Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.

CE218138 – PSoC 6 MCU with BLE Connectivity: BLE Thermometer (RTOS)

Objective

This code example demonstrates interfacing PSoC® 6 MCU with a thermistor circuit to read temperature information and sending the data over Bluetooth Low Energy Health Thermometer Service (HTS) to a mobile device running CySmart™ mobile application.

Overview

This code example demonstrates interfacing PSoC 6 MCU with BLE Connectivity (PSoC 6 MCU) with a thermistor circuit to read temperature information and sending the temperature data as BLE HTS indications to a mobile device running CySmart mobile application. In addition, PSoC 6 MCU's real time clock (RTC) generates alarms (interrupts) at every minute to show temperature information on the E-INK display when BLE is not connected.

In more detail:

- BLE connectivity using Health Thermometer Service and Device Information Service
- ADC scans two differential channels and averages multiple samples without the need for CPU intervention for accurate temperature measurement from a thermistor circuit
- An "always-on" E-INK display that shows the instructions to use the code example. In addition, the E-INK display refreshes at one minute intervals to show the temperature if BLE is not connected.
- Low power operation using the Deep-Sleep mode with MCWDT, RTC and GPIO interrupts

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) 3.0.1

Programming Language: C (Arm® GCC 5.4.1)

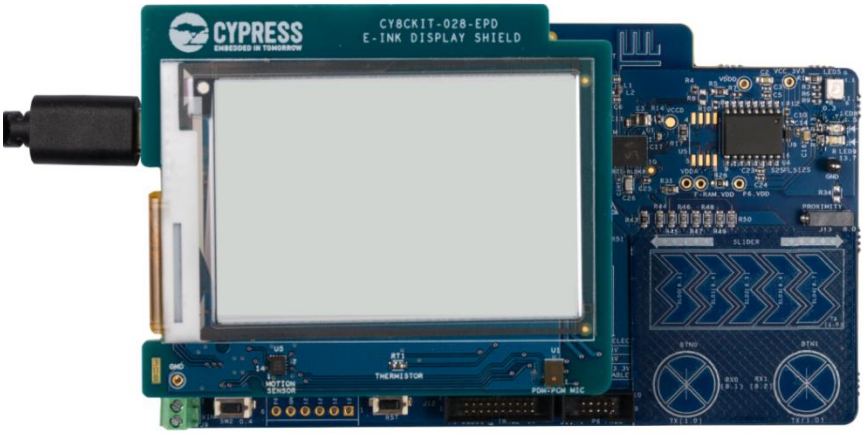
Associated Parts: All PSoC 6 MCUs with BLE Connectivity

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 on the Pioneer Board as shown in Table 1.

Table 1. Switch and Jumper Selection

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

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.

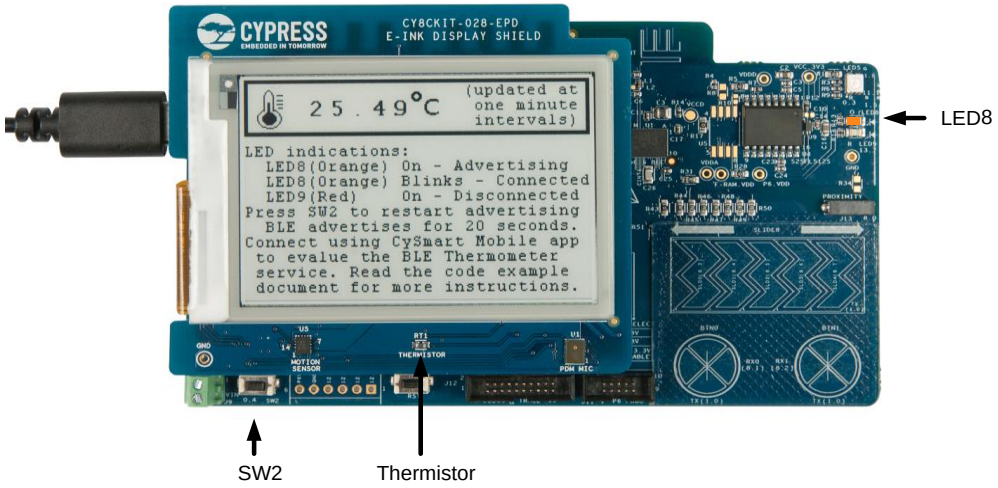
Operation

To verify this code example using the CySmart mobile application (see the CySmart Mobile App webpage), follow these steps:

1. Install the CySmart app.
2. Power the Pioneer Board through the USB connector **J10**.
3. Program the Pioneer Board with the CE220567_BLE_Thermometer project. See the Pioneer Kit guide for details on how to program firmware into the device.

