CE223820 – PSoC 6 MCU IPC Pipes

Objective

This example demonstrates how to use the inter-processor communication (IPC) driver to implement a message pipe in PSoC® 6 MCU. The pipe is used to send messages between CPUs.

Requirements

Tool: PSoC Creator™ 4.2, Peripheral Driver Library (PDL) 3.1.0

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

Associated Parts: All PSoC 6 MCU parts with dual CPUs

Related Hardware: CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit and CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit

Overview

In this example, the PSoC 6 MCU device works as a prompt calculator. The operands and operator are received as ASCII strings from a terminal emulator. The result is displayed back to the terminal. In firmware, the CM0+ CPU handles the UART communication with the terminal. The CM4 CPU parses the ASCII strings, converts operand strings to floating point numbers, and identifies the operator. It then executes the operation, if valid.

Messages containing ASCII strings are sent between the CPUs using the IPC pipe. A pipe has two endpoints, one on each CPU. Each endpoint contains a dedicated interrupt, which executes registered callbacks. The CM0+ CPU simply passes the ASCII string from the UART to the CM4 CPU, and from the CM4 to the UART.

Hardware Setup

This example uses the kit's default configuration. Refer to the kit guide to ensure that the kit is configured correctly.

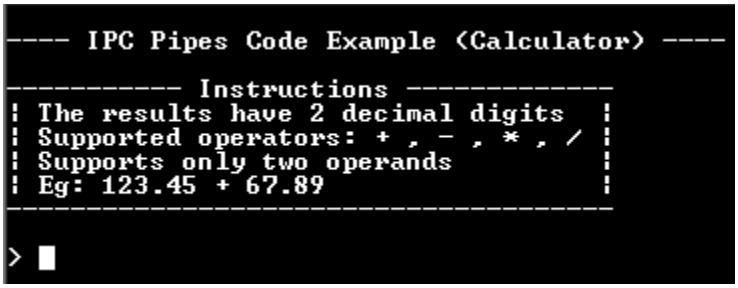
Software Setup

If necessary, install a terminal emulator like Tera Term or PuTTY on your computer. The example uses the terminal window to send and display messages.

Operation

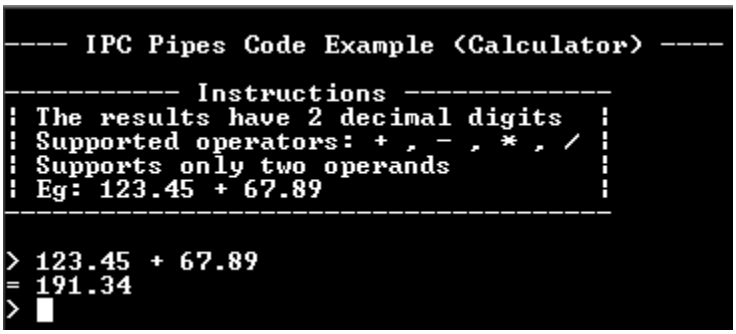
1. Plug the CY8CKIT-062 kit board into your computer's USB port.
2. Build the project and program it into the PSoC 6 MCU device. Choose **Debug** > **Program**. For more information on device programming, see PSoC Creator Help. Flash for both CPUs is programmed in a single program operation.
3. Open a serial port communication program such as Tera Term and select the corresponding COM port. Configure the terminal with 115200 baud rate, 8 data bits, 1 stop bits, and with parity and flow control set to none. Disable local echo.
4. The terminal should print the message shown in Figure 1. Press the reset button on the board in case it is not shown.

Figure 1. Initial Print Message in the Terminal



5. Type an expression using two operands and one operator. The calculator supports floating point numbers, and addition, subtraction, multiplication, and division operators.
6. Press Enter after typing the expression. The terminal should display the result, as shown in Figure 2.

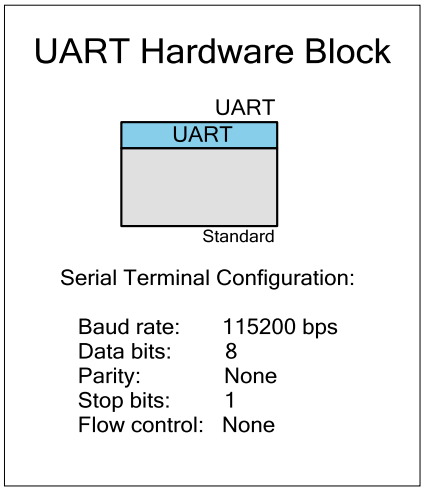
Figure 2. Result Printed in the Terminal



Design and Implementation

The design shown in Figure 3 has a UART Component. The UART is configured in RX+TX mode to transmit/receive data at 115200 baud rate.

