上海艾拉比智能科技有限公司	文档编号	ABUP-IOTD-CO-R&D-0890	密级	商密
	项目名称	Abup FOTA 用户手册		

Abup FOTA 用户手册

2019-07-17

文档变更履历				
日期	**版本**	**修订说明**	**修改人**	**备注**
2019/11/15	01	Abup FOTA 用户手册	沈瑞	

目 录

1. 文档概述

本文档描述了 MCU FOTA 的过程，以 STM32L452RE 为例，涵盖了 APP,BootLoader 的移植，内存空间的配智，关键的宏定义与函数接口，开发包是一个针对 MCU 芯片的完整解决方案（HTTP+CoAP、wosun+lusun）。

2. 关键宏定义

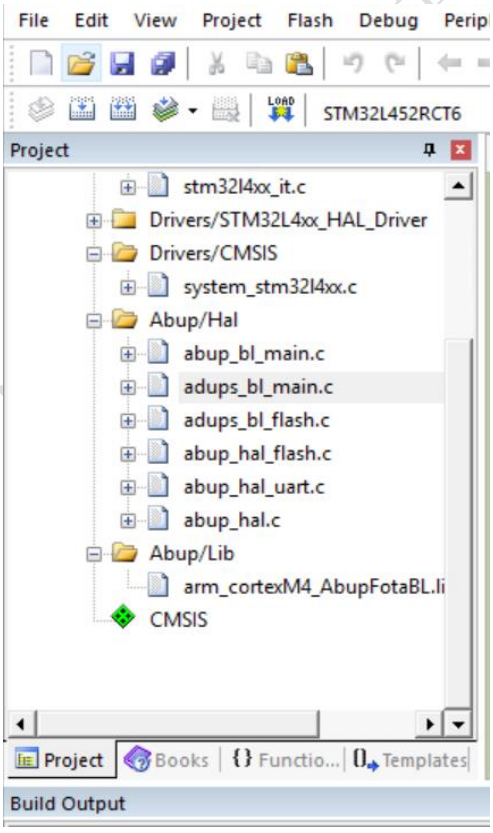
```
//艾拉比 FOTA downloader 开关
#define PKG_USING_ABUP_FOTA
//选择固件下载器使用 HTTP/CoAP 协议，1 是 CoAP，2 是 HTTP
#define ABUP_DEFAULT_NETWORK_PROTOCOL 1
//0 是 lusun 算法，1 是 wosun 算法
#define ABUP_FOTA_ALGORITHM 1
//Abup FOTA 还原算法版本
#define ADUPS_FOTA_WOSUN_VERSION "IOT4.0_R42641"
//Application 分区名称
#define ABUP_RTTHREAD_APP "App"
//Abup 下载分区名称
#define ABUP_RTTHREAD_FLASH "Abup"
//下载长度索引，len = 16X2^index
#define ABUP_DEFAULT_SEGMENT_SIZE_INDEX 3
// flash 最小单位大小
#define ABUP_DEFAULT_SECTOR_SIZE 0x00000800
//bootloader 大小
#define ABUP_BL_SIZE 0x00004000
//app 大小
#define ABUP_APP_SIZE 0x10000
//Abup 分区绝对地址
#define ABUP_UPDATE_ADDR 0x08014000
//Abup 分区大小
#define ABUP_UPDATE_SIZE 0xC000
//OEM(艾拉比服务器上项目信息)
#define ADUPS_FOTA_SERVICE_OEM "G070RB"
```

```
//设备型号(艾拉比服务器上项目信息)
#define ADUPS_FOTA_SERVICE_MODEL "G070RB"
//Product ID(艾拉比服务器上项目信息)
#define ADUPS_FOTA_SERVICE_PRODUCT_ID "1562662709"
//Product Secret(艾拉比服务器上项目信息)
#define ADUPS_FOTA_SERVICE_PRODUCT_SEC "d42a103a639f4b5d94d97c3bd7bc9ba5"
//设备类型(艾拉比服务器上项目信息)
#define ADUPS_FOTA_SERVICE_DEVICE_TYPE "box"
//平台(艾拉比服务器上项目信息)
#define ADUPS_FOTA_SERVICE_PLATFORM "stm32l0"
//App 版本
#define ABUP_FIRMWARE_VERSION "1.0"
```

