传输协议设计

# 目录

[1. 目录 1](#_Toc198896936)

[2. 协议说明 2](#_Toc198896937)

[2.1. 协议格式定义 2](#_Toc198896938)

[2.2. 命令类型说明 3](#_Toc198896939)

[2.3. 各命令详细说明 4](#_Toc198896940)

[2.3.1. 开始采集命令 4](#_Toc198896941)

[2.3.2. 停止采集命令 4](#_Toc198896942)

[2.3.3. 数据上传命令 5](#_Toc198896943)

[2.3.4. 设置参数命令 6](#_Toc198896944)

[2.3.5. 读取参数命令 7](#_Toc198896945)

[2.3.6. 固件升级命令 8](#_Toc198896946)

[2.4. 数据类型定义 10](#_Toc198896947)

[2.4.1. 参数ID定义 10](#_Toc198896948)

[2.4.2. 上传数据类型定义 12](#_Toc198896949)

[2.4.3. 采样率定义 12](#_Toc198896950)

[2.4.4. 激光功率定义 13](#_Toc198896951)

[2.4.5. 低通滤波定义 13](#_Toc198896952)

[2.4.6. 高通滤波定义（暂定） 14](#_Toc198896953)

[2.4.7. 速度量程定义（暂定） 14](#_Toc198896954)

[2.4.8. 位移量程定义（暂定） 14](#_Toc198896955)

[2.4.9. 加速度量程定义（暂定） 15](#_Toc198896956)

[2.4.10. 模拟输出类型定义 15](#_Toc198896957)

[2.4.11. 模拟输出开关定义 15](#_Toc198896958)

[2.4.12. 前端滤波定义 15](#_Toc198896959)

[2.4.13. 前端滤波算法定义 16](#_Toc198896960)

[2.5. CRC32校验算法 16](#_Toc198896961)

[3. 通信流程 18](#_Toc198896962)

[3.1. 通讯方式 18](#_Toc198896963)

[3.2. 如何发送命令 18](#_Toc198896964)

[3.3. 查询设备命令 18](#_Toc198896965)

[4. 错误代码 19](#_Toc198896966)

[5. 示例实现 19](#_Toc198896967)

# 协议说明

用于设备进行数据传输

## 协议格式定义

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *N-Byte* | *4-Byte* |

* 协议头：协议标识(SCZN)
* 版本：协议版本号
* 命令：命令类型
* 命令ID：命令ID（用于上位机设置标识，下位机返回时带上相同的ID）
* 数据长度：数据区的长度
* 数据：数据区
* 校验值：CRC32校验值（对协议头~ 数据区的所有数据进行计算，如计算资源有限可以不校验 返回0x00000000即可）

## 命令类型说明

|  |  |  |
| --- | --- | --- |
| 地址（*1-Byte*） | 说明 | 备注 |
| *0x00* | 开始采集 | 上位机 🡪 下位机 |
| *0x01* | 回复开始采集 | 下位机 🡪 上位机 |
| *0x02* | 停止采集 | 上位机 🡪 下位机 |
| *0x03* | 回复停止采集 | 下位机 🡪 上位机 |
| *0x04* | 数据上传 | 下位机 🡪 上位机 |
|  |  |  |
| *0x06* | 写入参数 | 上位机 🡪 下位机 |
| *0x07* | 回复写入参数结果 | 下位机 🡪 上位机 |
| *0x08* | 读取参数 | 上位机 🡪 下位机 |
| *0x09* | 回复读取参数结果 | 下位机 🡪 上位机 |
|  |  |  |
|  |  |  |
| *0xFC* | 启动升级 | 上位机 🡪 下位机 |
| *0xFD* | 设备应答 启动升级 | 下位机 🡪 上位机 |
| *0xFE* | 固件传输 | 上位机 🡪 下位机 |
| *0xFF* | 设备应答 固件传输 | 下位机 🡪 上位机 |

## 各命令详细说明

### 开始采集命令

**发送数据格式：0x00**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *0-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0x00 | 0x0001 | 0x00000000 | / | 0x00000000 |

**接收数据格式：0x01**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *2-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0x01 | 0x0001 | 0x00000000 | **0x0000** | 0x00000000 |

返回结果0为成功、其他为失败（参考错误码）

### 停止采集命令

**发送数据格式：0x02**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *0-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0x02 | 0x0001 | 0x00000000 | / | 0x00000000 |

