CE222604 – PSoC 6 MCU with BLE Connectivity: RTC with Current Time Service (RTOS)

Objective

This code example demonstrates accurate time keeping with PSoC® 6 MCU's real-time clock (RTC), which is synchronized with a current time server such as an iPhone using the BLE current time service (CTS).

Overview

This code example demonstrates accurate time keeping with the RTC of PSoC 6 MCU with BLE Connectivity (PSoC 6 MCU), which also generates alarms (interrupts) at every one minute to show time information on an E-INK display. In addition, a BLE CTS is used to synchronize time and date with a current time server such as an iPhone.

This code example assumes that you are familiar with the PSoC 6 MCU and the PSoC Creator™ Integrated Design Environment (IDE). If you are new to PSoC 6 MCU, you can find introductions in the application note AN210781 – Getting Started with PSoC 6 MCU with Bluetooth Low Energy (BLE) Connectivity.

This code example uses FreeRTOS. See PSoC 6 101: Lesson 1-4 FreeRTOS training video to learn how to create a PSoC 6 FreeRTOS project with PSoC Creator. Visit the FreeRTOS website for documentation and API references of FreeRTOS.

Requirements

Tool: PSoC Creator 4.2; Peripheral Driver Library (PDL) 3.0.4

Programming Language: C (Arm® GCC 5.4.1 and Arm MDK 5.22)

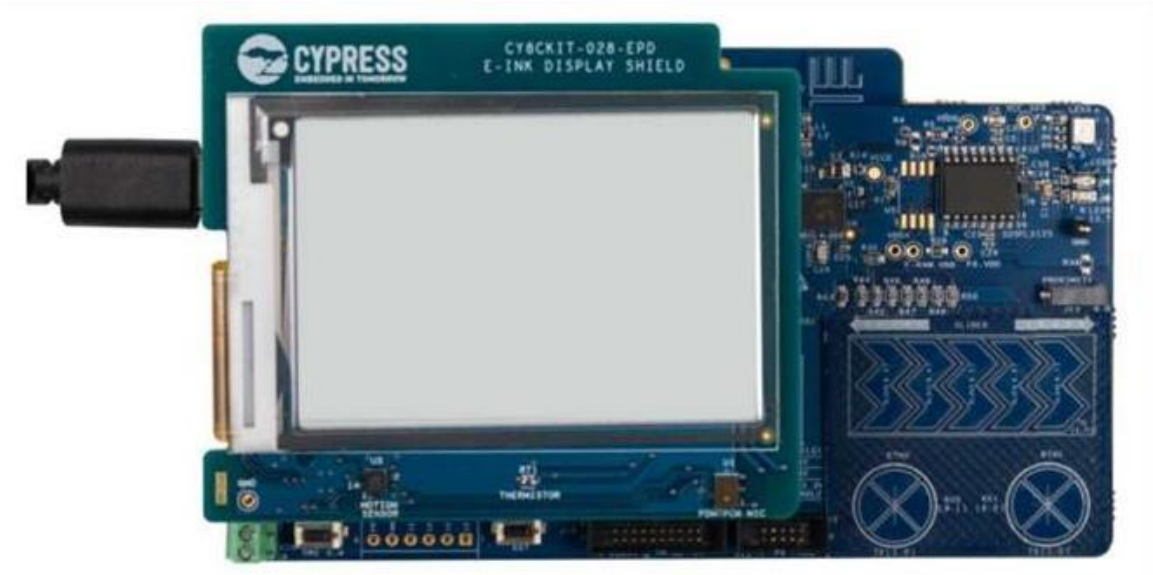
Associated Parts: All PSoC 6 MCUs with BLE Connectivity

Related Hardware: CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit

Hardware Setup

Plug in the E-INK display shield on to the Pioneer board as Figure 1 shows.

Figure 1. Hardware Setup



Set the switches and jumpers as shown in Table 1.

Table 1. Switch and Jumper Selection

Switch / Jumper	Position	Location
SW5	3.3V	Front
SW6	PSoC 6 BLE	Back
SW7	V_{DDD} / KitProg2	Back
J8	Installed	Back

Note: This code example does not support supply voltages other than 3.3 V due to limitations on the voltage required for the inertial measurement unit (IMU) and RGB LED.

