

## user uboot demo 使用说明 v1.1.1

目的：客户可自行修改 uboot 实现串口升级功能。

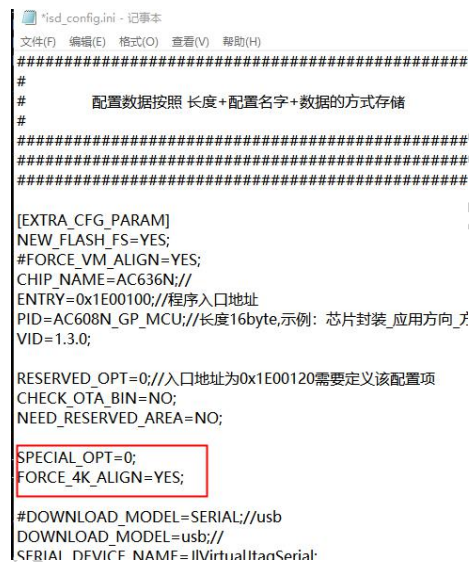
注意：1.样机已有程序 **uboot** 与 升级文件 **uboot** 要一致。

2. 建议将程序强制设置成 4K 对齐，避免不必要的麻烦。操作方法为在 ini 文件里添加以下语句：

**SPECIAL\_OPT=0;**

**FORCE\_4K\_ALIGN=YES;**

如图：

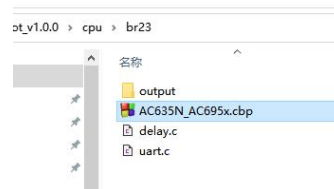


```
#####  
#  
# 配置数据按照 长度+配置名字+数据的方式存储  
#  
#####  
#####  
#####  
[EXTRA_CFG_PARAM]  
NEW_FLASH_FS=YES;  
#FORCE_VM_ALIGN=YES;  
CHIP_NAME=AC636N;//  
ENTRY=0x1E00100;//程序入口地址  
PID=AC608N_GP_MCU;//长度16byte,示例：芯片封装_应用方向_子  
VID=1.3.0;  
  
RESERVED_OPT=0;//入口地址为0x1E00120需要定义该配置项  
CHECK_OTA_BIN=NO;  
NEED_RESERVED_AREA=NO;  
  
SPECIAL_OPT=0;  
FORCE_4K_ALIGN=YES;  
  
#DOWNLOAD_MODEL=SERIAL;//usb  
DOWNLOAD_MODEL=usb;//  
SERIAL_DEVICE_NAME=\\Virtual\\ftanSerial
```

若无法使用 4k 对齐(代码空间不够)，请确保升级用的 **bin** 文件，是在用[强制升级工具连接样机下载代码时生成的](#)。

### 一、如何使用 user\_boot sdk

根据使用程序 SDK 选择对应 cbp 文件，比如使用的是 AC635N 系列芯片，则选择 \cpu\br23\AC635N\_AC695x.cbp，其他系列类似。如果不清楚对应关系的，可以咨询相关支持人员。

