CE195313 – PSoC Emulated EEPROM

Objective

This example demonstrates how to use the Em_EEPROM middleware in PSoC® 6 MCU devices using ModusToolbox™.

Requirements

Tool: ModusToolbox™ 1.1

Programming Language: C

Associated Parts: All PSoC 6 MCU parts

Related Hardware: PSoC 6 WiFi-BT Pioneer Kit, PSoC 6 BLE Pioneer Kit, PSoC 6 WiFi-BT Prototyping Kit

Overview

In this example, a counter is read from the emulated EEPROM (Em_EEPROM), incremented, written back to Em_EEPROM, and printed over UART. This occurs at every device reset or power cycle. As a result, the UART prints out an incrementing value at every reset.

Hardware Setup

This example uses the PSoC 6 WiFi-BT Pioneer Kit's default configuration. Refer to the kit guide to ensure that the kit is configured correctly. You can also use other PSoC 6 kits by importing the application for that kit.

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**; section "PSoC 6 MCU KitProg Firmware Loader". If you do not upgrade, you will see an error like "unable to find CMSIS-DAP device" or "KitProg firmware is out of date".

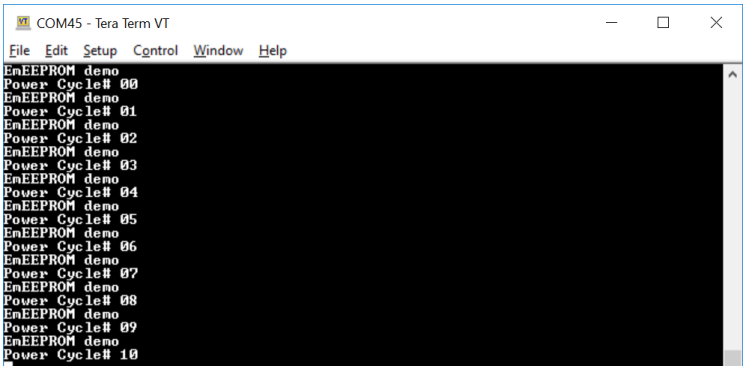
Software Setup

This example uses a terminal emulator program. Install one on your PC if you don't have one. The instructions use Tera Term.

Operation

1. Plug the kit into your computer's USB port.
2. Open a terminal program and select the KitProg3 COM port. Set the serial port parameters to 8N1 and 115,200 baud.
3. Import the application into a new workspace. See KBA225201.
4. Program the PSoC 6 MCU device. Select the 'mainapp' project. In the **Quick Panel**, scroll down, and click **Program (Kitprog3)**.
5. After programming the device, observe the counter value in the serial terminal. It increments each time you reset or cycle power. A sample output is shown in Figure 1.

Figure 1. Output window



Debugging

You can debug the example to step through the code. Use Debug (KitProg3) configuration in the Quick Panel. See KBA224621 in the Cypress community to learn how to start a debug session with ModusToolbox IDE.

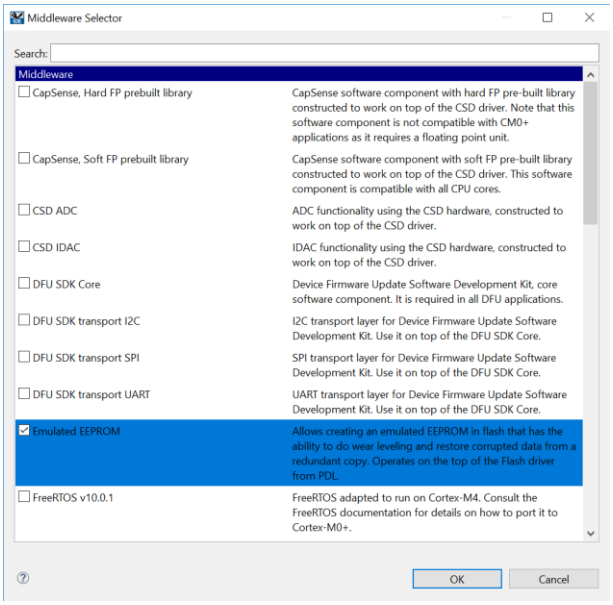
Design and Implementation

This example demonstrates how to use Em_EEPROM middleware. The application also uses a Serial Communication Block (SCB) resource, configured as UART. On startup, the example initializes the SCB and the Em_EEPROM block in flash. Then, a read operation is performed to verify whether the data stored in EEPROM is valid. If valid, the counter is incremented by one and the new value of the counter is written back to the Em_EEPROM. Otherwise, the expected valid data is written to the Em_EEPROM. The firmware then reads the value in the Em_EEPROM and prints it to a terminal window via UART. Every time the device is reset or power is cycled, the counter is incremented and printed on serial terminal.

