AT32F415 IAP应用指南

1. IAP 在线升级原理概述

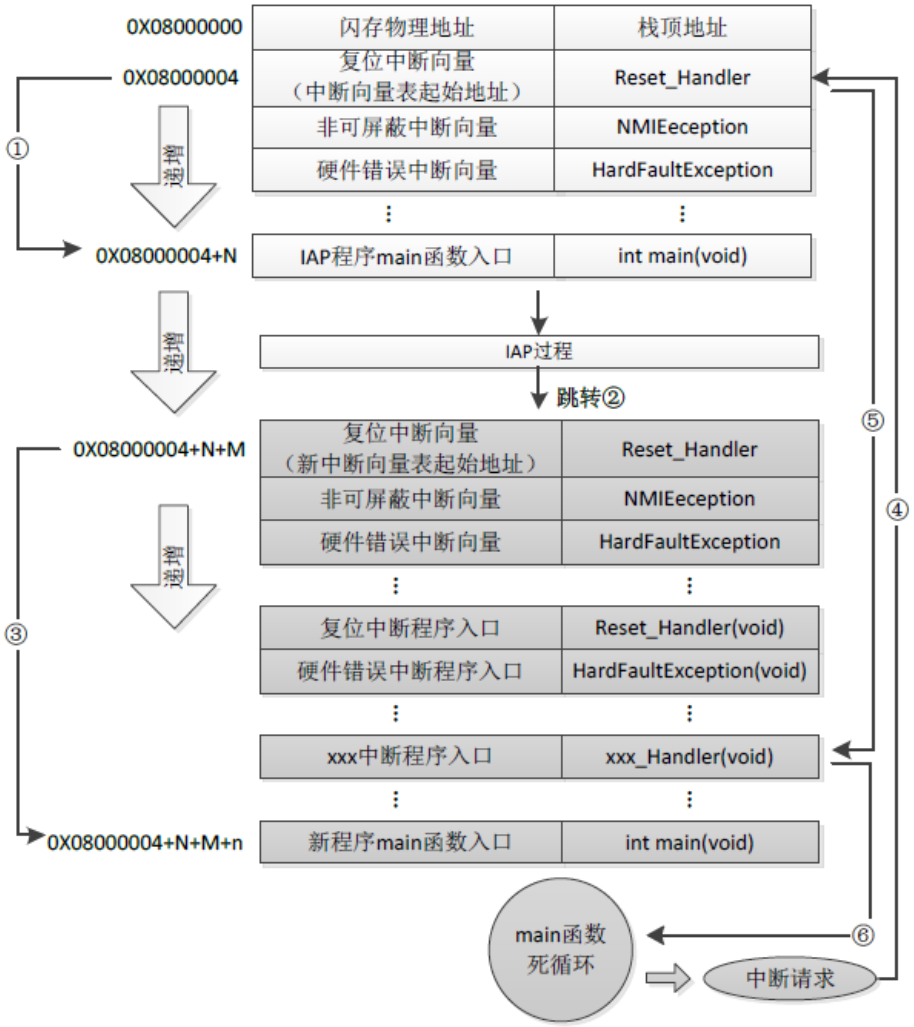
IAP（In Application Programming）即在线应用编程，IAP是用户自己的程序在运行过程中对User Flash的部分区域进行烧写，目的是为了在产品发布后可以方便地通过预留的通信口对产品中的固件程序进行更新升级。通常实现IAP功能时，即用户程序运行中作自身的更新操作，需要在设计固件程序时编写两个项目代码，第一个项目程序不执行正常的功能操作，而只是通过某种通信方式(如USB、USART)接收程序或数据，执行对第二部分代码的更新；第二个项目代码才是真正的功能代码。这两部分项目代码都同时烧录在User Flash中，当芯片上电后，首先是第一个项目代码开始运行，它作如下操作：