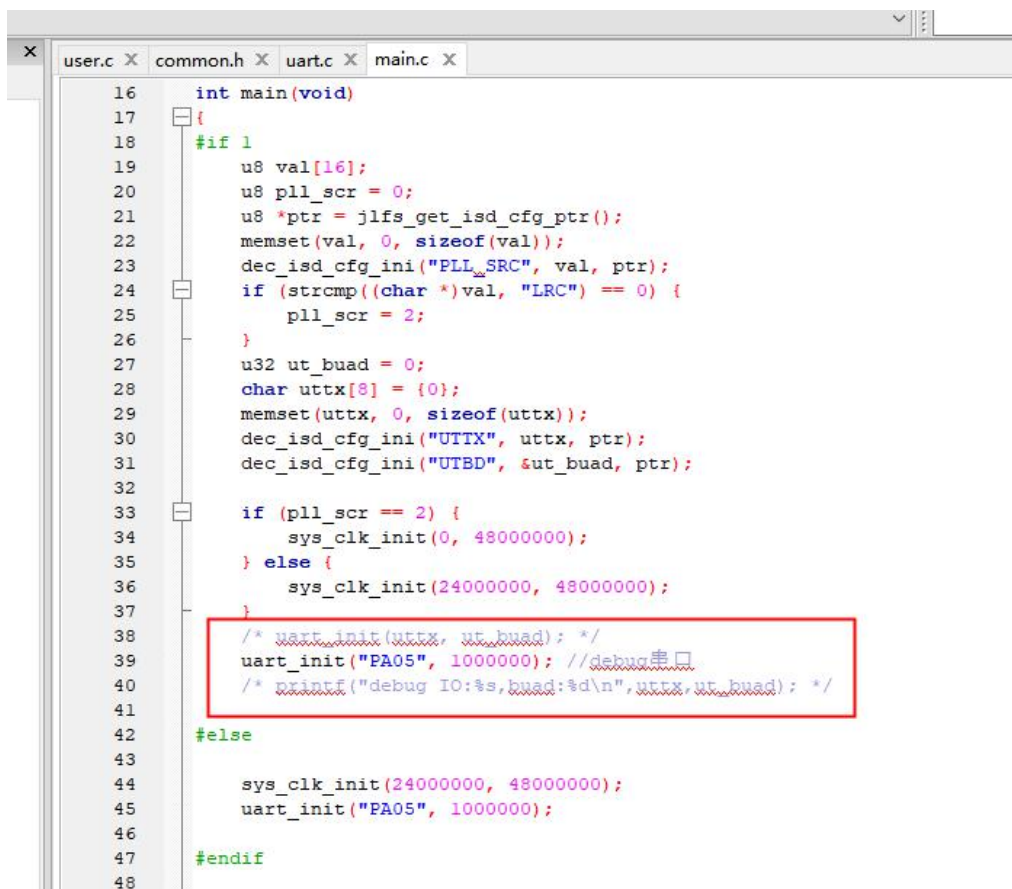


### 二、uboot 程序部分说明

#### 1、调试打印串口初始化

如图所示，打印调试宏为 APP\_DEBUG，默认关闭，在 common.h 定义，调试完成后最好关闭打印

APP\_DEBUG 及调试打印初始化。



```
16 int main(void)
17 {
18     #if 1
19         u8 val[16];
20         u8 pll_scr = 0;
21         u8 *ptr = jlfs_get_isd_cfg_ptr();
22         memset(val, 0, sizeof(val));
23         dec_isd_cfg_ini("PLL_SRC", val, ptr);
24         if (strcmp((char *)val, "LRC") == 0) {
25             pll_scr = 2;
26         }
27         u32 ut_buad = 0;
28         char uttx[8] = {0};
29         memset(uttx, 0, sizeof(uttx));
30         dec_isd_cfg_ini("UTTX", uttx, ptr);
31         dec_isd_cfg_ini("UTBD", &ut_buad, ptr);
32
33         if (pll_scr == 2) {
34             sys_clk_init(0, 48000000);
35         } else {
36             sys_clk_init(24000000, 48000000);
37         }
38         /* uart_init(uttx, ut_buad); */
39         uart_init("PA05", 1000000); //debug串口
40         /* printf("debug IO:%s,buad:%d\n", uttx, ut_buad); */
41
42     #else
43
44         sys_clk_init(24000000, 48000000);
45         uart_init("PA05", 1000000);
46
47     #endif
48 }
```