To add Emulated EEPROM middleware to an application, use the middleware selector. Right-click **<application_name>_mainapp** and select **ModusToolbox Application Middleware Selector** from the drop-down menu. Select the Emulated EEPROM middleware as shown in Figure 2. The required middleware source files can be found at

<application_name>_mainapp\psoc6sw-1.1\components\psoc6mw\em_eeprom

Figure 2. Middleware Selector



The application code runs on the CM4 CPU.

The firmware includes the declaration of EEPROM storage and details of the EEPROM configuration and context structures. (In PSoC Creator™, these tasks were handled by the Em_EEPROM Component.)

In this example, EEPROM storage can be declared in either the application flash (user flash) or in the section of flash dedicated for Em_EEPROM. If the data is written to the user flash, a blocking write must be used. This is because a write to and read/execute from the same flash sector at the same time while using non-blocking writes may cause a HardFault exception. Either blocking or non-blocking write will work for the Em_EEPROM flash, because it is in a different flash sector. For more details, see PDL documentation on Flash System Routine (Flash).

The Serial Communication Block resource is configured as a UART at 115200 baud, 8N1. It is connected to a clock operating at 923 kHz to generate the correct baud rate. The RX is on pin P5[0] and TX is on P5[1], to match the pin usage on the kit.

Resources and Settings

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

Table 1. ModusToolbox Resources

Resource	Alias	Purpose	Non-default Settings
SCB	KIT_UART	Prints a message to a terminal window.	See Figure 3.
Digital Output Pin	KIT_UART_TX	Used for UART transmit (Tx).	See Figure 4.
Digital Input Pin	KIT_UART_RX	Used for UART receive (Rx).	See Figure 5.

Figure 3, Figure 4, and Figure 5 show configuration settings for the resources used in this example.

