摘 要

本技术说明文档提供关于 CSU38F20 固件升级的应用指导，旨在帮助客户通过 IIC 接口在应用编程，用芯海上位机或其它 MCU 主控与 CSU38F20 的 Bootloader 程序通讯，更新 CSU38F20 的 App 程序。

适用范围

类型	适用产品型号或系列	说明
应用笔记	CSU38F20	

版 本

历史版本	修改内容	日期
V1.0	初版生成	2023-05-08
V1.1	格式调整	2023-05-25

目 录

1.概述

1.1 CSU38F20 升级概述

本说明文档介绍 CSU38F20 固件升级的通讯协议，以及阐述协议帧数据定义（ 协议中数据域的部分采用数字加密）。CSU38F20 芯片 ROM 为 8K*16 bits Flash 空间，人为分成了两个区域，Bootloader 区域和 App 区域，其空间地址分配如下图所示，Bootloader 区域为 0000~03FFH，App 区域为 0400~1FFFH，主机（上位机或其它 MCU 主控）通过 IIC 接口给 CSU38F20 发送通讯协议，与 Bootloader 程序进行通讯，把接收到的固件程序，依次写到 App 区域，如果程序已经运行在 App 区域，主机发送复位指令，让芯片复位进入 Bootloader 区域即可，以下为 CSU38F20 Flash 空间分配图：

图 1　CSU38F20 ROM 区域分配

CSU38F20 App 程序开发验证完后，需要在 CSU-IDE 中把 App 程序地址做相应的偏移，目前 Bootloader 程序空间为 1K Words（**汇编语言开发**），App 在 Flash 中存放起始地址为 0x0400, 在 CSU-IDE 中操作方法如下图所示，App 地址偏移后需重新编译程序：

