軟體實作 AES算法

NuMicro® 32 位系列微控制器範例代碼介紹

文件資訊

代碼簡述	用軟體實作 Advanced Encryption Standard 算法
BSP 版本	NUC230/240 Series BSP v3.01.002
開發平台	NuTiny–EVB–NUC240–LQFP100 V1.0

The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.

Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.

All data and specifications are subject to change without notice.

For additional information or questions, please contact: Nuvoton Technology Corporation.

www.nuvoton.com

1 功能介紹

1.1 簡介

此示例代碼演示如何實現軟件AES（高級加密標準）以及AES加密/解密的工作原理。

1.2 原理

AES的是在一個4x4 Byte的矩陣上運作的算法，主要包含了四個步驟。第一個步驟是將資料與金鑰做XOR運算，定義為Add round key。第二步驟則是透過 S-Box(置換函數)將資料的每一個Byte做轉換，定義為Substitutes bytes。第三個步驟則對於矩陣的每一橫列進行移位的動作，定義為Shift rows。第四個步驟則是對於矩陣的每一個直行與一個多項式進行多項式乘法，定義為Mix columns。AES的加密和解密，都需要先將金鑰做展開的動作，此展開的動作稱為Key Expansion；再用展開的金鑰來執行加解密的算法，亦即上述之四個步驟。

AES加密的流程，如圖1-1所示。假設加密流程有N個Round，首先，先對金鑰做展開的動作。金鑰展開後，在第一個Round執行Add round key；而後從第二個Round至第(N-1)個Round，分別依序執行Substitutes bytes、Shift rows、 Mix columns以及Add round key。最後一個Round則依序執行Substitutes bytes、Shift rows以及Add round key，即完成AES加密流程。

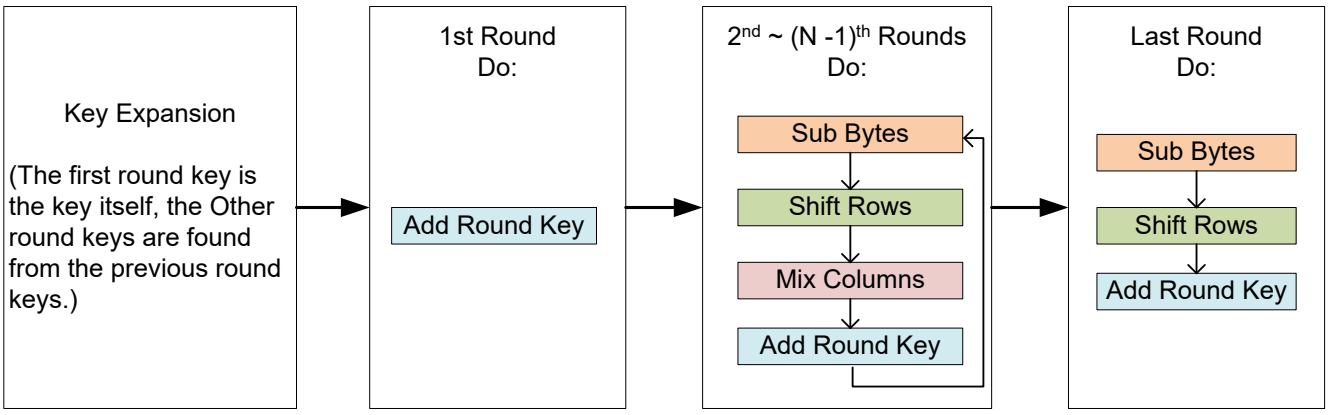

圖 1-1 AES 加密流程

AES解密的流程，如圖1-2所示。相對於加密的Substitutes bytes、Shift rows、 Mix columns算法，解密另外分別實作了Invers substitutes bytes、Inverse shift rows、Inverse mix columns的算法，以供解密流程使用。而解密的流程，的假設解密流程有N個Round。首先，先對金鑰做

展開的動作。金鑰展開後，在第一個Round執行Add round key；而後從第(N-1)個Round至第二個Round，分別依序執行Inverse shift rows、Inverse substitutes bytes、Add round key以及Inverse mix columns。最後一個round則依序執行Inverse shift rows、Inverse substitutes bytes以及Add round key，即完成AES解密流程。