**接收数据格式：0x03**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *2-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0x03 | 0x0001 | 0x00000000 | **0x0000** | 0x00000000 |

返回结果0为成功、其他为失败（参考错误码）

### 数据上传命令

发送开始采集命令成功后，下位机连续进行采集，并定时传输数据到上位机

**接收数据格式：0x04**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *N-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0x04 | 0x0001 | 0x00000000 | ***N*** | 0x00000000 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 上传数据类型 | 数据长度 | 点1-速度 | 点2-速度 |  |
| *4-Byte* | *4-Byte* | *4-Byte* | *4-Byte* |  |
| 0x00000000 | 代表一路数据的上传点数 | 0x0000 | 0x0000 |  |

如数据是I、Q信号 4个字节为2个字节的I路信号和为2个字节的Q路信号组成

### 设置参数命令

**发送数据格式：0x06**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *N-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0x06 | 0x0001 | 0x00000011 | ***N*** | 0x00000000 |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 数据 | | | | | | | |
| 参数数量 | 参数1\_ID | 参数1\_长度 | 参数1\_值 | 参数2\_ID | 参数2\_长度 | 参数2\_值 | …. |
| *4-Byte* | *4-Byte* | *4-Byte* | *1-Byte* | *4-Byte* | *4-Byte* | *1-Byte* | *…* |
| 0x00000002 | 0x00000000 | 0x00000001 | 0x00 | 0x00000001 | 0x00000001 | 0x00 | … |

举例：设置参数

这边设置采样率（0x00000000）和上传数据类型（0x00000001）两个参数

**接收数据格式：0x07**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *N-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0x07 | 0x0001 | 0x0000000E | ***N*** | 0x00000000 |

|  |
| --- |
| 参数2\_结果 |
| *2-Byte* |
| **0x0000** |

返回结果0为成功、其他为失败（参考错误码）

### 读取参数命令

**发送数据格式：0x08**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *N-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0x08 | 0x0001 | 0x00000011 | ***N*** | 0x00000000 |

|  |  |  |  |
| --- | --- | --- | --- |
| 数据 | | | |
| 参数数量 | 参数1\_ID | 参数2\_ID | …. |
| *4-Byte* | *4-Byte* | *4-Byte* | *…* |
| 0x00000002 | 0x00000000 | 0x00000001 | … |

举例：读取参数

这边读取采样率（0x00000000）和上传数据类型（0x00000001）两个参数

**接收数据格式：0x09**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *N-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0x09 | 0x0001 | 0x0000000E | ***N*** | 0x00000000 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 数据 | | | | | | | | | |
| 参数数量 | 参数1\_ID | 参数1\_状态 | 参数1\_长度 | 参数1\_值 | 参数2\_ID | 参数2\_状态 | 参数2\_长度 | 参数2\_值 | …. |
| *4-Byte* | *4-Byte* | *2-Byte* | *4-Byte* | *1-Byte* | *4-Byte* | *2-Byte* | *4-Byte* | *1-Byte* | *…* |
| 0x00000002 | 0x00000000 | 0x0000 | 0x00000001 | 0x01 | 0x00000001 | 0x0000 | 0x00000001 | 0x00 | … |

根据参数的ID来返回对应长度的值，这里演示的是返回采样率（0x00000000）和上传数据类型（0x00000001）两个参数 每个参数id加入了获取是否成功的字段 0为获取成功、其他为失败（参考错误码）

### 固件升级命令

1. **启动升级：**

**PC端发送0xFC**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *8-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0xFC | 0x0001 | 0x00000008 | ***参考下表*** | 0x00000000 |

数据内容

|  |  |
| --- | --- |
| 子命令类型 | 固件大小 |
| *4-Byte* | *4-Byte* |
| 0x00000000（表示当前传输的是固件大小） | 0x500000（填入实际固件大小） |

**设备端应答 0xFD**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *8-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0xFD | 0x0001 | 0x00000008 | ***参考下表*** | 0x00000000 |

数据内容

|  |  |
| --- | --- |
| 子命令类型 | 应答 |
| *4-Byte* | *2-Byte* |
| *0x00* | *0表示成功，1表示失败* |

1. **固件传输**

**PC端发送：0xEF**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *N-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0xEF | 0x0002 | 0x0C + N | ***参考下表*** | 0x00000000 |

