**移动互联大赛模块数据通信协议**

**V20160517**

**此文档比赛现场会提供，所涉及内容无需强行记忆，但要熟练了解；**

[1.数据通信协议说明概述 3](#_Toc3662)

[1.1 总体说明 3](#_Toc19967)

[1.2 0x03功能码的数据格式 3](#_Toc14459)

[1.3 0x10功能码的数据格式 4](#_Toc28109)

[2.各模块数据通信协议说明 5](#_Toc23345)

[2.1 温湿度传感节点指令 5](#_Toc26598)

[2.2 烟雾传感节点指令 5](#_Toc3157)

[2.3 光照传感节点指令 5](#_Toc9166)

[2.4 人体传感器节点指令 5](#_Toc25029)

[2.5 风扇（直流电机）节点指令 6](#_Toc8877)

[2.6 数码管输出节点指令 6](#_Toc6080)

[2.7 灯光蜂鸣器控制节点指令 6](#_Toc22676)

[2.8 PM2.5传感节点指令 7](#_Toc8092)

[2.9 窗帘（步进电机）控制节点指令 7](#_Toc12522)

[3.高频RFID节点指令举例说明 8](#_Toc259)

[3.1 读卡号操作 8](#_Toc7963)

[3.2 读卡的块数据操作 8](#_Toc2647)

[3.3 写卡的块数据操作 9](#_Toc26123)

[4.蓝牙射频板配置指令说明 11](#_Toc28902)

# 1.数据通信协议说明概述

注意：此部分内容比赛期间不会涉及

## 1.1 总体说明

本数据通信协议是使用通用的ModBus--RTU数据格式，用到其中两个功能码，分别是0x03和0x10，其中0x03是读取传感器等数据用，0x10是控制相关的直流电机、步进电机等设备用,但是高频RFID节点是用另外的数据格式。

## 1.2 0x03功能码的数据格式

**发送：**

地址+0x03+StartAddr\_Hi+StartAddr\_Lo+No.of Registers\_Hi+No.ofRegisters\_Lo+CRC\_Lo+CRC\_Hi；

地址：ModBus--RTU数据格式的地址，一般是不同的，在这里我们固定为0x01;

0x03: 读取数据的功能码，表示读取数据；

StartAddr\_Hi：读取数据地址的高8字节

StartAddr\_Lo：读取数据地址的低8字节

No.of Registers\_Hi：读取数据寄存器的个数的高8字节

No.of Registers\_Lo：读取数据寄存器的个数的低8字节

注意：这里说的寄存器不是我们CPU里面描述的那种寄存器。

CRC\_Lo：把前面的数据（地址+0x03+StartAddr\_Hi+StartAddr\_Lo+No.of Registers\_Hi+No.of Registers\_Lo）用CRC16的方式进行校验，产生的校验码低8位；

CRC\_Hi：把前面的数据（地址+0x03+StartAddr\_Hi+StartAddr\_Lo+No.of Registers\_Hi+No.of Registers\_Lo）用CRC16的方式进行校验，产生的校验码高8位；

**返回：**

地址+0x03+Count+Reg\_Hi+Reg\_Lo+CRC\_Lo+CRC\_Hi

地址：ModBus--RTU数据格式的地址，一般是不同的，在这里我们固定为0x01；

0x03: 读取数据的功能码，表示读取数据；

Count：表示读取数据字节的个数，Count = 2X（No.of Registers\_Hi+No.of Registers\_Lo）；

Reg\_Hi：表示读取到的数据高8位字节；

Reg\_Lo：表示读取到的数据低8为字节；

注意：读取到的数据的个数由Count确定，也即可能是多个数据，每个数据又分为高8位字节和低8位字节，所有的数据的高8位字节加上低8位字节的个数就是Count；

CRC\_Lo：把前面的数据（地址+0x03+Count+Reg\_Hi+Reg\_Lo）用CRC16的方式进行校验，产生的校验码低8位；

CRC\_Hi：把前面的数据（地址+0x03+Count+Reg\_Hi+Reg\_Lo）用CRC16的方式进行校验，产生的校验码高8位；

**举例：**

发送：01 03 00 14 00 02 84 0f

表示要从地址为01处读取数据，读取数据的开始地址为00 14，读取数据寄存器个数为00 02，也就是读取2个寄存器数据，共4个字节；84 0f为前面数据（01 03 00 14 00 02）CRC16的校验码。

