

SYD8811 AES 硬件加密

在硬件加密这块 SYD8811 完全遵循 SYD8801 的操作方式和流程，这里请参考 SYD8801 的 AES 硬件加密文章：<https://blog.csdn.net/chengdong1314/article/details/77692902>
该文章摘录如下：

SYD8801 硬件 AES 加密

aes 函数表示如下：encryptedData=e(key,plaintextData)

SYD8801 对应的 API：

```
void smp_aes_encrypt(uint8_t *k, uint8_t* p, uint8_t* c);
```

参数 k: aes 算法中的 key, 也就是密码

p: aes 算法中的 plaintextData, 也就是明文

c: aes 算法中的 encryptedData, 也就是密文

SYD8801 硬件 AES 加密应用层函数调用方式如下：

```
if(get_key()==1){  
    //PowerDown();  
    smp_aes_encrypt(kEY,plaintext,encrypted);  
    dbg_hexdump("aes key:\r\n",kEY,16);  
    dbg_hexdump("plaintext:\r\n",plaintext,16);  
    dbg_hexdump("encrypted:\r\n",encrypted,16);  
}
```

这里按下按键 1 就会调用 aes 加密函数进行加密。

其中的变量的定义如下：

```
1 int main()  
2 {  
3     uint8_t plaintext[]={0x00, 0xff, 0xee, 0xdd, 0xcc, 0xbb, 0xaa, 0x99, 0x88, 0x77, 0x66, 0x55, 0x44, 0x33, 0x22, 0x11};  
4     uint8_t encrypted[]={0x13, 0x02, 0xf1, 0xe0, 0xdf, 0xce, 0xbd, 0xac, 0x79, 0x68, 0x57, 0x46, 0x35, 0x24, 0x13, 0x02};  
5     uint8_t kEY[]={0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};  
}
```

上面只是一个例子，这个例子来自于规范 4.0 提及到的例子，这里的计算结果如下：

```
SYD Inc.  
aes key:  
[0000] 00 00 00 00 00 00 00 00  
[0008] 00 00 00 00 00 00 00 00  
plaintext:  
[0000] 00 ff ee dd cc bb aa 99  
[0008] 88 77 66 55 44 33 22 11  
encrypted:  
[0000] 62 a0 6d 79 ae 16 42 5b  
[0008] 9b f4 b0 e8 f0 e1 1f 9a
```

规范 4.0 中提及到的例子:

The least significant octet of $r2'$ becomes the least significant octet of r' and the most significant octet of $r1'$ becomes the most significant octet of r' .

For example, if the 64-bit value $r1'$ is 0x1122334455667788 and $r2'$ is 0x99AABBCCDDEEFF00 then r' is 0x112233445566778899AABBCCDDEEFF00.

The output of the key generation function $s1$ is:

$$s1(k, r1, r2) = e(k, r')$$

The 128-bit output of the security function e is used as the result of key generation function $s1$.

For example if the 128-bit value k is

0x00000000000000000000000000000000

and the 128-bit value r' is

0x112233445566778899AABBCCDDEEFF00

then the output from the key generation function $s1$ is

0x9a1fe1f0e8b0f49b5b4216ae796da062.

1963 / 2302

这里和 SYD8801 计算出来的结果是一样的, aes 加密正确!

注意: 因为 SYD8801 只有一套 AES 硬件, 所以当底层正在进行蓝牙配对流程后应用层不能够再使用这个硬件加密, 这里说的是正在进行配对的时候要有这个提示, 也就是说当配对完成后或者在只连接不配对以及广播的时候这个模块都是可以使用的!

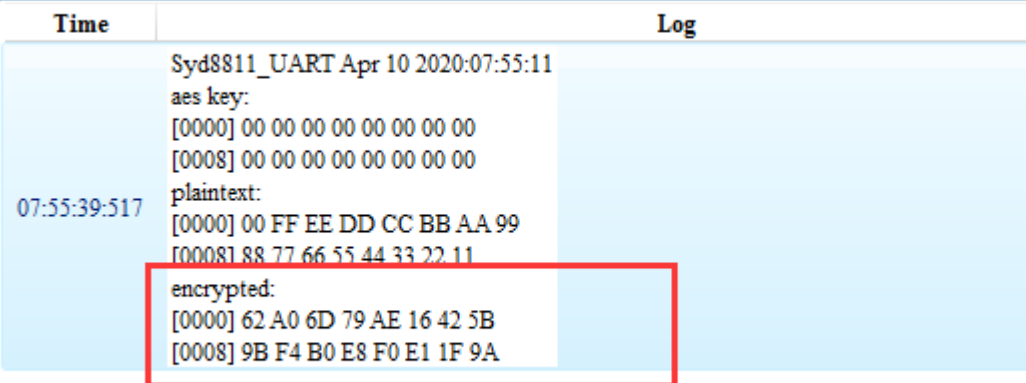
[这里上传上本博客使用到的工程:](#)

<http://download.csdn.net/download/chengdong1314/9956449>

对于 SYD8811:例程在 “Source Code\SYD8811_peripheral_misc\SYD8811_BLE_AES” 关键代码如下，打印信息如下：

```
__enable_irq();
{
    uint8_t plaintext[]={0x00, 0xff, 0xee, 0xdd, 0xcc, 0xbb, 0xaa, 0x99, 0x88, 0x77, 0x66, 0x55, 0x44, 0x33, 0x22, 0x11};
    uint8_t encrypted[]={0x13, 0x02, 0xf1, 0xe0, 0xdf, 0xce, 0xbd, 0xac, 0x79, 0x68, 0x57, 0x46, 0x35, 0x24, 0x13, 0x02};
    uint8_t key[]={0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00};
    smp_aes_encrypt(key, plaintext, encrypted);
    DBGHEXDUMP("aes key:\r\n", key, 16);
    DBGHEXDUMP("plaintext:\r\n", plaintext, 16);
    DBGHEXDUMP("encrypted:\r\n", encrypted, 16);
}

while(1)
```



Time	Log
07:55:39:517	Syd8811_UART Apr 10 2020:07:55:11 aes key: [0000] 00 00 00 00 00 00 00 00 [0008] 00 00 00 00 00 00 00 00 plaintext: [0000] 00 FF EE DD CC BB AA 99 [0008] 88 77 66 55 44 33 22 11 encrypted: [0000] 62 A0 6D 79 AE 16 42 5B [0008] 9B F4 B0 E8 F0 E1 1F 9A
07:58:37:515	RTCEVT_185S
08:01:42:106	RTCEVT_185S

从上面看出加密功能是正常的！

注意：这个工程只有在 syd8811_ble_lib20200410_075415.lib 之后的 lib 才有用