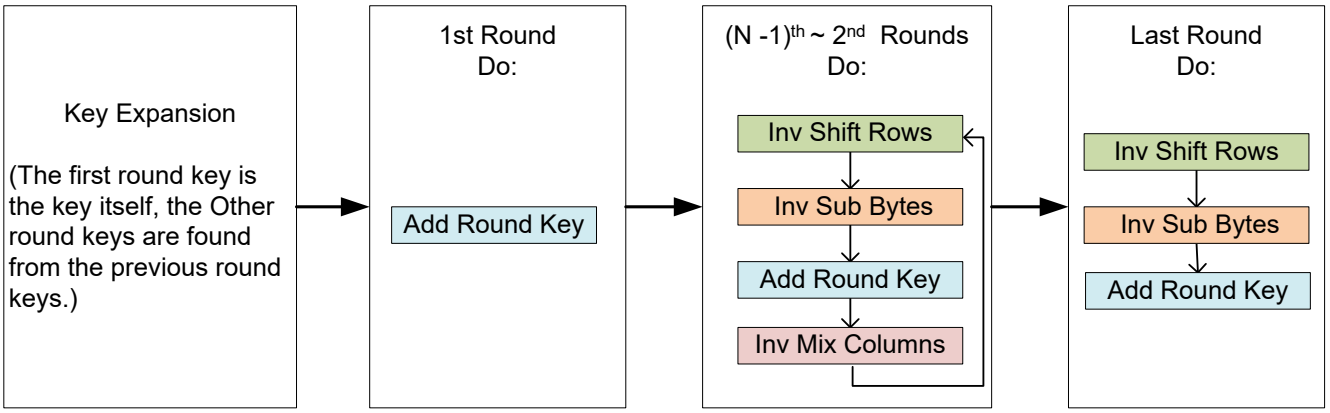

圖 1-2 AES 解密流程

1.3 執行結果

執行結果可以在UART上輸出，鮑率115200，如圖1-3所示：

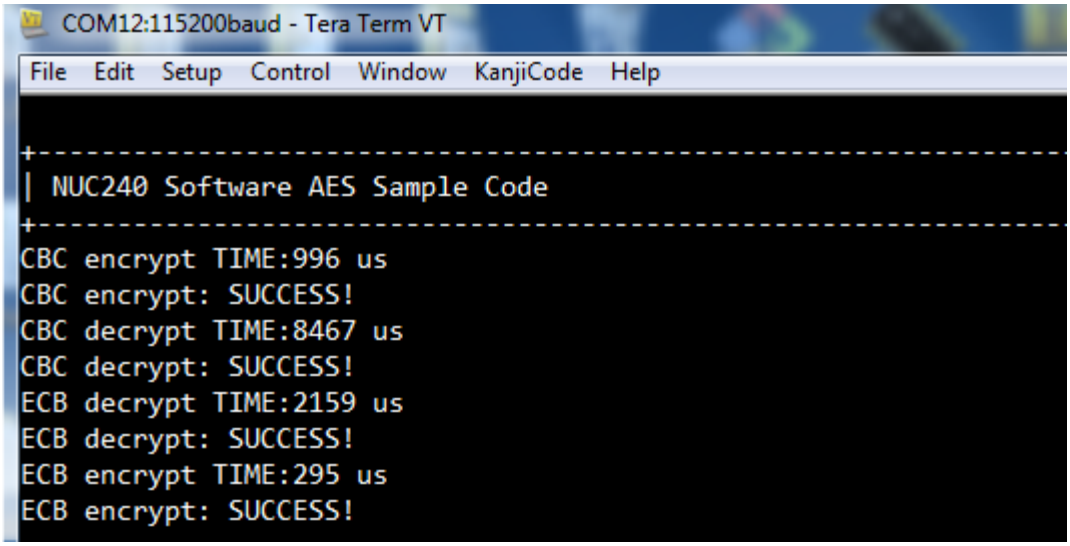

圖 1-3 UART 上打印執行結果

2 代碼介紹

下方的函式為金鑰展開的流程：

```
/* This function produces NB(Nr+1) round keys. The round keys are used in each round to
decrypt the states. */
static void KeyExpansion(void)
{
    uint32_t i, j, k;
    uint8_t au8Tempa[4]; /* Used for the column/row operations */

    /* The first round key is the key itself. */
    for (i = 0; i < NK; ++i)
    {
        s_au8RoundKey[(i * 4) + 0] = s_pu8Key[(i * 4) + 0];
        s_au8RoundKey[(i * 4) + 1] = s_pu8Key[(i * 4) + 1];
        s_au8RoundKey[(i * 4) + 2] = s_pu8Key[(i * 4) + 2];
        s_au8RoundKey[(i * 4) + 3] = s_pu8Key[(i * 4) + 3];
    }

    /* All other round keys are found from the previous round keys. */
    for (; (i < (NB * (NR + 1))); ++i)
    {
        for (j = 0; j < 4; ++j)
        {
            au8Tempa[j] = s_au8RoundKey[(i - 1) * 4 + j];
        }
        if (i % NK == 0)
        {
            /* This function rotates the 4 bytes in a word to the left once. */
            /* [a0,a1,a2,a3] becomes [a1,a2,a3,a0]                            */

            /* Function RotWord() */
            {
                k = au8Tempa[0];
                au8Tempa[0] = au8Tempa[1];
                au8Tempa[1] = au8Tempa[2];
                au8Tempa[2] = au8Tempa[3];
                au8Tempa[3] = k;
            }
```

```
            /* SubWord() is a function that takes a four-byte input word and          */
            /* applies the S-box to each of the four bytes to produce an output word. */

            /* Function Subword()                                                     */
            {
                au8Tempa[0] = GetSBoxValue(au8Tempa[0]);
                au8Tempa[1] = GetSBoxValue(au8Tempa[1]);
                au8Tempa[2] = GetSBoxValue(au8Tempa[2]);
