**传感网数据包协议格式**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 包头 | 目标地址 | 源地址 | 包序号 | 功能字 | 数据长度 | 首部校验和 | 数据域 | 数据校验和 |
| Frame | Target | Source | Index | Function | DataLength | HeadSum | DataArea | CheckSum |
| 4Byte | 2Byte | 2Byte | 2Byte | 1Byte | 2Byte | 1Byte | NByte | 1Byte |
| EF02AAAA | 0--0xFFFF | 0--0xFFFF | 0--0xFFFF | 0xF0—0xFF | LEN | SUM | JSON | SUM |

**详细指令介绍**：

**包头**: 包头固定为四字节(0xEF,0x02,0xAA,0xAA)，每个数据包内容开头均为此字节；

**目标地址:** 此包将发往的节点地址，主要用于转发。低字节在前。

**源地址:** 此包的产生来源地址，主要用于回应。低字节在前。

**包序号:** 用以在接收应答时确认应答对应的包，序号本身循环递增。低字节在前。

**功能字**: 从0xF0至0xFF,现暂定16种功能字，用以标识包的用途，如注册设备、传递数据、设置属性等。(详细定义见附件)

**数据长度**:共两字节，以指示数据域的字节长度，因为包头、校验和等字段的长度均已固定，因此这里的的数据长度**仅是指数据域的长度**，**不包含**包头、功能字、数据长度及校验和的字节长度。低字节在前。

**首部校验和**:对前半部分固定字节的首部全部字节做一次校验，主要用于检查数据长度项是否正确。

**数据域**: 数据域为实际用户需要传输的数据内容，本协议中采用JSON格式包装原始数据，将设备名称、地址、数据项与值等包装成一个JSON对象，以直接在应用层通信，单片机与上位机均直接解析该JSON对象(单片机端使用cJSON库)。其中在使用键值对时，本协议统一化键名，因此解析时只需按照固定的键名获取对应的值。(详细定义见附件)

**数据校验和**: 本协议中校验和分两部分检验，前半部分的首部固定字节由首部校验和保证，数据校验和只对数据域的数据所有字节做校验。

如无特殊说明，以下数值均指HEX码。

**功能字详细定义表**

|  |  |  |
| --- | --- | --- |
| 设备 ---> 终端 | | |
| 字节 | 功能 | 说明 |
| F0 | 握手 | 标示此包为握手包，设备启动时要先跟终端握手。 |
| F1 | 应答 | 除应答包外若无特殊说明，收到的所有包均要给予应答已确认收到。 |
| F2 | 注册设备 | 用设备名与地址在终端注册一个设备设备。 |
| F3 | 休眠 | 标示此设备即将进入休眠状态(无需应答包)。 |
| F4 | 启动 | 标示此设备已经启动(无需应答包)。 |
| F5 | 数据 | 标示此包为上传设备传感器的数据项。 |
| F6 | 注销 | 请求从终端的注册列表中注销。 |
| F7 | 重启 | 标示设备即将重启。 |
| F8 | 关闭 | 标示设备即将关闭。 |

|  |  |  |
| --- | --- | --- |
| 终端 ---> 设备 | | |
| 字节 | 功能 | 说明 |
| F0 | 握手 |  |
| F1 | 应答 |  |
| F2 | 休眠 |  |
| F3 | 唤醒 |  |
| F4 | 请求数据 |  |
| F5 | 设置属性 |  |
| F6 | 注销 |  |
| F7 | 重启 |  |
| F8 | 关闭 |  |

如无特殊说明，以下均指utf-8编码。

**数据域详细定义说明**

数据域中数据采用JSON封装，成为一条消息，例如：

{"InfoType":"Data","Owner":"admin","Device":"Contiki STM32","Address":65535," Temperature":null}

JSON说明：[JSON格式解析](http://blog.csdn.net/g1036583997/article/details/41008023)

1) 本协议中部分键名同一固定，解析数据时均按照固定键名解析。

2) 每条消息中必须有设备名、地址。

3) 消息类型可以不写明，若不写明则默认为数据项。

**JSON固定键名定义说明**

|  |  |  |  |
| --- | --- | --- | --- |
| 键名 | 值 | 值类型 | 说明 |
| Device | 设备名称 | string |  |
| Address | 设备地址 | 2Byte |  |
| InfoType | 消息类型 | 1Byte | 0x01 设备信息  0x02 数据项  0x03 设置项  0X04 命令 |
| Owner | 设备主人 | string |  |
|  |  |  |  |
|  |  |  |  |

例如终端机向设备发送一条JSON消息：

{

"Device":"Contiki via STM32F103",

“Owner”:”””admin”,

"Address":65535,

"InfoType":””Data”,

"SensorType1":null,

"SensorType2":null,

"SensorType3":null

}

服务器数据库中对每个用户存储如下：

{

"UserName":"admin",

"Password":"c21f969b5f03d33d43e04f8f136e7682",

"Terminal":[

{

"Address":1,

"Place":"监测点东",

"SensorList":[

{"SensorType":"LightIntensity","DisplayName":"光照强度","QuantityUnit":"Lux"},

{"SensorType":"Temperature","DisplayName":"温度" ,"QuantityUnit":"°C"},

{"SensorType":"Humidity","DisplayName":"湿度" ,"QuantityUnit":"%"},

{"SensorType":"SmogPercentage","DisplayName":"烟雾浓度" ,"QuantityUnit":"%"}

]

},

{

"Address":2,

"Place":"监测点西",

"SensorList":[

{"SensorType":"Humidity","DisplayName":"湿度" ,"QuantityUnit":"%"},

{"SensorType":"SmogPercentage","DisplayName":"烟雾浓度" ,"QuantityUnit":"%"},

{"SensorType":"WaveDistance","DisplayName":"超声波测距" ,"QuantityUnit":"CM"}

]

}

]

}

1. 非固定项键名如**数据名称**、**命令名称**、**设置项名**称都根据不同设备而改变；
2. 同一种设备的数据项键名应该相同统一，如” LightIntensity”,” Temperature”, “Humidity”,“distance”,”anyone”等；
3. 命令项键名也相同统一，如”SetBaudRate”,”SetDataDelay”,”Reboot”,”Shutdown”等；
4. 所有的数据传输都应整理为键值对的形式封装进JSON对象统一传输；
5. 服务器的”Address”地址为0x0001，其余终端均从2开始编号。