3.Bootloader

3.1Bootloader 移植

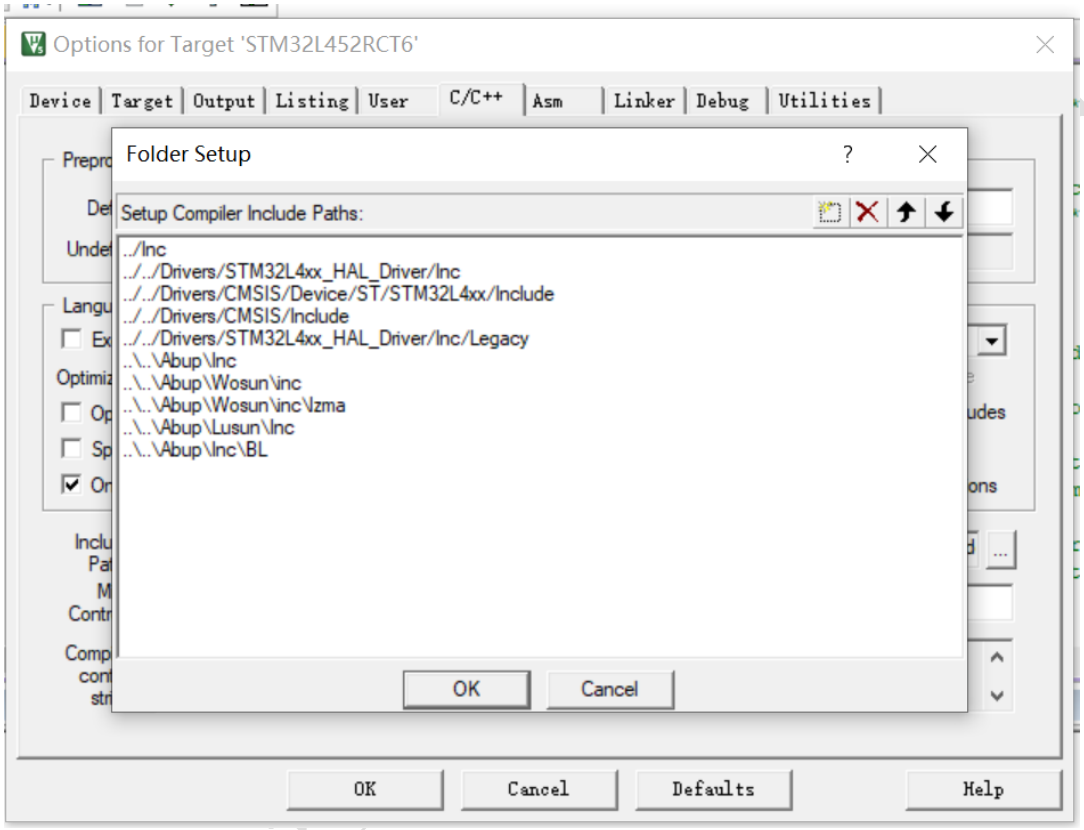
3.1.1 C 文件移植



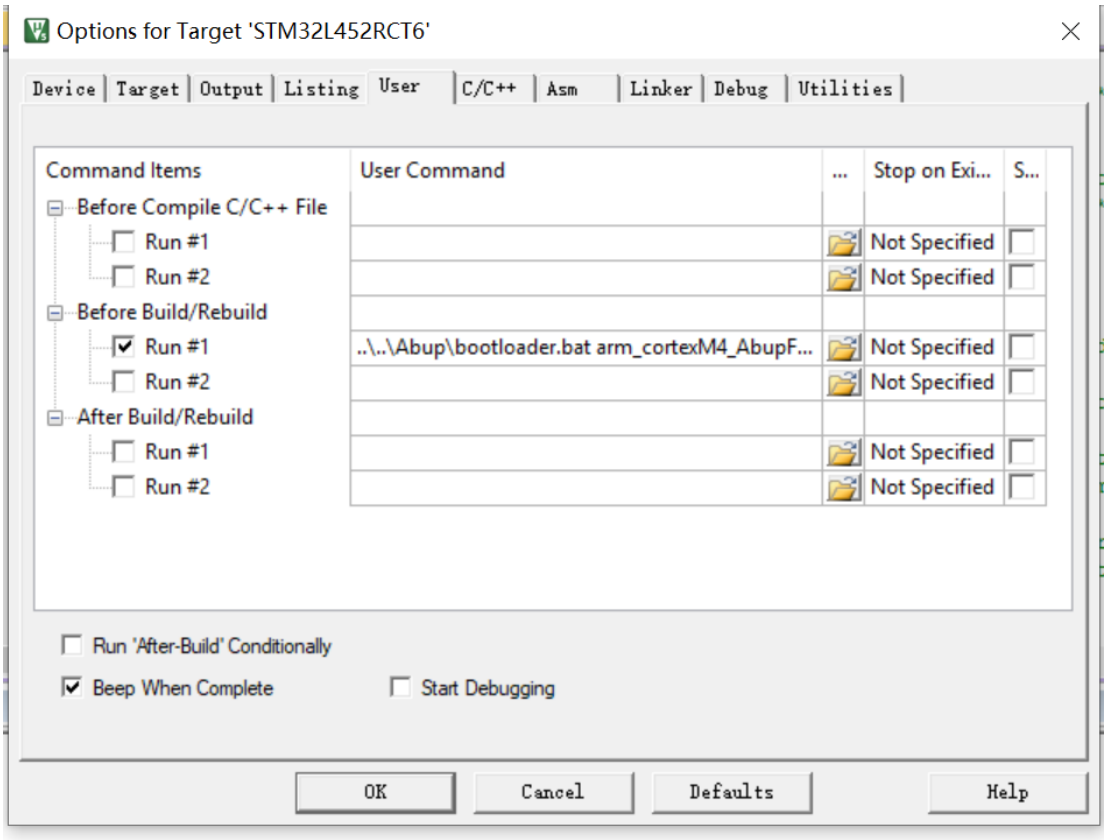
生成 MCU-Nucelo 工程，在该目录下新建 Abup/HAL,Abup/Lib 文件目录，abup_bl_main.c,adups_bl_main.c,adups_bl_flash.c,abup_hal_flash.c,abup_hal_uart.cabup_hal.c

3.1.2 头文件路径添加

将..\..\Abup\inc，..\..\Abup\Wosun\inc，..\..\Abup\Wosun\inc\lzma，..\..\Abup\lusun\inc，..\..\Abup\inc\BL 头文件路径加入项目工程头文件路径中



3.1.3 lib 库添加



新建 Abup/Lib 目录，将默认库 arm_cortexM4_Abup_Fota.lib 添加到该目录下。同时在 Options-for Target->User 中添加 bat 执行脚本，该脚本第一个参数为默认库名称，第二个参数为库文件路径。Lib 库版本的选择由 abup_os.h 中宏 ABUP_BOOTLOADER_DEBUG 来决定。当该宏值为 1 时，开启 debug,将选择使用 arm_cortexM4_Abup_Fota_Debug.lib；若该值为 0,开启 User 模式，使用 arm_cortexM4_Abup_Fota_Release.lib 库执行编译。