Software Setup

Install the CY8CKIT-62-BLE PSoC 6 BLE Pioneer Kit software, which contains all the required software to evaluate this code example. No additional software setup is required.

Operation

To verify the code example using an iOS device, follow these steps:

Note: This code example requires an iOS device with iOS 8 or a later version to evaluate. Android devices do not support the Current Time Service.

1. Power the Pioneer Board through the USB connector **J10**.
2. Program the Pioneer Board with the project. See the Pioneer Kit guide for details on how to program firmware into the device.

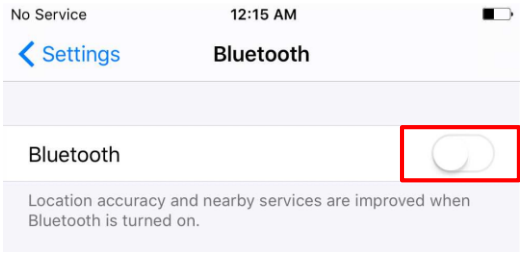
   After programming successfully, the E-INK display will refresh and show the default time and date, and the instructions to use this project. BLE will start advertising with an advertising timeout of 20 seconds. The orange LED (**LED8**) remains ON during this period to indicate the BLE advertising state.

Figure 2. BLE Advertising



3. If BLE advertisement has timed out (**LED8** is OFF), press **SW2** to restart advertisement.
4. Open Settings on your iOS device, and select **Bluetooth**. From the Bluetooth settings, turn ON Bluetooth as Figure 3 shows.

Figure 3. Turning ON Bluetooth



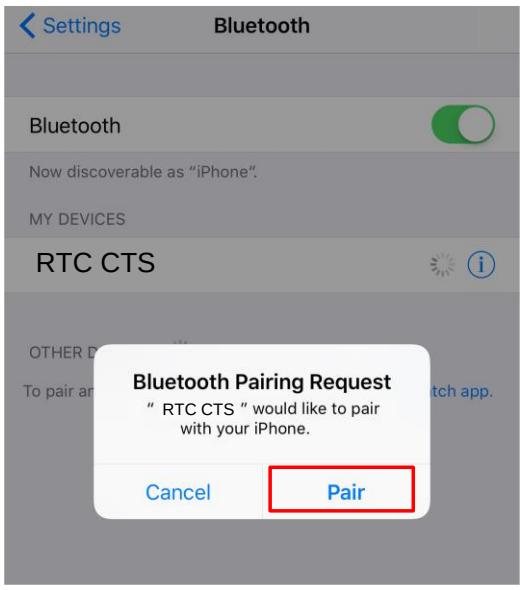
5. After Bluetooth is turned ON, the application will automatically search for available devices and will list them. Select the **RTC CTS** device as shown in Figure 4. A successful connection is indicated by **LED8** continuously blinking at half second intervals.

Figure 4. Selecting the Device



6. When connected, the PSoC 6 MCU sends a pairing request to the iOS device. Accept the pairing request as Figure 5 shows.

Figure 5. PSoC 6 MCU Pairing Request



7. Upon pairing, PSoC 6 MCU receives the current time and date from the iOS device and updates the RTC accordingly. The E-INK display will refresh to show the updated time and date.

Figure 6. Display Update



8. After the device is disconnected, the red LED (LED9) will turn ON to indicate a disconnect event.

Figure 7. Disconnect Indication



9. The display will keep updating at one-minute intervals. Press SW2 to restart the advertisement, if required.

Design and Implementation

PSoC 6 MCUs have a fully featured RTC that keeps track of the current time and date independent of the CPU. The RTC is clocked from an accurate 32768-Hz watch crystal oscillator (WCO). The RTC has a programmable alarm feature, which generates interrupts at a specified time and date with the capability to wake up the system from low power modes. In this code example, the alarm feature is used to generate interrupts at one-minute intervals to update an E-INK display with the current time and date. E-INK displays consume no power for display retention; therefore, the power supply of the display is turned off after an update to reduce the average power consumption. See code example CE218136 - PSoC 6 MCU E-INK Display with CapSense (RTOS).