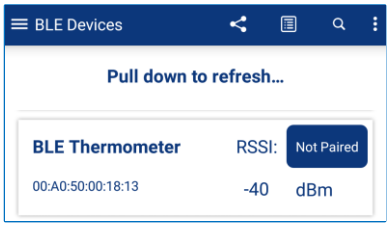
   After programming successfully, the E-INK display refreshes and shows the instructions to use this project and the current temperature. After this event, BLE starts advertising. The advertising timeout is configured to be 20 seconds. The orange LED (**LED8**) remains ON during this period to indicate the BLE advertising state.

Figure 2. BLE Advertising



4. If the BLE advertisement has timed out (**LED8** is OFF), press **SW2** to restart advertisement.
5. Open the CySmart app on the mobile device. If Bluetooth is not enabled on the device, the application asks you to enable it.
6. After Bluetooth is enabled, the CySmart mobile application automatically searches for available devices and lists them. Select the **BLE Thermometer** peripheral as shown in Figure 3. A successful connection is indicated by **LED8** continuously blinking at half-second intervals.

Figure 3. BLE Thermometer Peripheral



7. When connected, the CySmart mobile application lists the services supported by the device. Scroll and select the Device Information Service icon, as shown in Figure 4. The application will now show list of device related information.

Figure 4. Device Information Service

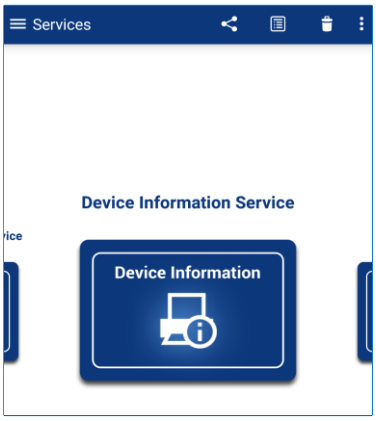
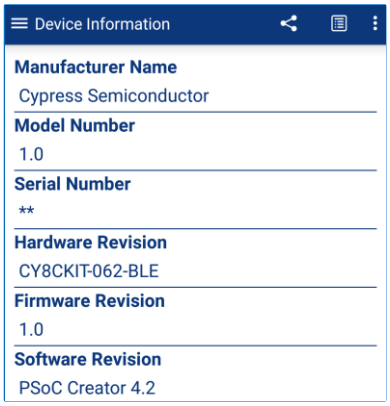


Figure 5. Device Information Service Page



8. Press the back button to return to the service selection page. Scroll and tap on the Health Thermometer service.

Figure 6. Health Thermometer Service

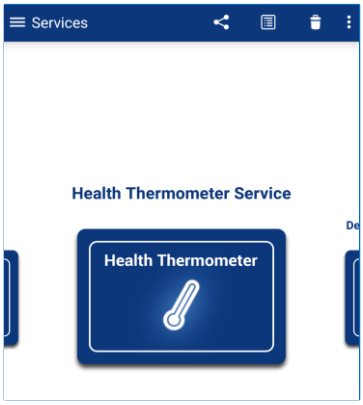
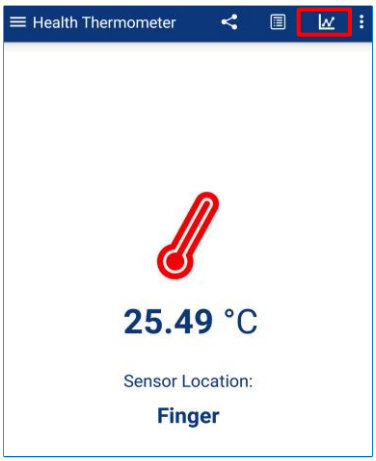
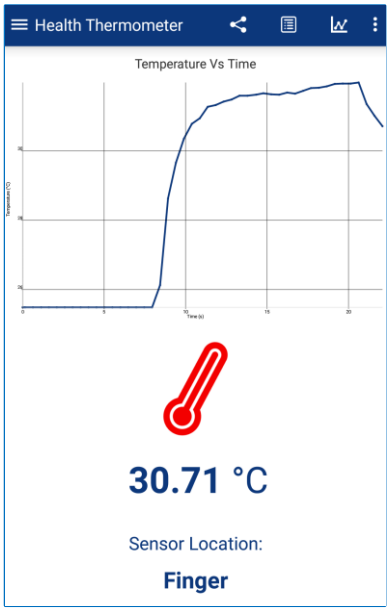


