软体实作 AES

NuMicro® 32 位系列微控制器范例代码介绍

文件信息

代码简述	用软体实作 Advanced Encryption Standard 算法
BSP 版本	NUC230/240 Series BSP v3.01.002
开发平台	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 功能介绍

1.1 简介

此示例代码演示如何实现软件AES（高级加密标准）以及AES加密/解密的工作原理。

1.2 原理

AES的是在一个4x4 Byte的矩阵上运作的算法，主要包含了四个步骤。第一个步骤是将资料与金钥做XOR运算，定义为Add round key。第二步骤则是透过 S-Box(置换函数)将资料的每一个Byte做转换，定义为Substitutes bytes。第三个步骤则对于矩阵的每一横列进行移位的动作，定义为Shift rows。第四个步骤则是对于矩阵的每一个直行与一个多项式进行多项式乘法，定义为Mix columns。 AES的加密和解密，都需要先将金钥做展开的动作，此展开的动作称为Key Expansion；再用展开的金钥来执行加解密的算法，亦即上述之四个步骤。

AES加密的流程，如图1-1所示。假设加密流程有N个Round，首先，先对金钥做展开的动作。金钥展开后，在第一个Round执行Add round key；而后从第二个Round至第(N-1)个Round，分别依序执行Substitutes bytes、Shift rows、 Mix columns以及Add round key。最后一个Round则依序执行Substitutes bytes、Shift rows以及Add round key，即完成AES加密流程。

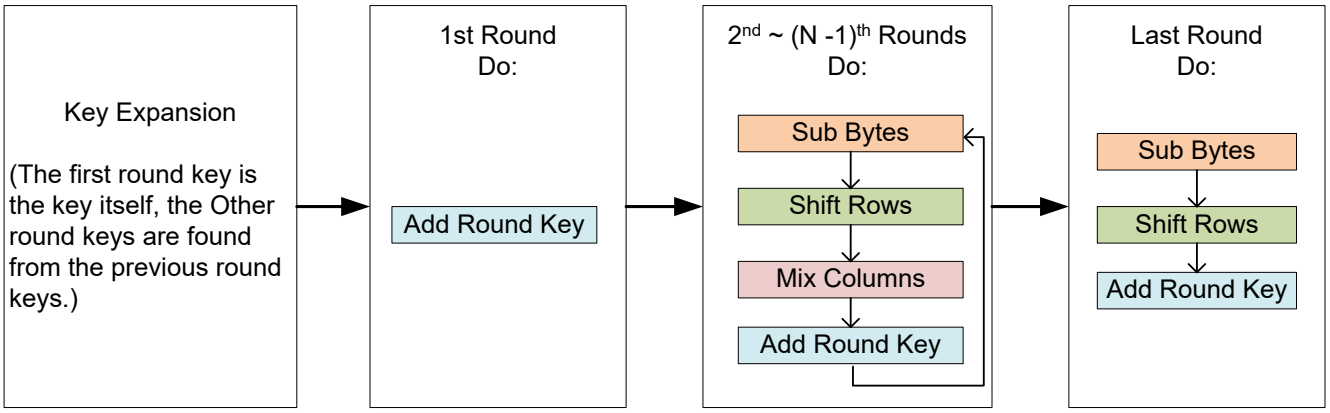

图 1-1 AES 加密流程

AES解密的流程，如图1-2所示。相对于加密的Substitutes bytes、Shift rows、 Mix columns算法，解密另外分别实作了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，的假设解密流程有N个Round。首先，先对金钥做展开的动作。金钥展开后，在第一个Round执行Add round key；而后从第(N-1)个Round至第

二个Round，分别依序执行Inverse shift rows、Inverse substitutes bytes、Add round key以及Inverse mix columns 。最后一个round则依序执行Inverse shift rows、Inverse substitutes bytes 以及Add round key，即完成AES解密流程。

