CE220465 – PSoC 6 MCU Cryptography: AES Demonstration

Objective

This code example demonstrates encryption and decryption of data using the Advanced Encryption Standard (AES) algorithm in PSoC® 6 MCU.

Requirements

Tool: ModusToolbox™ IDE 1.0

Programming Language: C

Associated Parts: All PSoC 6 MCU parts

Related Hardware: PSoC 6 BLE Pioneer Kit, PSoC 6 WiFi-BT Pioneer Kit, PSoC6 WiFi-Prototyping Kit

Overview

This code example encrypts and decrypts user input data using the AES algorithm using a 128-bit key. The encrypted and decrypted data are displayed on a UART terminal emulator.

Hardware Setup

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

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 If you do not upgrade, you will see an error like "unable to find CMSIS-DAP device" or "KitProg firmware is out of date".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

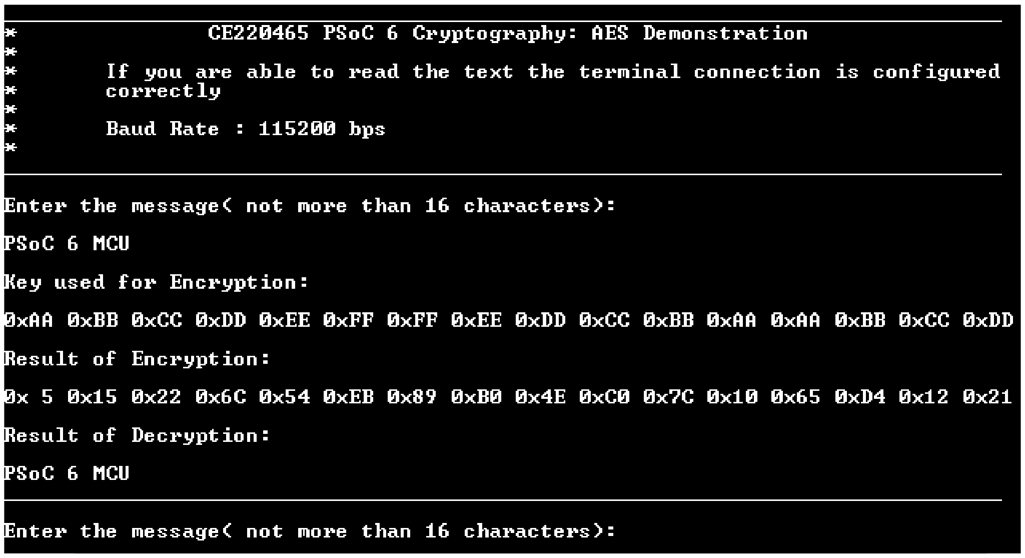
This example uses a terminal emulator to display the encryption and decryption results. If you don't have one, install one. The example uses Tera Term.

Operation

1. Connect the kit to your PC using the provided USB cable.
2. Import the code example into a new workspace. See KBA225201
3. Build the application. **Choose Project** > **Build All**.
4. Program the PSoC 6 MCU device. In the project explorer, select the **mainapp** project. In the Quick Panel, scroll to the Launches section and click the **Program (KitProg3)** configuration. Program configurations also build the code.
5. Open your terminal software and select the KitProg COM port. Set the other serial port parameters as follows:
    a. Baud rate: 115200 bps
    b. Data: 8 bit
    c. Parity: None
    d. Stop: 1 bit
    e. Flow control: None
6. Press the reset button on the kit and enter the message (not more than 16 characters) to be encrypted.
7. Observe the key and the results in the terminal window.

Note that the crypto block in PSoC 6 MCU accepts 128 bits (16 bytes) as input data. If you need to enter a message of more than 16 characters, modify the firmware to process the data in blocks of 16 characters. Figure 1 shows a sample output as displayed on Tera Term.