## 2、升级串口初始化

如图，设置升级需要的 tx/rx IO 口。

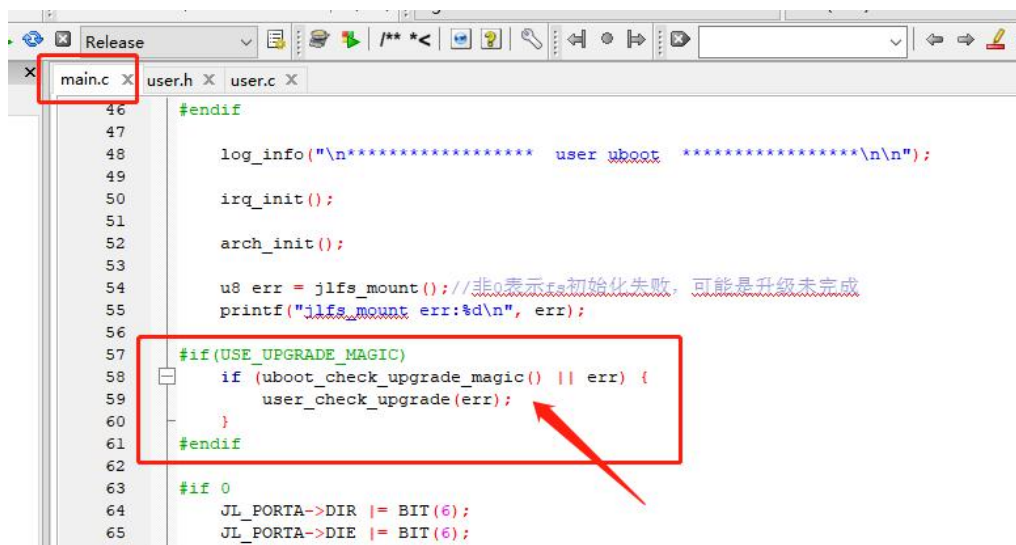
```
check_upgrade(u32 jls_err): u0
user.c X common.h X uart.c X main.c X
314
315
316 JLFS_FILE file_copy = {
317     .addr = flash_uboot_offset,
318     .size = flash_app_base_addr,
319 };
320 jlfs_fseek(&file_paste, 0);
321 jlfs_erase(&file_paste, file_paste.size);
322 u32 len;
323 u32 l = flash_alignsize;
324 u8 *p = get_global_buffer(l);
325 jlfs_fseek(&file_copy, 0);
326 jlfs_fseek(&file_paste, 0);
327 do {
328     len = jlfs_fread(&file_copy, p, l);
329     if (!len) {
330         break;
331     }
332     jlfs_write(&file_paste, p, len);
333 } while (1);
334
335 u8 err = jlfs_fopen_by_name(&upgrade_file, "app_dir_head", 0);
336 if (err) {
337     log_info("file open error !\n");
338     return 1;
339 }
340 log_info("%s() %x %d\n", __func__, upgrade_file.addr, upgrade_file.size);
341
342 #if (_HID_UART_MODE_SEL==0)
343     usb_device_mode(0, HID_CLASS);
344 #else
345     ut_device_mode("PA00", "PA01", 1000000);
346 #endif
347
348 upgrade_loop();
349 return 0;
350
351
```

### 3、升级触发

如图，利用 io 检测状态触发，用户可自行修改。命令接收处理在 upgrade\_loop() 中。

```
main.c X
49
50 irq_init();
51
52 arch_init();
53
54 u8 err = jlfs_mount(); //非:表示初始化失败,可能是升级未完成
55 printf("jlfs_mount err:%d\n", err);
56
57 #if (USE_UPGRADE_MAGIC)
58     if (uboot_check_upgrade_magic() || err) {
59         user_check_upgrade(err);
60     }
61 #endif
62
63 #if 0
64     JL_PORTA->DIR |= BIT(6);
65     JL_PORTA->DIE |= BIT(6);
66     JL_PORTA->PU |= BIT(6);
67     udelay(100);
68     if (((JL_PORTA->IN & BIT(6)) == 0) || err) {
69         user_check_upgrade(err);
70     }
71 #endif
72
73 jl_check_upgrade(err);
74
75 sfc_mode_boot();
```

第二种触发方式，可以从 app 层经软复位（不断电）进入到 uboot 进行升级触发。



```

46 #endif
47
48 log_info("\n***** user uboot *****\n\n");
49
50 irq_init();
51
52 arch_init();
53
54 u8 err = jifs_mount(); //非0表示fs初始化失败, 可能是升级未完成
55 printf("jifs_mount err:%d\n", err);
56
57 #if (USE_UPGRADE_MAGIC)
58 if (uboot_check_upgrade_magic() || err) {
59     user_check_upgrade(err);
60 }
61 #endif
62
63 #if 0
64 JL_PORTA->DIR |= BIT(6);
65 JL_PORTA->DIE |= BIT(6);

