

**SPRS组件**

说明文档

|  |  |  |
| --- | --- | --- |
|  | 当前版本： | V4.0.0 |
| 文件状态： | 创 建 人： | 翟海青 |
| [√]草稿 | 创建日期： | \_\_\_2020\_年 5 月 19 日 |
| [ ]正在修改 | 审 核 人： | 翟海青 |
| [ ]正式发布 | 审核日期： | \_\_\_2020\_年 5 月 23 日 |
|  | 密 级： | ■内部资料 □公开资料 |

版权声明和保密须知：

本文件出现的任何文字叙述、文档格式、照片、插图、方法、过程等内容，除特别声明外，版权均归属北京源清慧虹信息科技有限公司所有，受到有关产权及版权法保护。

Copyright© 2014.All rights reserved 北京源清慧虹信息科技有限公司

目录

[1. 说明 4](#_Toc41384393)

[2. 资源占用 4](#_Toc41384394)

[2.1. 硬件资源 4](#_Toc41384397)

[2.2. 软件资源 5](#_Toc41384398)

[3. 配置文件 5](#_Toc41384399)

[3.1. 配置文件说明 5](#_Toc41384401)

[3.2. 配置文件内容 5](#_Toc41384402)

[4. 设备表 6](#_Toc41384403)

[4.1. 设备表说明 6](#_Toc41384405)

[4.2. 存储 6](#_Toc41384406)

[4.3. 结构 6](#_Toc41384407)

[4.4. 管理 7](#_Toc41384408)

[4.4.1. 增加设备 7](#_Toc41384409)

[4.4.2. 删除设备 8](#_Toc41384410)

[4.4.3. 修改设备 9](#_Toc41384411)

[4.4.4. 查询设备 10](#_Toc41384412)

[5. 指令表 10](#_Toc41384413)

[5.1. 说明 10](#_Toc41384414)

[5.2. 存储 10](#_Toc41384415)

[5.3. 指令块与设备编号 10](#_Toc41384416)

[5.4. 结构 10](#_Toc41384417)

[5.5. 构成 11](#_Toc41384418)

[6. SPRS组件 12](#_Toc41384419)

[6.1. 环境依赖 12](#_Toc41384425)

[6.2. 配置说明 12](#_Toc41384426)

[6.3. 功能说明 12](#_Toc41384427)

[6.4. 接收消息规范 13](#_Toc41384428)

[6.5. 发送消息规范 14](#_Toc41384429)

[6.6. 线程 15](#_Toc41384430)

[6.7. 总体流程图 16](#_Toc41384431)

[6.8. 设备管理 17](#_Toc41384432)

[6.9. 子设备访问 18](#_Toc41384433)

[7. 错误表 19](#_Toc41384434)

# 说明

**本文详细说明了SPRS组件的功能、特点、访问方式、节点表、指令表等组织、管理、存储、采集方法等，嵌入式程序将按照此文进行设计。**

SPRS组件是基于Modbus-RTU协议且工作在TIA/EIA-485-A(RS-485)总线上的通用采集组件，对符合Modbus-RTU规范的各类子传感器/设备进行访问、获取数据，在总线上作为主设备，主要功能包含子设备访问、设备表管理等功能。

能够正常工作的SPRS组件需要在文件系统根目录（”/”）下包含以下三个文件：

1. SPRS.mo：程序主体
2. SPRS.ini：配置文件
3. EQMA.CFG：设备表
4. INMA.CFG：指令表

配置文件、设备表、指令表可独立放入文件系统中，其中SPRS组件可以对设备表进行增、删、改、查等管理。

以下对组件各个部分及工作进行详细的说明。

# 资源占用



## 硬件资源

SPRS组件运行在HCB\_V1.0.0主板上，同时需要连接上HMAC\_V1.0.0采集模块，组件使用了CPU（STM32L452REI6）的资源有：PC4，PA1，PA2、PA3，其中：

PC4：模块电源控制引脚，推挽输出模式；

PA1：RS485（半双工通信）通信方向控制引脚，推挽输出模式；

PA2：USART2数据发送引脚；

PA3：USART2数据接收引脚。

## 软件资源

SPRS运行在新框架环境下，在该架构下，创建两个线程和一个软件定时器：

线程1：rthread\_downstream

功能：用于接收RS485数据，并进行分帧；

栈空间：1024；

线程优先级：20；

线程2：rthread\_busdevice\_access

功能：用于访问设备或管理总线节点；

栈空间：3072；

线程优先级：21；

软件定时器1：sprs\_timer1;

