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.

CE223468 – PSoC 6 MCU: Interfacing BMI160 (I2C) in FreeRTOS

Objective

This code example demonstrates how to interface PSoC® 6 MCU with a BMI160 Motion Sensor over I^2C interface from a FreeRTOS task. This example reads raw motion data and estimates the orientation of the board.

Overview

This code example includes I2C interface configuration for BMI160, FreeRTOS configuration on Arm® Cortex®-M4 and FreeRTOS task implementation for BMI160 sensor interface over I^2C. The BMI160 sensor task configures the I^2C PDL interrupt callback where the task resume/wakeup is handled. The task wakes up periodically, gets data from BMI160 (suspends while the I2C hardware retrieves the data), converted to indicate the orientation of the sensor and then the orientation information is displayed on terminal application using UART interface.

This code example assumes that you are familiar with the PSoC 6 MCU device and the PSoC Creator™ Integrated Design Environment (IDE). If you are new to PSoC 6 MCU, see 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 and Arm MDK 5.22)

Associated Parts: All PSoC 6 MCU parts

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	3.3 V	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

This code example requires a PC terminal emulator to display orientation information.

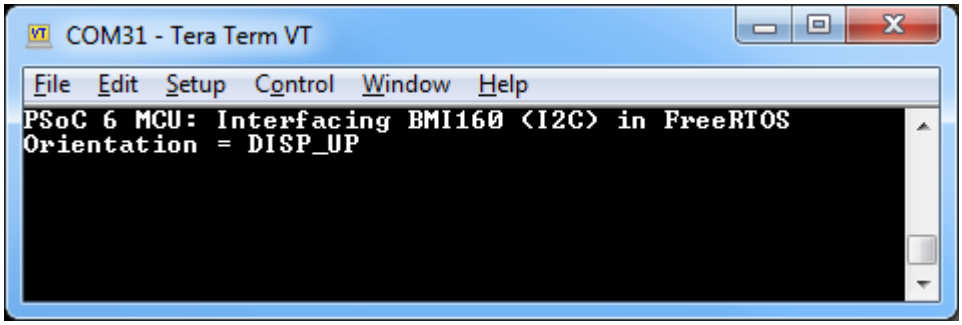
Operation

1. Connect the Pioneer Board to your PC using the provided USB cable through the USB connector (J10).
2. Open a terminal software such as Tera Term and select the KitProg2's COM port with a baud rate setting of 115200 bps. Use the default settings for other serial port parameters.
3. Build the project and program it into the PSoC 6 MCU. Choose **Debug** > **Program**. For more information on device programming, see the CY8CKIT-062-BLE kit guide. Flash for both CPUs is programmed in a single program operation.

Note: During the build process, do not replace *stdio_user.h* and *FreeRTOSConfig.h* file if prompted by PSoC Creator.

On successful programming, confirms that the terminal application displays the code example title and the initial orientation as shown in Figure 2.

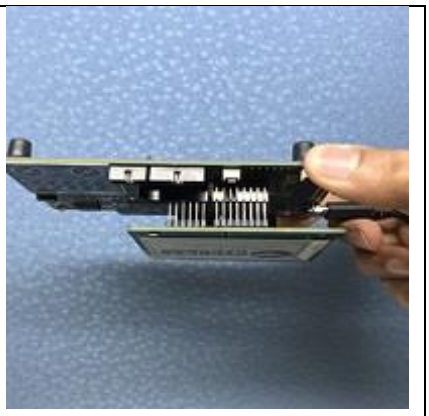
Figure 2. Terminal Application Displaying Startup Message



Note: If the terminal displays an error message, check the connection of the Motion Sensor or E-INK Display shield with the Pioneer Baseboard.

4. The accelerometer sensor data is used to estimate the board's orientation. The terminal application display shows one of the six orientation states as shown in Table 2.

Table 2. Orientation States

 Orientation = DISP_UP	 Orientation = DISP_DOWN
 Orientation = RIGHT_EDGE	 Orientation = LEFT_EDGE
 Orientation = TOP_EDGE	 Orientation = BOTTOM_EDGE

Design and Implementation

The E-INK Display Shield (CY8CKIT-028-EPD) contains a BMI160 motion sensor (U5), which is a low-power inertial measurement unit (IMU) providing 3-axis acceleration and 3-axis gyroscopic measurements. PSoC 6 MCU interfaces with this sensor and reads motion data, which is converted into Orientation outputs, and displayed on the terminal application.