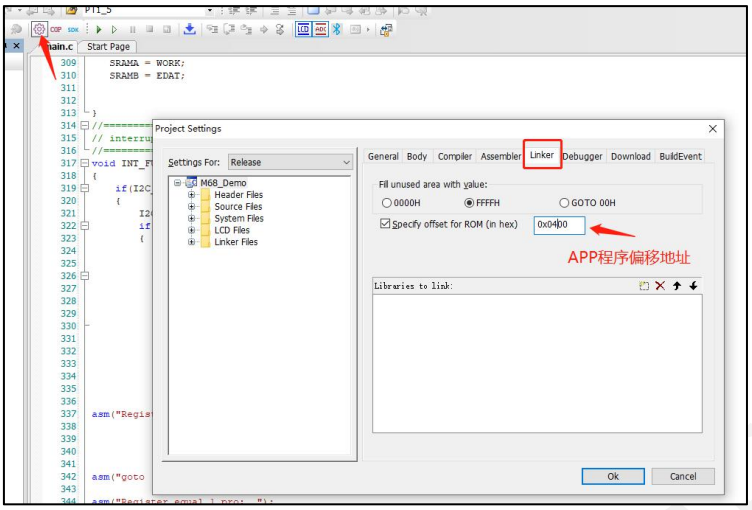

图 2　CSU38F20 ROM 区域分配

1.2 CSU38F20 升级流程

芯片正常上电或复位后，将由 BootLoader 引导启动，启动流程如下：

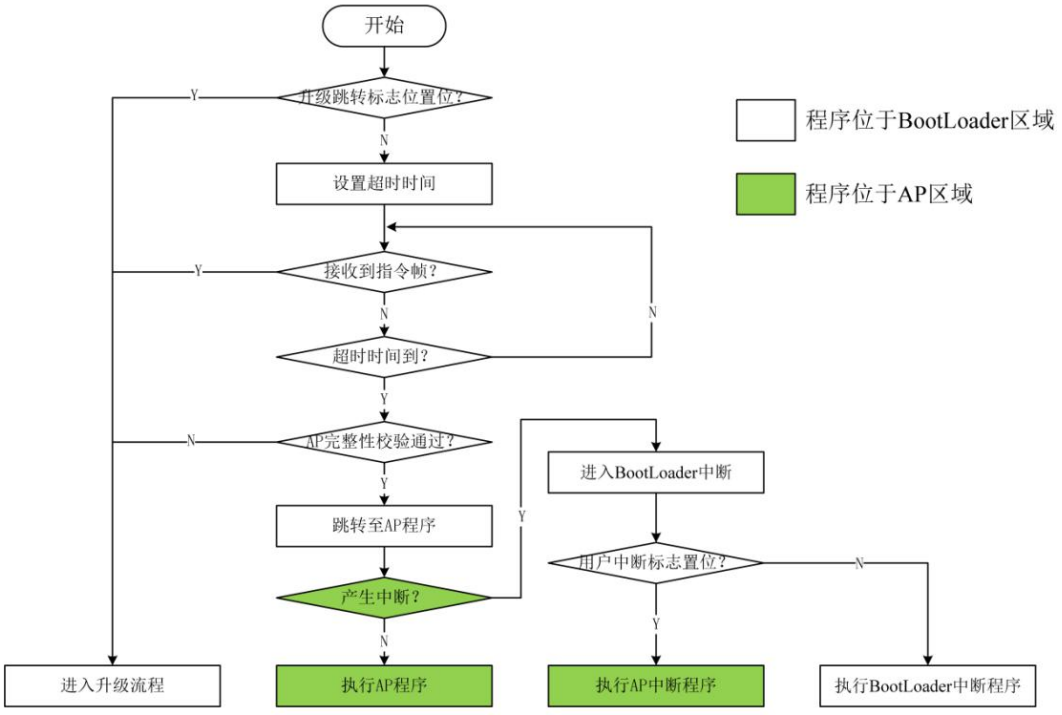

图 3　CSU38F20 固件升级流程

整体升级流程如下：

图 4　CSU38F20 固件升级流程

在由 BootLoader 跳转至 AP 前，将会对 AP 固件的完整性进行检查，在进入升级模式后，相应的完整性标志位被清除，接收到“结束升级固件”命令后将会根据帧内容刷新标志位。

2.CSU38F20 升级协议

2.1 帧格式

协议帧根据传输方向可分为两类：

第一类为上位机下发至芯片端的指令帧，其帧格式如下表所示：

Head	Length	Command	Reserve	Data	CK
0xAA	包长度	命令码	0x00	数据	帧校验码

第二类协议帧为芯片端上传至上位机的响应帧，其帧格式如下表所示：

Head	Length	Command	Execute	Data	CK
0xAA	包长度	命令码	执行状态	数据	帧校验码

注意：

❖ *Length 为 2 字节，指明协议帧的整体长度（包含帧校验码），其低字节在前，高字节固定为 0；*

❖ *Data 字段在不同命令帧中字节数不同（可以不包含该字段），上位机下发时需先对 Data 字段加密，再计算帧校验值；下位机接收到数据后先进行帧校验，后再对 Data 字段（若存在）解密使用；*

❖ *CK 为单字节，其值为以 Head 为首、长度为（Length - 1）的所有数据累加和不计溢出。*

2.2 命令类型表

表 1　命令类型表

序　号	名　称	命令码	命令描述
1	开始升级固件	0x01	下位机收到命令后，需进行初始化动作
2	升级固件数据	0x02	下位机收到命令后，需将数据写入
3	结束升级固件	0x03	下位机收到命令后，结束升级动作
4	跳转命令	0x5A	下位机收到命令后，需从AP跳转至BOOT，或从BOOT跳转至AP
5	设备识别命令	0xA5	上位机发送设备识别命令，相应的下位机设备进行应答

2.3 执行状态表

表 2　执行状态表

序　号	名　称	状态码值	状态描述
1	命令执行成功	0x00	命令执行成功
2	校验码错误	0x01	校验码错误
3	命令类型错误	0x02	不支持的命令类型
4	升级模式错误	0x03	设备不处于升级模式，无法升级固件
5	写 ROM 数据失	0x04	无法写入 ROM 数据
6	未知错误	0x05	未知错误