3.1.4 代码修改

main.c 中添加如下代码

```
abup_int main(void)
{
  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  LL_Init();

  /* Configure the system clock */
  SystemClock_Config();

  /* Initialize all configured peripherals */
  MX_USART2_UART_Init();
```

```
    abup_memset(progress,0,sizeof(progress));
    abup_memset(progress,'*',20);
    progress[20] = '\r';
    progress[21] = '\n';
    abup_bl_main_printf(progress);
    abup_bl_main_printf(" ENTER BOOTLOADER...\r\n");
    abup_bl_main_printf(progress);
    #if (ABUP_FOTA_ALGORITHM == 0)
    AbupProcedure();
    #else
    AUDPSProcedure();
    #endif
    jump_to_app();
}
```

3.2 主要接口定义

3.2.1 adups_bl_main.c

【原型】**void AUDPSProcedure(void)**

【描述】wosun 升级启动接口，跳转到 App 前使用

【参数详情】无

【返回值】无

3.2.2 abup_bl_main.c

【原型】**void AbupProcedure (void)**

【描述】lusun 算法升级启动接口，跳转到 App 前使用

【参数详情】无

【返回值】无

【原型】**void abup_patch_progress(abup_uint16 total,abup_uint16 current)**

【描述】打印升级 APP 总 SECTOR/BLOCK 数和当前正升级的 SECTOR/BLOCK，如果需要显示进度可以在此函数里添加。

【参数详解】

Parameter	In/Out	Type	Description	Note
total	in	abup_uint16	APP 总 SECTOR 数量	

current	in	abup_uint16	当前正在升级 SECTOR	

【返回值】无

3.2.3 abup_hal_uart.c

【原型】**void abup_bl_main_printf(abup_char *data)**

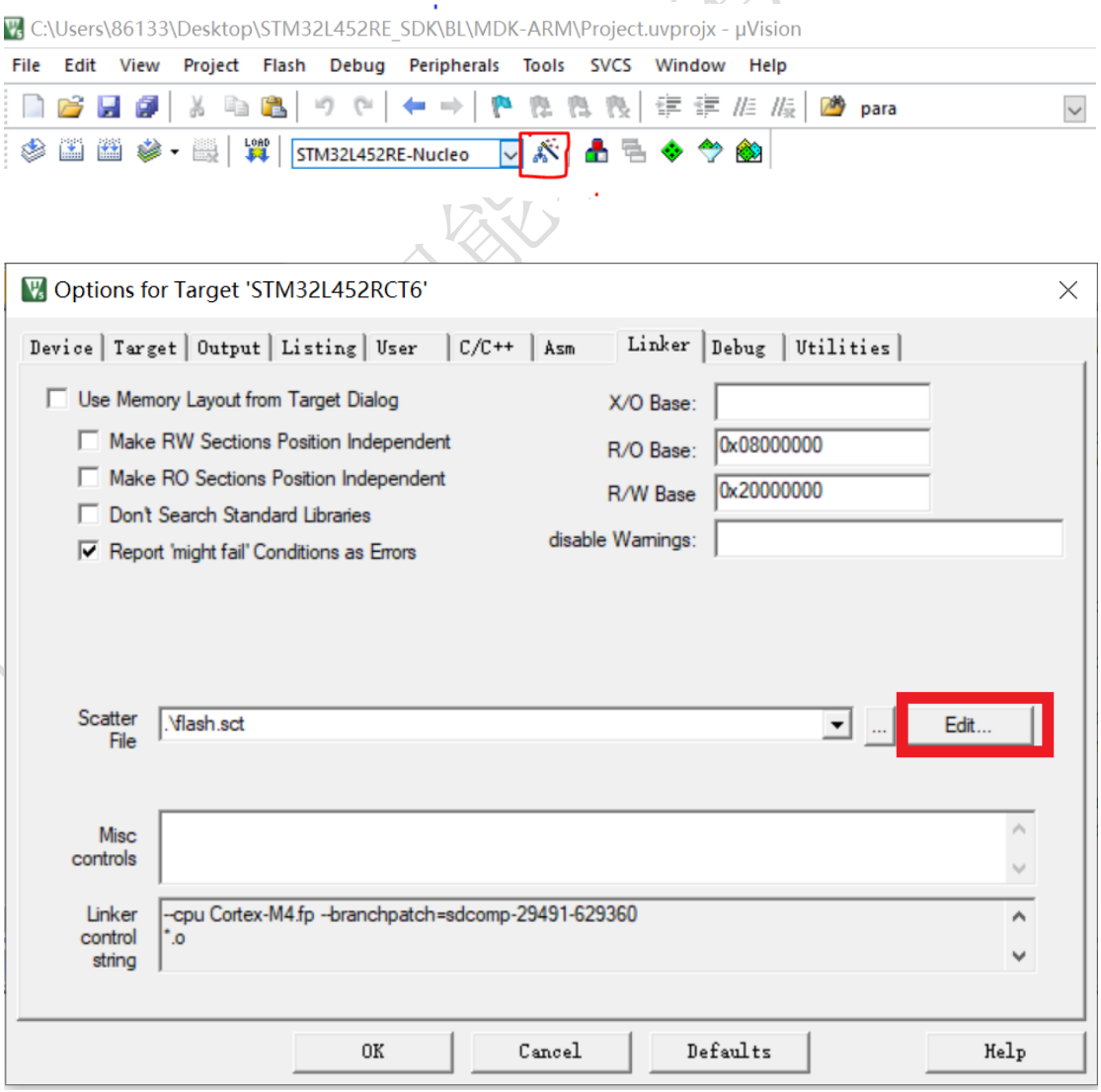
【描述】配置 log 打印。abup_bl_debug_printf ()函数仅当宏 ABUP_BL_PRINT_MAXLEN 定义时将会调用该函数打印串口接收的 log 进行打印