Figure 7. Health Thermometer Service Page



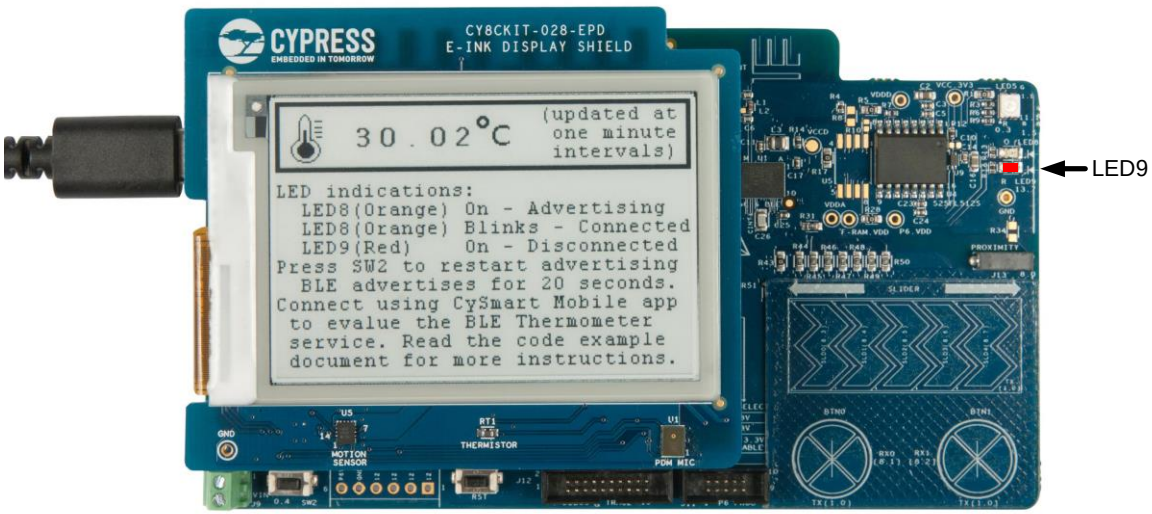
9. Enable the Temperature vs Time graph by touch the graph icon at the top-right corner of the page. Touch the Thermistor on the CY8CKIT-028-EPD E-INK Display Shield with your finger and observe an increase in Temperature indicated on the Health Thermometer Service Page (see Figure 8).

Figure 8. Health Thermometer Graph



10. On the service selection page, there is also a “GATT DB” selection, which allows you to examine the GATT database directly. From this page, you can read and write characteristics, as well as enable and disable indications.
11. If the CySmart app is closed, or Bluetooth is turned OFF, the red LED (**LED9**) will turn ON for three seconds to indicate a disconnect event. If BLE is disconnected, the E-INK display will refresh at one minute intervals to show the ambient temperature. Press **SW2** to restart the advertisement, if required.

Figure 9. Disconnect Indication



Design and Implementation

The BLE profile in this code example consists of HTS and the Device Information Service. Figure 10 shows the configuration of HTS Temperature Measurement characteristic. This characteristic contains one byte of flags and four bytes of temperature information in IEEE-11073 floating point format. The temperature measured by the ADC circuit at regular intervals is indicated to the GATT client device using the Temperature Measurement characteristic. The flags of Temperature Measurement characteristic, other characteristics of HTS, and the Device Information Service data are set during Component configuration. These are not updated during run-time. The service and characteristics configurations of the BLE Component can be found under the **GATT Settings** tab. See the GATT service specification of Health Thermometer Service and Device Information Service for more details. Note that Intermediate Temperature and Measurement Interval characteristics are not implemented in this code example.