功能：用于判断RS485接收的一帧数据已经结束；

周期：2Tick；

# 配置文件



## 配置文件说明

SPRS.INI是组件的配置文件，当前主要用于配置串口。

## 配置文件内容

配置文件内容：

[portcfg]

buad= 9600 #Value:2400,4800,9600,19200,38400,57600,115200

databits=8 #Value:5,6,7,8,9

stopbits=1 #Value:1=STOP\_BITS\_1,2=STOP\_BITS\_2,3=STOP\_BITS\_3,4=STOP\_BITS\_4

bufsize=256 #Value:32-512

parity=0 #Value:0=PARITY\_NONE,1=PARITY\_ODD,2=PARITY\_EVEN

文件包含一个[portcfg]的节，该节用于配置通信串口参数，参数取值如上所示，需要说明的是当组件读取ini文件失败或参数错误，则模式初始化为9600、8、1、256、0。

# 设备表



## 设备表说明

设备表是记录总线挂载的设备的信息表，SPRS组件通过设备表得到某一地址下的设备类型、属性、分组信息，并据此进行设备的访问。

## 存储

设备表存储在文件系统根目录下（”/”），文件为“EQMA.CFG”。

## 结构

设备表包含设备数量和设备参数，每一种设备的参数在设备表中有四种属性：地址、类型、属性、分组，其结构如下：



设备数量：用于记录总线上的设备数量，当前最大值为128；

地址：设备N在总线上的标识地址，SPRS组件将根据此地址进行设备访问（默认为0，可设置范围：1-247）；

类型：设备N对应的设备类型，SPRS组件将根据此类型值在指令表中查找对应的指令集（默认为0，可设置范围：1-255）；

属性：设备N的附加属性，具体值由上层应用定义，SPRS组件不使用该值，仅将其作为一个参数进行传递（默认为0，范围：0-255），eg：HCF710设备，0可以代表节点，1可以代表基准点；

分组：设备N的分组信息，具体值由上层应用定义，SPRS组件不使用改值，仅将其作为一个参数进行传递（默认为0，范围：0-255），eg：HCF710设备，可将多台设备分为不同沉降监测线路。

设备表中的值均采用unsigned char类型进行存储。

## 管理

SPRS组件可以对设备表进行增、删、改、查管理，SPRS将根据所要操作的设备的地址号，在设备表中查找该地址号对应的存储位置，然后对具体的参数进行相应的操作，具体如下：

### 增加设备

SPRS组件接收到上层组件传递的增加设备命令和设备的参数（地址、类型、属性、分组），进行校验，参数正确时，SPRS组件将该设备增加到设备表中，参数错误时，返回错误。

例如：设备表中包含地址为1、2、5的三台设备，它们均是设备类型为2的设备，当向设备表中新增一台地址为3，类型为2的设备时，增加设备就会成功，如下图所示:



而如果向设备表中新增一台设备地址为2的设备时，由于地址2的设备已经存在，此时增加就不会成功，SPRS将返回错误，如下图所示:



### 删除设备

SPRS组件接收到上层组件传递的删除设备命令和设备的地址参数，进行校验，参数正确时，SPRS组件将该设备删除，并更新设备表，参数错误时，返回错误。

例如：设备表中包含地址为1、2、5的三台设备，它们均是设备类型为2的设备，当删除设备表中一台地址为2的设备时，删除设备就会成功，如下图所示:



而如果删除地址为3的设备时，由于记录地址的存储区的地址号已经为0，即地址为3的设备不存在，此时删除就不会成功，SPRS将返回错误，如下图所示:



### 修改设备

SPRS组件接收到上层组件传递的修改设备命令和设备的参数（地址、类型、属性、分组），进行校验，参数正确时，SPRS组件将根据地址变更设备参数，参数错误时，返回错误。

例如：设备表中包含地址为1、2、5的三台设备，它们均是设备类型为2的设备，当修改设备表中地址为2的设备类型为1时，修改设备就会成功（设备存在，且传入的参数正确），如下图所示:



而如果修改地址为3的设备时，由于记录地址的存储区的地址号已经为0，即地址为3的设备不存在，此时修改就不会成功，SPRS将返回错误，如下图所示:



同时如果传入的参数错误（设备类型1-255），修改也不会成功。

### 查询设备

SPRS组件接收到上层组件传递的查询设备命令和设备的地址，进行校验，参数正确时，SPRS组件将返回该地址设备的参数，参数错误时，返回错误。

注：具体错误类型见附表1-1。

# 指令表

## 说明

指令表是子传感器的指令集文件，最大支持255种不同类型的设备，作为独立的文件进行维护更新，新支持的设备指令顺序添加在指令表中，SPRS组件仅能访问指令表中已存在的类型设备。