The BLE Component provides a CTS that allows a GATT time client to get the current time and date information from a GATT time server. iOS devices have built-in BLE time servers that allow BLE GATT time clients to connect to them and extract the time information. This code example utilizes the time server feature of iOS devices to fetch the current time value on establishing a BLE connection and then initializes the RTC with the time information.

Figure 8 shows the functional block diagram of this code example.

Figure 8. BLE Current Time Service Configuration

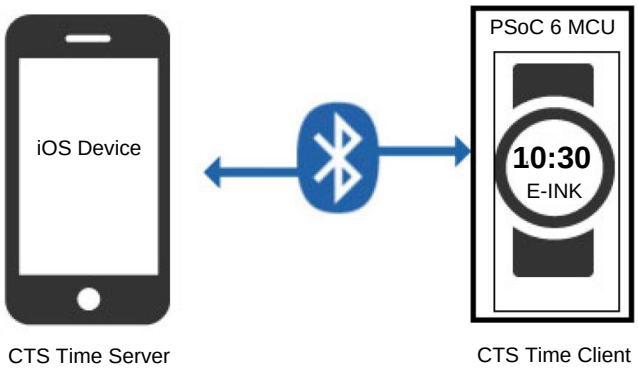


Figure 9 and Figure 10 show the TopDesign schematic of this code example. In addition to the BLE, RTC, and the E-INK display, this code example includes two LEDs that are used to show BLE status, a multi-counter watchdog timer and associated interrupt that controls LED timing, and a GPIO interrupt that is used to restart BLE advertisement.

