

Objective

This example demonstrates the RTC Alarm function of the PSoC® 6 MCU Real-Time Clock (RTC) using ModusToolbox™ IDE.

Requirements

Tool: [ModusToolbox™ IDE 1.0](#)

Programming Language: C

Associated Parts: All [PSoC 6 MCU](#) parts

Related Hardware: [PSoC 6 BLE Pioneer Kit](#), [PSoC 6 WiFi-BT Pioneer Kit](#), [PSoC6 WiFi-Prototyping Kit](#)

Overview

This code example demonstrates how to configure RTC registers for a daily alarm using the RTC driver API in the Peripheral Driver Library (PDL). A GPIO output is included for an LED to notify alarm expiration. A UART is used to show the current and alarm times. In this code example daily alarm is configured for Monday to Friday.

Hardware Setups

This example uses the PSoC 6 WiFi-BT Pioneer Kit in default configuration. Refer to the kit guide to ensure the kit is configured correctly.

Note: The PSoC 6 BLE Pioneer kit and the PSoC 6 WiFi-BT Pioneer kit ship with KitProg2. ModusToolbox only works with KitProg3. Before using this code example, make sure that the kit is upgraded to KitProg3. See [ModusToolbox Help > ModusToolbox IDE Documentation > User Guide, PSoC 6 MCU KitProg Firmware Loader](#) section. If you do not upgrade, you will see an error like “unable to find CMSIS-DAP device” or “KitProg firmware is out of date”.

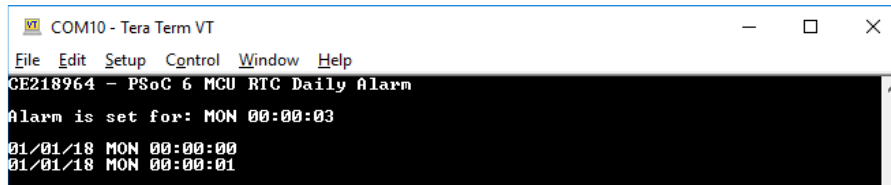
Software Setup

This code example requires a PC terminal software. Install one on your PC if you don't have one. The instructions use [Tera Term](#).

Operation

1. Connect the Pioneer board to your PC using the provided USB cable through the USB connector.
2. Open your terminal software and select the KitProg COM port. Set the other serial port parameters as follows:
 - a. Baud rate: 115200bps
 - b. Data: 8-bit
 - c. Parity: None
 - d. Stop: 1-bit
 - e. Flow control: None
3. Import the code example into a new workspace. See [KBA225201](#).
4. Program the PSoC 6 MCU device. In the project explorer, select the **mainapp** project. In the Quick Panel, scroll to the **Launches** section and click the **Program (KitProg3)** configuration.
5. Confirm that the terminal program is working. A message should be printed every one second in the terminal window as shown in [Figure 1](#).

Figure 1. UART Display Start Message

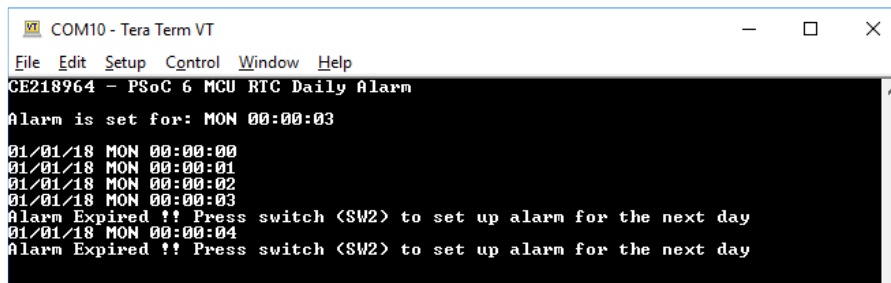


```

COM10 - Tera Term VT
File Edit Setup Control Window Help
CE218964 - PSoC 6 MCU RTC Daily Alarm
Alarm is set for: MON 00:00:03
01/01/18 MON 00:00:00
01/01/18 MON 00:00:01
  
```

- The following message should be printed after three second "Alarm Expired !! Press SW2 for to set alarm for next day"(Figure 2).