## 存储

指令表存储在文件系统根目录下（”/”），文件为“INMA.CFG”。

## 指令块与设备编号

SPRS在获取指令集时，根据设备类型编号在指令表中查找对应的指令块，因此，指令表的维护，需要将设备类型为N的指令集放入对应的指令块N中。

## 结构

指令表由指令表信息块和若干个指令块两部分组成。其中指令表信息由16个字节组成，用于记录指令表版本，支持的设备类型等信息，单个指令块由59个字节组成，如下所示：



* eq\_type：设备类型编号；
* check\_type：校验类型，改值为保留值，当前默认为Modbus-CRC16，不支持其他的校验类型；
* Res\_time：响应时间（无符号32位整型，小端格式），单位ms，传感器的指令响应时间，超过该时间，返回超时错误；
* Eq\_in\_type：指令类型，设置为1时，设备为双指令，设置为0时，设备为单指令；
* Eq\_inX\_len：指X的长度；
* inX\_dat[0]:inX\_dat[11]：指令X缓存区，单条指令最长为12字节；

手动维护指令表时，需要维护一个excel表格，在表格中对指令进行可视化管理，表格格式请查看附件1-1。

## 构成

单条指令的缓存大小为12B，因此一条指令长度不能超过12个字节，超过的请使用透传指令。

Modbus指令由地址+指令码+附加码+CRC校验组成，在SPRS组件中，一条指令由地址（1B）+[指令码+附加码]+CRC校验(2B)三部分组成。指令存储时，地址和CRC校验可以放入任何数，不用关心。

在进行子设备访问时，SPRS会将地址替换为正确的地址号，同时进行Modbus-CRC16算法，生成2B的校验码进行替换。



需要说明的是在发送透传指令时，SPRS不会进行地址替换和CRC校验替换，而是将收到的数据直接发送到总线上，透传指令最大长度为256字节。

# SPRS组件



## 环境依赖

SPRS运行在新框架环境下。

## 配置说明

暂缺（配置串口引脚、串口号、bps、停止位等）

组件ID配置为8

## 功能说明

SPRS组件具备设备表管理、子设备访问两大部分功能，具体的功能列表如下所示：

enum SUB\_FUNCTION\_CODE

{

  ADD\_EQ\_NODE = 1, //增加设备

  DEL\_EQ\_NODE = 2, //删除设备

  CHA\_EQ\_NODE = 3, //修改设备

  GET\_EQ\_NODE = 4, //查询设备

  GET\_EQ\_INDEX\_INFO = 9,  //获取指令表索引信息

  GET\_EQ\_CFG\_INFO = 10,   //获取配置信息

  GET\_EQ\_GET\_SDAT = 11,   //获取采样数据

  EQ\_CUSTOM\_COMMAND = 12, //自定义指令

  SEN\_EQ\_PAS = 13,        //透传指令

};

其中GET\_EQ\_INDEX\_INFO用于获取指令表的信息块（16B）。

## 接收消息规范

组件通信规范如下表所示定义：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **示例（业务组件下行消息）** | | | | | | | | |
| mq\_type | dat\_len | dat[0] | dat[1] | dat[2] | dat[3] | dat[4] | … | dat[n] |
| 1：增加节点 | 4 | 节点地址 | 节点类型 | 节点属性 | 节点分组 | NULL | … | NULL |
| 2：删除节点 | 1 | 节点地址 | NULL | NULL | NULL | NULL | … | NULL |
| 3：修改节点 | 4 | 节点地址 | 节点类型 | 节点属性 | 节点分组 | NULL | … | NULL |
| 4：查询节点 | 1 | 节点地址 | NULL | NULL | NULL | NULL | … | NULL |
| 9：查询指令表 | 0 | NULL | NULL | NULL | NULL | NULL | … | NULL |
| 10：获取配置信息 | 1 | 地址 | NULL | NULL | NULL | NULL | … | NULL |
| 11：获取数据 | 1 | 地址 | NULL | NULL | NULL | NULL | … | NULL |
| 12：自定义 | 1 | 地址 | NULL | NULL | NULL | NULL | … | NULL |
| 13：透传指令 | n+1 | D0 | D1 | D2 | D3 | D4 | … | Dn |

由于组件间采用消息队列进行通信，因此需要将相应的数据进行封包，消息队列包含了以下信息：

typedef struct

{

  D\_CMD\_STRU d\_cmd;

  unsigned char d\_src;

  unsigned char d\_des;

  void \*d\_p;

  unsigned short d\_dl;

  rt\_sem\_t psem;  //信号量

  RESP\_STRU \*r\_p; //返回数据指针

} \_\_attribute\_\_((packed)) GMS\_STRU;