                au8Tempa[3] = GetSBoxValue(au8Tempa[3]);
            }

            au8Tempa[0] = au8Tempa[0] ^ s_au8Rcon[i / NK];
        }
        else if (NK > 6 && i % NK == 4)
        {
            /* Function Subword() */
            {
                au8Tempa[0] = GetSBoxValue(au8Tempa[0]);
                au8Tempa[1] = GetSBoxValue(au8Tempa[1]);
                au8Tempa[2] = GetSBoxValue(au8Tempa[2]);
                au8Tempa[3] = GetSBoxValue(au8Tempa[3]);
            }
        }
        s_au8RoundKey[i * 4 + 0] = s_au8RoundKey[(i - NK) * 4 + 0] ^ au8Tempa[0];
        s_au8RoundKey[i * 4 + 1] = s_au8RoundKey[(i - NK) * 4 + 1] ^ au8Tempa[1];
        s_au8RoundKey[i * 4 + 2] = s_au8RoundKey[(i - NK) * 4 + 2] ^ au8Tempa[2];
        s_au8RoundKey[i * 4 + 3] = s_au8RoundKey[(i - NK) * 4 + 3] ^ au8Tempa[3];
    }
}
```

下方的函式為 Add round key 的流程：

```
/* This function adds the round key to state.                 */
/* The round key is added to the state by an XOR function.    */
static void AddRoundKey(uint8_t round)
{
    uint8_t i, j;
    for (i = 0; i<4; ++i)
    {
        for (j = 0; j < 4; ++j)
        {
            (*s_pu8State)[i][j] ^= s_au8RoundKey[round * NB * 4 + i * NB + j];
```

```
        }
    }
}
```

下方的函式為加解密的 Substitute bytes 流程：

```
/* The SubBytes Function Substitutes the values in the */
/* state matrix with values in an S-box.               */
static void SubBytes(void)
{
    uint8_t i, j;
    for (i = 0; i < 4; ++i)
    {
        for (j = 0; j < 4; ++j)
        {
            (*s_pu8State)[j][i] = GetSBoxValue((*s_pu8State)[j][i]);
        }
    }
}
/* The SubBytes Function Substitutes the values in the   */
/* state matrix with values in an S-box.                 */
static void InvSubBytes(void)
{
    uint8_t i, j;
    for (i = 0; i<4; ++i)
    {
        for (j = 0; j<4; ++j)
        {
            (*s_pu8State)[j][i] = GetSBoxInvert((*s_pu8State)[j][i]);
        }
    }
}
```

