远程升级协议及方案

V1.4版

|  |  |  |
| --- | --- | --- |
| 版本号 | 说明 | 修改人 |
| V1.0 | 初版 | 高屹 |
| V1.1 | 新增复旦微Boot协议方案 | 高屹 |
| V1.2 | 新增升级数据回复命令2020.2.5 | 高屹 |
| V1.3 | 新增复旦微协议头改为A6 | 高屹 |
| V1.4 | 新增差分升级协议头A7 | 高屹 |

# 一、网络端协议格式

|  |  |  |
| --- | --- | --- |
| 内容 | 长度 | 说明 |
| 协议头 | 1 Bety | 固定0xA5(430)/0xA6(复旦微)/0Xa7(差分升级) |
| 协议总长度 | 2 Bety | 低字节在前 |
| 表号 | 4 Bety | 低字节在前 |
| 数据 | 变长 |  |
| 校验和 | 1 Bety | 从协议头开始累加到校验和之前 |
| 协议尾 | 1 Bety | 固定0x5A |

注：全部数据为HEX

# 二、数据内容

## 1.开始升级

### 1.1.1服务器下行（MSP430/复旦微）

|  |  |  |  |
| --- | --- | --- | --- |
| 内容 | | 长度 | 说明 |
| 协议头 | | 1 Bety | 固定0xA5(430)/0xA6(复旦微) |
| 协议总长度 | | 2 Bety | 低字节在前 |
| 表号 | | 4 Bety | 低字节在前 |
| 数据 | 命令 | 1 Bety | 固定0X10 |
| 版本号 | 1 Bety | 0-0xFF |
| 总包数 | 2 Bety | 低字节在前 |
| 分包数 | 2 Bety | 低字节在前，固定0 |
| 校验和 | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | 1 Bety | 固定0x5A |

### 1.1.2服务器下行（差分升级）

|  |  |  |  |
| --- | --- | --- | --- |
| 内容 | | 长度 | 说明 |
| 协议头 | | 1 Bety | 固定0xA7(差分升级) |
| 协议总长度 | | 2 Bety | 低字节在前 |
| 表号 | | 4 Bety | 低字节在前 |
| 数据 | 命令 | 1 Bety | 固定0X10 |
| 版本号 | 1 Bety | 0-0xFF |
| 其他信息 | 10 Bety | 用于区分不同表的校验信息 |
| 总包数 | 2 Bety | 低字节在前 |
| 分包数 | 2 Bety | 低字节在前，固定0 |
| 校验和 | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | 1 Bety | 固定0x5A |

### 1.2.1表端应答（MSP430/复旦微）

|  |  |  |  |
| --- | --- | --- | --- |
| 内容 | | 长度 | 说明 |
| 协议头 | | 1 Bety | 固定0xA5(430)/0xA6(复旦微) |
| 协议总长度 | | 2 Bety | 低字节在前 |
| 表号 | | 4 Bety | 低字节在前 |
| 数据 | 响应 | 1 Bety | 1. 成功0x11，可以下发数据包； 2. 失败0x12，重发前命令包 3. 请求断点重传0x13，下发对应分包的数据包； 4. 未知命令错误0x1A，重发前命令包 5. 校验和错误0x1B，重发前命令包 6. 表端放弃升级0xCC   注：如果回复任何失败或错误码连续3次则表端自动放弃升级 |
| 版本号 | 1 Bety | 0-0xFF |
| 总包数 | 2 Bety | 低字节在前 |
| 分包数 | 2 Bety | 低字节在前，表端回复已接收到的对应版本号已存的分包数 |
| 校验和 | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | 1 Bety | 固定0x5A |