Figure 2. UART Display Message After RTC Alarm

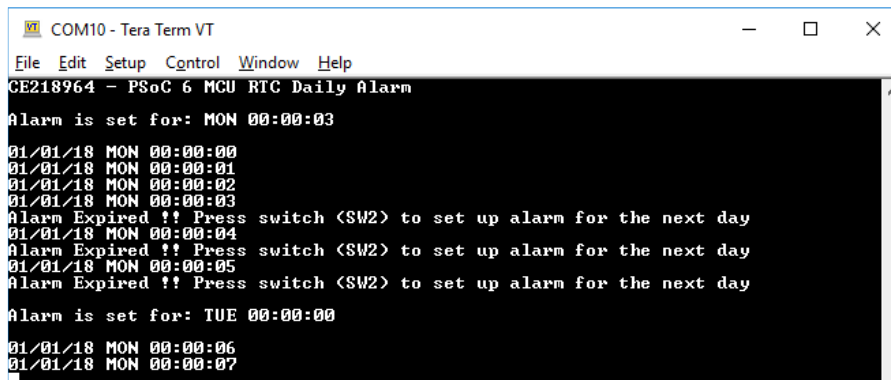


```

COM10 - Tera Term VT
File Edit Setup Control Window Help
CE218964 - PSoC 6 MCU RTC Daily Alarm
Alarm is set for: MON 00:00:03
01/01/18 MON 00:00:00
01/01/18 MON 00:00:01
01/01/18 MON 00:00:02
01/01/18 MON 00:00:03
Alarm Expired !! Press switch <SW2> to set up alarm for the next day
01/01/18 MON 00:00:04
Alarm Expired !! Press switch <SW2> to set up alarm for the next day
  
```

- Confirm Red LED (KIT_LED2) toggles every second.
- Press switch SW2 (P0[4]). It should show the next alarm information as shown in Figure 3.

Figure 3. UART Display Message After Switch Is Pressed



```

COM10 - Tera Term VT
File Edit Setup Control Window Help
CE218964 - PSoC 6 MCU RTC Daily Alarm
Alarm is set for: MON 00:00:03
01/01/18 MON 00:00:00
01/01/18 MON 00:00:01
01/01/18 MON 00:00:02
01/01/18 MON 00:00:03
Alarm Expired !! Press switch <SW2> to set up alarm for the next day
01/01/18 MON 00:00:04
Alarm Expired !! Press switch <SW2> to set up alarm for the next day
01/01/18 MON 00:00:05
Alarm Expired !! Press switch <SW2> to set up alarm for the next day
Alarm is set for: TUE 00:00:00
01/01/18 MON 00:00:06
01/01/18 MON 00:00:07
  
```

Design and Implementation

This code example features the Real Time Clock resource, one GPIO for LED alarm indicator, one GPIO for user switch and one UART resource.

The PSoC 6 MCU RTC is a hardware-based function; the alarm time can be configured by the alarm register fields. The daily alarm needs to enable the hour, minute, and second time fields. Each alarm field is paired with its own enable field. For example, the sec (second) field is paired with the secEn (second enable); the dayOfWeekEn field is paired with the dayOfWeekEn field. If an enable field is set, the field value will be used for matching the alarm time; otherwise the field value will be ignored.

In this code example, the alarm function uses the RTC alarm 1 interrupt. After an alarm has expired, the code prints the alarm expiration message and toggles the red LED (KIT_LED2) every second until the SW2 button is pressed. After the switch is pressed the alarm is configured for the next day.

