

Customer training workshop: Cryptography_AES_Demonstration for KIT_T2G-B-H_EVK

TRAVEO™ T2G CYT4BF series Microcontroller Training
V1.0.0 2022-06



Please read the [Important Notice and Warnings](#) at the end of this document

Scope of work

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

- › Device
 - The TRAVEO™ T2G CYT4BFBCH device is used in this code example.

- › Board
 - The TRAVEO™ T2G KIT_T2G-B-H_EVK board is used for testing.

Introduction

› The cryptography block has following features

- Advanced Encryption Standard (AES) functionality according to FIPS 197:
The AES component can be used to encrypt/decrypt data blocks of 128-bit length and supports programmable key length (128/192/256-bit key).
- CHACHA20 functionality according to RFC 7539:
CHACHA20 is a stream cipher, which produces output consisting of 512-bit random-looking bits. These random bits can be XORed with plain text to produce cipher text.
- Triple Data Encryption Standard (TDES):
The TDES component can be used to encrypt/decrypt data blocks of 64-bit length using a 64-bit key.
- Secure Hash Algorithm (SHA) functionality according to FIPS 180-4/FIPS-202:
This component can be used to produce a fixed-length hash (also called “message digest”) of up to 512 bits from a variable length input data (called “message”). SHA1, SHA2, and SHA3 hashes are supported.
- Cyclic Redundancy Check (CRC) functionality:
This component performs a cyclic redundancy check with a programmable polynomial of up to 32 bits.
- String (STR) functionality:
This component can be used to efficiently copy, set, and compare memory data.

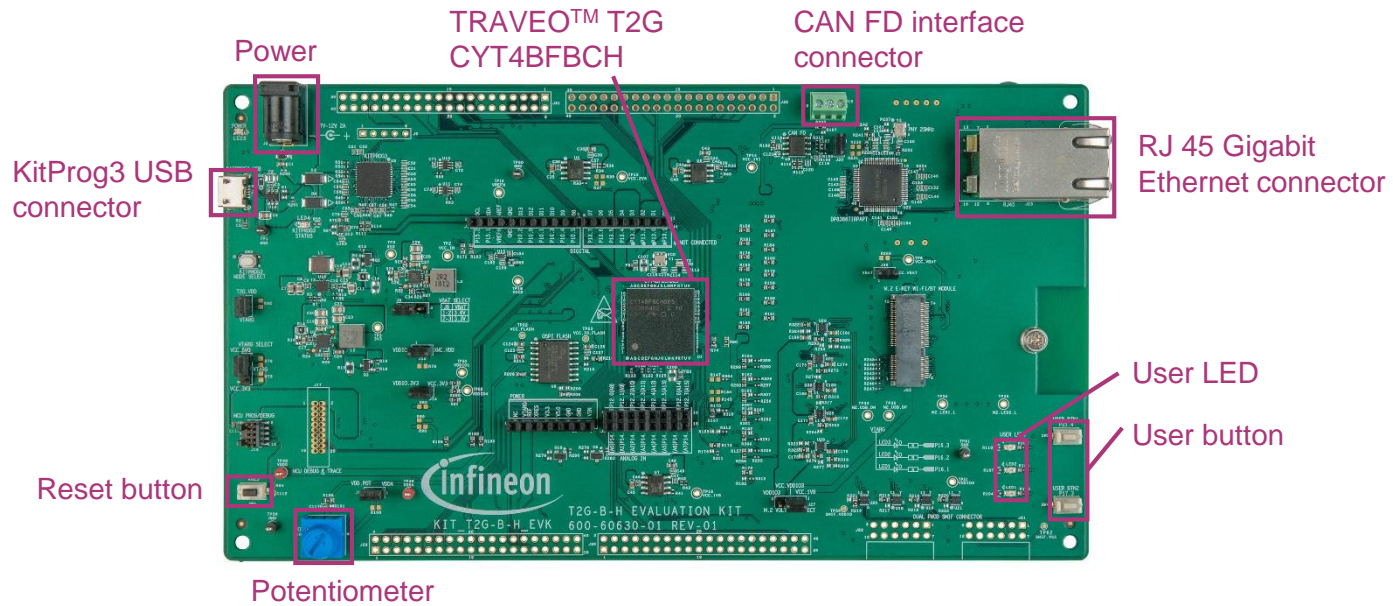
Introduction (contd.)