【参数详解】

Parameter	In/Out	Type	Description	Note
data	in	abup_char *	输出字符串信息	

【返回值】无

3.3 Bootloader 代码空间配置



```
; *************************************************************
; *** Scatter-Loading Description File generated by uVision ***
; *************************************************************

LR_IROM1 0x08000000 0x000D000  {    ; load region size_region
  ER_IROM1 0x08000000 0x000D000  {  ; load address = execution addres
   *.o (RESET, +First)
   *(InRoot$$Sections)
   .ANY (+RO)
  }
  RW_IRAM1 0x20000000 0x00028000  {  ; RW data
   *.o (RAMCODE)
   .ANY (+RW +ZI)
  }
}
```

起始地址，

bootloader大小

RAM运行代码

配置需同 ABUP_FLASH_BASE_ADDR、ABUP_BL_SIZE 等宏一致。

4. RT-Thread APP

4.1 RT-Thread App 移植

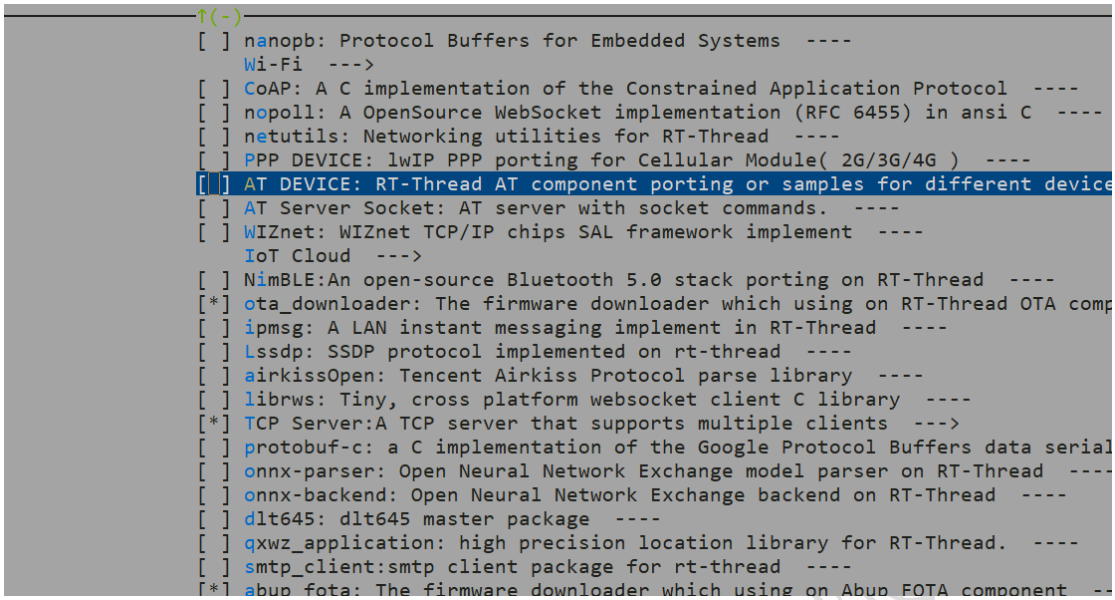
4.1.1 ENV 配置

如图所示，参考 readme 说明。

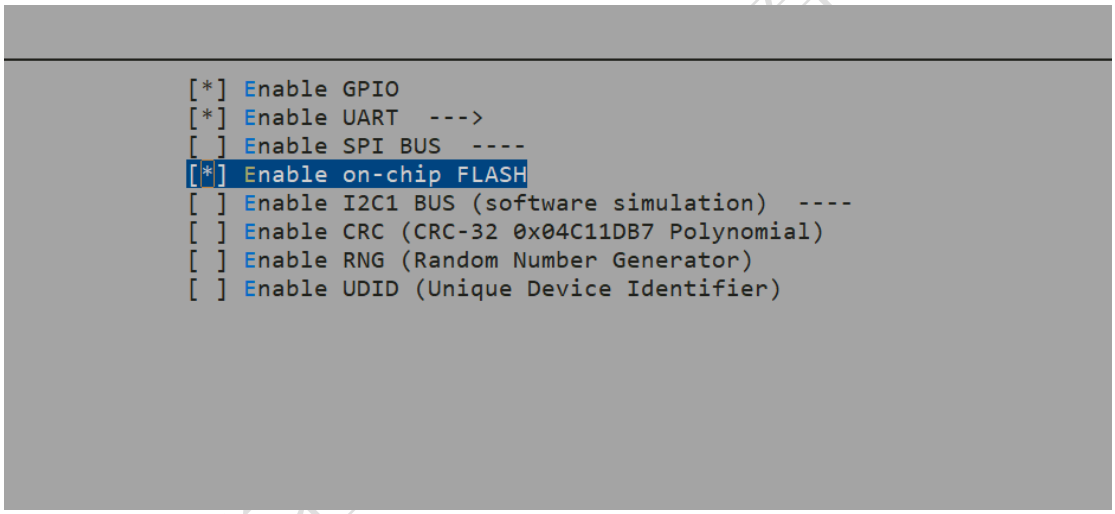
```
-*- abup_fota: The firmware downloader which using on Abup FOTA component
      Enable HTTP/CoAP FOTA (CoAP)  --->
      Recover algorithm (wosun)  --->
[ ]   Enable Abup FOTA Downloader debug
(App) Application area name
(Abup) Data area name
(0x40) Flash minimum unit total
(0x00000800) Flash minimum unit size
(0x08000000) Flash start address
(0x00004000) Bootloader size
(0x10000) Application size
(0x08014000) Abup partition address
(0xC000) Abup partition size
(G070RB) OEM
(G070RB) Device Name
(1562662709) Product ID
(d42a103a639f4b5d94d97c3bd7bc9ba5) Product Secret
(box) Device Type
(stm32l0) Platform
(1.0) firmware version
      Version (latest)  --->
```

