
SOC烧录器

滚码烧录原理及应用说明

使用手册

深圳市集贤科技有限公司

电话：0755-82571152 传真：0755-88373753

网址：<http://www.uascent.com>

地址：深圳市南山区科技园北区朗山路 11 号同方信息港 A 栋 4 楼

版本记录:

Version	Date	Author (s)	Description
0.1	2017.10.30	beiley	Initial version
0.2	2018.10.16	beiley	Initial version



用户在使用我司烧录器的时候有时候会用到滚码烧录，但又对这个原理不是很理解，结果可能会造成在开发板上可以正常通信，结果用我们 soc 烧录器烧录后却没有功能。接下来以 bk2461 来重点讲解下滚码烧录的原理及应用。

原理：滚码烧录是将 5 个 byte（或者 6 个 byte）的值烧录到 flash 或者 otp 芯片对应的存储位置，目前我们不同的芯片型号对应的烧录位置都不一样，大概给大家列一些常用芯片的滚码地址：

bk2461 的滚码烧录地址为 “0x1ff3-0x1ff7”

bk2535 的滚码地址为 “0x7ff7-0x7ffB”

bk3431s 的滚码地址为 “0x7ff7-0x7ffB”



这些对应的地址就是实际存储 5 个 byte（或者 6 个 byte）的地址。怎么调用呢？当然是在程序里将对应地址的值读出来即可。这里以

bk2461 举例：

```
//ID
#define ID0 CBYTE[0x1FF3]
#define ID1 CBYTE[0x1FF3+1]
#define ID2 CBYTE[0x1FF3+2]
#define ID3 CBYTE[0x1FF3+3]
#define ID4 CBYTE[0x1FF3+4]
```

定义地址对应的地址：

```
rf_id[0]=ID0;
rf_id[1]=ID1;
rf_id[2]=ID2;
rf_id[3]=ID3;
rf_id[4]=ID4;
```

将对应地址的值赋给数组：rf_id[]

```
memcpy(&TRX_RX_ADDR_P0_0, rf_id, 5);
memcpy(&TRX_RX_ADDR_P1_0, rf_id, 5);
memcpy(&TRX_TX_ADDR_0, rf_id, 5);
```

将 rf_id[] 的值赋给

了 2.4G 发射接收端的通信地址。

接下来给大家说明下滚码烧录的应用范围，不知道大家有没有留意到烧录器上有这么个选项：



“地址自动加 1”的功能是确保烧录的芯片都有一个唯一的码,该码的应用范围就很广了,既可以做蓝牙地 mac 地址码,以实现每个蓝牙设备有独立的 mac 地址。又可以做 2.4G 的地址码,可用于 2.4G 实现 1 对 1 通信等等。

最后,在以 bk3431 的滚码地址应用

```
uint8 bt_addr_a[6];  
void read_env_config()  
{  
    ENV_CONFIG_P env_config=(ENV_CONFIG_P)env_config_buff;  
    flash_set_line_mode(1);  
    flash_read_data (bt_addr_a, 0x7fff7, 6); // read address  
    flash_read_data (env_config_buff, 0x40000, sizeof(ENV_CONFIG_T)); // 0x40000  
    flash_read_data (direct_advertising_address, 0x41000, 6);  
#ifdef OAD_IMAGE_A  
    flash_read_data (device_fw_version, 0X2028, 2);  
#else  
    flash_read_data (device_fw_version, 0X42024, 2);  
#endif  
  
    flash_set_line_mode(4);  
    if( (bt_addr_a[0]!=0xff || bt_addr_a[1]!=0xff  
        || bt_addr_a[2]!=0xff || bt_addr_a[3]!=0xff  
        || bt_addr_a[4]!=0xff || bt_addr_a[5]!=0xff  
        )  
    {  
        memcpy(bd_addr, bt_addr_a, 6);  
    }  
}
```

大家可以看到,

```
flash_read_data (bt_addr_a, 0x7fff7, 6);
```

这句代码的意思就是从 0x7fff7 读取对应的值到 bt_addr_a, 然后又赋给了 bd_addr。