Figure 1. Sample Output Showing AES Encryption



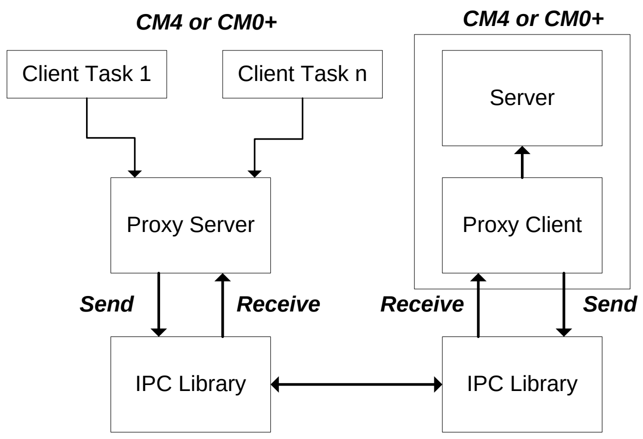
```
*                 CE220465 PSoC 6 Cryptography: AES Demonstration
*
*        If you are able to read the text the terminal connection is configured
*        correctly
*
*        Baud Rate : 115200 bps
*

Enter the message( not more than 16 characters):

PSoC 6 MCU

Key used for Encryption:

0xAA 0xBB 0xCC 0xDD 0xEE 0xFF 0xFF 0xEE 0xDD 0xCC 0xBB 0xAA 0xAA 0xBB 0xCC 0xDD

Result of Encryption:

0x 5 0x15 0x22 0x6C 0x54 0xEB 0x89 0xB0 0x4E 0xC0 0x7C 0x10 0x65 0xD4 0x12 0x21

Result of Decryption:

PSoC 6 MCU

Enter the message( not more than 16 characters):
```

Design and Implementation

AES is a symmetric block cipher data encryption algorithm. The AES operation works on 128-bit block size. AES is a symmetric algorithm which means that it uses the same key for encryption and decryption of data. The AES algorithm uses keys of 128 bits, 192 bits, or 256 bits of length.

Cryptography in PSoC 6 MCU is based on a Client-Server model. The firmware initializes and starts the Crypto server. In this example, the server runs on the CM0+ core. Access to the server is through the Inter-Processor Communication (IPC) driver. Figure 2 shows the Client-Server model for the Crypto block in PSoC 6 MCU.

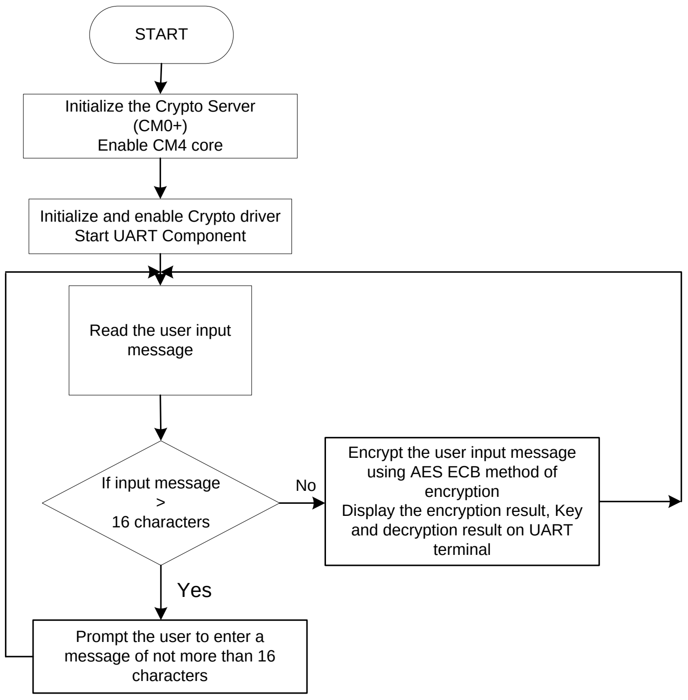
Figure 2. Crypto Client-Server Architecture in PSoC 6 MCU



The Crypto client typically runs on the CM4 CPU. The firmware initializes and starts the client. The firmware then provides the configuration data required for AES encryption technique and requests the crypto server to run the cryptographic operation.

The user input message is read from the UART terminal and encrypted using the AES algorithm with a key length of 128 bits. The 128-bit encrypted data is displayed on the UART terminal. Then, you can view the decrypted message on the UART terminal and verify that the decryption operation produces the same original encrypted message. Figure 3 shows the firmware flowchart.

