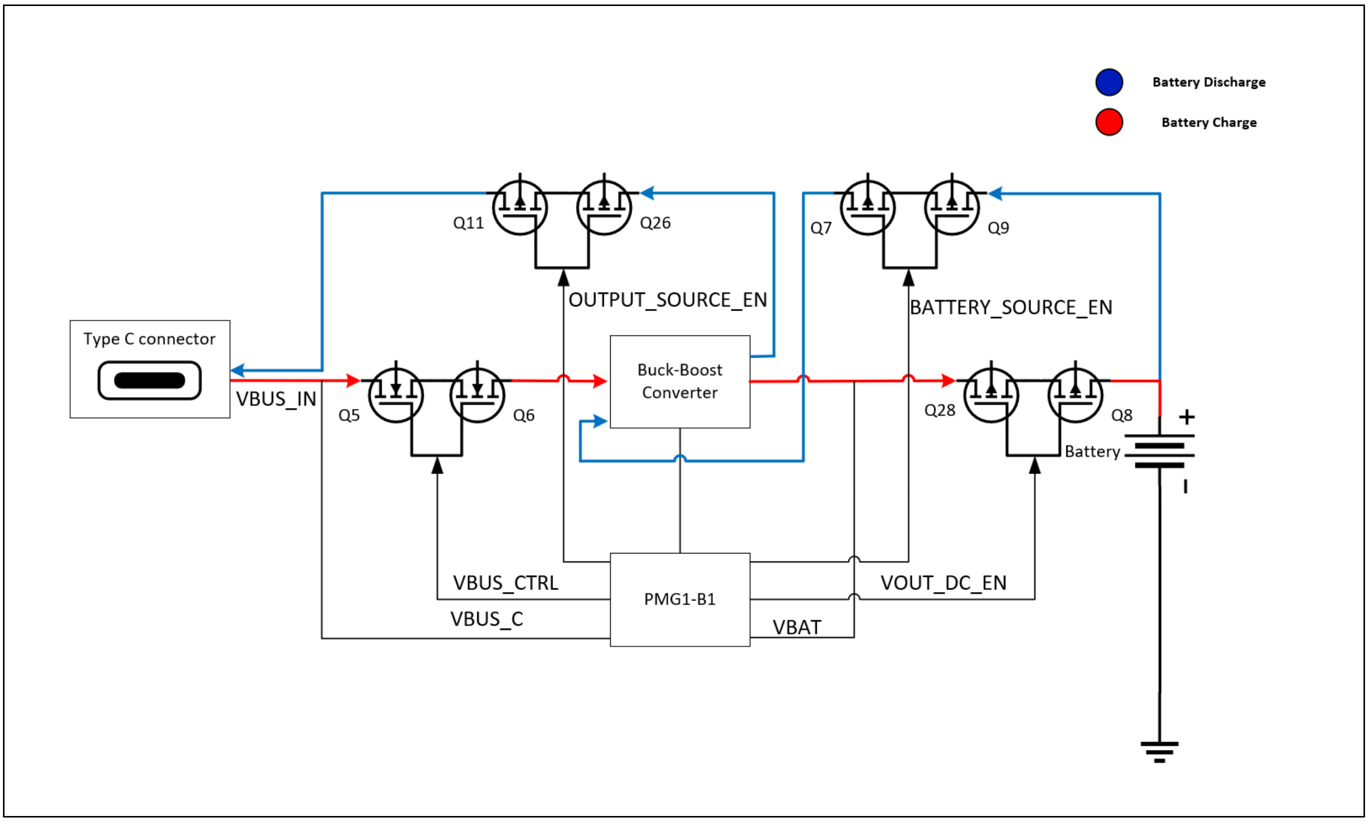

Figure 31 EZ-PD™ PMG1-B1 battery charge and battery discharge path with external FETs

8 Design specifications

8.1 Operating ranges and maximum ratings for the controllers

Table 7 Operating ranges and specification of the battery charger controller firmware

Parameter	Min	Max	Unit
Total number of Li-ion battery cells supported	2	5	–
Buck boost output voltage	3.3	21.5	V
Buck boost output CSA series Rsense resistor	5	10	mΩ
Buck boost output battery charging current (5 mΩ Rsense)	300	6500	mA
Buck boost output battery charging current (10 mΩ Rsense)	150	3200	mA
USB-C PDO contact voltage for battery charging	5	20	V
USB-C PDO contract current for battery charging	1.5	5	A
Buck boost PWM frequency	150	600	kHz
Battery individual cell voltage	2.0	4.3	V
Battery pack voltage (e.g., 5-cell design)	10.0	21.5	V

References

[1] Infineon Technologies AG: *EZ-PD™ PMG1 USB-C high voltage MCUs webpage*; Available online

[2] Infineon Technologies AG: *EZ-PD™ PMG-B1 USB-C high voltage MCUs webpage*; Available online

Acronyms/abbreviations

Table 8 Acronyms/abbreviations

Acronym	Description
BMC	biphase mark code
BMS	battery monitoring system
BSP	board support package
CC	constant current
CCM	continuous conduction mode
CSA	current sense amplifier
CV	constant voltage
DCM	discontinuous conduction mode
DRP	dual-role port
EA	error amplifier
ILO	internal low-speed oscillator
IMO	internal main oscillator
Li-ion	lithium-ion
OC	overcurrent
OV	overvoltage
PASS	programmable analog subsytem
PCMC	peak current mode controlled
PD	Power Delivery
PWM	pulse-width modulator
RC	reverse-current
SC	short-circuit
SCB	Serial Communication Block
SDK	software development kit
SoC	state of charge
UV	undervoltage
UVOV	undervoltage overvoltage
WDT	watchdog timer
ZCD	zero-crossing detection

Revision history

Document revision	Date	Description of changes
**	2024-06-19	Initial release

Edition 2024-06-19

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Email: erratum@infineon.com

Document reference

002-40084 Rev. **