Figure 9. TopDesign Schematic: BLE, RTC, Interrupts, and LEDs

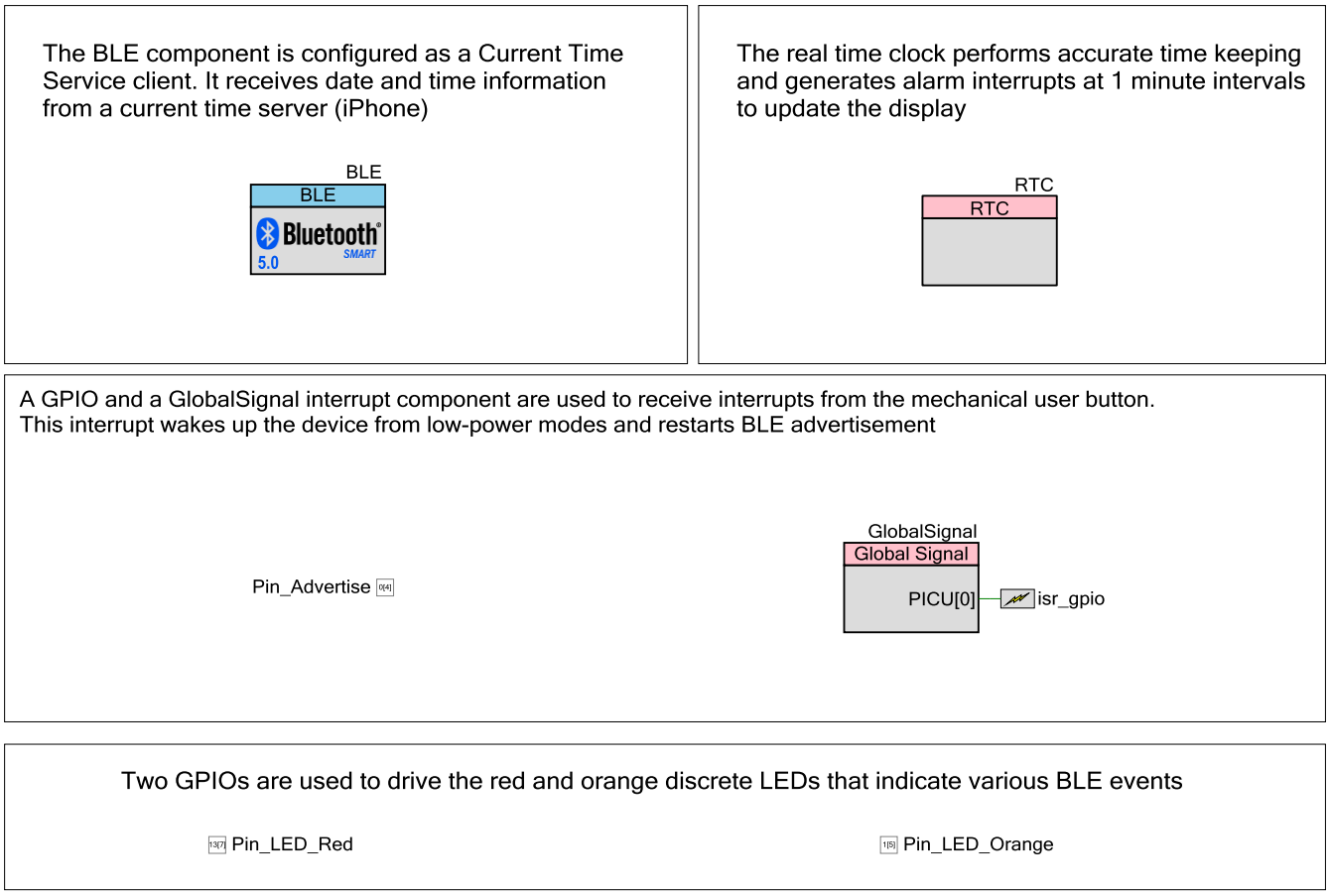
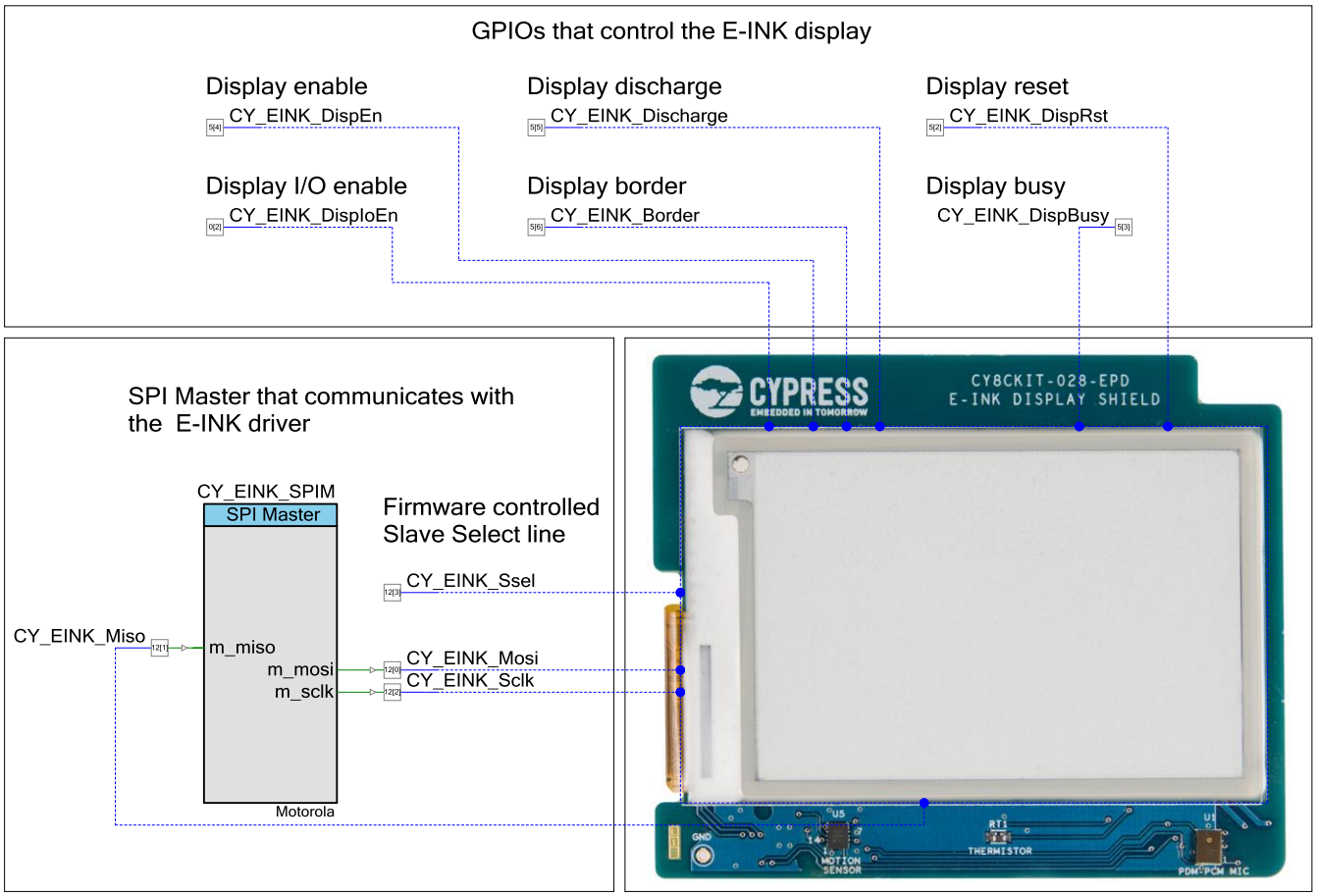