Figure 3. Firmware Flow


START
Initialize the Crypto Server (CM0+) Enable CM4 core
Initialize and enable Crypto driver Start UART Component
Read the user input message
If input message > 16 characters
No
Encrypt the user input message using AES ECB method of encryption Display the encryption result, Key and decryption result on UART terminal
Yes
Prompt the user to enter a message of not more than 16 characters

Resources and Settings

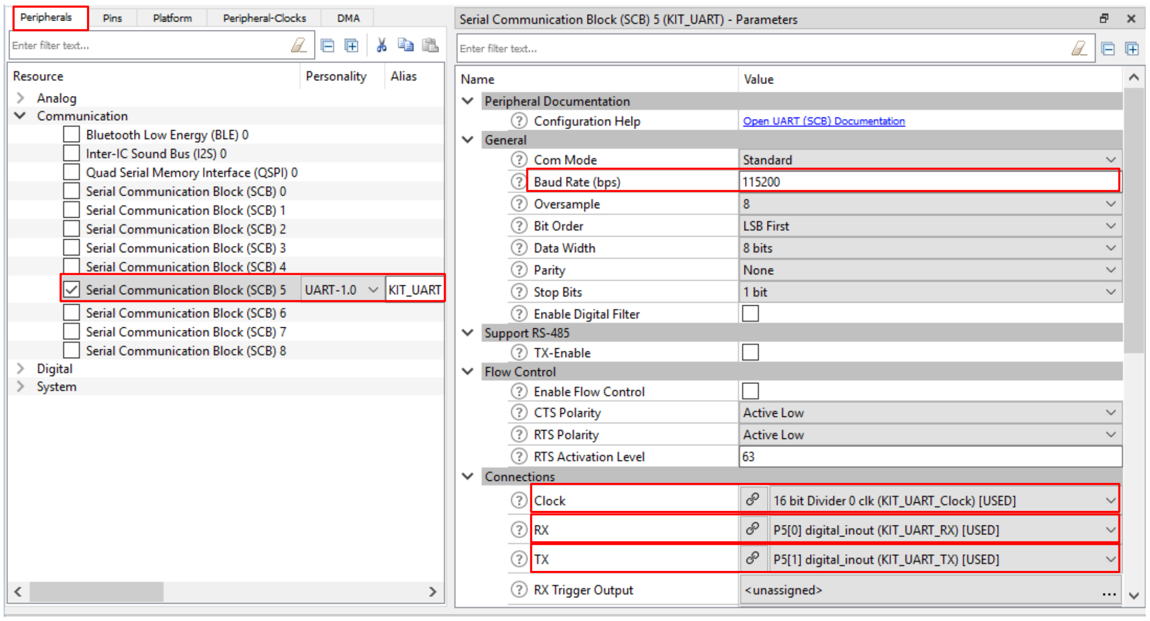
Table 1 lists the ModusToolbox resources used in this example, and how they are used in the design. For pin usage and configuration, open the Pins tab of the *design.modus* file.

Table 1. ModusToolbox Resources

Resource	Alias	Purpose
Serial Communication Block (SCB) – UART mode	KIT_UART	Send to and receive data from the UART Terminal.

Figure 4 highlights the non-default settings for the SCB (UART mode).