Resources and Settings

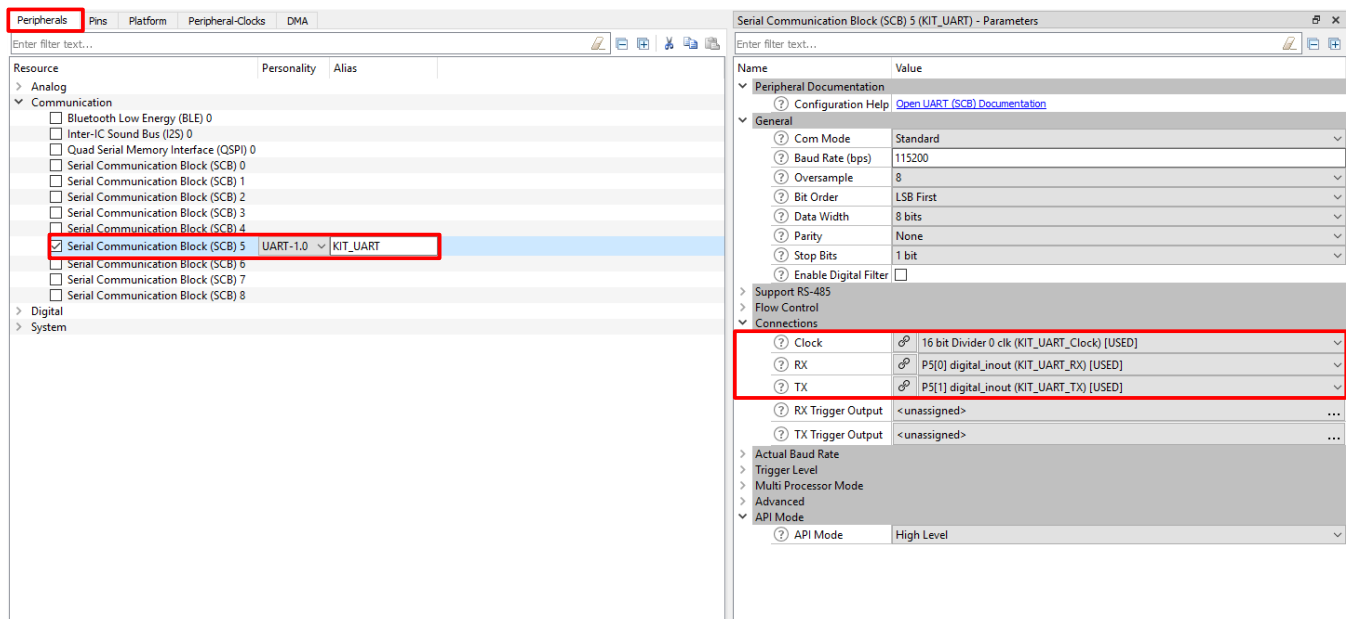
Table 1 lists the ModusToolbox resources used in this example, and how they are used in the design

Table 1. ModusToolbox Resources

Resources	Alias	Purpose	Non-default Settings
Real Time Clock	RTC	Provide date and time information	Default
SCB	KIT_UART	Used for printing terminal messages	See Figure 4
Digital Output Pin	KIT_UART_TX	Used for UART transmit (Tx)	See Figure 5
	KIT_LED2	Provide visual feedback	See Figure 6
Digital Input Pin	KIT_UART_RX	Used for UART receive (Rx)	See Figure 7
	KIT_BTN2	Provide user interaction	See Figure 8

Figure 4 to Figure 8 shows the non-default configuration settings for the ModusToolbox resources.

Figure 4. UART Configuration



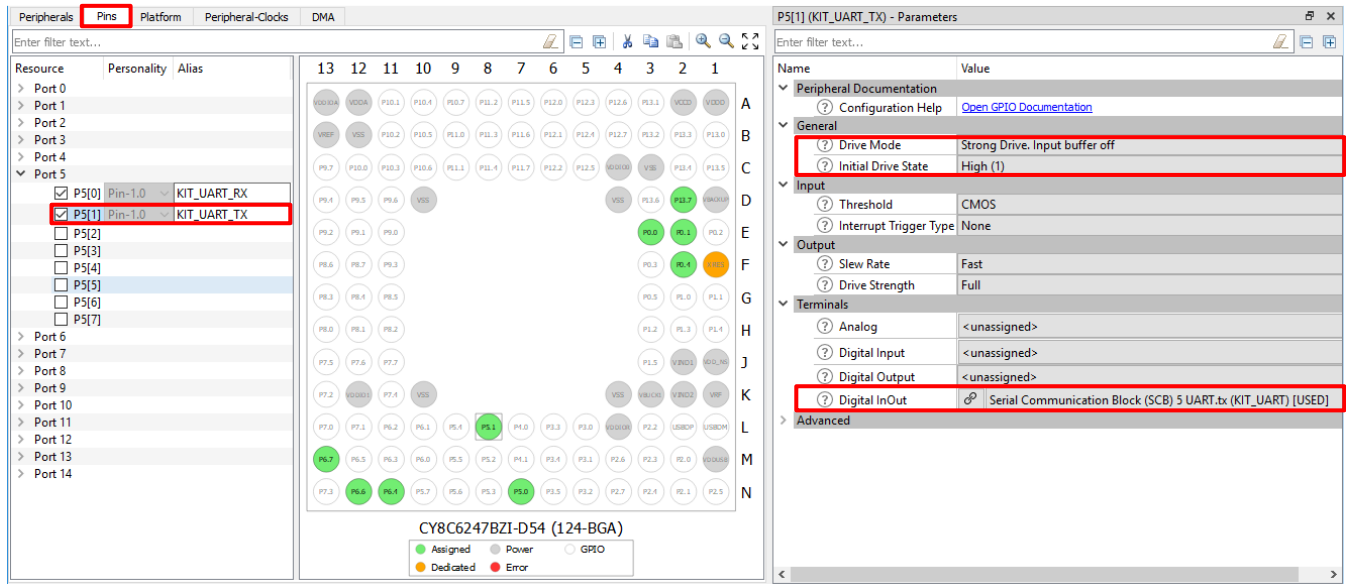
The screenshot displays the ModusToolbox interface for configuring the UART peripheral. On the left, the 'Peripherals' tab is active, and 'Serial Communication Block (SCB) 5' is selected under the 'Communication' category. The right pane shows the configuration parameters for 'Serial Communication Block (SCB) 5 (KIT_UART)'. The 'Connections' section is highlighted with a red box, showing the following connections:

- Clock:** 16 bit Divider 0 clk (KIT_UART_Clock) [USED]
- RX:** PS[0] digital_inout (KIT_UART_RX) [USED]
- TX:** PS[1] digital_inout (KIT_UART_TX) [USED]

Other configuration parameters visible include:

- General:**
 - Com Mode: Standard
 - Baud Rate (bps): 115200
 - Oversample: 8
 - Bit Order: LSB First
 - Data Width: 8 bits
 - Parity: None
 - Stop Bits: 1 bit
 - Enable Digital Filter: ☐
- Support RS-485:** ☐
- Flow Control:** ☐
- API Mode:** High Level

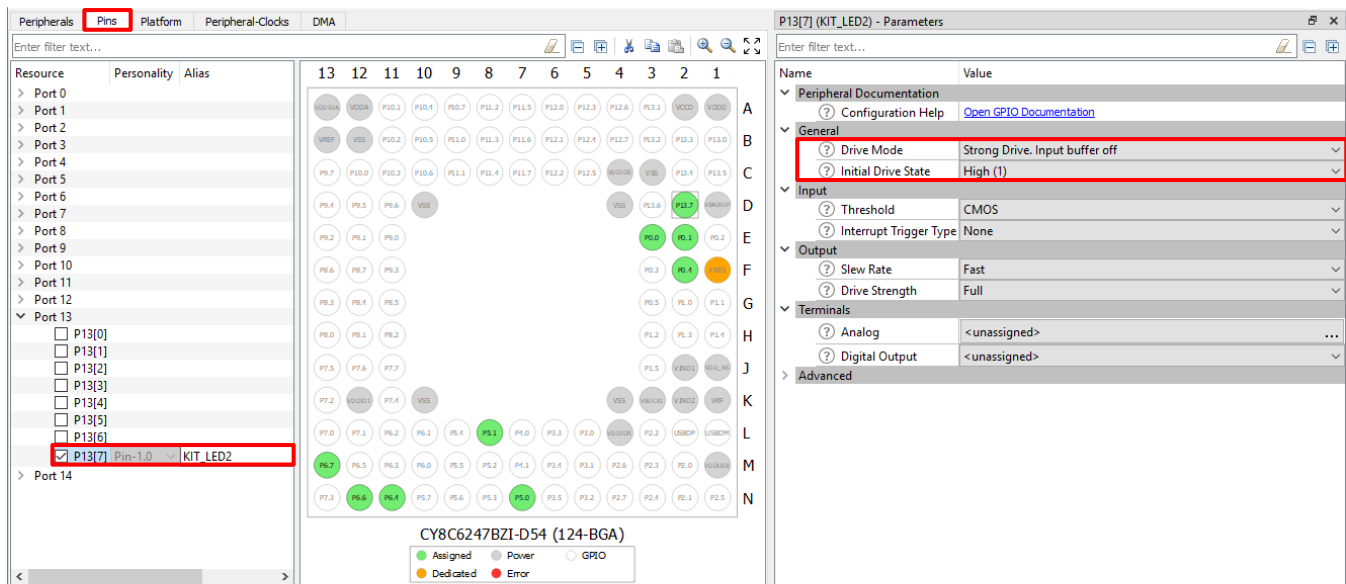
Figure 5. GPIO Pin Configuration for UART Tx