返回：01 03 04 02 84 01 26 3B E8

表示从地址为01处返回的数据，返回的数据一共有00 04个，即4个字节的数据，这4个字节的数据就是02 84 01 26，3B EB为前面数据（01 03 04 02 84 01 26）CRC16的校验码。

## 1.3 0x10功能码的数据格式

**发送:**

地址+0x10+StartAddr\_Hi+StartAddr\_Lo+Quantity of Registers\_Hi+Quantity of Registers\_Lo + Count + RegVal\_Hi+RegVal\_Lo+CRC\_Lo+CRC\_Hi

地址：ModBus--RTU数据格式的地址，一般是不同的，在这里我们固定为0x01;

0x10：写数据的功能码，在这里表示控制直流电机、步进电机等模块的功能码；

StartAddr\_Hi+StartAddr\_Lo ： 要写的数据存放的地址，分别为高8位字节和低8位字节；

Quantity of Registers\_Hi+Quantity of Registers\_Lo ： 将要写的数据寄存器个数；

Count ：要写数据的字节个数，就是数据寄存器的2倍；

RegVal\_Hi+RegVal\_Lo：要写的数据高8位字节和低8位字节，这些数据数据个数可能不止一个，具体由寄存器决定；

CRC\_Lo+CRC\_Hi ：前面数据（地址+0x10+StartAddr\_Hi+StartAddr\_Lo+Quantity of Registers\_Hi+Quantity of Registers\_Lo + Count + RegVal\_Hi+RegVal\_Lo）的CRC16校验码；

**返回：**

发什么返回什么，即发什么数据就原样返回什么样的数据；

举例：01 10 00 48 00 01 02 00 01 68 18

表示往地址为01处写数据，要写的数据开始存放的地址为00 48，要写的数据一共有00 01个寄存器，即一个寄存器，要写的数据字节数为02，要写的数据为00 01,前面的数据的CRC16校验码为68 18；

# 2.各模块数据通信协议说明

**注意：各模块的通信及解析方式是比赛重点，需要参赛选手熟练掌握，此文档在比赛现场电脑上会有提供，所以以下内容无需强行记忆。**

下面串口数据通信时的串口参数为9600 8 N 1，即波特率为9600，8位数据位，无检验，1位停止位，16进制数据收发。每条数据的后两位均为CRC16校验位。

## 2.1 温湿度传感节点指令

发送：01 03 00 14 00 02 84 0f

返回：01 03 04 01 FF 00 FF 8b bf

01 FF：就是采集的湿度值

00 FF：就是采集的温度值

湿度计算：（01 \* 256） + 0xFF = 511, 则湿度的实际值为51.1RH%

温度计算：（00 \* 256） + 0xFF = 255，则温度的实际值为25.5℃

用手触摸温湿度传感器，一般情况下湿度和温度的值会上升，湿度值上升快。

## 2.2 烟雾传感节点指令

发送：01 03 00 34 00 01 c5 c4

返回：01 03 02 00 47 F8 76

00 47：就是采集的烟雾值，即烟雾值为71，如果在烟雾传感器旁边有新增加烟雾，那么这个值一般会上升，安全起见用口持续吹气给烟雾传感器也可以看到烟雾值变化，这里的烟雾值仅供参考。

## 2.3 光照传感节点指令

发送：01 03 00 2A 00 01 a5 c2

返回：01 03 02 02 5E 38 DC

02 5E ：就是采集的光照值，即光照值为606，用手盖住光敏元件，那么这个值会变化，这里的光照值仅供参考。

## 2.4 人体传感器节点指令

发送：01 03 00 3c 00 01 44 06

返回有人：01 03 02 00 01 79 84

返回无人：01 03 02 00 00 B8 44

注意：检测时候，人体传感节点上面有个球状的白色透镜，透镜前面能看到透镜10米内不能有人，这时候当有人或者用手在这个白色透镜前面移动，发送检测指令，就能检测到有人，而且这个有人状态会持续8秒钟左右；如果透镜前面没有移动的人则检测无人。