The BMI160 motion sensor is interfaced with PSoC 6 MCU using an I^2C interface and two interrupt pins. BMI160 has a hardware-selectable I^2C slave address, depending on the logic driven on the SDO pin. On the E-INK Display Shield, the SDO pin is pulled to GND, which selects the slave address 0b1101000 (0x68).

BMI160 provides two output pins (INT1 and INT2) to which various interrupt events can be assigned. In this example, Orientation interrupt is assigned to INT2. On the E-INK Display Shield, INT1 and INT2 pins are connected to pin 2 and 1 respectively of J3. On the Pioneer Baseboard, INT1 and INT2 connects to P13[1] and P13[0] of PSoC 6 MCU. See the BMI160 datasheet for more details on interrupt outputs. The INT2 output is configured to provide a rising-edge signal with a pulse width of 2.5 ms. On PSoC 6 MCU, P13[0] is configured as an input pin and is internally pulled down. The interrupt is used to detect when there is a change in orientation. Raw accelerometer data is read and processed on orientation interrupt to compute the orientation.

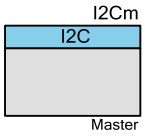
In PSoC Creator, an SCB-based I^2C Component is used to implement the I^2C Master interface to BMI160. The I^2C Data Rate is set to 400 kbps. Configuration of the motion sensor and reading accelerometer information are performed over this interface.

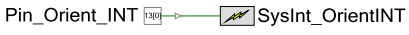
The Motion Task is suspended after initiating an I2C transfer and resumes the task after the transfer is complete. This example configures the I2C PDL interrupt callback where the task resume/wakeup is handled.

Interrupt callbacks of the I2C PDL is configured a FreeRTOS task based I2C sensor interface. The example uses BMI160 available in CY8CKIT-028-EPD shield to demonstrate the implementation. The task wakes up on BMI160 sensor interrupt, gets data from BMI160 (suspends while the I2C hardware retrieves the data), processes the data and then puts the data out through UART.

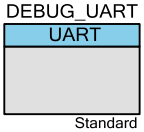
Figure 3. PSoC Creator Schematic Showing Motion Sensor Interface and Debug Outputs

<table>
<tr>
<td>I2C configured as master (400kbps)
Reads from BMI160 Motion Sensor (slave address 0x68)


</td>
<td>Motion Sensor is configured to provide interrupts on INT2 pins PSoC 6 MCU pin 13[0] is configured to detect interrupt on change in orientation


</td>
</tr>
<tr>
<td colspan="2">UART is used to transmit debug information. The UART debug can be enabled by right-clicking the component and selecting "enable".

See uart_debug.h file for additional steps.