1） 检查是否需要对第二部分代码进行更新

2） 如果不需要更新则转到4）

3） 执行更新操作

4） 跳转到第二部分代码执行



在上图所示流程中，AT32复位后，还是从0x08000004地址取出复位中断向量的地址，并跳转到复位中断服务程序，在运行完复位中断服务程序之后跳转到IAP的main函数，如图标号①所示，此部分同图52.1.1一样；在执行完IAP以后（即将新的APP代码写入AT32的FLASH，灰底部分。新程序的复位中断向量起始地址为0x08000004+N+M），跳转至新写入程序的复位向量表，取出新程序的复位中断向量的地址，并跳转执行新程序的复位中断服务程序，随后跳转至新程序的main函数，如图标号②和③所示，同样main函数为一个死循环，并且注意到此时AT32的FLASH，在不同位置上，共有两个中断向量表。

在main函数执行过程中，如果CPU得到一个中断请求，PC指针仍强制跳转到地址0x08000004中断向量表处，而不是新程序的中断向量表，如图标号④所示；程序再根据我们设置的中断向量表偏移量，跳转到对应中断源新的中断服务程序中，如图标号⑤所示；在执行完中断服务程序后，程序返回main函数继续运行，如图标号⑥所示。

通过以上两个过程的分析，我们知道IAP程序必须满足两个要求：

1） 新程序必须在IAP程序之后的某个偏移量为x的地址开始

2） 必须将新程序的中断向量表相应的移动，移动的偏移量为x

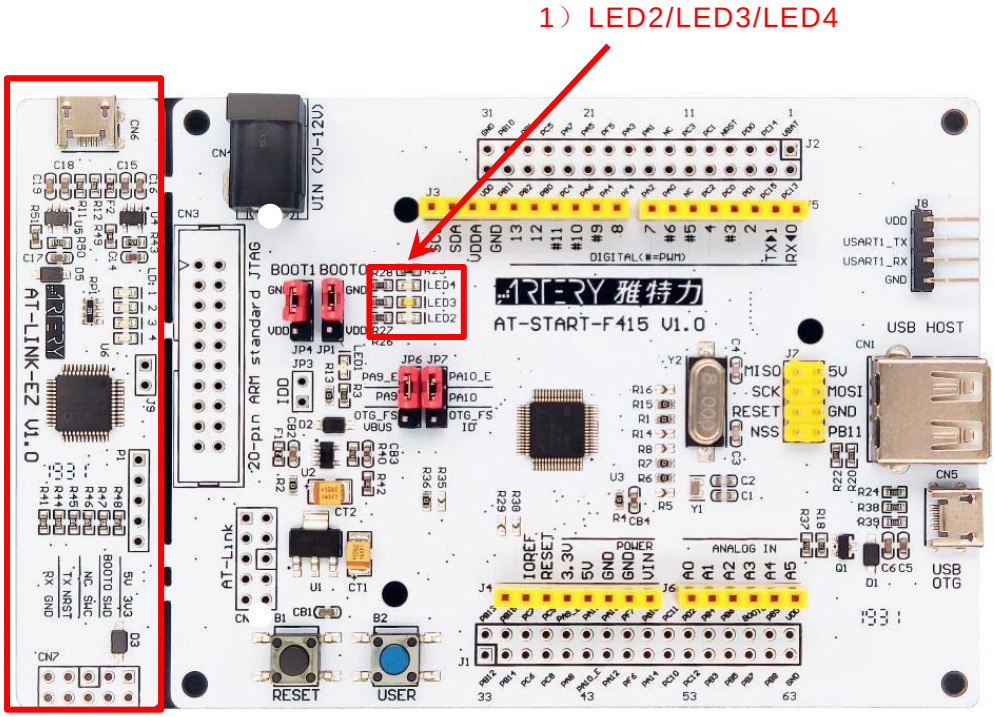
2 AT32串口IAP快速使用方法

2.1 硬件资源

1） 指示灯LED2/LED3/LED4

2） USART1 (PA9/PA10)

3） AT-START-F415 V1.0 开发板



Note:

1. 该IAP Demo是基于AT32F415的硬件条件，若使用者需要在AT32其他型号上使用，请修改相应配置即可
2. 使用AT-Link-EZ 连接开发板时，在上位机的设备管理器可看到ATLink-USART串口埠，此串口埠连接到芯片的USART1，请使用此串口埠与上位机通讯