```

app 层可以通用添加以下内容来实现该功能。

```

extern u32 nvram_list[];

#define NV_RAM_LIST_ADDR nvram_list

static u8 uboot_uart_upgrade_magic[8] = {
    'u', 'b', 'o', 'o', 't', 0x5a, 's', 't',
};

static u8 uboot_uart_upgrade_succ_magic[8] = {
    'u', 'b', 'o', 'o', 't', 0xa5, 'o', 'k',
};

// 须在 memory_init(); 前检测该标志
void check_uboot_uart_upgrade()
{
    if (memcmp((char *)NV_RAM_LIST_ADDR, uboot_uart_upgrade_succ_magic,
sizeof(uboot_uart_upgrade_succ_magic)) == 0) {
        memset((char *)NV_RAM_LIST_ADDR, 0, sizeof(uboot_uart_upgrade_succ_magic));
        log_info("uboot uart upgrade succ\n");
    }
}

```

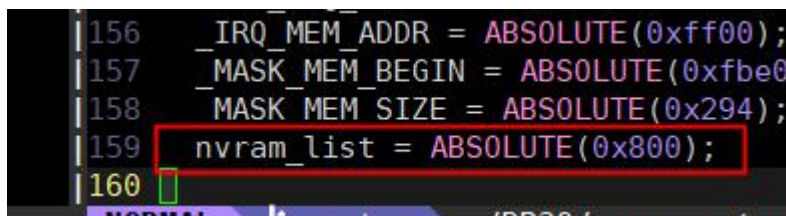
```
void hw_mmu_disable(void);

void chip_reboot_entry_uboot_uart_upgrade_mode()
{
    hw_mmu_disable();

    memcpy((char *)NV_RAM_LIST_ADDR, uboot_uart_upgrade_mode_magic,
sizeof(uboot_uart_upgrade_mode_magic));

    cpu_reset();
}
```

其中 nvram\_list[];的定义在 maskrom\_stubs.ld（不同 SDK 位置可能不一样，搜索出来）



各个系列值如下：

BR23: nvram\_list = ABSOLUTE(0x10800);  
BR25: nvram\_list = ABSOLUTE(0x10880);  
BR34: nvram\_list = ABSOLUTE(0x28800);  
BD19: nvram\_list = ABSOLUTE(0x800);

如果还有其他系列要用，请咨询相关人员。

使用时，调用 chip\_reboot\_entry\_uboot\_uart\_upgrade\_mode(); 复位进入 uboot，uboot 检测到标志进入升级模式。从 uboot 升级完，可以用 check\_uboot\_uart\_upgrade()检测升级是否成功（注意要放在 memory\_init();前）