Figure 10. TopDesign Schematic: E-INK Display Library



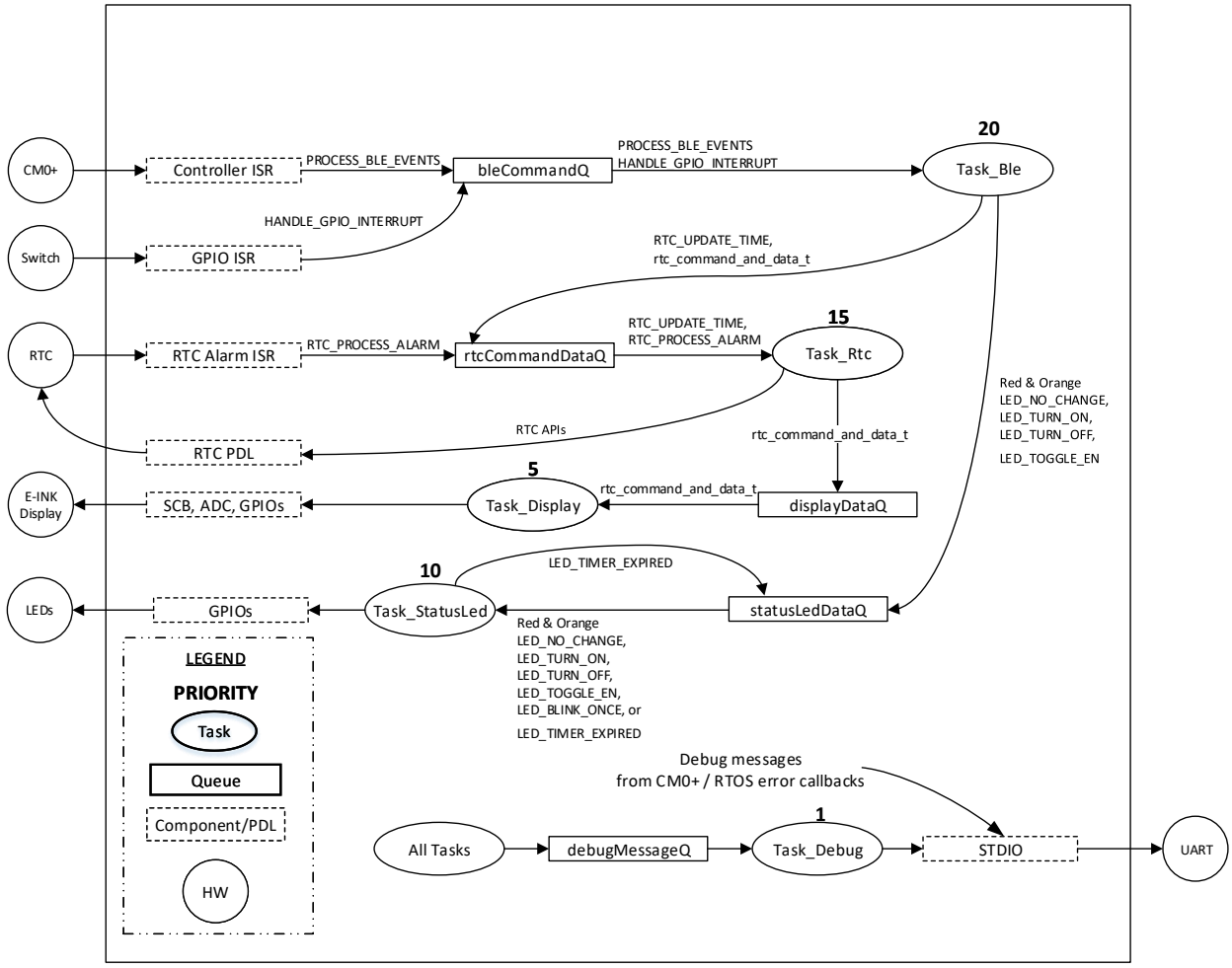
The code example consists of the following files:

- *FreeRTOSConfig.h* contains the FreeRTOS settings and configuration. Non-default settings are explained with in-line comments.
- *main_cm0p.c* contains functions that starts up the BLE controller, starts up CM4, and services BLE stack events.
- *main_cm4.c* contains the main CM4 function, which is the entry point and execution of the firmware application. The main function sets up user tasks and then starts the RTOS scheduler.
- *ble_task.c/.h* contain the task and associated functions that handle BLE communication and operation.
- *display_task.c/.h* contain the task that initialize the E-INK display and show the instructions to use code example at startup[1].
- *rtc_task.c/.h* contain the task that initialize RTC and process RTC interrupts.
- *status_led_task.c/.h* contain the task that controls status LED indications.
- *uart_debug.c/.h* contain the task and functions that enable UART based debug message printing.
- *screen_contents.c/.h* contain the text and background images used by the display module.

Figure 11 shows the RTOS firmware flow of this code example.

[1] For a detailed list of files included in the E-INK Library, see the code example, CE218133 PSoC 6 MCU E-INK Display with CapSense.

Figure 11. Firmware Flow



PSoC Creator Components

See the PSoC Creator project for details of PSoC Component configurations and system wide resource settings.

Table 2. List of PSoC Creator Components

Component	Instance Name	Function
BLE	BLE	The BLE Component is configured as a current-time-service client. It can receive date and time information from a current time server such as an iPhone.
RTC	RTC	The real-time clock performs accurate time keeping and generates alarm interrupts at 1-minute intervals to update the display.
MCWDT	MCWDT	The MCWDT Counter0 is configured to generate periodic interrupts at 0.5 second intervals. MCWDT interrupts are used to control the status LEDs and turn them off when not required, to save power.
Digital Output Pin	Pin_LED_Red Pin_LED_Orange	These GPIOs are configured as firmware controlled digital output pins that control status LEDs.
Digital Input Pin	Advertise	This pin is configured as a digital input pin that is used to generate interrupts when the user button (**SW2**) is pressed.
Global Signal Reference	GlobalSignal	The Global Signal Component is configured to extract interrupts from **Advertise** pin.

Note: See the code example CE218136 - PSoC 6 MCU E-INK Display with CapSense (RTOS) for more details on components used by E-INK library. See the PSoC Creator project for more details of PSoC Component configurations and design wide resource settings.

Related Documents

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 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
AN219434 – Importing PSoC Creator Code into an IDE for a PSoC 6 MCU Project	Describes how to import the code generated by PSoC Creator into your preferred IDE
PSoC Creator Component Datasheets	
BLE	Supports I^2C slave, master, and master-slave operation configurations using SCB
RTC	Provides real-time clock
UART	Provides asynchronous communication interface using SCB hardware
Pins	Supports connection of hardware resources to physical pins
Interrupt	Provides Interrupt component settings
Device Documentation	
PSoC 6 MCU: PSoC 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
BMI160 Motion Sensor datasheet	PSoC 6 MCU: PSoC 62 Datasheet
Development Kit Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	

Document History

Document Title: CE222604 – PSoC 6 MCU with BLE Connectivity: RTC with Current Time Service (RTOS)

Document Number: 002-22604

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6093651	AJYA	03/30/2018	Initial public release version
*A	6305473	AJYA	09/11/2018	Refined description for operation section

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 Spansion 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, Spansion, the Spansion 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.