数据内容

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类型 | 包id | 包长度 | 包数据 |
| *4-Byte* | *4-Byte* | *4-Byte* | *最大一帧为1280Bytes，最后一帧小于1280Bytes* |
| 0x00000001（表示当前传输固件包数据） | 0x00000001 | 0x500 | N |

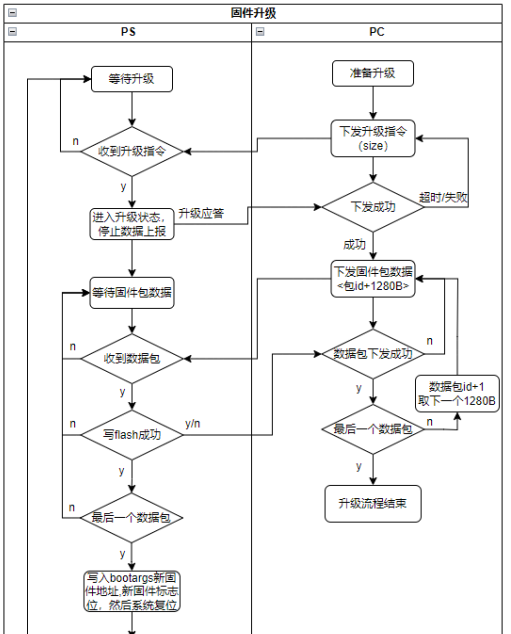
**设备端应答0xFF**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 协议头 | 版本 | 命令 | 命令ID | 数据长度 | 数据 | 校验值 |
| *4-Byte* | *1-Byte* | *1-Byte* | *2-Byte* | *4-Byte* | *13-Byte* | *4-Byte* |
| 0x53435A4E | 0x00 | 0xFF | 0x0001 | 0x0000000d | ***参考下表*** | 0x00000000 |

数据内容

|  |  |  |  |
| --- | --- | --- | --- |
| 数据类型 | 包id | 包长度 | 应答 |
| *4-Byte* | *4-Byte* | *4-Byte* | *2-Byte* |
| 0x00000001 | 0x00000001 | 0x500 | 0表示成功，1表示失败 |

**交互流程参考（PS指设备端）**



## 数据类型定义

### 参数ID定义

|  |  |  |
| --- | --- | --- |
| 地址（*4-Byte*） | 长度 | 说明 |
| 0x00000000 | *1-Byte* | 采样率 |
| 0x00000001 | *1-Byte* | 上传数据类型 |
| 0x00000002 | *1-Byte* | 激光功率 |
| 0x00000003 | *1-Byte* | 信号强度 |
| 0x00000004 | *N-Byte* | 硬件信息（具体根据实际数据传输） |
| 0x00000005 | *1-Byte* | 低通滤波 |
| 0x00000006 | *1-Byte* | 高通滤波 |
| 0x00000007 | *1-Byte* | 速度量程 |
| 0x00000008 | *1-Byte* | 位移量程 |
| 0x00000009 | *1-Byte* | 加速度量程 |
| 0x0000000A | *1-Byte* | 模拟输出类型（模拟输出口1） |
| 0x0000000B | *1-Byte* | 模拟输出开关（模拟输出口1）0为关闭 1为打开 |
| 0x0000000C | *1-Byte* | 模拟输出类型（模拟输出口2） |
| 0x0000000D | *1-Byte* | 模拟输出开关（模拟输出口2）0为关闭 1为打开 |
| 0x0000000E | *1-Byte* | 触发采样类型（自由触发、软件触发、硬件触发、调试触发模式） |
| 0x0000000F | *1-Byte* | 触发采样-触发采样模式-软（上升沿、下降沿） |
| 0x00000010 | *1-Byte* | 触发采样-触发采样模式-硬（触发电平） |
| 0x00000011 | *1-Byte* | 触发采样-触发通道 |
| 0x00000012 | *4-Byte* | 触发采样-数据采集的长度 |
| 0x00000013 | *4-Byte* | 触发采样-前置点数 |
|  |  |  |
| 0x10000000 | *1-Byte* | 前端滤波值选择 |
| 0x10000001 | *1-Byte* | 前端滤波算法类型 |
| 0x10000002 | *1-Byte* | 前端滤波开关0为关闭 1为打开 |
| 0x10000003 | *1-Byte* | 去直流开关0为关闭 1为打开 |
| 0x10000004 | *1-Byte* | 正交性修正开关0为关闭 1为打开 |
| 0x10000005 | *4-Byte* | 数据处理分段长度int32 |
| 0x10000006 | *1-Byte* | 速度低通滤波开关 对应上面的低通滤波0为关闭 1为打开 |
| 0x10000007 | *1-Byte* | 位移低通滤波开关 对应上面的低通滤波0为关闭 1为打开 |
| 0x10000008 | *1-Byte* | 加速度低通滤波开关 对应上面的低通滤波0为关闭 1为打开 |
| 0x10000009 | *4-Byte* | 速度振幅修正 float |
| 0x1000000A | *4-Byte* | 位移振幅修正float |
| 0x1000000B | *4-Byte* | 加速度振幅修正float |
| 0x10000000~0x20000000 | *?-Byte* | 具体算法定型后再定 |

