# 一、上位机握手协议结构说明

整段协议称作“一帧”，在demo中称作frame（后文记作frame），每个frame都有自己的帧头和帧尾，不同的frame之间是完全独立的，每个frame只能携带一类数据。

**一个frame的结构：**

|  |  |  |
| --- | --- | --- |
| 帧头 | 数据帧 | 帧尾 |

## 1.1、帧头结构

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | Byte | 功能 | 说明 |
| 0 | **0x3F** | **上位机接收的地址** | **——** |
| 1 | **0x0-** | **本帧携带的数据类型** | **0x01：**传感器/用户数据；  **0x02：**来自01接口的转发数据；  **0x03：**来自02接口的转发数据；（0x02和0x03在陀螺仪上有效，对上位机无效）；  **0x04：**日志数据； |
| 2 | **0x--** | **本帧的数据长度** | **请保证frame最大大小不超过255** |
| 3 | **0x--** | **计数器低8位** | 这是一个不断自增的计数器，用以确保波形与上传同步； |
| 4 | **0x--** | **计数器高8位** |
| 5 | **0x--** | **CRC8** | **——** |

## 1.2、数据帧结构

数据帧依照载入顺序排列，在这里只对消息类型是传感器/用户数据的frame的数据帧结构进行说明：

frame的数据帧中，首先有一个count(1 byte)记录有几个“数据包”，在demo中称作“packet”（后文记作packet），同一个frame中至少存在1个，最多存在20（默认）个packet：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| count | Packet0 | Packet1 | Packet2 | …… | Packetn |

每个packet可装载同样大小的一组数据，上传几“组”数据就有几个packet，不同packet则其按照添加顺序排列，不同packet之间也许数据长度不一致，但每个packet的结构是一致的。（协议这么设计的用意是：相同大小且相似用途的数据放在一个packet中，不同大小的数据放在不同的packet中，用利于在有限的数据带宽中提升字节利用率）

**单个packet的结构：**

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | Byte | 功能 | 说明 |
| 0 | **0x--** | **本组数据对应的基地址** | **基地址决定了波形在上位机显示的是从哪条曲线，**如本pakcet内有4个数据，基地址设为0，则4个数据分别对应曲线0、1、2、3；  **[注]：**v2.2.5的版本的上位机最多支持32个曲线显示，请不要超过这个限制。 |
| 1 | **0x--** | **包含了本组数据的配置信息** | **Bit0—Bit1：**单个数据的**大小**（字节）对2取对数，如一个数据4字节，则此处为0x02<<0；  **Bit2—Bit5：**packet中同类型数据的**个数**，如有m个数据，则此处为0x0m<<2；  **Bit6—Bit7：**数据特性（暂未开放） |
| 2 | **n\*Byte** | **数据0** | **单个数据占n个字节大小，同一个packet内的数据要确保大小一致！** |
| 3 | **n\*Byte** | **数据1** |
| … | **…** | **…** |
| n\*m+2 | **n\*Byte** | **数据m** |

## 1.3、帧尾结构

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | Byte | 功能 | 说明 |
| 0 | **0x--** | **CRC16低8位** | **——** |
| 1 | **0x--** | **CRC16高8位** | **——** |

# Demo使用说明

## 文件说明

Demo主要包含以下文件：



其中：

driver\_crc.h/c是crc校验驱动；如果工程中已有请用自己的文件进行替换；custom\_broadcast.h/c包含了追加packet的函数（若无必要，请不要进行改动），头文件中的联合体和限幅宏定义如果已有，请用自己的数据结构/函数进行替换；

broadcast\_demo.h/c包含了组装frame和数据发送的主循环demo函数，可根据自己的需要进行更改；

## 主要服务函数说明

1、void resetCustomFrame(customFrame\_t \*frame);

功能：清空一个customFrame，在每次重新对frame追加packet前使用；在broadcast\_demo.c的uploadCustomDataLoop()函数有其使用的演示；

2、void appendCustomCell(customFrame\_t \*frame, void \*data, uint8\_t baseIndex,\ uint8\_t cellSize, uint8\_t cellCount, uint8\_t properties);

功能：在特定用户数据框架frame后追加一组传感器数据packet；在broadcast\_demo.c的uploadCustomDataLoop()函数有其使用的演示；

3、void addCustomFrameToBroadcast(customFrame\_t \*frame, uint8\_t \*array, uint16\_t \*ptr);

功能：将一个用户数据框架追加到为准备发送的报文数组中，就是将所有packet翻译成数据帧中的一串数组；在broadcast\_demo.c的formatPrepare()函数有其使用的演示；

4、void initCustomContent(void);

功能：初始化数据结构，给发送数组分配地址，**在使用发送任务之前，必须进行初始化**；

5、void uploadCustomDataLoop(void);

功能：发送用户数据到上位机的循环函数demo，其发送3个不同周期的正弦波和余弦波到上位机；在本函数中，请使用自己的发送函数将准备好的数组通过USBVCP或串口发出；

## 上位机说明

1、在使用demo时请注意预留足够的heap（至少256byte），因为发送的数组是通过calloc函数申请的内存；

2、uploadCustomDataLoop()函数中的demo是在1kHz的周期任务中进行测试的；

3、在使用串口进行发送时，请确保单片机的串口波特率和上位机的波特率一致（建议至少使用460800bps）；

4、在使用虚拟串口时，只需安装有stm32官方的虚拟串口驱动即可，波特率任选，在找到设备后（如未找到设备请点击“刷新设备”），点击连接设备，切换到波形页面，如果移植正常，即可看到波形图了。