› **The cryptography block has the following features**

- Pseudo Random Number Generator (PRNG):
This component generates pseudo random numbers in a fixed range. This generator is based on three Linear Feedback Shift Registers (LFSRs).
- True Random Number Generator (TRNG):
This component generates true random numbers of up to 32 bits using ring oscillators.
- Vector Unit (VU):
This component acts as a coprocessor to offload asymmetric key operations from the main processor.
- AHB master-interface:
This allows to fetch operands directly from the system memory.
- Device key functionality:
The device key has its usage restricted to specific functionality; they cannot be accessed by the software that implements that functionality. Two independent device keys are supported.

Hardware setup

- › This code example has been developed for the KIT-T2G-B-H-EVK board.
- › Connect your PC to the board using the provided USB cable through the KitProg3 USB connector.



Implementation

- › AES is a symmetric block cipher data encryption algorithm, which means that it uses the same key for encryption and decryption of data. The AES operation works on the 128-bit block size and uses keys of 128 bits, 192 bits, or 256 bits of length. In this example, 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.
- › Press the reset button on the kit and enter the message to be encrypted.
Note that the maximum message size is restricted to 100 characters in this example.

Follow these steps to configure the code example:

- › STDIN / STDOUT setting
- › Display the initial message to terminal
- › Enable the cryptography block
- › Disable data cache
- › Get the message from terminal
- › Encrypt message
- › Decrypt message

Implementation (contd.)

STDIN / STDOUT setting

- › Initialization of the GPIO for UART is done using the [**cy_retarget_io_init\(\)**](#) function
 - Initialize P13.1 as UART TX, P13.0 as UART RX (these pins are connected to the KitProg3 COM port)
 - The serial port parameters change to 8N1 and 115200 baud

Display the initial message to terminal

- › The terminal can be displayed by **printf()**
 - Display data is specified **CLEAR_SCREEN** and **SCREEN_HEADER**

Enable the cryptography block

- › Enabling the cryptography block is done using the [**Cy Crypto Core Enable\(\)**](#) function
 - Enable crypto hardware

Disable data cache

- › Data cache in CM7 CPU is disabled by the **SCB_DisableDCache()**¹
 - This ensures that the cryptograph block can read the string on SRAM entered by user via STDIN

¹: The CPU cache operation instructions are provided by Cortex microcontroller software interface standard (CMSIS) with intrinsic functions.

Implementation (contd.)

Get the message from the terminal

- › Getting the character from the terminal is done using the [cyhal_uart_getc\(\)](#) function
 - This is a blocking call which waits till a character is received or till **UART_TIMEOUT_MS** has elapsed
- › UART RX activation is checked by the [cyhal_uart_is_rx_active\(\)](#) function
 - If RX is active, send the received character to the terminal by calling the [cyhal_uart_putc\(\)](#) function
 - This is a blocking call, which waits till the character is sent out from the UART completely.
 - When the received character is “Enter” key, the message input is completed.
 - If the message exceeds 100 characters, the message input is requested again.
- › If you want to increase the message size, you can update the **MAX_MESSAGE_SIZE** macro in the *main.c* file.

Implementation (contd.)

Encrypt message

- › The message encryption is done using the ***encrypt_message()*** function. This function runs the following:
 - Calculates the number of AES blocks from the message size using ***AES128_ENCRYPTION_LENGTH***
 - Initializes the AES operation using the ***Cy Crypto Core Aes Init()*** function
 - Using Key and Key length for initialization
 - Performs AES encryption using the ***Cy Crypto Core Aes Ecb()*** function
 - It performs the AES operation on a single block.
 - The ***Cy Crypto Core WaitForReady()*** function waits for completion of the cryptography operation
 - It waits until all instructions in FIFO are completed.
 - Sending a character to the terminal is done using the ***printf()*** and ***print_data()*** function
 - The key used for encryption and the encryption results are displayed on the terminal
 - You can change the ***aes_key*** in the *main.c* file to change the key used for encryption.
 - After all encryption, ***Cy Crypto Core Aes Free()*** clears the AES operation context

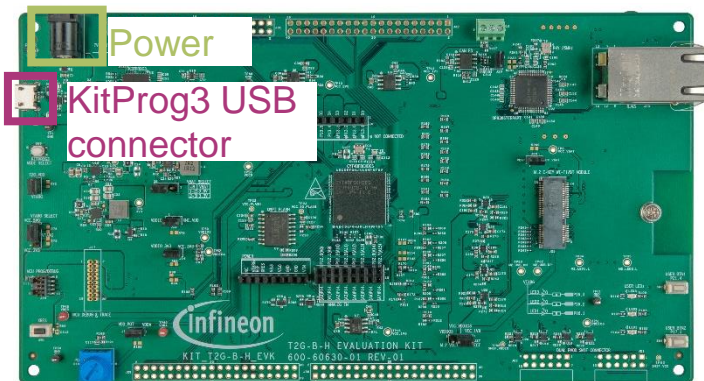
Implementation (contd.)