## 2.5 风扇（直流电机）节点指令

打开发送：01 10 00 48 00 01 02 00 01 68 18

打开返回：01 10 00 48 00 01 02 00 01 68 18

关闭发送：01 10 00 48 00 01 02 00 02 28 19

关闭返回：01 10 00 48 00 01 02 00 02 28 19

把风扇的两P插头插到风扇（直流电机）控制节点左上角的两P接头上面去。

## 2.6 数码管输出节点指令

数码管输出节点上面有一个带小数点的四位数码管，数码管显示分带小数点和不带小数点显示两种，数码管从左往右的位分别用四个字节数据表示，每位数据的高4位为1表示不带小数点显示，高4位为0则表示带小数点显示，每位数据的低4位就是在该位显示的数据，低4位数据可以显示为0、1、2、3、4、5、6、7、8、9。

例如要显示213.7 可以表示为：12 11 03 17

发送：01 10 00 5E 00 02 04 12 11 03 17 62 9C

返回：01 10 00 5E 00 02 04 12 11 03 17 62 9C

例如要显示9876 可以表示为：19 18 17 16

发送：01 10 00 5E 00 02 04 19 18 17 16 7E 7A

返回：01 10 00 5E 00 02 04 19 18 17 16 7E 7A

## 2.7 灯光蜂鸣器控制节点指令

灯光蜂鸣器控制节点上面有一个蜂鸣器和一个RGB LED灯，用4个字节来分别表示蜂鸣器、LED灯的Red、LED灯的Green、LED灯的Blue，每个字节为01则表示相应的设备启动，00则表示不启动。

例如只打开蜂鸣器：01 00 00 00

发送：01 10 00 5A 00 02 04 01 00 00 00 77 10

返回：01 10 00 5A 00 02 04 01 00 00 00 77 10

例如要打开蜂鸣器和LED灯的Red：01 01 00 00

发送：01 10 00 5A 00 02 04 01 01 00 00 26 d0

返回：01 10 00 5A 00 02 04 01 01 00 00 26 d0

例如关闭蜂鸣器，打开LED灯的Red，打开LED灯的Green，打开LED灯的Blue：00 01 01 01

发送：01 10 00 5A 00 02 04 00 01 01 01 e7 7c

返回：01 10 00 5A 00 02 04 00 01 01 01 e7 7c

例如单独开LED灯的Green：00 00 01 00

发送：01 10 00 5A 00 02 04 00 00 01 00 77 7c

返回：01 10 00 5A 00 02 04 00 00 01 00 77 7c

## 2.8 PM2.5传感节点指令

把PM2.5传感器的6P的插头插到PM2.5传感节点右上角的6P接头上。

发送：01 03 00 58 00 01 05 d9

返回：01 03 02 00 12 38 49

返回数据中00 12就是PM2.5的测量值，即PM2.5= 18微克每立方米，用嘴往PM2.5传感器中间圆孔吹气或者有烟雾通过则PM2.5的值会变化，一般是上升，这里的PM2.5值仅供参考。

## 2.9 窗帘（步进电机）控制节点指令

把步进电机的5P插头插到窗帘（步进电机）控制节点左上角5P接头上。用两个字节分别表示正反转和转的步数；第一个字节为02表示正转，为01表示反转，第二个字节表示转多少步，步数范围：1~~255步。注意：每条指令发送以后，等待步进电机转到停止以后再发下一条指令。