下方的函式為加解密的Shift rows流程：

```
/* The ShiftRows() function shifts the rows in the state to the left.   */
/* Each row is shifted with different offset.                           */
/* Offset = Row number. So the first row is not shifted.                */
static void ShiftRows(void)
{
    uint8_t u8Temp;

    /* Rotate first row 1 columns to left */
    u8Temp = (*s_pu8State)[0][1];
```

```
    (*s_pu8State)[0][1] = (*s_pu8State)[1][1];
    (*s_pu8State)[1][1] = (*s_pu8State)[2][1];
    (*s_pu8State)[2][1] = (*s_pu8State)[3][1];
    (*s_pu8State)[3][1] = u8Temp;

    /* Rotate second row 2 columns to left */
    u8Temp = (*s_pu8State)[0][2];
    (*s_pu8State)[0][2] = (*s_pu8State)[2][2];
    (*s_pu8State)[2][2] = u8Temp;

    u8Temp = (*s_pu8State)[1][2];
    (*s_pu8State)[1][2] = (*s_pu8State)[3][2];
    (*s_pu8State)[3][2] = u8Temp;

    /* Rotate third row 3 columns to left */
    u8Temp = (*s_pu8State)[0][3];
    (*s_pu8State)[0][3] = (*s_pu8State)[3][3];
    (*s_pu8State)[3][3] = (*s_pu8State)[2][3];
    (*s_pu8State)[2][3] = (*s_pu8State)[1][3];
    (*s_pu8State)[1][3] = u8Temp;
}

static void InvShiftRows(void)
{
    uint8_t u8Temp;

    /* Rotate first row 1 columns to right */
    u8Temp = (*s_pu8State)[3][1];
    (*s_pu8State)[3][1] = (*s_pu8State)[2][1];
    (*s_pu8State)[2][1] = (*s_pu8State)[1][1];
    (*s_pu8State)[1][1] = (*s_pu8State)[0][1];
    (*s_pu8State)[0][1] = u8Temp;

    /*  Rotate second row 2 columns to right */
    u8Temp = (*s_pu8State)[0][2];
    (*s_pu8State)[0][2] = (*s_pu8State)[2][2];
    (*s_pu8State)[2][2] = u8Temp;

    u8Temp = (*s_pu8State)[1][2];
    (*s_pu8State)[1][2] = (*s_pu8State)[3][2];
    (*s_pu8State)[3][2] = u8Temp;
```

```
    /* Rotate third row 3 columns to right */
    u8Temp = (*s_pu8State)[0][3];
    (*s_pu8State)[0][3] = (*s_pu8State)[1][3];
    (*s_pu8State)[1][3] = (*s_pu8State)[2][3];
    (*s_pu8State)[2][3] = (*s_pu8State)[3][3];
    (*s_pu8State)[3][3] = u8Temp;
}
```

下方的函式為加解密的Mix columns的流程：

```
/* MixColumns function mixes the columns of the state matrix */
static void MixColumns(void)
{
    uint8_t i;
    uint8_t u8Tmp, u8Tm, u8t;
    for (i = 0; i < 4; ++i)
    {
        u8t = (*s_pu8State)[i][0];
        u8Tmp = (*s_pu8State)[i][0] ^ (*s_pu8State)[i][1] ^ (*s_pu8State)[i][2] ^
(*s_pu8State)[i][3];
        u8Tm = (*s_pu8State)[i][0] ^ (*s_pu8State)[i][1];
        u8Tm = Xtime(u8Tm);
        (*s_pu8State)[i][0] ^= u8Tm ^ u8Tmp;
        u8Tm = (*s_pu8State)[i][1] ^ (*s_pu8State)[i][2];
        u8Tm = Xtime(u8Tm);
        (*s_pu8State)[i][1] ^= u8Tm ^ u8Tmp;
        u8Tm = (*s_pu8State)[i][2] ^ (*s_pu8State)[i][3];
        u8Tm = Xtime(u8Tm);
        (*s_pu8State)[i][2] ^= u8Tm ^ u8Tmp;
        u8Tm = (*s_pu8State)[i][3] ^ u8t;
        u8Tm = Xtime(u8Tm);
        (*s_pu8State)[i][3] ^= u8Tm ^ u8Tmp;
    }
}
/* MixColumns function mixes the columns of the state matrix.                    */
/* The method used to multiply may be difficult to understand for the inexperienced. */
/* Please use the references to gain more information.                           */
static void InvMixColumns(void)
{
    int i;
    uint8_t u8a, u8b, u8c, u8d;
    for (i = 0; i<4; ++i)
```

```
    {
        u8a = (*s_pu8State)[i][0];
        u8b = (*s_pu8State)[i][1];
        u8c = (*s_pu8State)[i][2];
        u8d = (*s_pu8State)[i][3];

        (*s_pu8State)[i][0] = Multiply(u8a, 0x0e) ^ Multiply(u8b, 0x0b) ^ Multiply(u8c,
0x0d) ^ Multiply(u8d, 0x09);
        (*s_pu8State)[i][1] = Multiply(u8a, 0x09) ^ Multiply(u8b, 0x0e) ^ Multiply(u8c,
0x0b) ^ Multiply(u8d, 0x0d);
        (*s_pu8State)[i][2] = Multiply(u8a, 0x0d) ^ Multiply(u8b, 0x09) ^ Multiply(u8c,
0x0e) ^ Multiply(u8d, 0x0b);
        (*s_pu8State)[i][3] = Multiply(u8a, 0x0b) ^ Multiply(u8b, 0x0d) ^ Multiply(u8c,
0x09) ^ Multiply(u8d, 0x0e);
    }
}
```

