

下位机程序模块说明

下位机程序模块说明

下位机程序模块包含三个文件夹，多个文件

代码块

```
1  src
2  |—— common
3  |   |—— crc16.c
4  |   |—— crc16.h
5  |—— IAP_BSP
6  |   |—— bsp_usb_hid_iap.c
7  |   |—— bsp_usb_hid_iap.h
8  |—— IAP
9      |—— IAP_916.c
10     |—— IAP_916.h
11     |—— IAP_Application.c
12     |—— IAP_Application.h
13     |—— IAP_Transport.c
14     |—— IAP_Transport.h
15     |—— IAP_Bootloader.c
16     |—— IAP_Bootloader.h
17     |—— IAP_Flash_WP.c
18     |—— IAP_Flash_WP.h
19     |—— IAP_FLASH_MAP.h
20     |—— IAP_UserDef.h
```

crc16.c

包含了一个软件实现的crc校验函数

bsp_usb_hid_iap.c

包含了IAP升级的所有描述符定义，以及USB hid数据接收的入口点

IAP_916.c

目前仅包含一个IAP功能总体初始化函数

IAP_Application.c

实现IAP协议的应用层（或者叫业务层）的逻辑，处理应用层协议包，包括【IAP_CMD_START】【IAP_CMD_FLASH_WRITE】【IAP_CMD_SWITCH_APP】，并返回【IAP_CMD_ACK】。

📖 IAP通讯协议

IAP_Transport.c

实现IAP协议的Transport层的逻辑，接收上位机的分包数据组并应用层协议包，或将应用层协议包分包发送给上位机程序。

IAP_Bootloader.c

内部包含一个二级bootloader的升级函数（program_fota_metadata()），升级最后一步会通过此函数来配置二级bootloader的升级元数据，之后程序复位二级bootloader就会自动完成新程序的升级。

代码块

```
1  static void IAP_DoUpdate_Delay_Timeout_Callback(void){
2      IAP_APP_ERROR("[CMD] do update [0x%08X]\n", cmdCtl.loadAddress);
3
4      /* 如果需要重新枚举的话，需要先关掉usb，再进行软复位 */
5      bsp_usb_disable();
6      vTaskDelay(pdMS_TO_TICKS(100));
7
8      /* program meta data */
9      fota_update_block_t ota_meta;
10     ota_meta.src = IAP_STORAGE_ADDR;
11     ota_meta.dest = cmdCtl.loadAddress;
12     ota_meta.size = cmdCtl.binSize;
13     program_fota_metadata(1, &ota_meta);
14
15     /* reset */
16     platform_reset();
17 }
```

IAP_Flash_WP.c

和写保护相关的函数，目前没用到。

IAP_FLASH_MAP.H

按照二级Bootloader的结构定义了相关的地址。

其中建议将用户参数区域规划到Flash顶部。

从APP_TOP_ADDR到APP_PARAM_START_ADDR之间的区域用来存储IAP升级时接收的新程序！，这一部分区域称作IAP_STORAGE，这一部分区域的大小如果不足以存储新程序的话，升级不应该成功。

程序会比较这块区域的大小和HEADER中的bin_size的值来判断能否升级。

代码块

```
1  #if (INGCHIPS_FAMILY == INGCHIPS_FAMILY_918)
2  #define EFLASH_UNIT_SIZE          (uint32_t)(EFLASH_PAGE_SIZE)
3  #elif (INGCHIPS_FAMILY == INGCHIPS_FAMILY_916)
4  #define EFLASH_UNIT_SIZE          (uint32_t)(EFLASH_SECTOR_SIZE)
5  #endif
6  #define ALIGN_TO(address, alignment) (((address) + (alignment) - 1) & ~
    ((alignment) - 1))
7  #define ALIGN_TO_EFLASH_UNIT(address) ALIGN_TO(address, EFLASH_UNIT_SIZE)
8  #define APP_ADDR                   (0x0202A000)
9  #define APP_SIZE                   (0x00005EE0)      // 24288Byte
10 #define APP_PARAM_ADDR             (0x0207C000)      // Param area: from
    APP_PARAM_ADDR to FLASH_TOP_ADDR
11 #define SECOND_BOOT_START_ADDR     (uint32_t)(0x02002000)
12 #define PLATFORM_START_ADDR        (uint32_t)(0x02003000)
13 #define APP_START_ADDR              (uint32_t)(APP_ADDR)
14 #define APP_TOP_ADDR                (uint32_t)(ALIGN_TO_EFLASH_UNIT(APP_ADDR
    + APP_SIZE))
15 #define APP_PARAM_START_ADDR        (uint32_t)(APP_PARAM_ADDR)
16 #define FLASH_TOP_ADDR              (uint32_t)(0x02080000)
17 #define SECOND_BOOT_CODE_SIZE       (PLATFORM_START_ADDR -
    2ND_BOOT_START_ADDR)
18 #define PLATFORM_CODE_SIZE          (APP_START_ADDR -
    PLATFORM_START_ADDR)
19 #define APP_CODE_SIZE               (APP_TOP_ADDR - APP_START_ADDR)
20 #define APP_PARAM_SIZE              (FLASH_TOP_ADDR -
    APP_PARAM_START_ADDR)
```