### 1.2.2表端应答（差分升级）

|  |  |  |  |
| --- | --- | --- | --- |
| 内容 | | 长度 | 说明 |
| 协议头 | | 1 Bety | 固定0xA7(差分升级) |
| 协议总长度 | | 2 Bety | 低字节在前 |
| 表号 | | 4 Bety | 低字节在前 |
| 数据 | 响应 | 1 Bety | 1. 成功0x11，可以下发数据包； 2. 失败0x12，重发前命令包 3. 请求断点重传0x13，下发对应分包的数据包； 4. 未知命令错误0x1A，重发前命令包 5. 校验和错误0x1B，重发前命令包 6. 表端放弃升级0xCC   注：如果回复任何失败或错误码连续3次则表端自动放弃升级 |
| 版本号 | 1 Bety | 0-0xFF |
| 其他信息 | 10 Bety | 用于区分不同表的校验信息 |
| 总包数 | 2 Bety | 低字节在前 |
| 分包数 | 2 Bety | 低字节在前，表端回复已接收到的对应版本号已存的分包数 |
| 校验和 | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | 1 Bety | 固定0x5A |

## 2.升级数据包

### 2.1.1服务器下行(用于MSP430MCU)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 内容 | | | 长度 | 说明 |
| 协议头 | | | 1 Bety | 固定0xA5(430) |
| 协议总长度 | | | 2 Bety | 低字节在前 |
| 表号 | | | 4 Bety | 低字节在前 |
| **数据** | 命令 | | 1 Bety | 固定0X20 |
| 版本号 | | 1 Bety | 0-0xFF |
| 总包数 | | 2 Bety | 低字节在前 |
| 分包数 | | 2 Bety | 低字节在前  从0开始计数 |
| Boot包长度 | | 2 Bety | 标红(Boot协议)的总长度 |
| Boot协议 | 协议头 | 1 Bety | 固定0x80 |
| 长度 | 2 Bety | 低字节在前（紫字部分长度） |
| 命令码 | 1 Bety | 固定0x10 |
| 数据地址 | 3 Bety | 低字节在前  AL ; AM ; AH |
| 数据 | 变长(1-240 )Bety | 分包数据最大0xF0 |
| CRC16 | 2 Bety | 低字节在前 |
| 校验和 | | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | | 1 Bety | 固定0x5A |

注:Boot协议根据不同bootloader可能发生改变

注:使用本Boot协议连同标绿底的数据最大长度为256，正好写满Flash一页，方便Flash程序设计

### 2.1.2服务器下行(用于复旦微MCU)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 内容 | | | 长度 | 说明 |
| 协议头 | | | 1 Bety | 固定0xA6(复旦微) |
| 协议总长度 | | | 2 Bety | 低字节在前 |
| 表号 | | | 4 Bety | 低字节在前 |
| **数据** | 命令 | | 1 Bety | 固定0X20 |
| 版本号 | | 1 Bety | 0-0xFF |
| 总包数 | | 2 Bety | 低字节在前 |
| 分包数 | | 2 Bety | 低字节在前  从0开始计数 |
| Boot包长度 | | 2 Bety | 标红(Boot协议)的总长度 |
| Boot协议 | 数据地址 | 4 Bety | 低字节在前  AL ; AML; AMH ; AH |
| 数据 | 变长(1-256 )Bety | 分包数据最大0x100 |
| CRC16 | 2 Bety | 低字节在前，从boot协议数据地址开始算直到最后一个数据 |
| 校验和 | | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | | 1 Bety | 固定0x5A |

### 2.1.3服务器下行(用于差分升级)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 内容 | | | 长度 | 说明 |
| 协议头 | | | 1 Bety | 固定0XA7 |
| 协议总长度 | | | 2 Bety | 低字节在前 |
| 表号 | | | 4 Bety | 低字节在前 |
| **数据** | 命令 | | 1 Bety | 固定0X20 |
| 版本号 | | 1 Bety | 0-0xFF |
| 总包数 | | 2 Bety | 低字节在前 |
| 分包数 | | 2 Bety | 低字节在前  从0开始计数 |
| 数据包长度 | | 2 Bety | 标红的总长度 |
|  | 是否有后续 | 1 Bety | 由于NB限制，只能下发512以下的数据量，而差分数据包可能大于512，因此对于大于512的数据包需要分包  1：有后续  0：无后续 |
| 数据 | 变长 | 差分包数据 |
| 校验和 | | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | | 1 Bety | 固定0x5A |