3.CSU38F20 升级主要协议示范

3.1 开始升级固件命令

芯片端接收到该命令后将会从后向前擦除整个 AP 区域数据（包含保留区域数据），进入升级模式，允许上位机向以 0x0400 为起始地址的 AP 区域写入数据，并解除完整性校验失败状态，在固件刷新后可再次校验跳转。执行完成后向上位机返回可允许的分段长度（用以指明“升级固件数据命令”帧中包含的固件数据长度）。

通信帧示例：

上位机下发数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5	Byte6
Head	Length	Command	Reserve	Data	CK
0xAA	0x0007	0x01	0x00	操作类型	

注：

- *Length 数据低字节在前，高字节在后并设置为 0；*
- *操作类型固定是 0x01（表示目标区域为 MCU 程序区），需加密后发送。*

芯片端回复数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5~6	Byte7
Head	Length	Command	Execute	Data	CK
0xAA	0x0008	0x01	0x00	分段长度	

注：

- *由于芯片使用 PageLatch 按页写入 ROM 数据，因此分段长度固定为 0x0040（64 字节长度），低字节在前，加密后回复。*

3.2 升级固件数据命令

芯片端接收到该命令后将会从第 31 页（地址范围 0x0400～0x041F）开始，按页递增的方式逐页写入数据，注意每一帧命令都要包含一页（即 64 字节）固件数据。

通信帧示例：

上位机下发数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5	Byte6~9	Byte10~11	Byte12~75	Byte76
Head	Length	Command	Reserve	Data	Data	Data	Data	CK
0xAA	0x004D	0x02	0x00	操作类型	Flash 地址	固件数据长度	固件数据	

注：

- *操作类型固定是 0x01（表示目标区域为 MCU 程序区），需加密后发送；*
- *芯片端不处理 Flash 地址（芯片按页写数据成功后即跳转至下一页），固件数据长度需为 0x0040，低字节在前，上位机可使用该字段记录已下发固件数据的信息，需加密后发送。*

芯片端回复数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5
Head	Length	Command	Execute	CK
0xAA	0x0006	0x02	0x00	0xB2

3.3 结束升级固件命令

芯片端接收到该命令后将会禁止上位机向 AP 区域写入数据，并根据帧信息更新 AP Checksum 以及固件状态信息。

通信帧示例：

上位机下发数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5	Byte6~9	Byte10~13	Byte14	Byte15
Head	Length	Command	Reserve	Data	Data	Data	Data	CK
0xAA	0x0010	0x03	0x00	操作类型	Checksum	代码长度	固件状态	

注：

- *操作类型固定是 0x01（表示目标区域为 MCU 程序区），需加密后发送；*
- *Checksum 是由上位机对 hex 数据进行 CRC 计算得到，芯片端不进行复验，只作为信息保存；*
- *芯片端不处理代码长度字段；*
- *固件状态可选参数为：*
  0x5A -- 固件刷新完成，允许跳转
  0xFF -- 固件未刷新完成，不允许跳转

芯片端回复数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5
Head	Length	Command	Execute	CK
0xAA	0x0006	0x03	0x00	0xB3

3.4 跳转命令

AP 接收到跳转命令后可通过看门狗复位、硬件复位等方式进入 BOOT，在 BOOT 中接收到此命令将会校验 AP 数据的完整性并执行跳转。

通信帧示例：

上位机下发数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5	Byte6
Head	Length	Command	Reserve	Data	CK
0xAA	0x0007	0x5A	0x00	跳转目标	

注：

- *跳转目标可选参数为：*
  0x5A -- 跳转至 AP
  0xFF -- 跳转至 BOOT

芯片端回复数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5
Head	**Length**	**Command**	**Execute**	**CK**
0xAA	0x0006	0x5A	0x00	0x0A