Figure 3. PSoC Creator Project Schematic



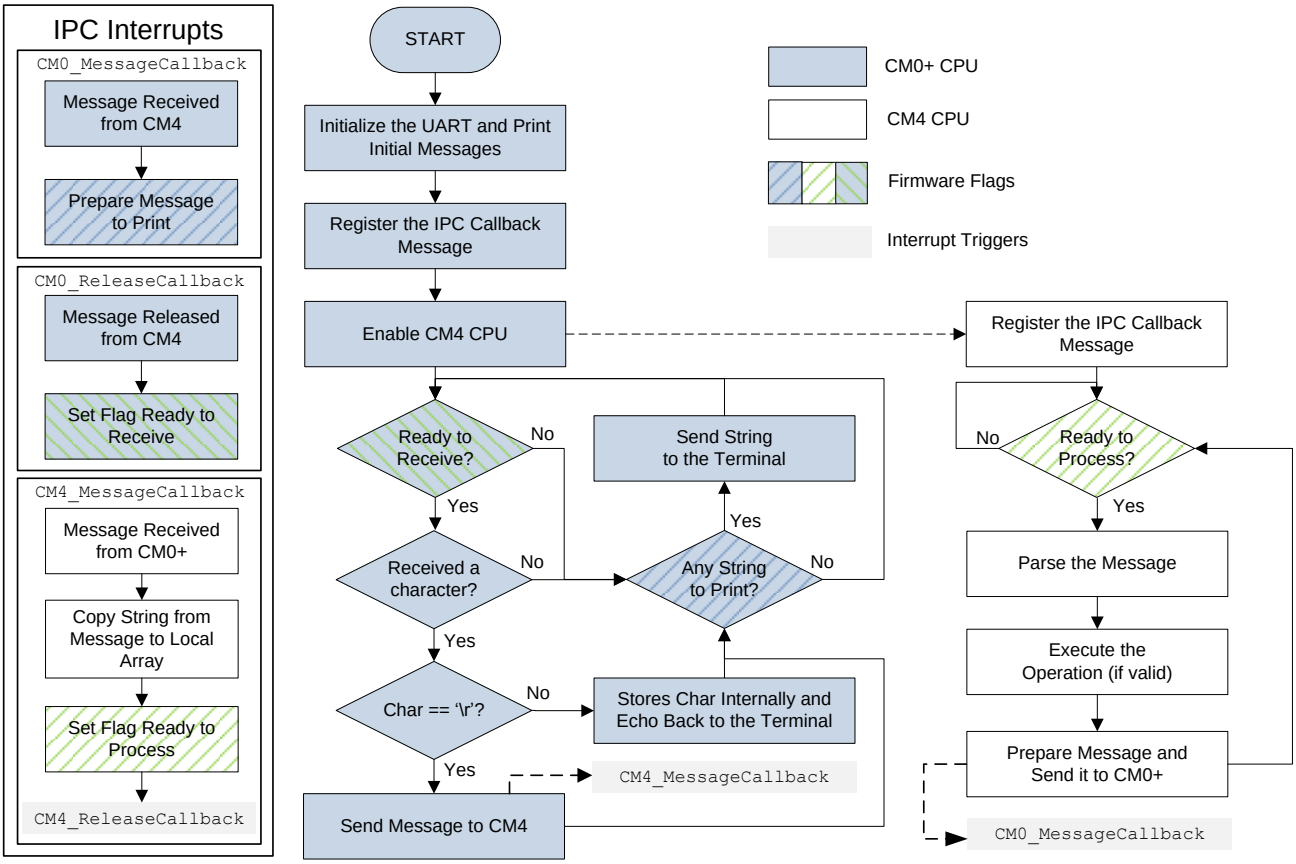
The CM0+ CPU initializes the UART and prints the initial messages to the terminal. It registers a callback to be executed when a message from the CM4 CPU is received. When registering this callback, an endpoint address and client ID are provided. The endpoint address designates which endpoint the callback is added to. The client ID determines the index in the callback array where the function pointer is saved. The CM4 CPU also registers its own callback to be executed when a message from the CM0+ CPU is received.

In the CM0+ CPU main loop, the firmware checks if any character was received. Once received, it echoes back to the terminal, so the user knows the exact characters transmitted. If the carriage return (Enter) character is received, CM0+ sends all the characters transmitted previously to CM4. When a message is received from CM4, CM0+ sends it to the terminal over UART.

When CM4 receives a message from CM0+, it parses it to extract the two operands and the operator. If the expression is valid, it calculates the result of the operation, converts it to a string, then sends it back to CM0+. If the expression is not valid, it sends an error message string to CM0+.

Figure 4 shows the firmware flowchart for the design.

Figure 4. Firmware Flowchart



Boxes filled with a diagonal pattern are related by firmware flags that are set in the IPC interrupt handlers. The main loop looks for the flag, acts if it is set, and then clear the flags.

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 Settings
UART	UART	Prints a message to a terminal window	All default

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

Because the example uses floating point numbers, the CM4 project is set up to include a floating point math library. To include the library in your own project, when using the GCC compiler, go to **Project** > **Build Settings** > **CM4 ARM GCC** > **Linker** > **Use newlib-nano Float Formatting**, and set to **True**.

Reusing This Example

This example is designed for the CY8CKIT-062-WiFi-BT pioneer kit. To port the design to a different PSoC 6 MCU device and/or kit, change the target device using the Device Selector and update the pin assignments in the Design Wide Resources Pins settings as needed.

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
Code Examples	
Visit the Cypress Code Example site for a comprehensive collection of code examples using PSoC Creator IDE.	
PSoC Creator Component Datasheets	
UART	Provides asynchronous serial communications
Device Documentation	
PSoC 6 MCU Datasheets	PSoC 6 MCU Technical Reference Manuals
Development Kit Documentation	
CY8CKIT-062-BLE PSoC 6 BLE Pioneer Kit CY8CKIT-062-WiFi-BT PSoC 6 WiFi-BT Pioneer Kit	

Document History

Document Title: CE223820 – PSoC 6 MCU IPC Pipes

Document Number: 002-23820

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6185420	RLOS	06/05/2018	New code example
*A	6438005	RLOS	01/10/2019	Updated document and project to use PDL 3.1.0

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, 2019. 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.