结构体内的变量为内核定义的消息规范。

SPRS组件定义了一个用于接收业务组件数据结构体，内容如下：

typedef struct \_\_attribute\_\_ ((\_\_packed\_\_))

{

    uint8\_t  mq\_type;                       //消息类型

    uint8\_t  dat\_len;                       //数据长度

    uint8\_t  dat[256];

}SPRS\_in\_struct\_type;

业务组件向SPRS组件传输命令/数据时，需要根据组件通信规范中的定义将数据放入结构体内，例如，增加节点命令（节点参数：3 2 0 0）时,需要将以下参数赋值给结构体：

SPRS\_in\_struct.mq\_type=ADD\_EQ\_NODE;   //命令

SPRS\_in\_struct.dat\_len=4;                   //数据长度

SPRS\_in\_struct.dat[0]=3;                    //数据

SPRS\_in\_struct.dat[1]=2;                    //数据

SPRS\_in\_struct.dat[2]=0;                    //数据

SPRS\_in\_struct.dat[3]=0;                    //数据

同时将该结构体的首地址赋值给消息队列的d\_p指针，将数据长度6赋值给d\_dl变量。

## 发送消息规范

SPRS组件接收到业务组件的消息，完成处理后，会向业务组件返回消息，返回的消息遵循以下规范：

typedef struct \_\_attribute\_\_((\_\_packed\_\_))

{

  uint8\_t src;           //消息源

  uint8\_t eq\_addr;       //传感器地址

  uint8\_t eq\_type;       //传感器类型

  uint8\_t eq\_par;        //传感器属性

  uint8\_t eq\_gro;        //传感器分组

  uint8\_t mq\_type;       //指令类型

  uint8\_t is\_mq\_success; //0成功，错误代码请查看错误规范

  uint8\_t d\_len;         //返回的数据长度

  uint8\_t \*dp;           //返回的数据指针

} SPRS\_output\_dat\_type;

其中：

1. src为上层业务组件的ID号；
2. eq\_addr为子设备的地址号；
3. eq\_type为子设备的设备类型；
4. eq\_par为子设备的附加属性；
5. eq\_gro为子设备的分组号；
6. mq\_type为执行命令的ID号；
7. is\_mq\_success为执行动作是否成功，如果执行成功，该值为0，如果有错误，则代表错误代码；
8. d\_len为子设备返回的数据长度；
9. dp为子设备返回的数据指针。

## 线程

SPRS组件在新框架的基础上创建两个线程，分别为串口数据接收线程（优先级20）和设备访问与管理线程（优先级21），SPRS组件整体的工作流程如下所示：



## 总体流程图

SPRS流程图如下：



## 设备管理

由于设备表管理在第二部分中即已经进行说明，本节不再对其进行详细说明。

## 子设备访问

子设备访问共包含为5部分功能：指令信息表获取、获取子设备配置信息、获取子设备数据信息、自定义指令、透传指令。

设备访问详细流程图如下所示：



由于设备采用了指令表的形式进行指令管理，并对相应功能进行抽象，事实上，用户可以将获取配置、获取数据、自定义指令灵活配置为任何符合规则的指令。

# 错误表

|  |  |  |  |
| --- | --- | --- | --- |
| 错误类型 | 错误项 | 错误代码 |  |
| 消息错误 | 不支持的功能号 | 0xF1 |  |
|  | 消息长度错误 | 0xF2 | 暂不支持 |
|  |  |  |  |
| 节点管理 | 节点表不存在 | 0xE1 |  |
|  | 地址异常 | 0xE2 |  |
|  | 新增节点时，地址号已存在 | 0xE3 |  |
|  | 新增节点时，地址号已满 | 0xE4 |  |
|  | ~~删除节点时，地址号不存在~~ | ~~0xE5~~ |  |
|  | 删除节点时，地址号已空 | 0xE6 |  |
|  | 修改节点时，地址号不存在 | 0xE7 |  |
|  | 查询节点时，地址号不存在 | 0xE8 |  |
|  | 节点表打开失败 | 0xE9 |  |
|  | 参数错误 | 0xEA |  |
|  |  |  |  |
| 设备访问 | 总线无此设备 | 0xD1 |  |
|  | 子设备无响应/超时 | 0xD2 |  |
|  | 子设备数据返回错误（仅校验地址和CRC，透传指令不校验） | 0xD3 |  |
|  |  |  |  |
| 指令操作 | 指令表不存在 | 0xC1 |  |
|  | 指令表不支持的设备 | 0xC2 | 暂不支持 |
|  | 指令表读取错误 | 0xC3 |  |