Figure 10. BLE Health Thermometer Service Configuration

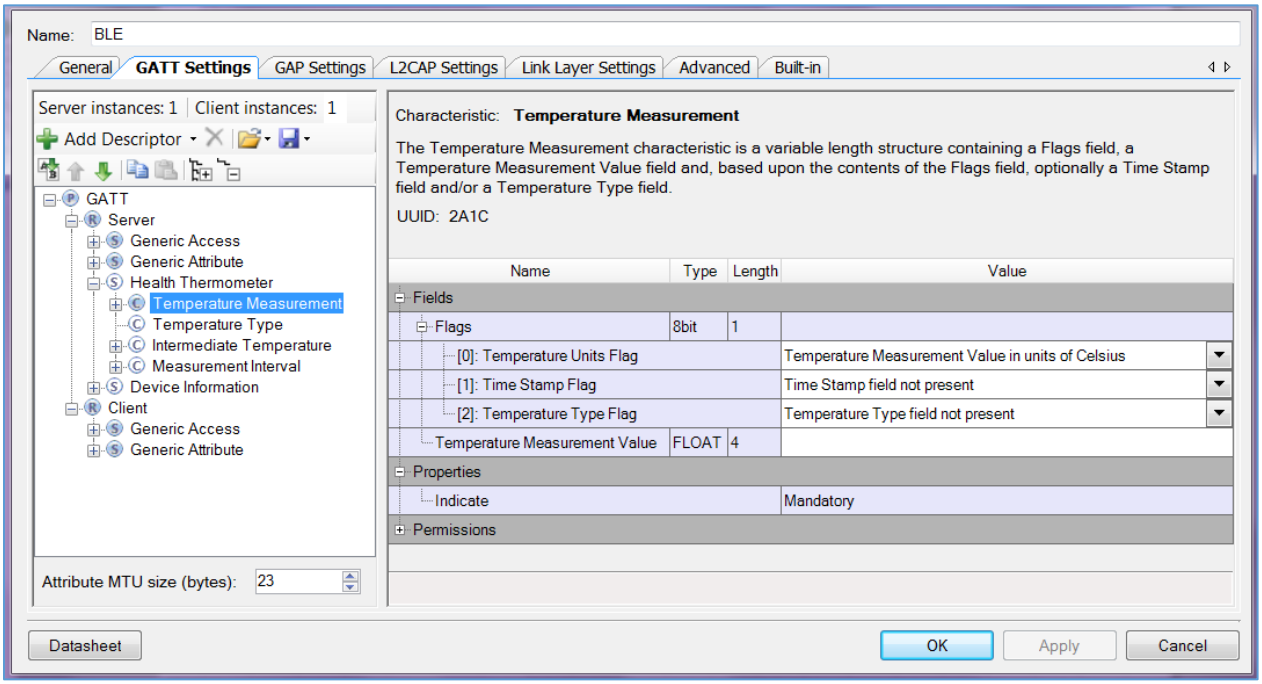


Figure 11,

Figure 12, and Figure 13 show the TopDesign schematic of this code example.

The ADC measures voltages across a thermistor and a series reference resistor using two differential channels. The ADC then uses its hardware post-processing block to average 256 samples to increase the SNR without the need for CPU intervention. The temperature value is then calculated from these two voltage readings using thermistor equation. See *temperature_task.c* source file for details of the calculations involved.