### 2.2表端应答

|  |  |  |  |
| --- | --- | --- | --- |
| 内容 | | 长度 | 说明 |
| 协议头 | | 1 Bety | 固定0xA5(430)/0xA6(复旦微)/0Xa7(差分升级) |
| 协议总长度 | | 2 Bety | 低字节在前 |
| 表号 | | 4 Bety | 低字节在前 |
| 数据 | 响应 | 1 Bety | 1. 成功0x21，可以下发后续包； 2. 失败0x22：重发当前分包。 3. 错误0x23：若回复写错误命令则该次升级完全失败，因为Flash写错任何一个字节都可能导致程序出现不可预知的错误，表端自动放弃升级。需要重发0x10开始升级命令。 4. 分包数不匹配:0x24,按照表端的分包数进行重发。 5. 表内版本号，总包数不等:0x25，重发当前分包 6. 升级包与当前APP处于同一地址，升级失败：0x26（为复旦微新增）返回该命令表端自动放弃升级 7. 未知命令0x2A，重发当前分包 8. 校验和错误0X2B，重发当前分包 9. 表端放弃升级0xCC   注：如果回复任何失败或错误码连续3次则表端自动放弃升级 |
| 版本号 | 1 Bety | 0-0xFF |
| 总包数 | 2 Bety | 低字节在前 |
| 分包数 | 2 Bety | 低字节在前  注：分包数最大值与总包数差1 |
| 校验和 | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | 1 Bety | 固定0x5A |

## 3.升级确认/取消升级

### 3.1服务器下行

|  |  |  |  |
| --- | --- | --- | --- |
| 内容 | | 长度 | 说明 |
| 协议头 | | 1 Bety | 固定0xA5(430)/0xA6(复旦微)/0Xa7(差分升级) |
| 协议总长度 | | 2 Bety | 低字节在前 |
| 表号 | | 4 Bety | 低字节在前 |
| 数据 | 命令 | 1 Bety | 固定0X30 |
| 版本号 | 1 Bety | 0-0xFF |
| 总包数 | 2 Bety | 低字节在前 |
| 分包数 | 2 Bety | 低字节在前 |
| 校验和 | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | 1 Bety | 固定0x5A |

### 3.2表端应答

|  |  |  |  |
| --- | --- | --- | --- |
| 内容 | | 长度 | 说明 |
| 协议头 | | 1 Bety | 固定0xA5(430)/0xA6(复旦微) |
| 协议总长度 | | 2 Bety | 低字节在前 |
| 表号 | | 4 Bety | 低字节在前 |
| 数据 | 响应 | 1 Bety | 1. 分包等于总包，成功0x31，不在发送数据，等待表端进行升级； 2. 表内分包数与总包数不等升级失败0x32 3. 未知命令0x3A，重发当前命令 4. 校验和错误0X3B，重发当前命令 5. 表端放弃升级0xCC   注：最后一包不在校验服务器下行的版本号，总包数，分包数  注：如果回复任何失败或错误码连续3次则表端自动放弃升级 |
| 版本号 | 1 Bety | 0-0xFF |
| 总包数 | 2 Bety | 低字节在前 |
| 分包数 | 2 Bety | 低字节在前  数值等于总包数 |
| 校验和 | | 1 Bety | 从协议头开始累加到最后一个数据 |
| 协议尾 | | 1 Bety | 固定0x5A |

# 三、表端升级方案

## 表端Flash操作

当表端接收到完整的开始升级命令时，判断是否有断点数据，如果没有，则直接清Flash并回复成功/失败。如果有，则请求断点数据。

当表端接收到完整的升级数据包命令时，将升级数据包命令的数据全部存入Flash中，使用Flash连续写函数判断是否成功，并回复成功/错误/失败。

当表端接收到完整的升级确认命令时，判断写入Flash的包数是否等于总包数，若相等回复成功然后跳转Bootloader，若不等回复失败则该次升级包写入完全失败需要重新写入升级包