The screenshot displays the PSoC Creator interface for configuring GPIO pins. The 'Pins' tab is selected, showing a pin map for the CY8C6247BZI-D54 (124-BGA) package. Pin P5[1] is highlighted in green and assigned to KIT_UART_TX. The parameters window for P5[1] (KIT_UART_TX) - Parameters is open, showing the following configuration:

- General:**
 - Drive Mode: Strong Drive, Input buffer off
 - Initial Drive State: High (1)
- Input:**
 - Threshold: CMOS
 - Interrupt Trigger Type: None
- Output:**
 - Slew Rate: Fast
 - Drive Strength: Full
- Terminals:**
 - Analog: <unassigned>
 - Digital Input: <unassigned>
 - Digital Output: <unassigned>
 - Digital In/Out: Serial Communication Block (SCB) 5 UART.bx (KIT_UART) [USED]

Figure 6. GPIO Pin Configuration for LED Pin



The screenshot displays the PSoC Creator interface for configuring GPIO pins. The 'Pins' tab is selected, showing a pin map for the CY8C6247BZI-D54 (124-BGA) package. Pin P13[7] is highlighted in green and assigned to KIT_LED2. The parameters window for P13[7] (KIT_LED2) - Parameters is open, showing the following configuration:

- General:**
 - Drive Mode: Strong Drive, Input buffer off
 - Initial Drive State: High (1)
- Input:**
 - Threshold: CMOS
 - Interrupt Trigger Type: None
- Output:**
 - Slew Rate: Fast
 - Drive Strength: Full
- Terminals:**
 - Analog: <unassigned>
 - Digital Output: <unassigned>

Peripherals **Pins** Platform Peripheral-Clocks DMA

Enter filter text...

Resource	Personality	Alias
> Port 0		
> Port 1		
> Port 2		
> Port 3		
> Port 4		
> Port 5		
> Port 6		
> Port 7		
> Port 8		
> Port 9		
> Port 10		
> Port 11		
> Port 12		
> Port 13		
> Port 14		

<input checked="" type="checkbox"/> P5[0]	Pin-1.0	KIT_UART_RX
<input checked="" type="checkbox"/> P5[1]	Pin-1.0	KIT_UART_TX
<input type="checkbox"/> P5[2]		
<input type="checkbox"/> P5[3]		
<input type="checkbox"/> P5[4]		
<input type="checkbox"/> P5[5]		
<input type="checkbox"/> P5[6]		
<input type="checkbox"/> P5[7]		