3.5 设备识别命令

该命令用于获取当前设备的固件信息，包括 BootLoader 版本号、AP 版本号及 Checksum 信息。为保护不同用户数据，读取信息需核对秘钥（即厂商信息），只有验证通过芯片端才会上报固件信息。

通信帧示例：

上位机下发数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5~12	Byte13
Head	**Length**	**Command**	**Reserve**	**Data**	**CK**
0xAA	0x000E	0xA5	0x00	秘钥	

注：

- *秘钥固化在 0x03E8 地址处，可在 BootLoader isp.asm 文件中修改。*

芯片端回复数据：

Byte0	Byte1~2	Byte3	Byte4	Byte5~8	Byte9~12	Byte13~40	Byte41	Byte42	Byte43	Byte44	Byte45
Head	**Length**	**Command**	**Execute**	**Data**	**Data**	**Data**	**Data**	**Data**	**Data**	**Data**	**CK**
0xAA	0x002E	0x5A	0x00	reserve	Checksum	reserve	AP 版本号	BOOT 版本号	设备类别	执行区域	

注：

- *Checksum 低字节在前，读取自 0x1FF1、0x1FF2 地址处；*
- *设备类别固定是 0x00（表示为终端设备）；*
- *执行区域可选参数为：*
  0x0A -- 程序运行在 AP 区域
  0x0B -- 程序运行在 BOOT 区域

4.Bootloader 使用注意事项

❖CSU38F20 的 IIC 接口为 PT5.2 （SDA）/PT5.3 （SCL），从机地址为 0x26。

❖如果芯片已有 App 程序，芯片正常上电，在 Bootloader 里运行时间为 2 秒，2 秒内未发送升级命令，自动跳到 App 程序里。

❖ 芯片复位后，通过判断 R_APToBoot1/R_APToBoot2 寄存器值为 0x03/0x09 进入 IAP 升级。

❖ IIC 通讯超时时间为 500 毫秒。

❖ 在中断程序中，通过判断 R_SysFlag 寄存器的 B_Int_Flag 位，来进行中断程序的处理，B_Int_Flag 为 0 时，进入 Bootloader 中断程序，B_Int_Flag 为 1 时，进入 App 中断程序，如下图所示：

```
;==========================================
; 中断处理函数
;==========================================
INT_FUNCTION_sec .section rom   ;
INT_FUNCTION:
    push                        ;将WORK和STATUS压栈处理
    movfw   BSR
    push                        ;将BSR压栈处理
    clrf    BSR
    btfss   R_SysFlag,B_Int_Flag ;中断入口标志为0标志BOOT中断，为1表示用户中断
    goto    L_INT_Timer0
    pop
    movwf   BSR                 ;恢复保护的BSR
    pop
    goto    0404h               ;跳转至用户中断入口

L_INT_Timer0:
    btfss   INTF,TM0IF          ;是否为定时器0中断
    goto    L_INT_IIC
    bcf     INTF,TM0IF
    goto    L_INT_Timer0_Deal
```

图 5　CSU38F20 Bootloader 程序中断跳转

❖上位机发送数据中数据域字段有加密，Bootloader 程序需要对接收到数据域的数据进行解密，解密时，需依次读取 Bootloader 0x03C0 地址的数据，与数据域的数据进行异或，数据域有几个字节，在 0x03C0 地址依次读取同样字节数据进行一一异或。

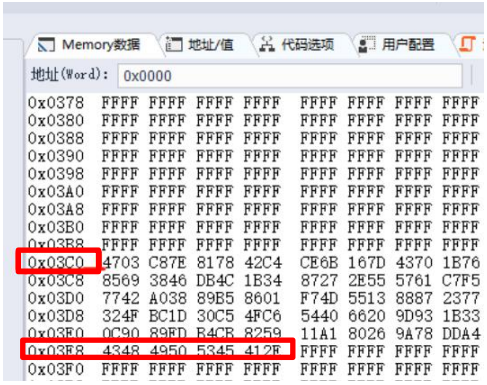

图 6　Bootloader 0x03C0 地址存放数据

注意：0x03E8 地址为厂商 ID，共 8 个字节

❖若 CSU38F20 只有 Bootloader 没有 App 程序时，上位机发送设备识别命令时，CSU38F20 应答的数据域全为 0xFF。

❖执行设备识别命令时，下位机会对秘钥（即厂商 ID 信息，8 个字节）进行校对，秘钥存放起始地址为 03E8h，长度为 8 字节，只有校对成功，才可获取升级权限。