例如：

正转BB步（即十进制187步）发送：01 10 00 4A 00 01 02 02 BB e9 29

正转BB步（即十进制187步）返回：01 10 00 4A 00 01 02 02 BB e9 29

反转88步（即十进制136步）发送：01 10 00 4A 00 01 02 01 88 a9 cc

反转88步（即十进制136步）返回：01 10 00 4A 00 01 02 01 88 a9 cc

# 3.高频RFID节点指令举例说明

**注意：各模块的通信及解析方式是比赛重点，需要参赛选手熟练掌握，此文档在比赛现场电脑上会有提供，所以以下内容无需强行记忆。**

注意：下面串口数据通信时的串口参数为19200 8 N 1，即波特率为19200，8位数据位，无检验，1位停止位，16进制数据收发。

把高频卡放在高频RFID节点天线区，高频读卡距离小于4CM。

数据格式：2字节的命令头 + 2字节的数据长度 + 数据 + 1字节检验码

2字节的命令头：固定为AA BB。

2字节的数据长度：数据长度后面所有的字节个数，即数据字节个数加1字节检验码之和。

1字节校验码：所有“数据”进行按位异或得出的校验码，注意2字节的命令头和2字节的数据长度不参与按位异或。

高频数据格式举例：

AA BB 06 00 00 00 01 02 52 51

AA BB：命令头

06 00：数据长度，低字节在前，高字节在后，表示06 00这两个字节后面所有的字节个数，即6个字节。

51：是数据长度后面的数据经过按位异或操作（XOR运算）的结果，

例如：00 XOR 00 XOR 01 XOR 02 XOR 52 按位异或运算结果为51.

## 3.1 读卡号操作

**第一步：**

发送：

寻卡：AA BB 06 00 00 00 01 02 52 51

返回：

寻卡正常：AA BB 08 00 00 00 01 02 00 04 00 07

寻卡异常：AA BB 06 00 00 00 01 02 14 14

**第二步：**

发送：

AA BB 06 00 00 00 02 02 04 04

返回：

寻卡号正常：AA BB 0A 00 00 00 02 02 00 **5A EF C6 04** 77

寻卡号异常：AA BB 06 00 00 00 02 02 0A 0A

其中5A EF C6 04 就是读取的卡号，不同的卡的卡号是不同的，这样卡号后面的检验码也不同。

## 3.2 读卡的块数据操作

**第一步：**

发送：

寻卡：AA BB 06 00 00 00 01 02 52 51

返回：

寻卡正常：AA BB 08 00 00 00 01 02 00 04 00 07

寻卡异常：AA BB 06 00 00 00 01 02 14 14

**第二步：**

发送：

AA BB 06 00 00 00 02 02 04 04

返回：

寻卡号正常：AA BB 0A 00 00 00 02 02 00 **5A EF C6 04** 77

寻卡号异常：AA BB 06 00 00 00 02 02 0A 0A

其中5A EF C6 04 就是读取的卡号，不同的卡的卡号是不同的，这样卡号后面的检验码也不同。

**第三步：**

发送：

选定卡：AA BB 09 00 00 00 03 02 **5A EF C6 04** 76

选定上一步读到的卡号，不同的卡的卡号是不同的，其后面的校验码一般也是不同的。

返回：

返回正常：AA BB 07 00 00 00 03 02 00 08 09

返回异常：AA BB 06 00 00 00 03 02 0A 0A

**第四步：**

发送选定块命令：AA BB 0D 00 00 00 07 02 60 01 FF FF FF FF FF FF 64

返回正常：AA BB 06 00 00 00 07 02 00 05

返回异常：AA BB 06 00 00 00 07 02 16 16

**第五步：**

发送读块命令：AA BB 06 00 00 00 08 02 01 0B

返回正常：AA BB 16 00 00 00 08 02 **00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00** 0A

其中00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 就是块数据，一般全新的卡的块数据会是这样，当然不同卡的块数据一般是不同的。