Figure 3. UART Configuration

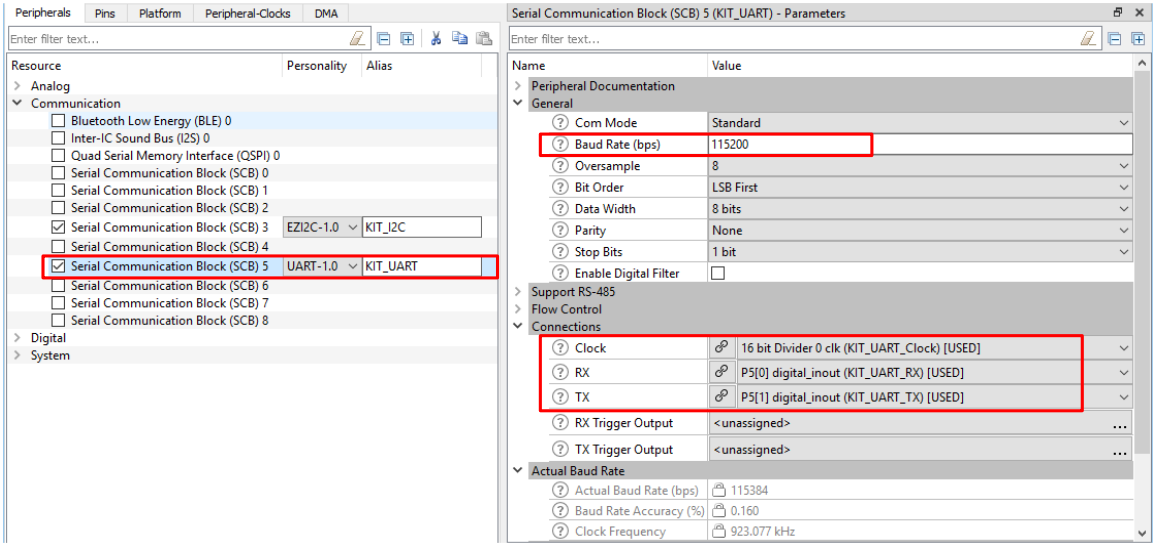


Figure 4. GPIO Pin Configuration for UART Tx

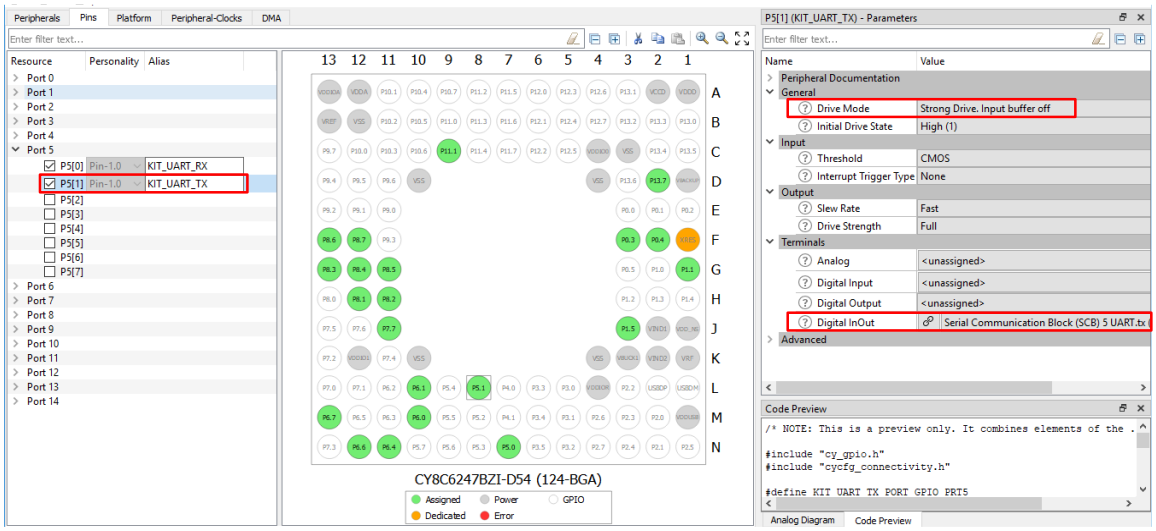


Figure 5. GPIO Pin Configuration for UART Rx

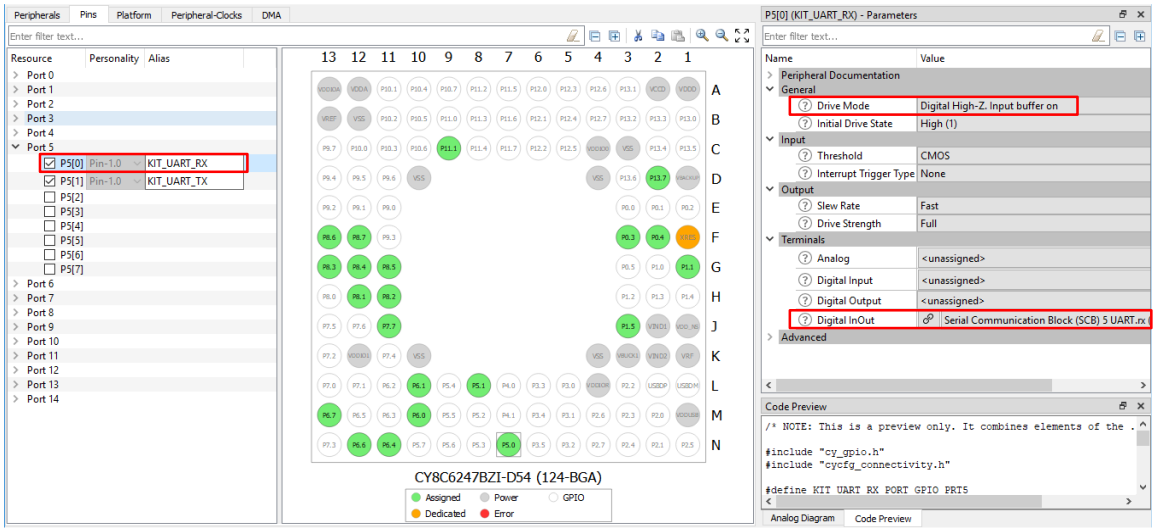
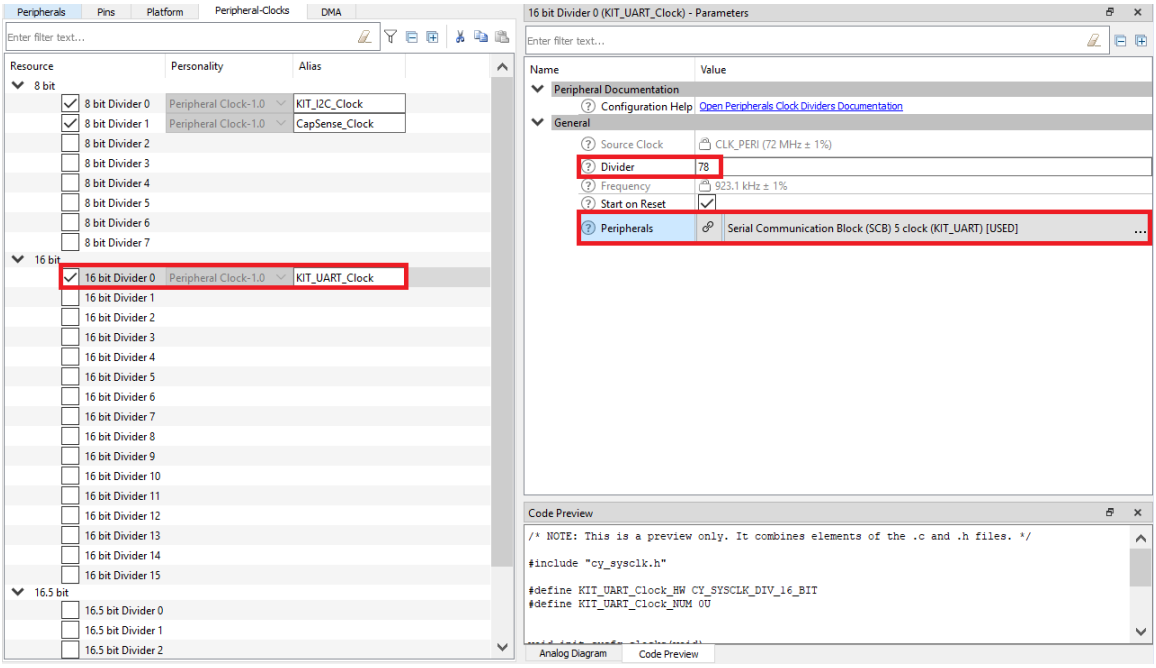


Figure 6 shows the Peripheral-Clock configuration for UART.

Figure 6. Peripheral-Clock Configuration for UART



Reusing This Example

This example is designed for the CY8CKIT-062-WIFI-BT Pioneer Kit. To use the design on a different PSoC 6 MCU kit, import the application for that kit. If you are unsure how to import an application, see KBA225201. If changing to a different kit, you may need to reassign pins. The kits below all use the same pins for UART communication.

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

Kit Name	Device Used	UART_RX	UART_TX
CY8CKIT-062-WiFi-BT	CY8C6247BZI-D54	P5[0]	P5[1]
CY8CKIT-062-BLE	CY8C6347BZI-BLD53	P5[0]	P5[1]
CY8CPROTO-062-4343W	CY8C624ABZI-D44	P5[0]	P5[1]

In some cases, a resource used by a code example (for example, a peripheral) is not supported on another device. In that case, the example will not work. If you build the code targeted at such a device, you will get errors. See the device datasheet for information on what a particular resource the device supports.

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 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	
Device Documentation	
PSoC 6 MCU: PSoC 62 Datasheet	PSoC 6 MCU: PSoC 62 Architecture Technical Reference Manual (TRM)
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 IDE	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: CE195313 - PSoC Emulated EEPROM

Document Number: 002-27683

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6605643	SBKR	07/01/2019	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 | Code Examples | Projects | Videos | Blogs | Training | Components

Technical Support

cypress.com/support

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