### 上传数据类型定义

|  |  |
| --- | --- |
| 数据（1*-Byte*） | 说明 |
| 0x00 | 速度 |
| 0x01 | 位移 |
| 0x02 | 加速度 |
| 0x03 | I、Q路信号 |

### 采样率定义

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 采样率 2kHz |
| 0x01 | 采样率 5kHz |
| 0x02 | 采样率 10kHz |
| 0x03 | 采样率 20kHz |
| 0x04 | 采样率 50kHz |
| 0x05 | 采样率 100kHz |
| 0x06 | 采样率 200kHz |
| 0x07 | 采样率 400kHz |
| 0x08 | 采样率 800kHz |
| 0x09 | 采样率 1MHz |
| 0x0A | 采样率 2MkHz |
| 0x0B | 采样率 4MHz |
| 0x0C | 采样率 5MkHz |
| 0x0D | 采样率 10MHz |
| 0x0E | 采样率 20MHz |

### 激光功率定义

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 功率0% |
| 0x01 | 功率1% |
| … | … |
| … | … |
| 0x63 | 功率99% |
| 0x64 | 功率100% |

### 低通滤波定义

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 100Hz |
| 0x01 | 500Hz |
| 0x02 | 1kHz |
| 0x03 | 2kHz |
| 0x04 | 5kHz |
| 0x05 | 10kHz |
| 0x06 | 20kHz |
| 0x07 | 40kHz |
| 0x08 | 80kHz |
| 0x09 | 100kHz |
| 0x0A | 160kHz |
| 0x0B | 320kHz |
| 0x0C | 500kHz |
| 0x0D | 1MHz |
| 0x0E | 3MHz |

### 高通滤波定义（暂定）

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 档位1 |
| 0x01 | 档位2 |
| … | … |
| … | … |
| 0x09 | 档位9 |
| 0x0A | 档位10 |

### 速度量程定义（暂定）

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 档位1 |
| 0x01 | 档位2 |
| … | … |
| … | … |
| 0x09 | 档位9 |
| 0x0A | 档位10 |

### 位移量程定义（暂定）

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 档位1 |
| 0x01 | 档位2 |
| … | … |
| … | … |
| 0x09 | 档位9 |
| 0x0A | 档位10 |

### 加速度量程定义（暂定）

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 档位1 |
| 0x01 | 档位2 |
| … | … |
| … | … |
| 0x09 | 档位9 |
| 0x0A | 档位10 |

### 触发采样-触发类型定义

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 自由触发 相当于不用触发模式 |
| 0x01 | 软件触发 |
| 0x02 | 硬件触发 |
| 0x03 | 自定义触发（调试模式） |

### 触发采样-触发采样模式-软

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 上升沿（高电平） |
| 0x01 | 下降沿（低电平） |

### 触发采样-触发电平-硬

|  |  |
| --- | --- |
| 数据（4*-Byte*） | 说明 |
| 0x1234 | 电压 float 单位mV |

### 触发采样-出发通道

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 1通道 |
| 0x01 | 2通道 |

### 触发采样-数据采集长度（时间）

|  |  |
| --- | --- |
| 数据（4*-Byte*） | 说明 |
| 0x1234 | 数据长度 |

### 触发采样-前置点数

|  |  |
| --- | --- |
| 数据（4*-Byte*） | 说明 |
| 0x1234 | 触发前保留的点数 |

### 模拟输出类型定义

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 速度 |
| 0x01 | 位移 |
| 0x02 | 加速度 |

### 模拟输出开关定义

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 关闭 |
| 0x01 | 打开 |

### 前端滤波定义

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | 100k |
| 0x01 | 300k |
| 0x02 | 500k |
| 0x03 | 1M |
| 0x04 | 5M |
| 0x05 | 8M |