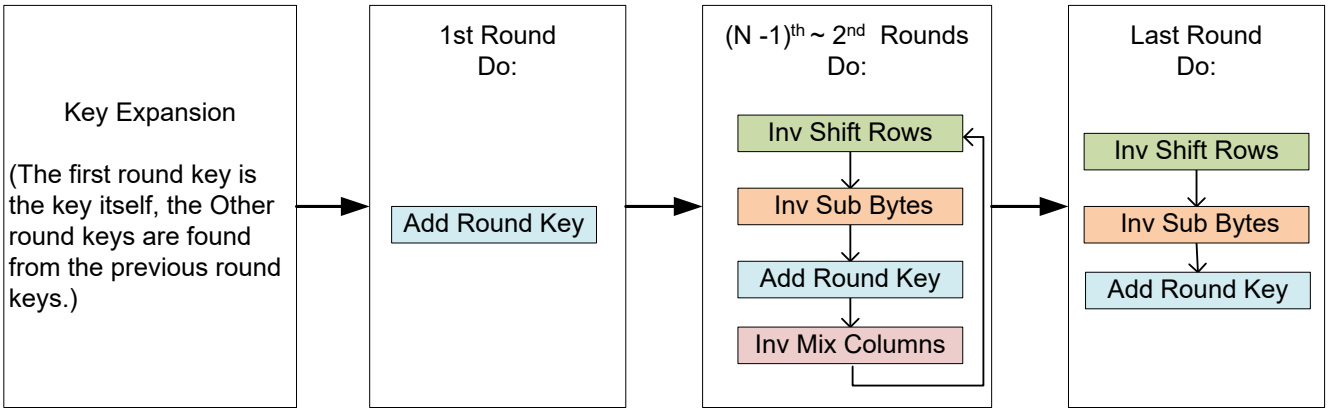

图 1-2 AES解密流程

1.3 执行结果

执行结果可以在 UART 上输出，鲍率 115200，如图 1-2 所示：

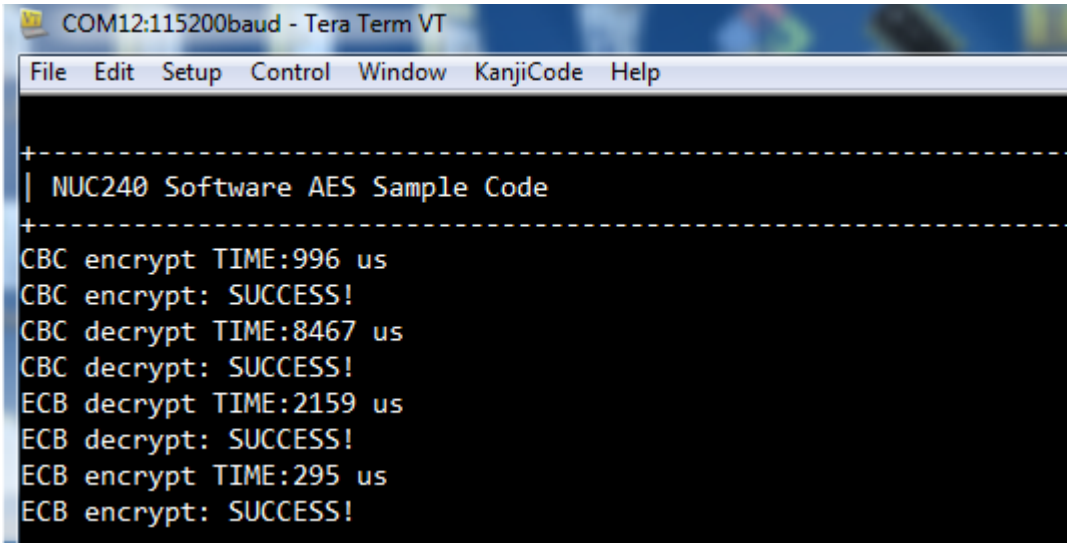

图 1-3 UART 上打印执行结果

2 代码介绍

下方的函式为金钥展开的流程：

```
/* 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];
    }
}
```

下方的函式为 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];
```

```
        }
    }
}
```

下方的函式为加解密的 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]);
        }
    }
}
```

下方的函式为加解密的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;
}
```

下方的函式为加解密的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);
    }
}
```

下方的函式为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);
}
```

下方的函式为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 软件与硬件环境

- **软件环境**
  - BSP 版本
    - NUC230/240 Series BSP CMSIS v3.01.002
  - IDE 版本
    - Keil uVersion 5.28
- **硬件环境**
  - 电路组件
    - NuTiny–EVB–NUC240–LQFP100 V1.0
  - 示意图
    - 将 UART TX（PB.1）pin 脚连接到 PC UART RX，以显示示例代码的执行结果于 PC 上。

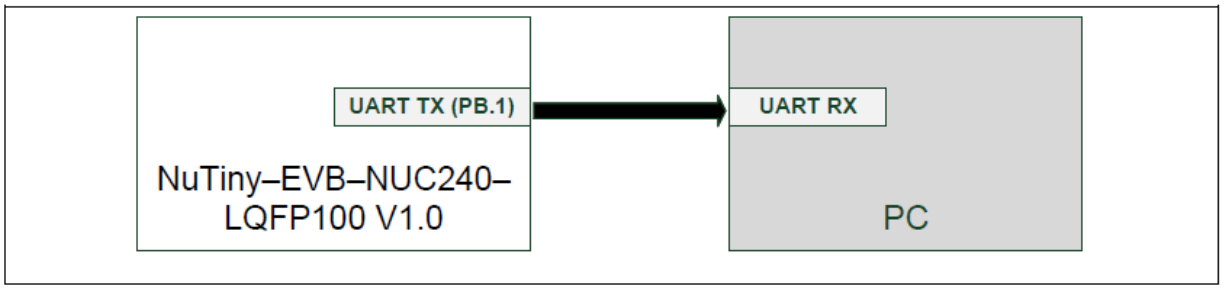

图 3-1 NUC240 UART TX 和 PC UART RX 连接

4 目录信息

- 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 如何执行范例程序

1. 根据目录信息章节进入 ExampleCode 路径中的 KEIL 文件夹，双击 Software_AES.uvproj。
2. 进入编译模式接口
   a. 编译
   b. 下载代码至内存
   c. 进入 / 离开除错模式
3. 进入除错模式接口
   a. 执行代码

6 修订纪录

Date	Revision	Description
Sep. 9, 2019	1.00	1. 初始发布.

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.