2.2 软件资源

1） APP_Release

- IAP_Programmer_V1.2.2.exe: PC上位机Tool，下载时LED2闪烁

2） SourceCode

- IAP_Bootloader: IAP_Bootloader源程序
- APP_LED3_TOGGLE: LED3闪烁源程序
- APP_LED4_TOGGLE: LED4闪烁源程序

3） Doc

- AT32F415 IAP应用指南

Note: IAP所有Project都是基于keil 5而建立，若有需要在其他编译环境上使用，请参考AT32F4xx_StdPeriph_Lib_V1.x.x\Project\AT_START_F4xx\Templates中各种编译环境（例如IAR6/7/8, keil 4/5）进行简单修改即可

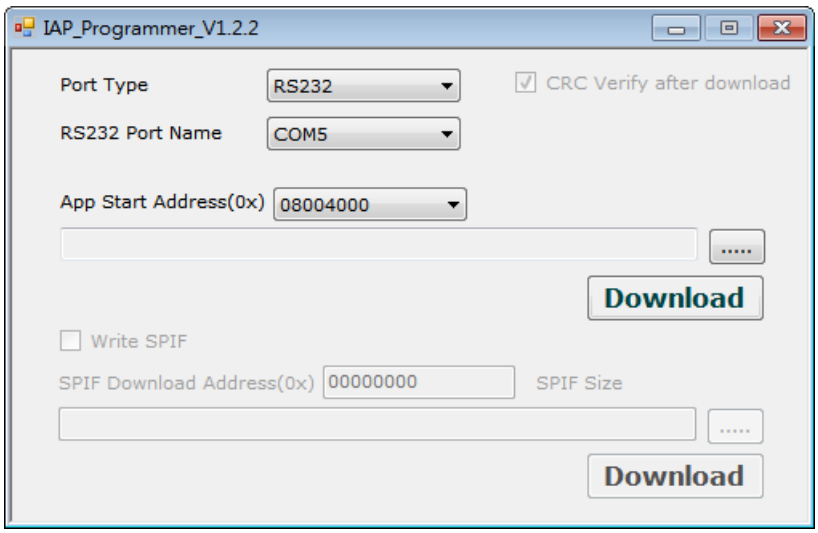
2.3 IAP Demo使用

1） 打开IAP_Bootloader Project源程序，编译后下载到实验板

2） 打开IAP_Programmer_V1.2.2.exe

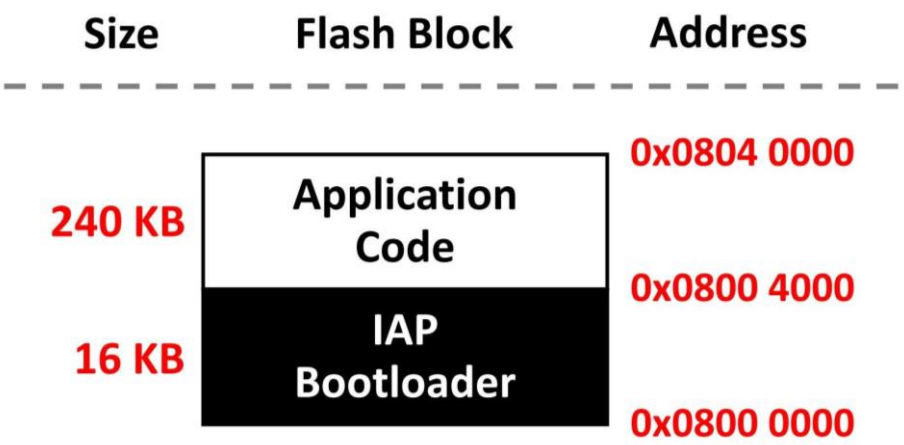
3） 如图，通讯串口选择RS232，选择对应串口埠后，APP下载地址选择0x08004000，点选即将更新之BIN文档，点击Download下载