4.1.2 依赖关系

需要开启 AT DEVICE 用以连接服务器：



开启 fal，用以保存数据：



4.1.3 代码修改

main.c:

```
22
23  extern int abup_init_update_result(void);
24  int main(void)
25  {
26      int count = 1;
27      /* set LED2 pin mode to output */
28      rt_pin_mode(LED2_PIN, PIN_MODE_OUTPUT);
29      fal_init();
30
31      abup_init_update_result();
32      while (count++)
33      {
34          rt_pin_write(LED2_PIN, PIN_HIGH);
35          rt_thread_mdelay(500);
36          rt_pin_write(LED2_PIN, PIN_LOW);
37          rt_thread_mdelay(500);
38      }
39
40      return RT_EOK;
```

system_stm32XXxx.c:

#include "abup_typedef.h"

#ifdef ABUP_BL_SIZE

#define VECT_TAB_OFFSET ABUP_BL_SIZE /*!< Vector Table base offset field.

This value must be a multiple of 0x200. */

#else

#define VECT_TAB_OFFSET 0x00U /*!< Vector Table base offset field.

This value must be a multiple of 0x200. */

#endif

at_socket_esp8266.c：

设置 mid（连接服务器的唯一标识）

```
    /* set netdev info */
    inet_aton(gateway, &ip_addr);
    netdev_low_level_set_gw(netdev, &ip_addr);
    inet_aton(netmask, &ip_addr);
    netdev_low_level_set_netmask(netdev, &ip_addr);
    inet_aton(ip, &ip_addr);
    netdev_low_level_set_ipaddr(netdev, &ip_addr);
    sscanf(mac, "%x:%x:%x:%x:%x:%x",
            &mac_addr[0], &mac_addr[1], &mac_addr[2], &mac_addr[3], &mac_addr[4],
    for (num = 0; num < netdev->hwaddr_len; num++)
    {
        netdev->hwaddr[num] = mac_addr[num];
    }
    abup_set_mid((char *)netdev->hwaddr);
    /* send dns server query command "AT+CIPDNS_CUR?" and wait response */
    if (at_obj_exec_cmd(device->client, resp, "AT+CIPDNS_CUR?") < 0)
    {
        LOG_W("please check and update device(%s) firmware to support the \"AT+CIP
```

fal_cfg.h：

```
32      ONCHIP_FLASH_DEV
33  }
34
35  /* ===================== Partition Configuration ========================== */
36  #ifdef FAL_PART_HAS_TABLE_CFG
37
38  #ifdef BSP_USING_ON_CHIP_FLASH
39  #define ONCHIP_FLASH_PATITION          {FAL_PART_MAGIC_WROD, ABUP_RTTHREAD_APP,   "onchip_flash",
40                                         {FAL_PART_MAGIC_WROD, ABUP_RTTHREAD_FLASH, "onchip_flash",
41  #else
42  #define ONCHIP_FLASH_PATITION
43  #endif
44
45  /* partition table */
46  #define FAL_PART_TABLE
47  {
```

4.2 主要接口定义

4.2.1 abup_client.c

【原型】**abup_int abup_init_update_result (void)**

【描述】初始化参数，有升级结果上报结果。

【返回值】RT_EOK

【原型】**void abupcv(void)**

【描述】检测新版本。

【参数详解】无

【返回值】无

【原型】**void AbupProgress(abup_int8 state)**

【描述】启动进程

【参数详解】

Parameter	In/Out	Type	Description	Note
state	in	abup_int8	状态	

【返回值】无

【原型】**abup_bool AbupCreateSOC(abup_int8 state,abup_read_cb read_cb)**

【描述】创建套接字连接服务器。

【参数详解】

Paraments	Type	Description	Note
state	abup_int8	状态	
read_cb	abup_read_cb	读取返回函数	

【返回值】成功或失败。

【原型】**abup_char *AbupGetURLPort(abup_int8 state,abup_int *port)**

【描述】获取 url 和 port。

【参数详解】

Paraments	Type	Description	Note
state	abup_int8	状态	
port	abup_int *	Port 指针	

【返回值】url 指针。

【原型】**void abup_closesocket(abup_int sock)**

【描述】关闭套接字。

【参数详解】

Paraments	Type	Description	Note
sock	abup_int	id 号	

【返回值】无。

【原型】**void abup_set_mid(abup_abup_char * mid)**

【描述】保存 mid。

【参数详解】

Paraments	Type	Description	Note
mid	abup_abup_char *	唯一识别码	

【返回值】无。

【原型】**abup_uint abup_rtt_callback(abup_uint8 state,abup_abup_char *buf, abup_uint**

len)

【描述】收取数据处理。

【参数详解】

Paraments	Type	Description	Note
state	abup_uint8	状态	
buf	abup_abup_char *	收取数据缓存	
len	abup_uint	收取数据缓存长度	

【返回值】收取数据处理结果。

【原型】**abup_bool abup_state_send(abup_int8* State,abup_int sock,struct sockaddr_in *addr)**

【描述】发送数据。

【参数详解】

Paraments	Type	Description	Note
state	abup_int8*	状态	
sock	abup_int	id 号	
addr	struct sockaddr_in *	套接字地址	