Port 6

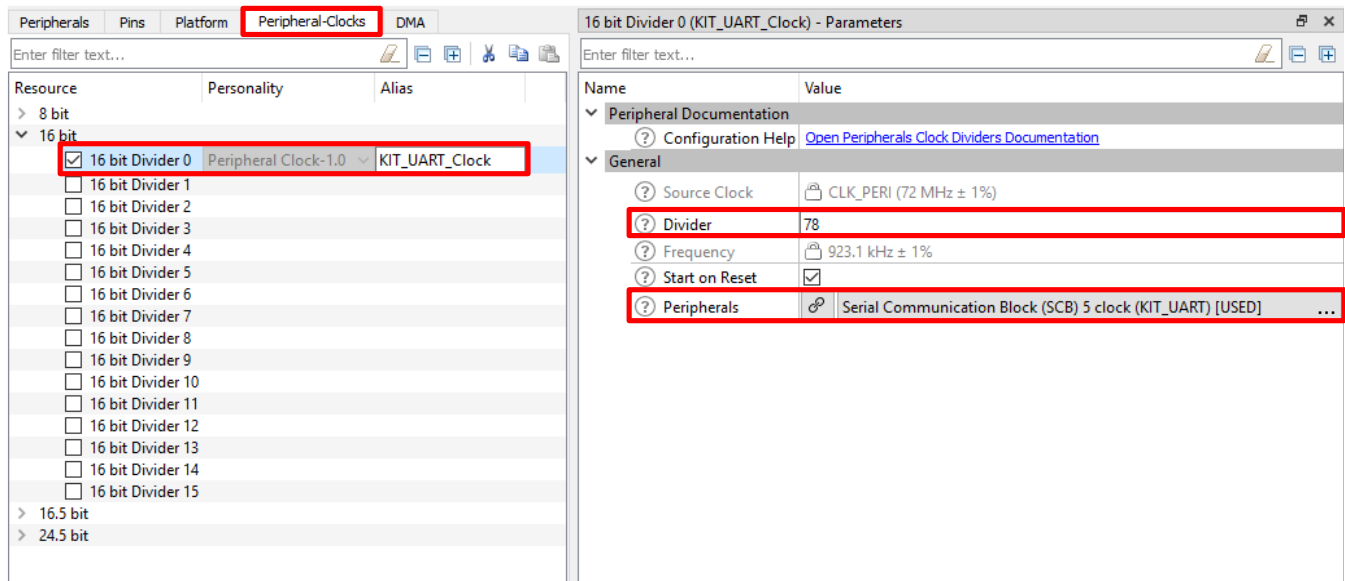
CY8C6247BZI-D54 (124-BGA)

Legend: ● Assigned ● Power ● Dedicated ● Error

The screenshot displays the Pinctrl tool interface for the CY8C6247BZI-D54 (124-BGA) device. On the left, the 'Pins' tab is active, showing a list of pins. Pin P0[4] is selected, and its alias is set to 'KIT_BTN2'. The main area shows a pin grid with P0[4] highlighted in orange, indicating it is dedicated. The right pane shows the 'Parameters' for P0[4], where 'Drive Mode' is set to 'Resistive Pull-Up, Input buffer on' and 'Interrupt Trigger Type' is set to 'Falling Edge'.

Figure 9 shows the Peripheral-Clock configuration for resource.

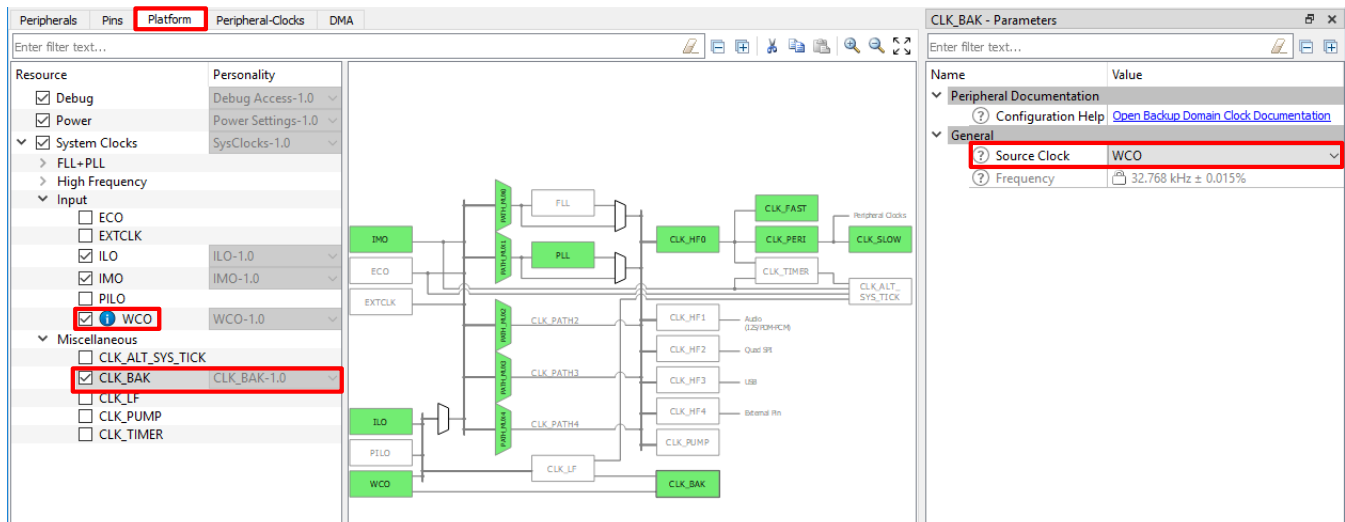
Figure 9. Peripheral-Clock Configuration for UART



Design Considerations

It is necessary to provide a 32.768-kHz clock for the RTC function in the backup power domain. For accurate RTC operation, it is recommended that you use a Watch Crystal Oscillator (WCO). Figure 10 shows the backup clock configuration used in this project.