4） 观察LED2/3/4闪烁，LED2闪烁-IAP_Bootloader工作，LED3闪烁-APP_LED3工作，LED4闪烁-APP_LED4工作



3 AT32 IAP程序设置

3.1 地址分布



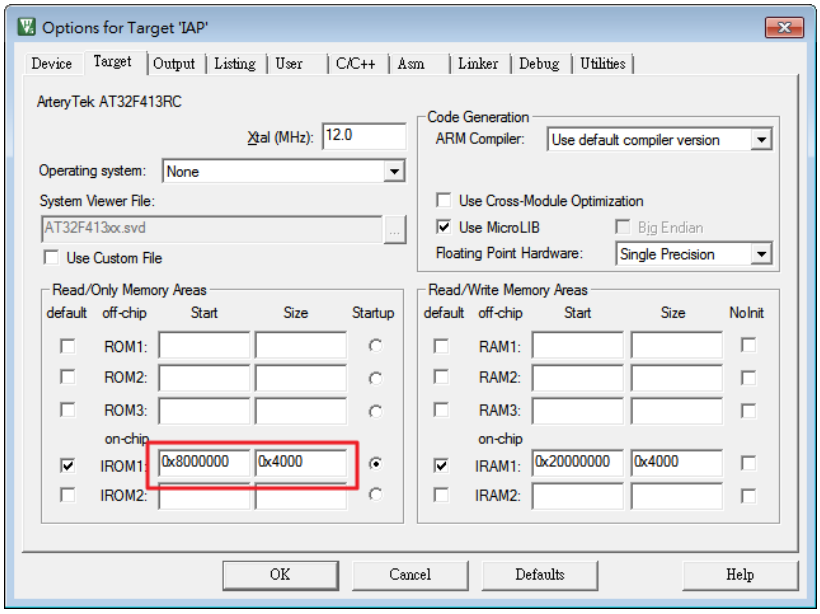
Note: IAP_Bootloader最后一个Page(2KB)，用于放置防掉电丢失的Flag，用户修改IAP时请勿操作该段地址

3.2 IAP Bootloader Project配置

1） Keil设置

-> Options for Target -> Target -> IROM1

OTA_Bootloader预留Code Size为16KB， Flash Size填入0x4000



2） IAP源程序修改

flash.h的FLASH_SIZE，开发板MCU为AT32F415RC， Flash Size为256KB

```
#define FLASH_SIZE 256        //所选AT32的FLASH容量大小(单位为KB)
```

Iap.h的FLASH_APP_ADDR，可修改APP程序的启始位址

```
#define FLASH_APP_ADDR          0x08004000                  // 0x08004000~0x0800FFFF的空间为
#define APP_UPGRADE_FLAG_ADDR   (FLASH_APP_ADDR-2048)       // IAP地址最后一个Page放Flag
#define APP_UPGRADE_FLAG        0x41544B38
```

3.3 APP Project配置

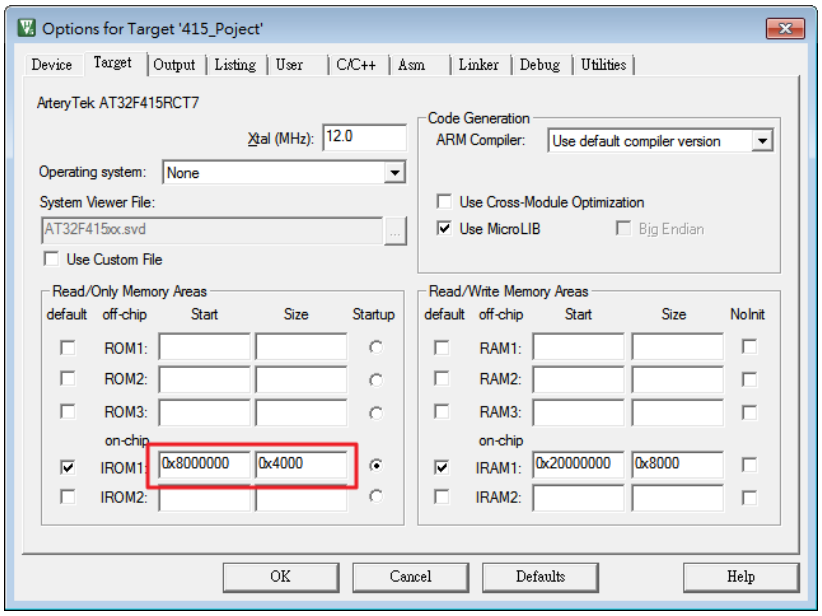
该Demo提供了2个APP程序供测试用，皆以Flash Addr 0x08004000为起始地址。APP1 LED3闪烁，APP2 LED4闪烁。设计步骤如下：