【返回值】成功或失败。

【原型】**abup_int8 abup_state_continue(abup_int8* State,abup_int8 end,abup_abup_int sock,struct sockaddr_in *addr,abup_abup_char *data,abup_abup_int len,abup_uint result)**

【描述】继续下一步。

【参数详解】

Paraments	Type	Description	Note
state	abup_int8*	状态	
end	abup_int8	结束状态	
sock	abup_int	id 号	
addr	struct sockaddr_in *	套接字地址	
data	abup_char *	收取数据缓存	
len	abup_uint	收取数据缓存长度	
result	abup_int	收取数据处理结果	

【返回值】大于 0 成功，否则失败。

【原型】**void abup_set_try_count(abup_int8* State)**

【描述】设置重试次数。

【参数详解】

Paraments	Type	Description	Note
State	abup_int8*	当前状态	

【返回值】无。

【原型】abup_int8 abup_dl_callback(abup_int8* State,abup_int sock,struct sockaddr_in *addr,abup_char *data,abup_int len)

【描述】下载进程处理。

【参数详解】

Paraments	Type	Description	Note
state	abup_int8*	状态	
sock	abup_int	id 号	
addr	struct sockaddr_in *	套接字地址	
data	abup_char *	收取数据缓存	
len	abup_uint	收取数据缓存长度	

【返回值】成功或失败。

【原型】abup_int8 abup_cv_callback(abup_int8* State,abup_int sock,struct sockaddr_in *addr,abup_char *data,abup_int len)

【描述】检测新版本进程处理。

【参数详解】

Paraments	Type	Description	Note
state	abup_int8*	状态	
sock	abup_int	id 号	
addr	struct sockaddr_in *	套接字地址	
data	abup_char *	收取数据缓存	
len	abup_uint	收取数据缓存长度	

【返回值】成功或失败。

【原型】abup_int8 abup_ru_callback(abup_int8* State,abup_int sock,struct sockaddr_in *addr,abup_char *data,abup_int len)

【描述】上报升级结果进程处理。

【参数详解】

Paraments	Type	Description	Note
state	abup_int8*	状态	
sock	abup_int	id 号	
addr	struct sockaddr_in *	套接字地址	
data	abup_char *	收取数据缓存	
len	abup_uint	收取数据缓存长度	

【返回值】成功或失败。

【原型】**abup_int8 abup_rd_callback(abup_int8* State,abup_int sock,struct sockaddr_in *addr,abup_char *data,abup_int len)**

【描述】上报下载结果进程处理。

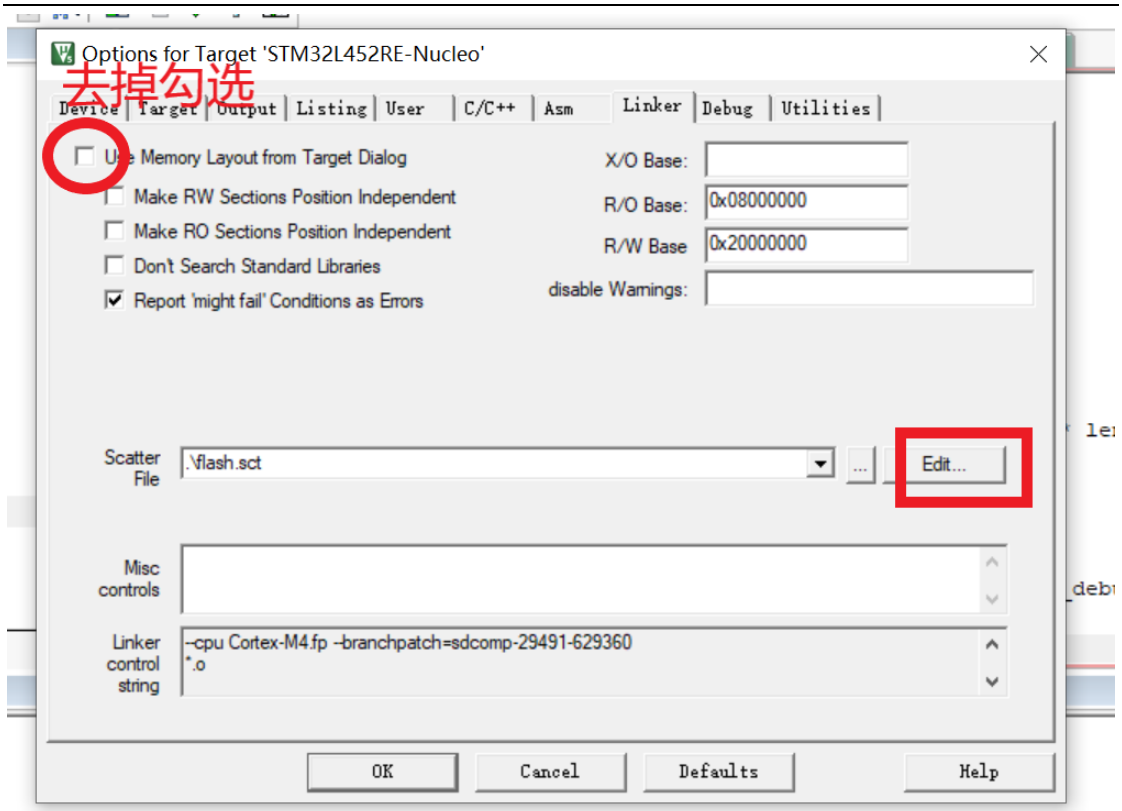
【参数详解】

Paraments	Type	Description	Note
state	abup_int8*	状态	
sock	abup_int	id 号	
addr	struct sockaddr_in *	套接字地址	
data	abup_char *	收取数据缓存	
len	abup_uint	收取数据缓存长度	