# 四、复旦微解包方法

数据格式为：llaaaatt[dd...]cc

ll=长度

aaaa=地址域

tt=记录类型

数据类型只有可能是如下6种：

00-数据记录

01-文件结束记录

02-扩展段地址记录

03-开始段地址记录

04-扩展线性地址记录

05-开始线性地址记录

dd=数据

cc=校验和

校验和公式：1+NOT（ll+aaaa+tt+dd...） 例：01h + NOT(02h + 00h + 00h + 04h + FFh + FFh)

关于checksum的算法

checksum = 0xFF - (从count段开始所有字节的总和) + 1。

1.3.1 数据记录

Intel HEX文件由任意数量以回车换行符结束的数据记录组成.

数据记录外观如下:

:10246200464C5549442050524F46494C4500464C33

其中:

10 是这个记录当中数据字节的数量.

2462 是数据将被下载到存储器当中的地址.

00 是记录类型(数据记录)

464C…464C是数据.

33 是这个记录的校验和.

1.3.2 扩展线性地址记录(HEX386)

扩展线性地址记录也叫作32位地址记录或HEX386记录.

这些记录包含数据地址的高16位.

扩展线性地址记录总是有两个数据字节,外观如下:

:02000004FFFFFC

其中:

02 是这个记录当中数据字节的数量.

0000 是地址域,对于扩展线性地址记录,这个域总是0000.

04 是记录类型 04(扩展线性地址记录)

FFFF 是地址的高16位.

FC 是这个记录的校验和,计算方法如下: 01h + NOT(02h + 00h + 00h + 04h + FFh + FFh).

当一个扩展线性地址记录被读取,存储于数据域的扩展线性地址被保存,它被应用于从Intel HEX文件读取来的随后的记录.

线性地址保持有效,直到它被另外一个扩展地址记录所改变.

通过把记录当中的地址域与被移位的来自扩展线性地址记录的地址数据相加获得数据记录的绝对存储器地址.

以下的例子演示了这个过程..

来自数据记录地址域的地址（2462）+ 扩展线性地址记录的数据域（FFFF） = 绝对存储器地址（FFFF2462）。

1.3.3 扩展段地址记录(HEX86) 扩展段地址记录也叫HEX86记录,它包括4-19位数据地址段.

扩展段地址记录总是有两个数据字节,外观如下:

:020000021200EA

其中:

02 是记录当中数据字节的数量.

0000 是地址域.对于扩展段地址记录,这个域总是0000.

02 是记录类型 02(扩展段地址记录)

1200 是地址段.

EA 是这个记录的校验和,计算方法如下: 01h + NOT(02h + 00h + 00h + 02h + 12h + 00h).

当一个扩展段地址记录被读取,存储于数据域的扩展段地址被保存,它被应用于从Intel HEX文件读取来的随后的记录.

段地址保持有效,直到它被另外一个扩展地址记录所改变.

通过把记录当中的地址域与被移位的来自扩展段地址记录的地址数据相加获得数据记录的绝对存储器地址.

以下的例子演示了这个过程..

来自数据记录地址域的地址（2462）+ 扩展段地址记录数据域（1200）= 绝对存储器地址（00014462）。

1.3.4 文件结束(EOF)记录

Intel HEX文件必须以文件结束(EOF)记录结束.

这个记录的记录类型域的值必须是01.

EOF记录外观总是如下:

:00000001FF

其中:

00 是记录当中数据字节的数量.

0000 是数据被下载到存储器当中的地址. 在文件结束记录当中地址是没有意义被忽略的.0000h是典型的地址.

01 是记录类型 01(文件结束记录)

FF 是这个记录的校验和,计算方法如下: 01h + NOT(00h + 00h + 00h + 01h).