1） Keil配置

-> Options for Target -> Target -> IROM1

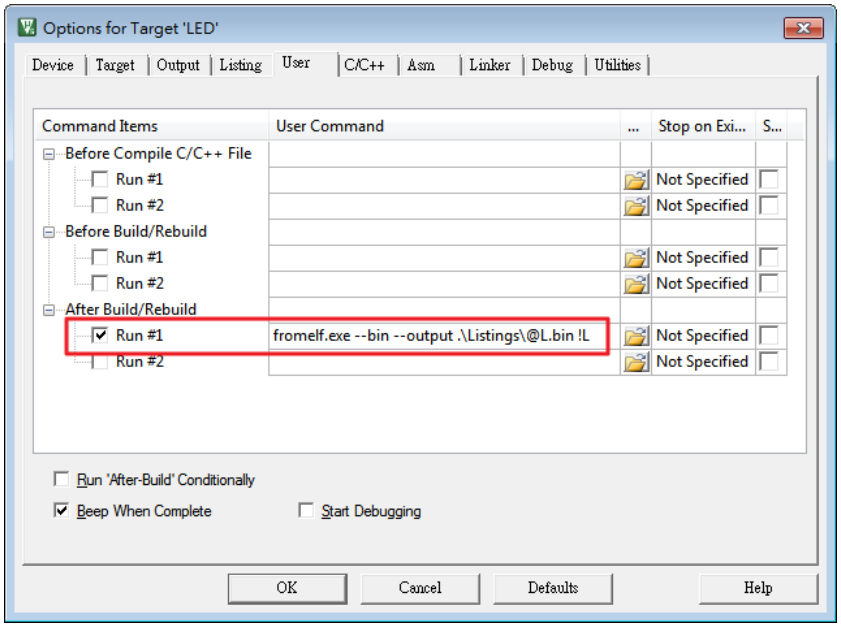
将Flash Start Addr修改为0x08004000，AT32F415RC Flash Size为256KB，填入0x40000

Note: IAP Bootloader预留之Code Size为16KB，剩余可用闪存容量为240KB



-> Options for Target -> User -> After Build/Rebuild -> Run#1

Keil中的User选项卡，设置编译后调用fromelf.exe，在User Command栏位键入fromelf.exe --bin --output .\Listings\@L.bin !L，即可编译生成BIN文档



2） IAP源程序修改

- 将IAP_Init() & IAP_Command_Handle()放入原始代码

```
int main(void)
{
  IAP_Init();
  LED_Init();                 // TIM2中断观察LED Toggle
  Uart_Init(115200);          // 用于与上位机通信 切换到IAP Update状态

  buf1=buffer[25*1024];       // 测试大文件下载

  while(1)
  {
    IAP_Command_Handle();
  }
}
```

- IAP_Init()，若IAP_UPGRADE_FLAG_ADDR为IAP_UPGRADE_FLAG，會擦除此位址

```
void IAP_Init(void)
{
  //SCB->VTOR = FLASH_BASE | 0x000004000;

  if((*(u32*)IAP_UPGRADE_FLAG_ADDR)==IAP_UPGRADE_FLAG)
  {
    FLASH_Unlock();
    FLASH_ErasePage(IAP_UPGRADE_FLAG_ADDR);
    FLASH_Lock();
  }
}
```

- 串口中断接收封包，若为0x5AA5，则IAP_Flag写IAP_REV_FLAG_DONE，IAP开始更新流程

```
void USART1_IRQHandler(void)
{
  u16 Res;

  if(USART_GetITStatus(USART1, USART_INT_RDNE) != RESET)   //接收中断
  {
    Res=USART_ReceiveData(USART1);   //读取接收到的数据

    if((Res==0x5A)&&(IAP_Flag==IAP_REV_FLAG_NO))
      IAP_Flag=IAP_REV_FLAG_5A;
    else if((Res==0xA5)&&(IAP_Flag==IAP_REV_FLAG_5A))
      IAP_Flag=IAP_REV_FLAG_DONE;
    else
      IAP_Flag=IAP_REV_FLAG_NO;
  }
}
```