【返回值】成功或失败。

4.5APP 代码空间配置

配置需同 ABUP_FLASH_BASE_ADDR、ABUP_APP_SIZE 等宏一致。




```
LR_IROM1 0x0800D000 0x00023000  {    ; load region size_region
  ER_IROM1 0x0800D000 0x00023000  {  ; load address = execution addres
   *.o (RESET, +First)
   *(InRoot$$Sections)
   .ANY (+RO)
  }
  RW_IRAM1 0x20000000 0x00028000  {  ; RW data
   *.o (RAMCODE)
   .ANY (+RW +ZI)
  }
}
```

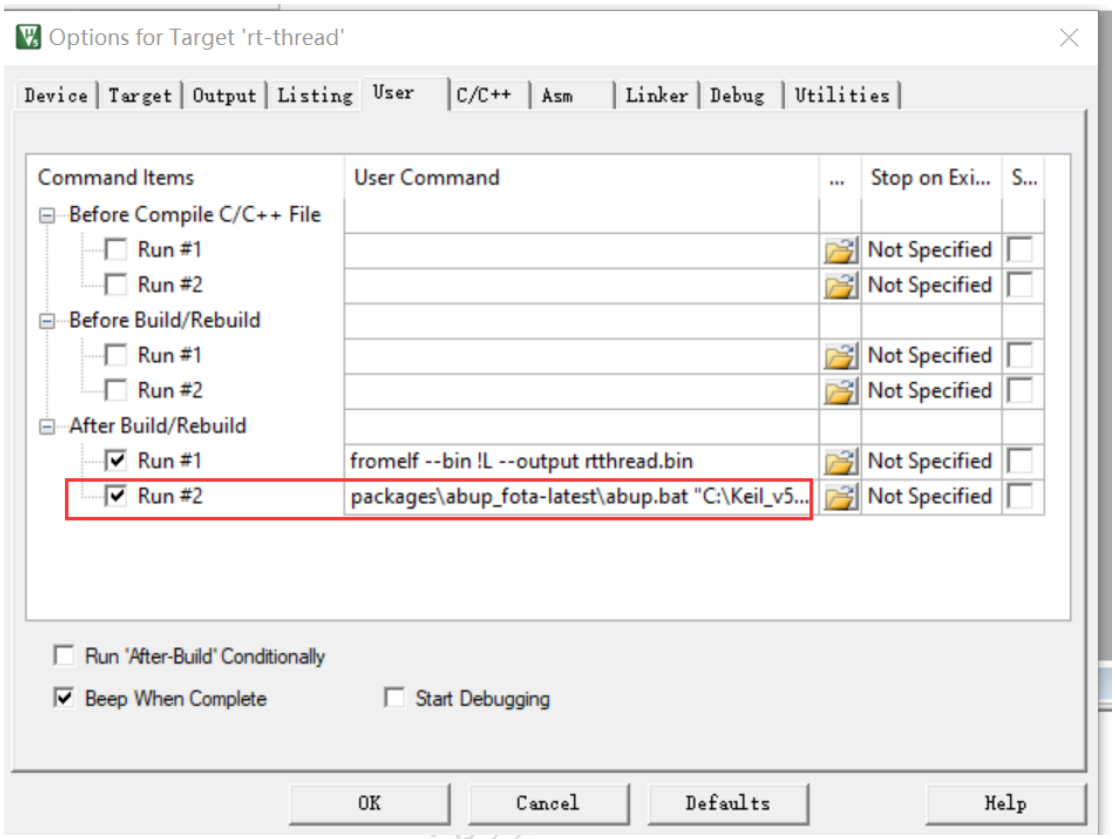
起始地址

APP大小

RAM运行代码

5.调试

5.1 打包



abup.bat 参数说明：

例 子 ： packages\abup_fota-latest\abup.bat "C:\Keil_v5\ARM\ARMCC\bin\fromelf.exe" "C:\Program Files\WinRAR\WinRAR.exe" rtconfig.h build