### 前端滤波算法定义

|  |  |
| --- | --- |
| 数据（*1-Byte*） | 说明 |
| 0x00 | hamming |
| 0x01 | hann |
| 0x02 | kaiser |

## 触发采样-交互逻辑

── 主机 → 设备 ── 【写入参数 0x06, ID=0x0000000E、0x0000000F、0x00000010、0x00000011、0x00000012, ID=0x1001】目前调试情况下只要设置【触发类型】为自定义触发模式即可进行数据采集

── 设备 → 主机 ── 【回复写参数结果 0x07, ID=0x1001】

── 主机 → 设备 ── 【开始采集 0x00, ID=0x1002】

── 设备 → 主机 ── 【回复开始采集 0x01, ID=0x1002】

— 等待硬件触发完毕 —

── 设备 → 主机 ── 【数据上传 0x04, ID=0x1003】

触发过程中可以发送【停止采集命令停止触发】

## CRC32校验算法

|  |
| --- |
| 1. **static** **const** unsigned **int** crc32tab[] = { 2. 0x00000000L, 0x77073096L, 0xee0e612cL, 0x990951baL, 3. 0x076dc419L, 0x706af48fL, 0xe963a535L, 0x9e6495a3L, 4. 0x0edb8832L, 0x79dcb8a4L, 0xe0d5e91eL, 0x97d2d988L, 5. 0x09b64c2bL, 0x7eb17cbdL, 0xe7b82d07L, 0x90bf1d91L, 6. 0x1db71064L, 0x6ab020f2L, 0xf3b97148L, 0x84be41deL, 7. 0x1adad47dL, 0x6ddde4ebL, 0xf4d4b551L, 0x83d385c7L, 8. 0x136c9856L, 0x646ba8c0L, 0xfd62f97aL, 0x8a65c9ecL, 9. 0x14015c4fL, 0x63066cd9L, 0xfa0f3d63L, 0x8d080df5L, 10. 0x3b6e20c8L, 0x4c69105eL, 0xd56041e4L, 0xa2677172L, 11. 0x3c03e4d1L, 0x4b04d447L, 0xd20d85fdL, 0xa50ab56bL, 12. 0x35b5a8faL, 0x42b2986cL, 0xdbbbc9d6L, 0xacbcf940L, 13. 0x32d86ce3L, 0x45df5c75L, 0xdcd60dcfL, 0xabd13d59L, 14. 0x26d930acL, 0x51de003aL, 0xc8d75180L, 0xbfd06116L, 15. 0x21b4f4b5L, 0x56b3c423L, 0xcfba9599L, 0xb8bda50fL, 16. 0x2802b89eL, 0x5f058808L, 0xc60cd9b2L, 0xb10be924L, 17. 0x2f6f7c87L, 0x58684c11L, 0xc1611dabL, 0xb6662d3dL, 18. 0x76dc4190L, 0x01db7106L, 0x98d220bcL, 0xefd5102aL, 19. 0x71b18589L, 0x06b6b51fL, 0x9fbfe4a5L, 0xe8b8d433L, 20. 0x7807c9a2L, 0x0f00f934L, 0x9609a88eL, 0xe10e9818L, 21. 0x7f6a0dbbL, 0x086d3d2dL, 0x91646c97L, 0xe6635c01L, 22. 0x6b6b51f4L, 0x1c6c6162L, 0x856530d8L, 0xf262004eL, 23. 0x6c0695edL, 0x1b01a57bL, 0x8208f4c1L, 0xf50fc457L, 24. 0x65b0d9c6L, 0x12b7e950L, 0x8bbeb8eaL, 0xfcb9887cL, 25. 0x62dd1ddfL, 0x15da2d49L, 0x8cd37cf3L, 0xfbd44c65L, 26. 0x4db26158L, 0x3ab551ceL, 0xa3bc0074L, 0xd4bb30e2L, 27. 0x4adfa541L, 0x3dd895d7L, 0xa4d1c46dL, 0xd3d6f4fbL, 28. 0x4369e96aL, 0x346ed9fcL, 0xad678846L, 0xda60b8d0L, 29. 0x44042d73L, 0x33031de5L, 0xaa0a4c5fL, 0xdd0d7cc9L, 30. 0x5005713cL, 0x270241aaL, 0xbe0b1010L, 0xc90c2086L, 31. 0x5768b525L, 0x206f85b3L, 0xb966d409L, 0xce61e49fL, 32. 