- IAP_Command_Handle()，若IAP_Flag为IAP_REV_FLAG_DONE，开始更新流程。会先擦除IAP_UPGRADE_FLAG_ADDR，并将IAP_UPGRADE_FLAG写入，随即重启进入IAP_Bootloader

```
void IAP_Command_Handle(void)
{
  if(IAP_Flag==IAP_REV_FLAG_DONE)
  {
    FLASH_Unlock();
    FLASH_ErasePage(IAP_UPGRADE_FLAG_ADDR);
    FLASH_ProgramWord(IAP_UPGRADE_FLAG_ADDR, IAP_UPGRADE_FLAG);
    FLASH_Lock();
    //Back_Ok();              // 向上位机返回OK
    NVIC_SystemReset();       // 跳到IAP区域
  }
}
```

- iap.c文档中IAP_Load_APP副函数上方，有一行#pragma O0代码，功能为使IAP_Load_APP这段程序不进行优化。部份编译器开启优化功能後，会省略跳转程序导致跳转失败；加上#pragma O0，可保证IAP_Load_APP不进行优化以及程序正确性，此行请勿删除

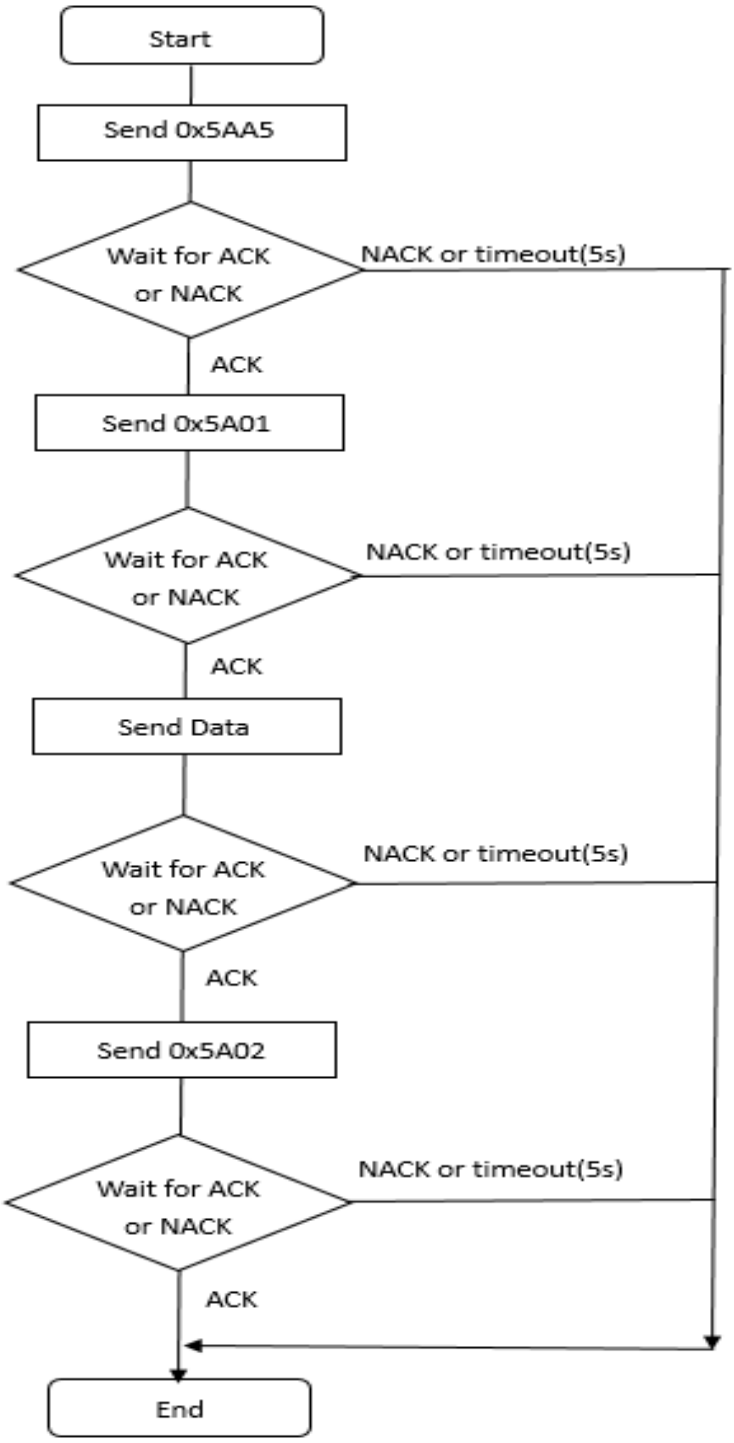
```
#pragma O0
void IAP_Load_APP(u32 appxaddr)
{
  if(((*(vu32*)appxaddr)&0x2FFE0000)==0x20000000)
  {
    jump2app=(iapfun)*(vu32*)(appxaddr+4);
    MSR_MSP(*(vu32*)appxaddr);
    jump2app();
  }
}
```