返回异常：AA BB 06 00 00 00 08 02 17 17

## 3.3 写卡的块数据操作

**第一步：**

发送：

寻卡：AA BB 06 00 00 00 01 02 52 51

返回：

寻卡正常：AA BB 08 00 00 00 01 02 00 04 00 07

寻卡异常：AA BB 06 00 00 00 01 02 14 14

**第二步：**

发送：

AA BB 06 00 00 00 02 02 04 04

返回：

寻卡号正常：AA BB 0A 00 00 00 02 02 00 **5A EF C6 04** 77

寻卡号异常：AA BB 06 00 00 00 02 02 0A 0A

其中5A EF C6 04 就是读取的卡号，不同的卡的卡号是不同的，这样卡号后面的检验码也不同。

**第三步：**

发送：

选定卡：AA BB 09 00 00 00 03 02 **5A EF C6 04** 76

选定上一步读到的卡号，不同的卡的卡号是不同的。

返回：

返回正常：AA BB 07 00 00 00 03 02 00 08 09

返回异常：AA BB 06 00 00 00 03 02 0A 0A

**第四步：**

发送选定块命令：AA BB 0D 00 00 00 07 02 60 01 FF FF FF FF FF FF 64

返回正常：AA BB 06 00 00 00 07 02 00 05

返回异常：AA BB 06 00 00 00 07 02 16 16

**第五步：**

假如我们要往块里面写“01 0102030405060708090A0B0C0D0E0F10”

发送写块命令：AA BB 16 00 00 00 09 02 01 0102030405060708090A0B0C0D0E0F10 1A

返回正常：AA BB 06 00 00 00 09 02 00 0B

返回异常：AA BB 06 00 00 00 09 02 18 18

# 4.蓝牙射频板配置指令说明

注意：此部分内容比赛期间**不会**涉及；

本固件是蓝牙实现主从一体的功能，可以通过AT命令进行配置模块的工作模式。模块启动默认初始化的工作参数为：

|  |  |  |  |
| --- | --- | --- | --- |
| 串口 | 设备名称 | 主从模式 |  |
| 9600，N，8，1 | FRO\_BLE4.0 | 从机 |  |

注意：模块的配置一旦修改就会保存到flash(断电保存)，所以下次开机就是你改过的配置。

如果模块配置成主机，则将自动扫描周围从机设备自动连接并保存连接的该从机设备；下次开机自动寻找该从机设备进行连接。如需连接其它从机，则需要发送AT+CLEAR\r\n清除主设备配对信息。

AT命令（注意：AT命令一定要以**回车换行**结束）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 指令功能 | 指令 | 应答 | 参数 | 举例 |
| AT测试 | AT | OK | 无 | AT\r\n |
| 查询串口波特率 | AT+BAUD? | OK+Get:[para1] | Para1：0～4  0=9600;1=19200;2=38400;3=57600;4=115200  Default：0（9600) | AT+BAUD?\r\n |
| 设置串口波特率 | AT+BAUD[para1] | OK+Set:[para1] | AT+BAUD1\r\n(设置为19200) |
| 查询设备名称 | AT+NAME? | OK+NAME[para1] | Para1：设备名称  最长 11 位数字或字母，  含中划线和下划线，不建  议用其它字符。  Default：FRO\_BLE4.0 | AT+NAME?\r\n |
| 设置设备名称 | AT+NAME[para1] | OK+Set:[para1] | AT+NAMEFRO123\r\n(蓝牙名称设置为FRO123) |
| 恢复默认设置 | AT+RENEW | OK+RENEW |  | AT+RENEW\r\n |
| 模块复位，重启 | AT+RESET | OK+RESET |  | AT+RESET\r\n |
| 查询主从模式 | AT+ROLE? | OK+ Get:[para1] | Para1: 0 ~ 1  1: 主设备  0: 从设备  Default: 0 | AT+ROLE?\r\n |
| 设置主从模式 | AT+ROLE[para1] | OK+ Set:[para1] | AT+ROLE1\r\n(设置成主机) |
| 清除主设备配对信息 | AT+CLEAR | OK+CLEAR |  | AT+CLEAR\r\n  (此指令只有在主设备时才有效；从设备时不接受此指令  ) |
| 查询软件版本 | AT+VERS? | 版本信息 |  | AT+VERS?\r\n |
| 读取 RSSI 信号值 | AT+RSSI? | OK+ RSSI:[para] | Para:信号强度，单位为 db  Para 是一个负数，绝对值越  小说明信号强度越大 | AT+RSSI?\r\n |
|  |  |  |  |  |