0x5edef90eL, 0x29d9c998L, 0xb0d09822L, 0xc7d7a8b4L, 33. 0x59b33d17L, 0x2eb40d81L, 0xb7bd5c3bL, 0xc0ba6cadL, 34. 0xedb88320L, 0x9abfb3b6L, 0x03b6e20cL, 0x74b1d29aL, 35. 0xead54739L, 0x9dd277afL, 0x04db2615L, 0x73dc1683L, 36. 0xe3630b12L, 0x94643b84L, 0x0d6d6a3eL, 0x7a6a5aa8L, 37. 0xe40ecf0bL, 0x9309ff9dL, 0x0a00ae27L, 0x7d079eb1L, 38. 0xf00f9344L, 0x8708a3d2L, 0x1e01f268L, 0x6906c2feL, 39. 0xf762575dL, 0x806567cbL, 0x196c3671L, 0x6e6b06e7L, 40. 0xfed41b76L, 0x89d32be0L, 0x10da7a5aL, 0x67dd4accL, 41. 0xf9b9df6fL, 0x8ebeeff9L, 0x17b7be43L, 0x60b08ed5L, 42. 0xd6d6a3e8L, 0xa1d1937eL, 0x38d8c2c4L, 0x4fdff252L, 43. 0xd1bb67f1L, 0xa6bc5767L, 0x3fb506ddL, 0x48b2364bL, 44. 0xd80d2bdaL, 0xaf0a1b4cL, 0x36034af6L, 0x41047a60L, 45. 0xdf60efc3L, 0xa867df55L, 0x316e8eefL, 0x4669be79L, 46. 0xcb61b38cL, 0xbc66831aL, 0x256fd2a0L, 0x5268e236L, 47. 0xcc0c7795L, 0xbb0b4703L, 0x220216b9L, 0x5505262fL, 48. 0xc5ba3bbeL, 0xb2bd0b28L, 0x2bb45a92L, 0x5cb36a04L, 49. 0xc2d7ffa7L, 0xb5d0cf31L, 0x2cd99e8bL, 0x5bdeae1dL, 50. 0x9b64c2b0L, 0xec63f226L, 0x756aa39cL, 0x026d930aL, 51. 0x9c0906a9L, 0xeb0e363fL, 0x72076785L, 0x05005713L, 52. 0x95bf4a82L, 0xe2b87a14L, 0x7bb12baeL, 0x0cb61b38L, 53. 0x92d28e9bL, 0xe5d5be0dL, 0x7cdcefb7L, 0x0bdbdf21L, 54. 0x86d3d2d4L, 0xf1d4e242L, 0x68ddb3f8L, 0x1fda836eL, 55. 0x81be16cdL, 0xf6b9265bL, 0x6fb077e1L, 0x18b74777L, 56. 0x88085ae6L, 0xff0f6a70L, 0x66063bcaL, 0x11010b5cL, 57. 0x8f659effL, 0xf862ae69L, 0x616bffd3L, 0x166ccf45L, 58. 0xa00ae278L, 0xd70dd2eeL, 0x4e048354L, 0x3903b3c2L, 59. 0xa7672661L, 0xd06016f7L, 0x4969474dL, 0x3e6e77dbL, 60. 0xaed16a4aL, 0xd9d65adcL, 0x40df0b66L, 0x37d83bf0L, 61. 0xa9bcae53L, 0xdebb9ec5L, 0x47b2cf7fL, 0x30b5ffe9L, 62. 0xbdbdf21cL, 0xcabac28aL, 0x53b39330L, 0x24b4a3a6L, 63. 0xbad03605L, 0xcdd70693L, 0x54de5729L, 0x23d967bfL, 64. 0xb3667a2eL, 0xc4614ab8L, 0x5d681b02L, 0x2a6f2b94L, 65. 0xb40bbe37L, 0xc30c8ea1L, 0x5a05df1bL, 0x2d02ef8dL 66. };  69. **static** unsigned **int** crc32( **const** unsigned **char** \*buf, unsigned **int** size) 70. { 71. unsigned **int** i, crc; 72. crc = 0xFFFFFFFF; 73. **for** (i = 0; i < size; i++) 74. crc = crc32tab[(crc ^ buf[i]) & 0xff] ^ (crc >> 8); 75. **return** crc^0xFFFFFFFF; 76. } |