Decrypt message

- › The message decryption is done in the ***decrypt_message()*** function. This function runs the following operations:
 - Calculates the number of AES blocks from the message size using ***AES128_ENCRYPTION_LENGTH***
 - Initializes the AES operation using the ***Cy Crypto Core Aes Init()*** function
 - Using Key and Key length for initialization
 - Performs AES decryption using the ***Cy Crypto Core Aes Ecb()*** function
 - It performs the AES operation on a single block.
 - The ***Cy Crypto Core WaitForReady()*** function waits for the completion of the cryptography operation
 - It waits until all instructions in FIFO are completed.
 - Sending the character to the terminal is done using the ***printf()*** function
 - The decryption result is displayed on the terminal
 - After all the decryption, ***Cy Crypto Core Aes Free()*** clears the AES operation context

Compiling and programming

1. Connect to power and USB cable
2. Use Eclipse IDE for ModusToolbox™ software for compiling and programming
3. Compile
 - a) Select the target application project in the Project Explorer
 - b) In the Quick Panel, scroll down, and click “Build Cryptography_AES_Demonstration Application” in Cryptography_AES_Demonstration (KIT-T2G-B-H-EVK)
4. Open a terminal program and select the KitProg3 COM port. Set the serial port parameters to 8N1 and 115200 baud.
5. Programming
 - a) Select the target application project in the Project Explorer
 - b) In the Quick Panel, scroll down, and click “Cryptography_AES_Demonstration Program (KitProg3_MiniProg4)” under Launches



-  Build Cryptography_AES_Demonstration Application
-  Clean Cryptography_AES_Demonstration Application

- ▼ Launches
 -  Cryptography_AES_Demonstration Debug (JLink)
 -  Cryptography_AES_Demonstration Debug (KitProg3_MiniProg4)
 -  Cryptography_AES_Demonstration Program (JLink)
 -  Cryptography_AES_Demonstration Program (KitProg3_MiniProg4)
 -  Generate Launches for Cryptography_AES_Demonstration

Run and test

1. After successful programming, you can observe the following message:

```

COM5 - Tera Term VT
File Edit Setup Control Window Help
-----
*                CE220465 PDL Cryptography: AES Demonstration
*
*   This code example demonstrates encryption and decryption of data using
*   the Advanced Encryption Scheme (AES) algorithm in MCU.
*
*   UART Terminal Settings: Baud Rate- 115200 bps, 8N1
*
*
-----
Enter the message:
  
```

2. When you input the message for encryption, the “Enter” key is pressed, and displays the key used for encryption, encryption result, and decryption result on the terminal.

```

Enter the message:
T2G-B-H MCU

Key used for Encryption:
0xAA 0xBB 0xCC 0xDD 0xEE 0xFF 0xFF 0xEE 0xDD 0xCC 0xBB 0xAA 0xAA 0xBB 0xCC 0xDD

Result of Encryption:
0x82 0xF1 0x2A 0x81 0x37 0x44 0x11 0x57 0xBE 0x4F 0x26 0xCA 0xDC 0xCA 0x2F 0xF0

Result of Decryption:
T2G-B-H MCU

-----
Enter the message:
  
```

References

Datasheet

- › [CYT4BF datasheet 32-bit Arm® Cortex® -M7 microcontroller TRAVEO™ T2G family](#)

Architecture technical reference manual

- › [TRAVEO™ T2G automotive body controller high family architecture technical reference manual](#)

Registers technical reference manual

- › [TRAVEO™ T2G Automotive body controller high registers technical reference manual](#)

PDL/HAL

- › [PDL](#)

- › [HAL](#)

Training

- › [TRAVEO™ T2G Training](#)

Revision History

Revision	ECN	Submission Date	Description of Change
**	7782101	2022/07/05	Initial release

Important notice and warnings

All referenced product or service names and trademarks are the property of their respective owners.

Edition 2021-06
Published by
Infineon Technologies AG
81726 Munich, Germany

© 2021 Infineon Technologies AG.
All Rights Reserved.

Do you have a question about this document?

Go to:
www.infineon.com/support

Document reference
002-35563 Rev. **

IMPORTANT NOTICE

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics (“Beschaffenhheitsgarantie”).

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer's compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer's products and any use of the product of Infineon Technologies in customer's applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (www.infineon.com).

WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.