Figure 11. TopDesign Schematic: Temperature Measurement

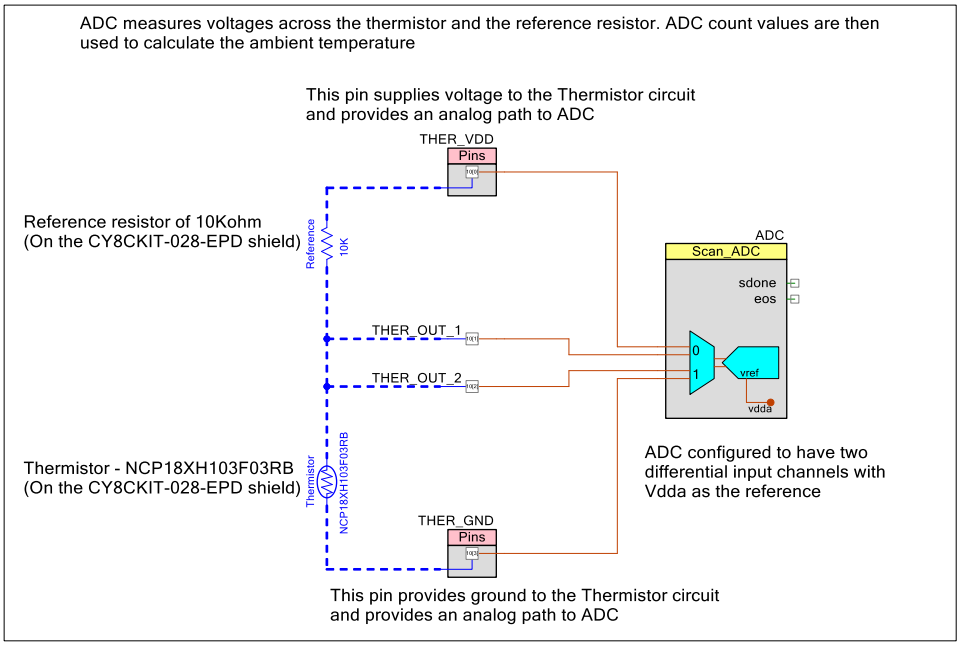


Figure 12. TopDesign Schematic: BLE and Indications

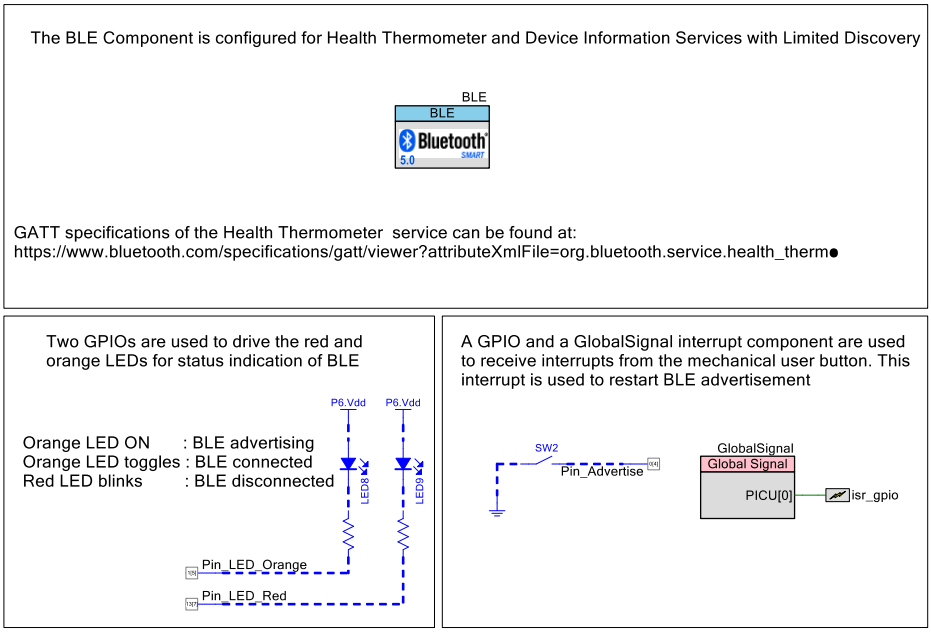
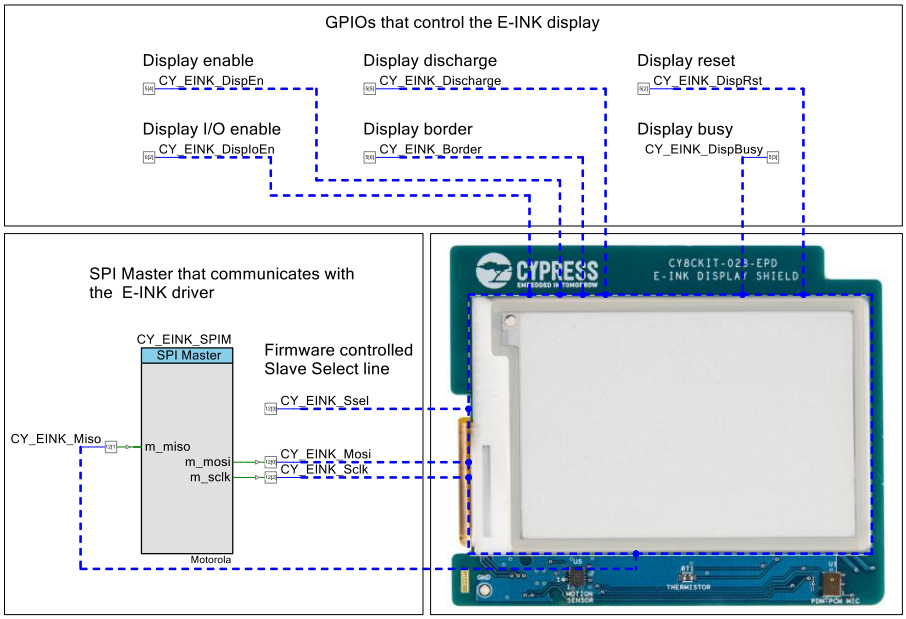