Figure 10. Backup Clock Configuration



Reusing This Example

This example is designed for the supported kits. To port the design to a different PSoC 6 MCU device, right-click an application project and choose **Change Device**. If changing to a different kit, you may need to reassign pins.

Table 2. Device and Pin Mapping Table Across PSoC 6 MCU Kits

Kit Name	Device Used	KIT_LED	KIT_UART_RX	KIT_UART_TX	KIT_BTN2
CY8CKIT-062-WiFi-BT	CY8C6247BZI-D54	P13[7]	P5[0]	P5[1]	P0[4]
CY8CKIT-062-BLE	CY8C6347BZI-BLD53	P13[7]	P5[0]	P5[1]	P0[4]
CY8CPROTO-062-4343W	CY8C624ABZI-D44	P13[7]	P5[0]	P5[1]	P0[4]

Related Documents

For a comprehensive list of PSoC 6 MCU resources, see [KBA223067](#) in the Cypress community.

Application Notes	
AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity	Describes PSoC 6 MCU with BLE Connectivity devices and how to build your first ModusToolbox application and PSoC Creator project
AN221774 – Getting Started with PSoC 6 MCU	Describes PSoC 6 MCU devices and how to build your first ModusToolbox application and PSoC Creator project
AN215656 – PSoC 6 MCU: Dual-CPU System Design	Describes the dual-CPU architecture in PSoC 6 MCU, and shows how to build a simple dual-CPU design
Code Examples	
Visit the Cypress GitHub site for a comprehensive collection of code examples using ModusToolbox IDE	
CE216825 - PSoC® 6 Real-Time Clock Basics	
CE218542 - PSoC 6 MCU Custom Tick Timer Using RTC Alarm	
Device Documentation	
PSoC 6 MCU: PSoC 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
Development Kit Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	
CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit	
CY8CPROTO-062-4343W PSoC 6 Wi-Fi BT Prototyping Kit	
Tool Documentation	
ModusToolbox	The Cypress IDE for IoT designers

Cypress Resources

Cypress provides a wealth of data at www.cypress.com to help you to select the right device, and quickly and effectively integrate the device into your design.

For the PSoC 6 MCU devices, see [KBA223067](#) in the Cypress community for a comprehensive list of PSoC 6 MCU resources.

Document History

Document Title: CE218964 - PSoC 6 MCU RTC Daily Alarm

Document Number: 002-25550

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6371319	AJYA	11/22/2018	New code example

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at [Cypress Locations](#).

Products

Arm® Cortex® Microcontrollers	cypress.com/arm
Automotive	cypress.com/automotive
Clocks & Buffers	cypress.com/clocks
Interface	cypress.com/interface
Internet of Things	cypress.com/iot
Memory	cypress.com/memory
Microcontrollers	cypress.com/mcu
PSoC	cypress.com/psoc
Power Management ICs	cypress.com/pmic
Touch Sensing	cypress.com/touch
USB Controllers	cypress.com/usb
Wireless Connectivity	cypress.com/wireless

PSoC® Solutions

[PSoC 1](#) | [PSoC 3](#) | [PSoC 4](#) | [PSoC 5LP](#) | [PSoC 6 MCU](#)

Cypress Developer Community

[Community](#) | [Projects](#) | [Videos](#) | [Blogs](#) | [Training](#) | [Components](#)

Technical Support

cypress.com/support

All other trademarks or registered trademarks referenced herein are the property of their respective owners.



Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

© Cypress Semiconductor Corporation, 2018. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spanion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. No computing device can be absolutely secure. Therefore, despite security measures implemented in Cypress hardware or software products, Cypress does not assume any liability arising out of any security breach, such as unauthorized access to or use of a Cypress product. In addition, the products described in these materials may contain design defects or errors known as errata which may cause the product to deviate from published specifications. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spanion, the Spanion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.