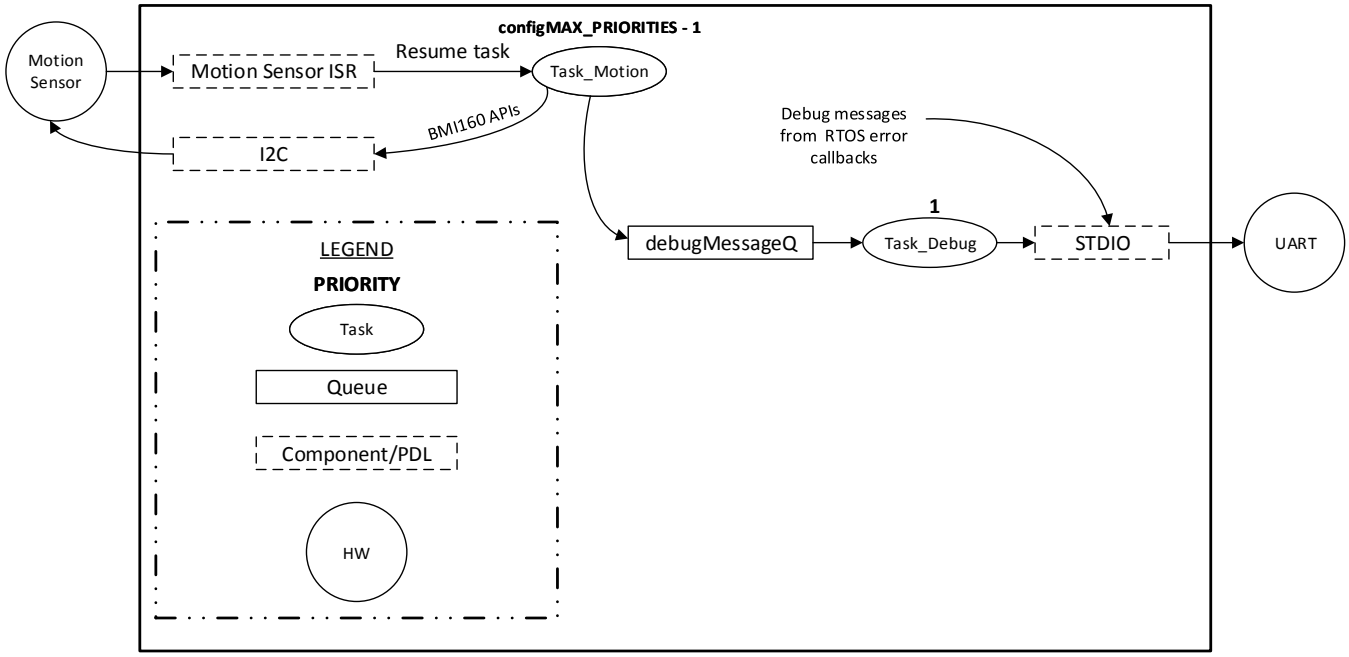
Set up the serial port terminal emulator with these settings:
Baud rate : 115200
Data size : 8-bit
Parity : None
Stop : 1-bit
Flow Control : None</td>
</tr>
</table>

The project 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 for C-M4, 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* enables CM4.
- *bmi160.c/.h*, *bmi160_def.h* contain the sensor driver for BMI160 motion sensor. For the latest version of this driver, visit the GitHub repository.
- *task_motion.c/.h* contain the task and macro definitions related to motion sensor application outputs. This includes the functions used to initialize and configure the motion sensor, set up interrupts, and compute motion outputs like orientation from raw accelerometer data.
- *i2cm_support.c/.h* contain the I^2C master read and write functions.
- *stdio_user.h* contain functions for the debug and UART functionality.
- uart_debug.c/h contain the task and functions that enable UART based debug message printing.

Figure 4 shows the RTOS firmware flow for this project.

Figure 4. Firmware Flowchart



Components and Settings

Table 3 lists the PSoC Creator Components used in this example, how they are used in the design, and the non-default settings required so they function as intended.

Table 3. PSoC Creator Components

Component	Instance Name	Purpose	Non-default settings
I2C (SCB)	I2Cm	I²C master for communicating with the motion sensor	Mode: Master Data Rate (kbps): 400
Digital Input Pin	Pin_Orient_INT	Pin connected to the motion sensor interrupt signal	**[General tab]** Drive mode: Resistive Pull Down **[Input tab]** Interrupt: Rising Edge
UART (SCB)	DEBUG_ UART	Serial communication block for debug output on terminal	Default
Interrupt	SysInt_OrientINT	Component to receive signal from Pin_Orient_INT	Interrupt Type: Rising-Edge Triggered

For information on the hardware resources used by a Component, see the Component datasheet.

Reusing This Example

This example is designed for the PSoC 6 BLE Pioneer Kit. To port the design to a different PSoC 6 MCU device, kit, or both, change the target device using the Device Selector and update the pin assignments in the Design Wide Resources Pins settings as needed. For single-core PSoC 6 MCU devices, port the code from *main_cm4.c* to *main.c*.

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	
I2C	Supports I²C slave, master, and master-slave operation configurations using SCB
SPI	Provides an industry-standard, 4-wire master SPI interface using SCB hardware
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: CE223468 – PSoC 6 MCU: Interfacing BMI160 (I2C) in FreeRTOS

Document Number: 002-23468

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6084060	AJYA	06/20/2018	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 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.