Figure 13 TopDesign Schematic: E-INK Display



The E-INK display shows the instructions to use this code example at startup. The display also refreshes at one minute intervals to show the temperature if BLE is not connected. Since E-INK displays consume no power to retain the display, supply to the display is tuned OFF during the interval between display refreshes. For more details on E-INK display, see the code example CE218136 – PSoC 6 MCU E-INK Display with CapSense (RTOS).

The code example consists of the following files:

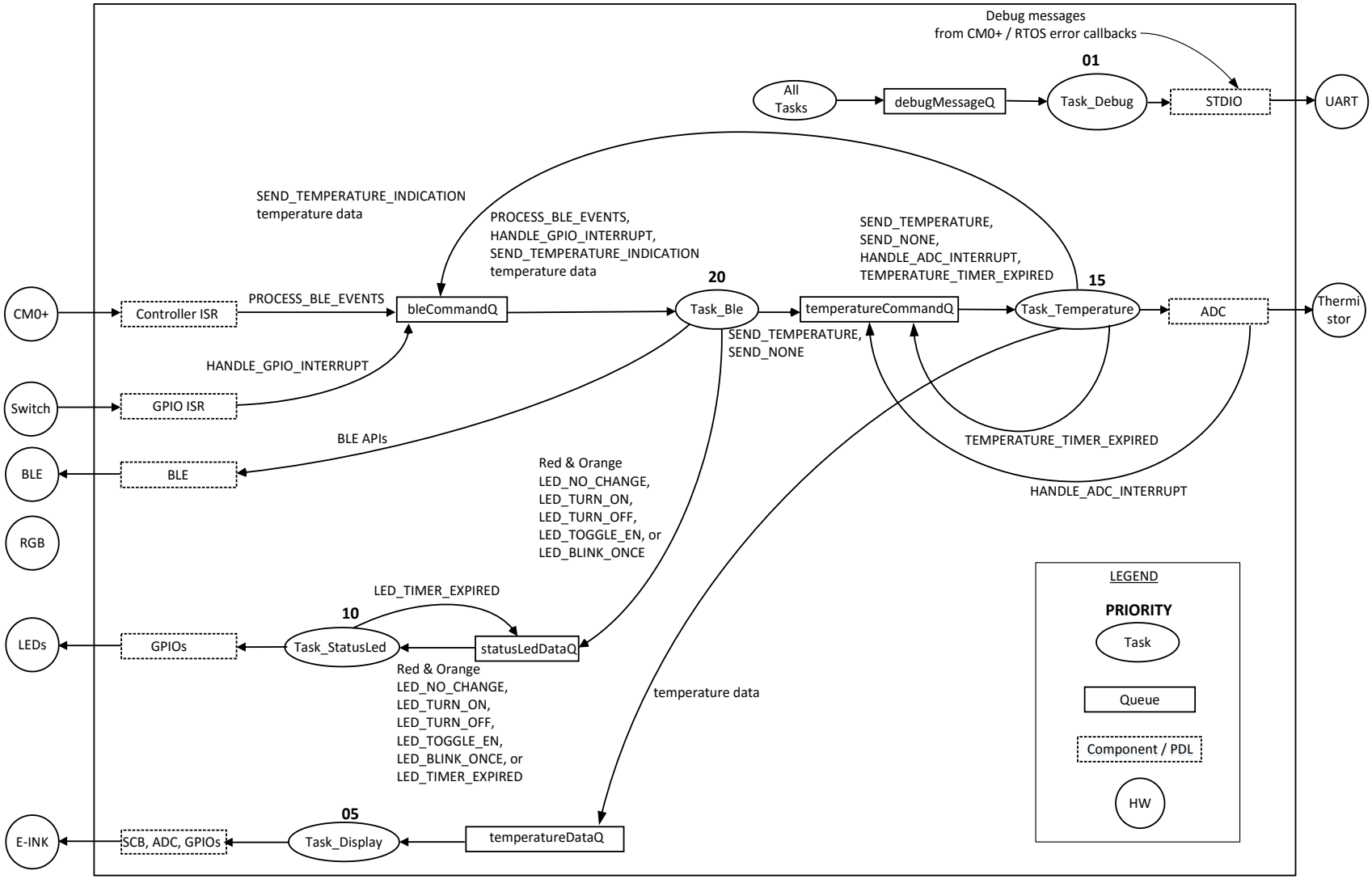
- *FreeRTOSConfig.h* contains the FreeRTOS settings and configuration. Non-default settings are explained with in-line comments.
- *main_cm4.c* contains the main function, which is the entry point and execution of the firmware application. The main function sets up user tasks and then starts the RTOS scheduler.
- *main_cm0p.c* contains functions that start up the BLE controller, start up the CM4, and continuously service BLE stack events.
- *ble_task.c/.h* contain the task and associated functions that handle BLE communication and operation.
- *ble_thermometer_config.h* contains the macros and datatypes used for the BLE health thermometer service.
- *temperature_task.c/h* contain the task that measures temperature.
- *rgb_led.c/.h* contain the task that initialize and control the RGB LED and intensity.
- *status_led_task.c/h* – contain the task that controls status LED indications.
- *display_task.c/.h* contain the task that initialize the E-INK display and show the instructions to use code example at startup[1].
- *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 14 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, CE218136 – PSoC 6 MCU E-INK Display with CapSense (RTOS)

Figure 14. RTOS Firmware Flow



Components

Table 2. List of PSoC Creator Components