Figure 4. SCB (UART mode) Configuration



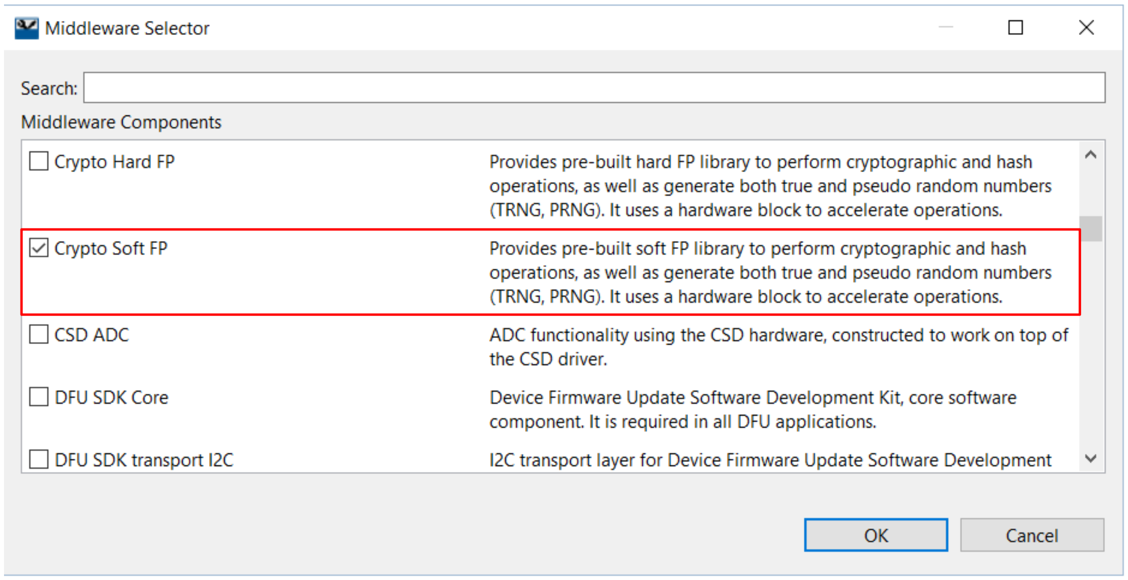
Resource	Personality	Alias
Analog		
Communication		
☐ Bluetooth Low Energy (BLE) 0		
☐ Inter-IC Sound Bus (I2S) 0		
☐ Quad Serial Memory Interface (QSPI) 0		
☐ Serial Communication Block (SCB) 0		
☐ Serial Communication Block (SCB) 1		
☐ Serial Communication Block (SCB) 2		
☐ Serial Communication Block (SCB) 3		
☐ Serial Communication Block (SCB) 4		
✓ Serial Communication Block (SCB) 5	UART-1.0	KIT_UART
☐ Serial Communication Block (SCB) 6		
☐ Serial Communication Block (SCB) 7		
☐ Serial Communication Block (SCB) 8		
Digital		
System		

Serial Communication Block (SCB) 5 (KIT_UART) - Parameters

Name	Value
Peripheral Documentation	
Configuration Help	Open UART (SCB) Documentation
General	
Com Mode	Standard
Baud Rate (bps)	115200
Oversample	8
Bit Order	LSB First
Data Width	8 bits
Parity	None
Stop Bits	1 bit
Enable Digital Filter	☐
Support RS-485	
TX-Enable	☐
Flow Control	
Enable Flow Control	☐
CTS Polarity	Active Low
RTS Polarity	Active Low
RTS Activation Level	63
Connections	
Clock	16 bit Divider 0 clk (KIT_UART_Clock) [USED]
RX	P5[0] digital_inout (KIT_UART_RX) [USED]
TX	P5[1] digital_inout (KIT_UART_TX) [USED]
RX Trigger Output	<unassigned>

To use the Crypto block of PSoC 6 MCU in your design, the Crypto driver must be enabled on each CPU (this example uses both the CPUs: Server on CM0+ and Client on CM4). To do this in your own project, right-click on the **<project name>_mainapp** and launch the **ModusToolbox Middleware Selector**. Enable the **Crypto Soft FP** from the Middleware Selector as shown in Figure 5. Repeat the steps for the CM0+ core to enable the Crypto driver for that core as well.

Figure 5 Enabling Crypto Middleware



Middleware Components

☐ Crypto Hard FP	Provides pre-built hard FP library to perform cryptographic and hash operations, as well as generate both true and pseudo random numbers (TRNG, PRNG). It uses a hardware block to accelerate operations.
✓ Crypto Soft FP	Provides pre-built soft FP library to perform cryptographic and hash operations, as well as generate both true and pseudo random numbers (TRNG, PRNG). It uses a hardware block to accelerate operations.
☐ CSD ADC	ADC functionality using the CSD hardware, constructed to work on top of the CSD driver.
☐ DFU SDK Core	Device Firmware Update Software Development Kit, core software component. It is required in all DFU applications.
☐ DFU SDK transport I2C	I2C transport layer for Device Firmware Update Software Development

Reusing This Example

This example is designed for the supported kits. To port the design to a different PSoC 6 MCU device, right click an application project and choose **Change Device**. If changing to a different kit, you may need to reassign pins.

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

Kit Name	Device Used	KIT_UART_RX	KIT_UART_TX
CY8CKIT-062-BLE	CY8C6347BZI-BLD53	P5[0]	P5[1]
CY8CKIT-062-WiFi-BT	CY8C6247BZI-D54	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, an IP block) 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 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.
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 63 with BLE Datasheet	PSoC 6 MCU: PSoC 63 with BLE Architecture Technical Reference Manual
Development Kits	
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: CE220465 – PSoC 6 MCU Cryptography: AES Demonstration

Document Number: 002-25573

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	6374033	VKVK	11/02/2018	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

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