下方的函式為AES加密的流程：

```
/* Cipher is the main function that encrypts the PlainText. */
static void Cipher(void)
{
    uint8_t u8Round = 0;

    /* Add the First round key to the state before starting the rounds. */
    AddRoundKey(0);

    /* There will be Nr rounds.                          */
    /* The first Nr-1 rounds are identical.              */
    /* These Nr-1 rounds are executed in the loop below.  */
    for (u8Round = 1; u8Round < NR; ++u8Round)
    {
        SubBytes();
        ShiftRows();
        MixColumns();
        AddRoundKey(u8Round);
    }

    /* The last round is given below.                        */
    /* The MixColumns function is not here in the last round. */
    SubBytes();
    ShiftRows();
    AddRoundKey(NR);
}
```

下方的函式為AES解密的流程：

```
static void InvCipher(void)
{
    uint8_t u8Round = 0;

    /* Add the First round key to the state before starting the rounds. */
    AddRoundKey(NR);

    /* There will be Nr rounds.                          */
    /* The first Nr-1 rounds are identical.              */
    /* These Nr-1 rounds are executed in the loop below. */
    for (u8Round = NR - 1; u8Round>0; u8Round--)
    {
        InvShiftRows();
        InvSubBytes();
        AddRoundKey(u8Round);
        InvMixColumns();
    }

    /* The last round is given below.                        */
    /* The MixColumns function is not here in the last round. */
    InvShiftRows();
    InvSubBytes();
    AddRoundKey(0);
}
```

3 軟體與硬體環境

- **軟體環境**
  - BSP 版本
    - NUC230/240 Series BSP CMSIS v3.01.002
  - IDE 版本
    - Keil uVersion 5.28
- **硬體環境**
  - 電路元件
    - NuTiny–EVB–NUC240–LQFP100 V1.0
  - 示意圖
    - 將 UART TX（PB.1）pin 腳連接到 PC UART RX，以顯示示例代碼的執行結果於 PC 上。

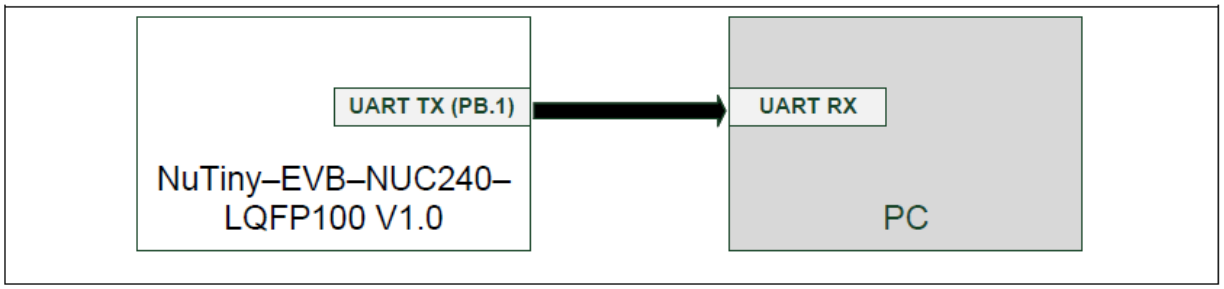

圖 3-1 NUC240 UART TX 和 PC UART RX 連接

4 目錄資訊

- EC_NUC240_Software_AES_V1.00
  - Library — Sample code header and source files
    - CMSIS — Cortex® Microcontroller Software Interface Standard (CMSIS) by Arm® Corp.
    - Device — CMSIS compliant device header file
    - StdDriver — All peripheral driver header and source files
  - SampleCode
    - ExampleCode — Source file of example code

5 如何執行範例程式

1. 根據目錄資訊章節進入 ExampleCode 路徑中的 KEIL 資料夾，雙擊 Software_AES.uvproj。
2. 進入編譯模式介面
   a. 編譯
   b. 下載代碼至記憶體
   c. 進入 / 離開除錯模式
3. 進入除錯模式介面
   a. 執行代碼

6 修訂紀錄

Date	Revision	Description
Sep. 9, 2019	1.00	1. 初始發布.

Important Notice

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

Please note that all data and specifications are subject to change without notice.
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.