Component	Instance Name	Function
BLE	BLE	The BLE Component is configured for Health Thermometer and Device Information Services with Limited Discovery.
Scanning SAR ADC	ADC	ADC measures the voltages across a thermistor and a series reference resistor using two differential channels to calculate the temperature.
MCWDT	MCWDT	The MCWDT Counter0 is configured to generate periodic interrupts at 0.5 second intervals. MCWDT interrupts provide timing to ADC scans. MCWDT interrupts are also used to control the status LEDs and turn them off when not required, to save power.
Real-time clock (RTC)	RTC	The real-time clock generates alarm interrupts at 1 minute intervals to update the display.
Analog Pin	THER_OUT_1, THER_OUT_2	These GPIOs connect thermistor circuit output to ADC input
Analog + Digital Output Pin	THER_VDD, THER_GND	These GPIOs provide power / ground to the thermistor circuit and provides an analog path to ADC
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 and temperature compensation.

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-Core CPU system Design	Describes the dual-core CPU architecture in PSoC 6 MCU, and shows how to build a simple dual-core 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	
Pins	Supports connection of hardware resources to physical pins
Bluetooth Low Energy	Supports BLE connectivity.
Scanning SAR ADC	Supports voltage measurement
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	
Training Videos	
PSoC 6 101: Lesson 1-4 FreeRTOS	

Document History

Document Title: CE218138 – PSoC 6 MCU with BLE Connectivity: BLE Thermometer (RTOS)

Document Number: 002-18138

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6096841	NIDH	03/13/2018	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 Forums | 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.