- APP程序启始位置调整为0x08004000，所以中断向量表的位址也必须一并调整。在system_at32f4xx.c文档的SystemInit()副程序中，请将SCB->VTOR做如下修改

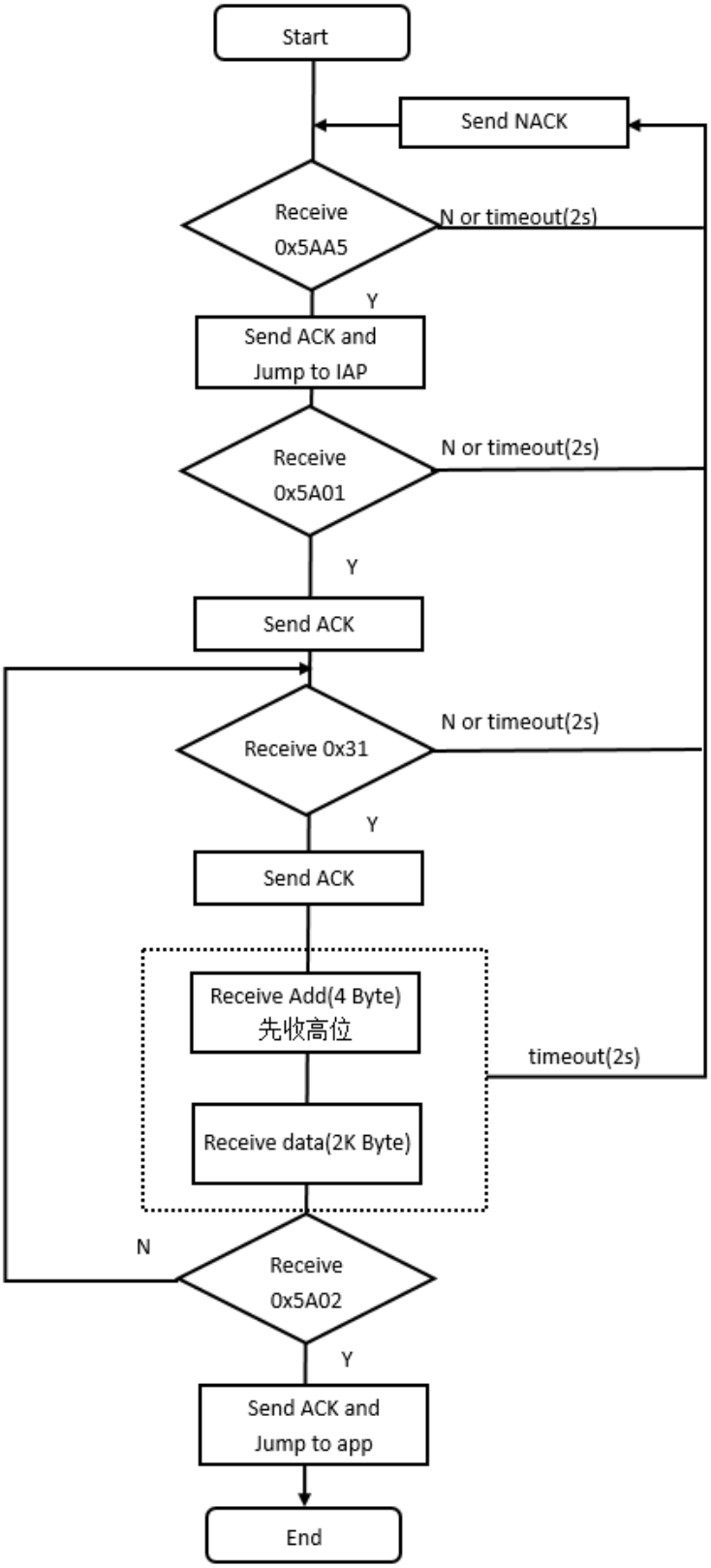
```
#ifdef VECT_TAB_SRAM
  SCB->VTOR = SRAM_BASE | VECT_TAB_OFFSET;
#else
  //SCB->VTOR = FLASH_BASE | VECT_TAB_OFFSET;
  SCB->VTOR = FLASH_BASE | 0x000004000;
#endif
```