代码块

```
1  // IAP Storage
2  #define IAP_STORAGE_ADDR (APP_TOP_ADDR)
3  #define IAP_STORAGE_SIZE (APP_PARAM_START_ADDR - APP_TOP_ADDR)
```

IAP_UserDef.h

IAP协议的宏定义部分

代码块

```
1
2  // Product config.
```

```

3  #define USER_FLASH_LOCK_EN          (0)          /* <= Enable flash protection
   function. The test phase can disable,
4
   and the MP phase is
   strongly recommended to enable. */
5  // Test config.
6  #define USER_ERASE_BOOT_PARAM_EN    (0)          /* <= For test. Enable erase
   boot params area for startup. */
7  #define USER_APP_CODE_CHECK_CRC_EN  (0)          /* <= For test. Enable
   checking CRC of APP before jumping to APP. */
8  // Log enable/disable
9  #define USER_IAP_MAIN_DEBUG_LOG_EN  (1)
10 #define USER_IAP_MAIN_ERROR_LOG_EN  (1)
11 #define USER_IAP_PARAM_DEBUG_LOG_EN (0)
12 #define USER_IAP_PARAM_ERROR_LOG_EN (1)
13 #define USER_IAP_APP_DEBUG_LOG_EN   (0)
14 #define USER_IAP_APP_ERROR_LOG_EN   (1)
15 #define USER_IAP_TRANSPORT_DEBUG_EN (0)
16 #define USER_IAP_TRANSPORT_ERROR_EN (1)
17 // version information.
18 #define USER_DEF_CHIP_ID              "ING91680C_TB"          // max =
   15bytes.
19 #define USER_DEF_ITEM_STR             "HS_DG_WH_V01"          // max =
   23bytes.
20 // boot usb information.
21 #define USER_DEF_USB_VID              (0x36B0)
22 #define USER_DEF_USB_PID              (0x3002)
23 #define USER_DEF_IAP_REPORT_ID        (0x3F)

```

如何集成到自己的项目中

1. 应用二级Bootloader

为 ING916 添加二级 Bootloader

[ING916 二级Bootloader 调试记录](#)

如果要详细了解二级Bootloader的实现原理，可以阅读上面的两个文档。

只是想在现有的工程上应用二级bootloader，只需要阅读调试记录->搭建的步骤->改造platform.bin，然后将事先准备好的二级bootloader拷贝过去直接用就行了

2. 集成软件模块

3. 将文件拷贝到自己的src目录中，在工程中添加相关源代码文件，同时配置编译选项中的include目录。
4. 将bsp_usb_hid_iap.c中的描述符，枚举过程，IAP端点发送接收数据的处理合入到自己的代码中。

157744/ 这里避免篇幅太长，只展示部分需要移植的代码，枚举过程的代码没有展示出来

```
2 USB_DEBUG("##USB recv end: ep(%d)\n", event->data.ep);
3 if(event->data.ep == EP_IAP_OUT){
4     #if 0
5     platform_printf("(%d)RCV[%d]: ",event->data.ep, event-
>data.size);printf_hexdump(DataRecvBuf, event->data.size);
6     #endif
7     /* Push rx data to user callback. */
8     bsp_usb_hid_iap_push_rx_data_to_user(&DataRecvBuf[1], event->data.size-1);
9     /* Start next rx proc. */
10    bsp_usp_hid_iap_rx_data_trigger(2);
11 }
12 /* If send ok, Clear busy status, and notify user. */
13 if(event->data.ep == EP_IAP_IN){
14     IapReport.sendBusy = U_FALSE;
15     bsp_usb_hid_iap_push_send_complete_to_user();
16 }
```

1. 修改设备信息

代码块

```
1 // INFO BEGIN.
2 static const char bUpgradeFlag[] = "INGCHIPS";
3 static const char bChipID[] = USER_DEF_CHIP_ID;
4 static const char bItemStr[] = USER_DEF_ITEM_STR;
5 static const char bHardware[] = "V0.0.0";
6 static const char bSoftware[] = "V0.0.0";
7 #define GET_STR_LEN(x) (strlen(x))
8 // INFO END.
```

1. 如果有用户Flash保存参数，请移动到Flash顶部，并且修改IAP_FLASH_MAP.H中的APP_PARAM_ADDR宏定义。
2. 若程序能编译通过了，需要查看程序的大小，并更新IAP_FLASH_MAP.H中的APP_SIZE宏定义，IAP程序会依此计算IAP_STORAGE区域的大小，IAP_STORAGE区域的大小决定了Flash能否容纳新程序。（每次更新程序都要检查更新APP_SIZE）。

代码块

```
1
2 #define APP_ADDR (0x0202A000)
3 #define APP_SIZE (0x00005EE0) // 24288Byte
4 #define APP_PARAM_ADDR (0x0207C000) // Param area: from
APP_PARAM_ADDR to FLASH_TOP_ADDR
```

3. 验证是否集成成功

使用USB IAP 工具验证

[📖 上位机工具使用指南（原版）](#)

1. 创建一个新版本程序，保证设备信息匹配，**IAP_STORAGE**区域大小能容纳新版程序
2. 使用上位机工具生成升级工具
3. 使用升级工具尝试升级