❖CSU38F20 的 App 程序升级不成功或完整性校验不成功，程序将一直在 Bootloader 里运行。

❖主机发送固件升级数据帧后，CSU38F20 应答的间隔时间约为 25mS，主机再次发送固件升级数据帧的间隔时间约为 4mS，供参考。

图 7　固件升级过程数据帧间隔示意图

5.App 程序注意事项

程序包括 IIC 初始化设置、IIC 读写以及异常处理、IAP 协议执行、ISP 读操作。开发过程中用户程序需修改地方：

❖在“CSU38F20.h”和“SysRegDefine.c”文件中增加 IAP 寄存器定义:

```
#define  _CSS38F20_H 1

extern volatile unsigned char bank0 APToBoot1              @ 0xfc;
extern volatile unsigned char bank0 APToBoot2              @ 0xfd;
extern volatile sbit bank0 flag_INT                        @ 0x80*8+0;
```

图 8　CSU38F20 H 文件特殊寄存器定义

```
//=============================================

 volatile unsigned char bank0 APToBoot1              @ 0xfc;
 volatile unsigned char bank0 APToBoot2              @ 0xfd;
 volatile sbit bank0 flag_INT                        @ 0x80*8+0;
```

图 9　CSU38F20 SysRegDefine.c 文件中特殊寄存器定义

❖ “BootLoader.c”和“BootLoader.h”文件中为 IAP 所用到的变量及标志位，直接添加到 App 程序中即可。

❖ “isp.c”和“isp.h”为 ROM 读取函数，用于读取保留信息区中的内容。

❖App 信息区地址起始为 0x1FF0，用户在 main 程序后添加即可。升级时，结束命令会选择升级是否完成，如果升级完成，上位机发送命令帧数据会包含 0x5A，未升级完成则发送 OxFF，下位机对这个数据不做修改，直接写进信息区，App 升级结束跳转前对 0x1FD5 这个数据进行校验。

```
asm("org 0x1FF0");      //AP信息区
asm("dw 0x0001");       //AP版本号
asm("dw 0x0000");       //CHECKSUM低位
asm("dw 0x0000");       //CHECKSUM高位
asm("dw 0x0000");       //保留
asm("dw 0x0000");       //保留
asm("dw 0x5A00");       //升级完成标志，用于合并hex时，此地址写0x5A00，用于升级文件时，此地址写0x00h
asm("dw 0x0000");       //保留
asm("dw 0x0000");       //保留
asm("dw 0x0000");       //保留
asm("dw 0x0000");       //保留
asm("dw 0x0000");       //保留
asm("dw 0x0000");       //保留
asm("dw 0x0000");       //保留
asm("dw 0x0000");       //保留
asm("dw 0x0000");       //保留
```

