CE221122 – PSoC 6 MCU Non-blocking Flash Write

Objective

These examples use a non-blocking Peripheral Driver Library (PDL) API function to write a flash row. This is done using polling and interrupt methods.

Overview

These examples implement the flash write using polling and interrupt to complete the flash write operation.

Requirements

Tool: PSoC Creator™ 4.2

Programming Language: C (Arm® GCC 5.4-2016-q2-update, Arm MDK 5.22)

Associated Parts: All PSoC 6 MCU parts

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

Hardware Setup

The code example works with the default settings of the CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit. If the settings are different from the default values, see the "Selection Switches" table in the kit guide to reset to the default settings.

Note that the green LED does not operate at 1.8 V. By default, the kit is set to 3.3 V.

Software Setup

None.

Operation

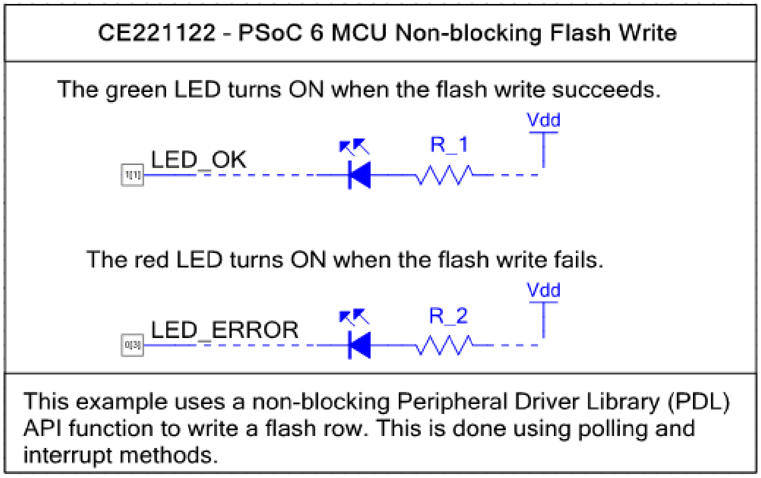
1. Connect CY8CKIT-062 BLE to a USB port on your PC.
2. For each example, build and program the application into CY8CKIT-062 BLE. For more information on building a project or programming a device, see PSoC Creator Help. The green LED turns ON if the flash write is successful.
3. Set the value of the macro `MAKE_FLASH_WRITE_FAIL` in the *main_cm4.c* file to 1, rebuild the project, reprogram the device, and observe that the red LED turns ON to indicate that the flash write operation failed.

Design

In these examples, the Arm Cortex®-M0+ core simply enables the Cortex-M4 core. The CM4 core executes the implementation.

These examples use a constant array with a size equaling the size of one flash row. The main function places this array at an address in flash such that it occupies one complete flash row. It calls the non-blocking flash write API function. The NonBlockingFlashWrite_Polling example uses a polling method to complete the flash write operation. The NonBlockingFlashWrite_Interrupt example uses an interrupt method to complete the flash write operation. It then verifies the flash data by comparing the flash data with the written data. If the flash write is successful, the green LED turns ON. Otherwise the red LED turns ON. Figure 1 shows the PSoC Creator schematic, which is used for both the code examples.

Figure 1. TopDesign Schematic



CE221122 - PSoC 6 MCU Non-blocking Flash Write

The green LED turns ON when the flash write succeeds.

The red LED turns ON when the flash write fails.

This example uses a non-blocking Peripheral Driver Library (PDL) API function to write a flash row. This is done using polling and interrupt methods.

Components and Settings

Table 1 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 1. PSoC Creator Components

Component	Instance Name	Purpose	Non-default Parameter Settings
Digital Output Pin	LED_OK, LED_ERROR	For showing success and failure information.	Uncheck the HW Connection box, check the External Terminal box

Design-Wide Resources

Table 2 shows the pin assignment for the code example.

Table 2. Pin Names and Location

Pin Name	Location
LED_ERROR	P0[3]
LED_OK	P1[1]

Reusing This Example

This code example is designed to run on CY8CKIT-062-BLE with the PSoC 6 MCU. To port the design to another PSoC 6 MCU device or kit, change the target device in Device Selector, and change the pin assignments in the *cydwr* settings. 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 BLE devices and how to build your first PSoC Creator project
PSoC Creator Component Datasheets	
GPIO	Supports connection of hardware resources to physical pins
Device Documentation	
PSoC 6 MCU: PSoC 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
Development Kit (DVK) Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit	

Document History

Document Title: CE221122 - PSoC 6 MCU Non-blocking Flash Write

Document Number: 002-21122

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5894792	VJYA	02/25/2018	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

Forums | WICED IOT 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

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.