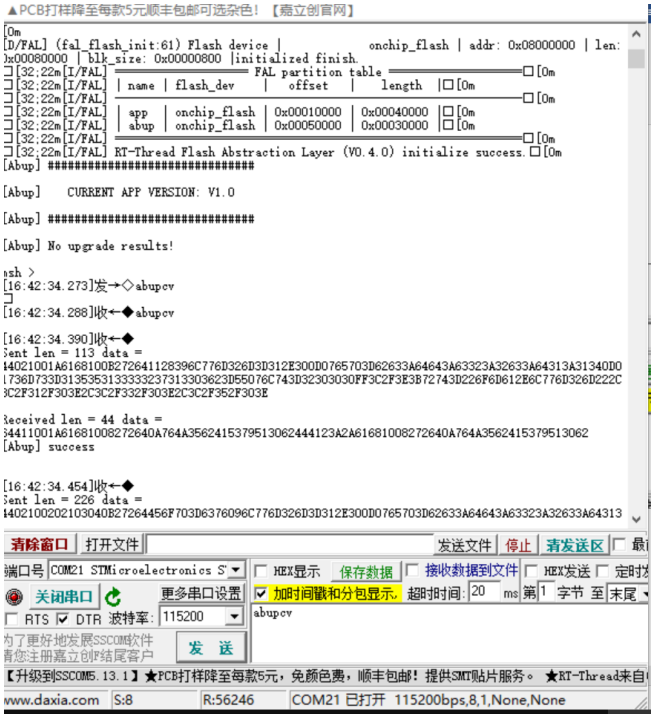
可以修改第二参数为 7ZIP 或 WINRAR。

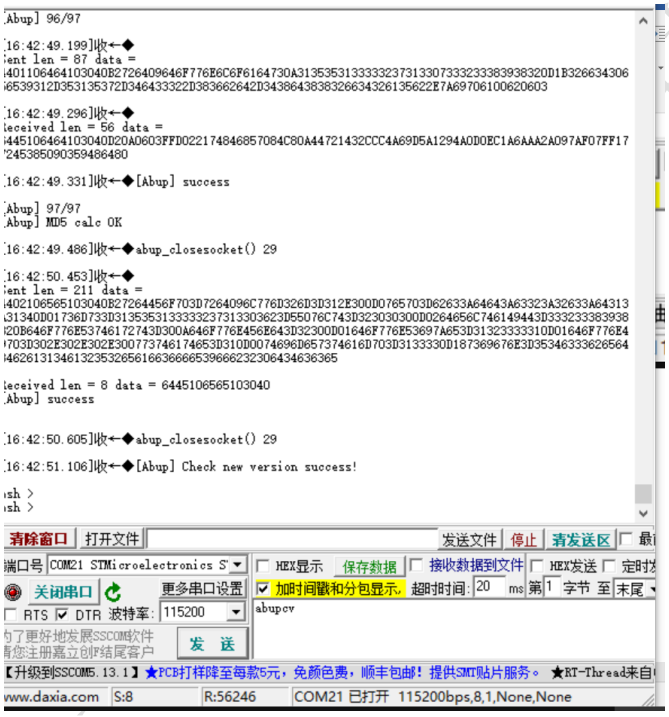
生成压缩包名为：版本号_日期_时间.zip

5.3 助手调试

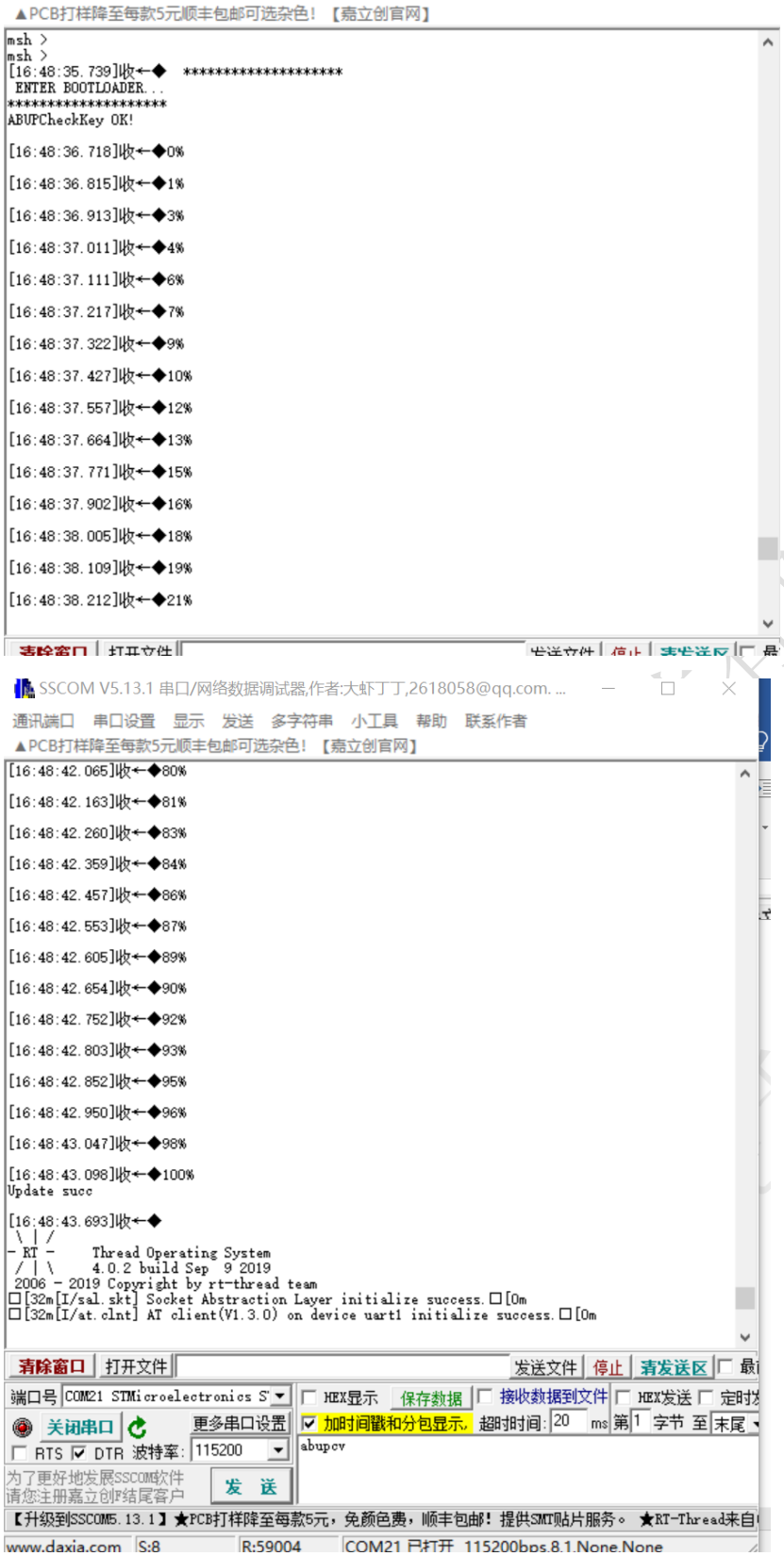
打开串口调试助手，配置好串口号和波特率等相关信息。

abupcv：





AT +BOOT:将会显示升级包还原的进程:



App 启动上报升级结果：