图 10　App 信息区数据

❖ 主程序及中断函数中，模拟用户程序为定时翻转 PT1.0 口，其余程序为 IAP 所需程序，用户直接复制添加即可。

6 附录

6.1CSU38F20 Bootloader 与 App 合成工具

此工具把 Bootloader 程序和 App 合成一个 HEX，可直接通过芯海烧录器写入 CSU38F20 芯片。

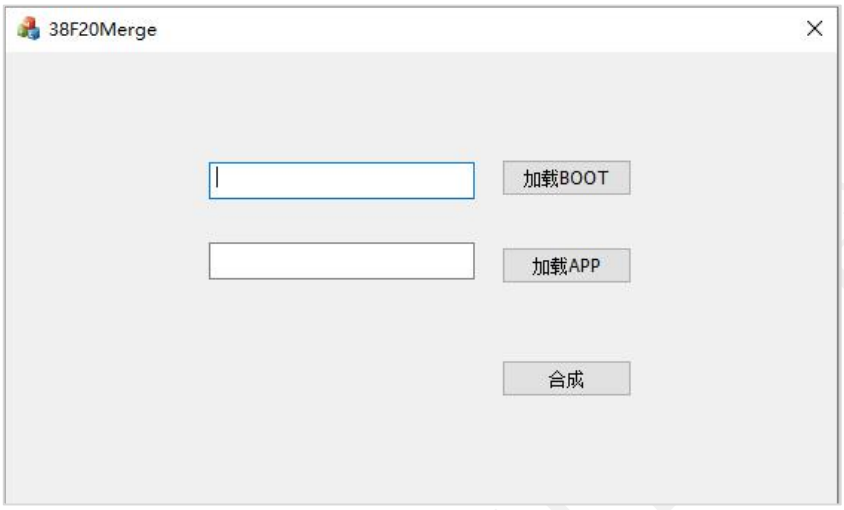

图 11　Bootloader 和 App 合成软件

6.2CSU38F20 上位机升级软件

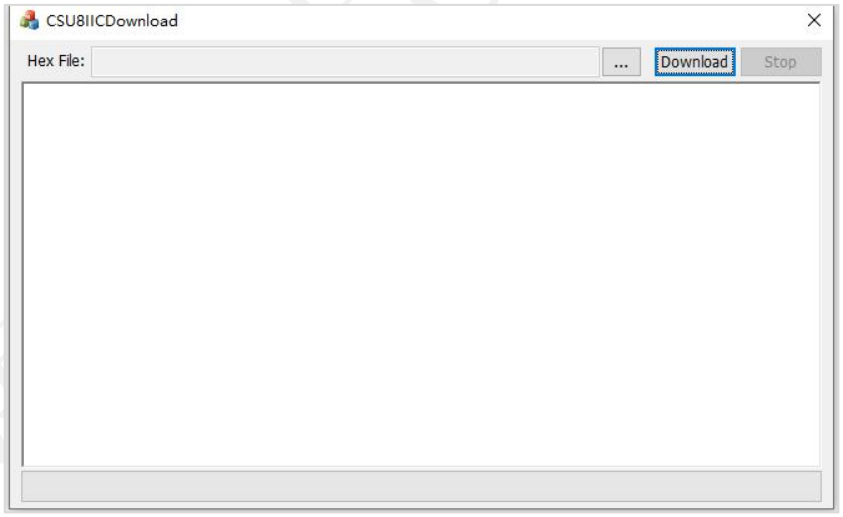

图 12　CSU38F20 上位机升级软件

6.3 上位机升级配套的工具

图 13　USB-IIC 工具

注意：此工具客户需自购，如果用其它 MCU 对 CSU38F20 升级，不需要采购此工具。

免责声明和版权公告

本文档中的信息，包括供参考的 URL 地址，如有变更，恕不另行通知。

本文档可能引用了第三方的信息，所有引用的信息均为“按现状”提供，芯海科技不对信息的准确性、真实性做任何保证。

芯海科技不对本文档的内容做任何保证，包括内容的适销性、是否适用于特定用途，也不提供任何其他芯海科技提案、规格书或样品在他处提到的任何保证。

芯海科技不对本文档是否侵犯第三方权利做任何保证，也不对使用本文档内信息导致的任何侵犯知识产权的行为负责。本文档在此未以禁止反言或其他方式授予任何知识产权许可，不管是明示许可还是暗示许可。

Wi-Fi 联盟成员标志归 Wi-Fi 联盟所有。蓝牙标志是 Bluetooth SIG 的注册商标。

文档中提到的所有商标名称、商标和注册商标均属其各自所有者的财产，特此声明。