3.4 IAP与上位机串口通信协议

1） IAP Programming上位机通信协议



2） IAP_Bootloader下位机通信协议



3） 通讯协议

ACK: 0xCCDD

NACK: 0xEEFF

封包组成: 0x31 + Addr + Data + Chenksum

Addr: 4Bytes，高位在前

最後一笔数据若不足2KB，上位机会自动传送0xFF补足2KB

Checksum: 1Byte，4Bytes的Addr與2KBytes数据加總和的Low-Byte

4 版本历史

日期	版本	变更
2019.4.12	1.0.0	最初版本
2019.4.24	1.1.0	● APP Bin Size缩小为100KB，可直接在xC与xB系列展示 ● 接收Update Flag后之ACK，为避免系统重启造成上位机接收错误封包，改在Bootloader回传 ● SYSCLK更改为192MHz ● Vector Table之位址宣告SCB->VTOR，改在system_at32f4xx.c副程式SystemInit(void)直接定义
2019.7.4	1.2.0	● LED_Init()移至main loop ● 将403/413整合在同一专案档 ● iap.h文档的#define命名，有做修改
2019.8.7	1.2.2	● flash.c改写 ● flash.h定义型号与页容量之对应关系，修正在页容量1KB下更新失败之问题 ● 将403/413/415整合在同一专案档 ● 新增IAR专案档
2020.3.2	1.2.3	● 修正2.1硬件资源的说明，使用AT-Link-EZ时，需使用ATLink-UART串口埠

重要通知 - 请仔细阅读

买方自行负责对本文所述雅特力产品和服务的选择和使用，雅特力概不承担与选择或使用本文所述雅特力产品和服务相关的任何责任。

无论之前是否有过任何形式的表示，本文档不以任何方式对任何知识产权进行任何明示或默示的授权或许可。如果本文档任何部分涉及任何第三方产品或服务，不应被视为雅特力授权使用此类第三方产品或服务，或许可其中的任何知识产权，或者被视为涉及以任何方式使用任何此类第三方产品或服务或其中任何知识产权的保证。

除非在雅特力的销售条款中另有说明，否则，雅特力对雅特力产品的使用和 / 或销售不做任何明示或默示的保证，包括但不限于有关适销性、适合特定用途（及其依据任何司法管辖区的法律的对应情况），或侵犯任何专利、版权或其他知识产权的默示保证。

雅特力的产品不得应用于武器。此外，雅特力产品也不是为下列用途而设计并不得应用于下列用途：（A）对安全性有特别要求的应用，例如：生命支持、主动植入设备或对产品功能安全有要求的系统；（B）航空应用；（C）汽车应用或汽车环境，且 / 或（D）航天应用或航天环境。如果雅特力产品不是为前述应用设计的，而采购商擅自将其用于前述应用，即使采购商向雅特力发出了书面通知，采购商仍将独自承担因此而导致的任何风险，雅特力的产品设计规格明确指定的汽车、汽车安全或医疗工业领域专用产品除外。根据相关政府主管部门的规定，ESCC、QML或JAN正式认证产品适用于航天应用。

经销的雅特力产品如有不同于本文档中提出的声明和 / 或技术特点的规定，将立即导致雅特力针对本文所述雅特力产品或服务授予的任何保证失效，并且不应以任何形式造成或扩大雅特力的任何责任。