# 通信流程

## 通讯方式

采用TCP/IP的方式通讯，上位机为服务端、下位机为客户端，下位机定时连接上位机，直到连接成功为止。

## 如何发送命令

* 构建数据包：发送方将需要传递的信息按协议格式构建数据包，指定命令类型和参数。
* 发送数据：通过 TCP/IP 通道发送该数据包。
* 解析与执行：接收方解析数据包，校验并执行命令。
* 响应结果：接收方构建并发送一个确认包，指示命令的执行结果。

## 查询设备命令

通过广播协议查找设备-具体待定

# 错误代码

|  |  |
| --- | --- |
| 地址（*2-Byte*） | 说明 |
| 0x0000 | 代表执行成功 |
| 0x0001 | 代表执行失败 |
| 其他 | 待定 |
|  |  |

# 示例实现

以下是示例代码，展示了如何封装和解析协议数据包：

|  |
| --- |
| 1. import struct 2. import crc32c 3. *# 定义协议固定部分（全部使用字节）* 4. PROTOCOL\_HEADER = b'\x53\x43\x5A\x4E'  *# "SCZN" 的字节形式* 5. VERSION = 0x00  *# 1 字节无符号整数* 6. def calculate\_crc32(data): 7. """计算 CRC32 校验值""" 8. return crc32c.crc32c(data) 9. def build\_packet(command: int, command\_id: int, data: bytes) -> bytes: 10. """ 11. 构建协议数据包（使用纯字节表示） 12. :param command: 命令类型 (1 字节) 13. :param command\_id: 命令 ID (2 字节) 14. :param data: 数据区 (字节) 15. :return: 完整数据包 16. """ 17. data\_length = len(data)  *# 数据区长度（字节数）* 19. *# 打包协议数据（头部 + 数据区 + 校验值）* 20. packet = struct.pack( 21. f">4s B B H I {data\_length}s",  *# 大端序* 22. PROTOCOL\_HEADER,               *# 4 字节协议头* 23. VERSION,                       *# 1 字节版本号* 24. command,                       *# 1 字节命令类型* 25. command\_id,                    *# 2 字节命令 ID* 26. data\_length,                   *# 4 字节数据长度* 27. data                           *# 数据区（可变长度）* 28. ) 30. *# 计算 CRC32 校验值* 31. crc = calculate\_crc32(packet) 33. *# 将校验值追加到数据包末尾* 34. return packet + struct.pack(">I", crc) 35. def parse\_packet(packet: bytes): 36. """ 37. 解析协议数据包（字节表示） 38. :param packet: 完整数据包 (字节) 39. :return: 解析后的内容 (字典形式) 40. """ 41. *# 解包固定头部部分* 42. header, version, command, command\_id, data\_length = struct.unpack(">4s B B H I", packet[:12]) 44. if header != PROTOCOL\_HEADER: 45. raise ValueError("Invalid protocol header") 47. *# 提取数据区和校验值* 48. data = packet[12:12 + data\_length] 49. received\_crc, = struct.unpack(">I", packet[12 + data\_length:]) 51. *# 校验 CRC32* 52. calculated\_crc = calculate\_crc32(packet[:-4]) 53. if received\_crc != calculated\_crc: 54. raise ValueError("CRC mismatch") 56. return { 57. "header": header, 58. "version": version, 59. "command": command, 60. "command\_id": command\_id, 61. "data\_length": data\_length, 62. "data": data, 63. "crc": received\_crc, 64. } 65. *# 示例用法* 66. if \_\_name\_\_ == "\_\_main\_\_": 67. *# 示例：构建一个停止采集命令* 68. command = 0x01       *# 停止采集* 69. command\_id = 0x0001  *# 命令 ID* 70. data = b""           *# 数据为空* 72. *# 构建数据包* 73. packet = build\_packet(command, command\_id, data) 74. print("构建的数据包:", packet.hex().upper()) 76. *# 解析数据包* 77. parsed = parse\_packet(packet) 78. print("解析后的内容:", parsed)   输出接口为：  构建的数据包: 53435A4E0001000100000000A57105A2  解析后的内容: {'header': b'SCZN', 'version': 0, 'command': 1, 'command\_id': 1, 'data\_length': 0, 'data': b'', 'crc': 2775647650} |
|  |