### 三、V1.1.1 pc 上位机 demo 说明

#### 1、上位机界面

使用 QT 编写，提供源码，用户也可以自行修改（版本号：5.9 以上，小于 6.0）。界面如图，对应说明如下：

- 1 选择对应的串口

- 2 串口波特率

- 3 升级 uboot（一般不进行此操作）

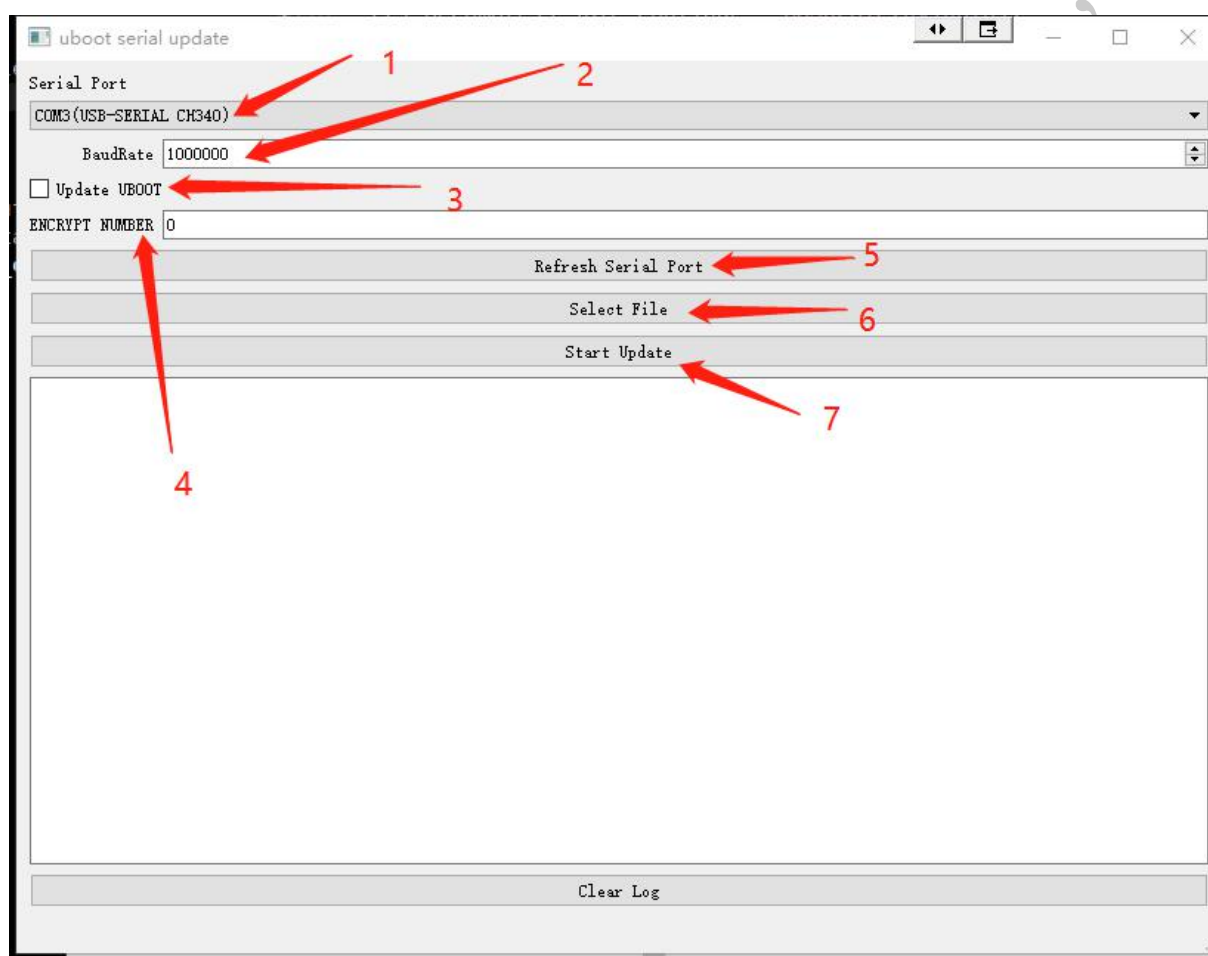
- 4 通信加密(注意：uboot sdk 默认带加密，密钥为 12345678（10 进），建议用户使用



自己的通信密钥)

```
154  
155 //10进制，注意要与上位机一致，不然无法升级  
156 u32 communication_key = 12345678;  
157 //加解密  
158 void enc_dec_data(u8 *buf, u32 l[n])
```

- 5 复位串口
- 6 升级文件选择
- 7 开始升级按键



注意：关于 0K/4K 文件

如果在 isd\_config.ini 有 EOFFSET=1;的配置，则需要在 ini 文件里加上 GENERATE\_TWO\_BIN = YES;的配置生成 0K/4K 文件，然后根据 upgrade\_eoffset（uboot 代码里有），等于 4k 就用 jl\_isd\_4K.bin,否则用 jl\_isd\_0K.bin。如果没有 EOFFSET=1;直接使用 jl\_isd.bin 文件即可。

#### 四、测试流程

- 1、先 build uboot 工程，生成新的 uboot.boot（位于\cpu\brXX\output），并把它复制到 SDK 的 \cpu\brXX\tools 文件夹(即是生成 jl\_isd.bin 的地方，不同的 SDK 里 uboot.boot 位置可能会有所不同)。
- 2、下载 SDK 到小机，此时生成的 jl\_isd.bin 为程序 A，先保存备份。
- 3、再修改 SDK（修改某些打印），再下载到小机，此时生成的 jl\_isd.bin 为程序 B。

上面的步骤得到小机上运行着程序 B，和待升级文件 jl\_isd.bin（程序 A）。

- 4、先把对应的串口通信线接好，小机上电，**检测进入串口升级模式**，进入该模式等待。
- 5、PC 上位机设置好对应的参数（com 口、波特率、密钥、选择升级文件）。
- 6、上位机按 start update 开始升级。

**注意：**升级用的 jl\_isd.bin 文件必须是下载到小机时生成的文件，否则部分芯片会产生对齐错误。