**UP-RFID-RT型综合教学平台**

**13.56MHz模块串口协议(14443A)**

**博创智联科技有限公司**

**2016-08-15**

**13.56MHz\_14443A串口通讯协议**

## **指令集合**

这部分主要介绍13.56MHz模块支持的14443协议下的指令汇总。按照13.56MHz模块的工作端分成两个部分介绍，ISO 14443A协议指令是13.56MHz模块和14443A标签之间的通信协议指令，串口协议指令是13.56MHz模块与串口终端（如PC端、ARM端）之间的通信协议指令。14443A国际标准协议是对通信规则的标准，没有对PICC（非接触式耦合卡）和PCD（耦合设备）之间通信规定特定的命令。下面14443A的命令是基于14443A协议并结合PHILIPS公司的芯片资料而提取出的命令。

## **ISO 14443A协议指令**

**表1.1 ISO 14443A协议指令**

|  |  |  |
| --- | --- | --- |
| **指令标识** | **指令代码（Hex）** | **说明** |
| PICC\_REQIDL | 26 | 寻天线区内未进入休眠状态 |
| PICC\_REQALL | 52 | 寻天线区内全部卡 |
| PICC\_ANTICOLL1 | 93 | 防冲撞（一级串联） |
| PICC\_ANTICOLL2 | 95 | 防冲撞（二级串联） |
| PICC\_AUTHENT1A | 60 | 验证A密钥 |
| PICC\_AUTHENT1B | 61 | 验证B密钥 |
| PICC\_READ | 30 | 读块 |
| PICC\_WRITE | A0 | 写块 |
| PICC\_DECREMENT | C0 | 扣款 |
| PICC\_INCREMENT | C1 | 充值 |
| PICC\_RESTORE | C2 | 调块数据到缓冲区 |
| PICC\_TRANSFER | B0 | 保存缓冲区中数据 |
| PICC\_HALT | 50 | 休眠（暂停） |
| PICC\_RESET | E0 | 复位 |

注：表1.1是读写器上STC芯片和LCRC632芯片之间的通信命令，而实际使用这些命令的是PICC和PCD之间的交互。这些通信命令与标签相关，该标签工作在ISO 14443A协议上，具体信息可参照标签资料（PHILIPS的资料）。

## **串口协议指令**

**表1.2 串口协议指令**

|  |  |  |
| --- | --- | --- |
| **指令标识** | **指令代码（Hex）** | **功能** |
| REQUEST\_A | 0201 | 寻卡A卡 |
| ANTICOLL | 0202 | A卡防冲突 |
| SELECT | 0203 | 锁定A卡 |
| SLEEP | 0204 | 休眠A卡 |
| AUTHENTICATION | 0207 | A卡密钥验证 |
| M1READ | 0208 | 读M1卡命令 |
| M1WRITE | 0209 | 写M1卡 |
| M1INITVAL | 020A | 初始化M1卡 |
| M1READVAL | 020B | 读取M1卡 |
| M1DECREAMENT | 020C | M1卡扣款 |
| M1INCREAMENT | 020D | M1卡充值 |
| ANT\_CONTROL | 010C | 天线控制 |
| GET\_HARDMODEL | 0104 | 获取版本号 |
| SET\_BAUDRATE | 0101 | 设置波特率 |

注:M1卡也就是A型卡的一种，这里的M1卡指模拟的一卡通、钱包等功能的卡。

## **指令说明**

## **ISO 14443A协议指令说明**

ISO 14443A协议使用的是国际标准，详细说明参照HF13.56M/国际标准/ISO 14443A部分的文档。

## **串口协议指令说明**

13.56MHz模块作为其他主机（如PC）的外设通过串口通信，串口默认波特率为19200。此处详细说明13.56MHz模块14443A协议下的串口指令以及各个指令帧结构。模块接收到的命令帧结构大致分为6部分，主要分为帧头、校验和、参数以及命令字，具体如表2.1所示。

**表2.1 命令帧数据结构**

|  |  |  |
| --- | --- | --- |
| **字段** | **值** | **说明** |
| SOF | 0xAABB | 主机（如PC）与13.56MHz模块通信的命令帧起始字 |
| Lenth | 0xXXXX | 一帧命令的长度，从DEV\_ID（包含）开始到FCS（包含）结束为止的总长度 |
| Dev\_ID | 0x0000 | 设备编号，用于扩展使用 |
| CMD | 0xXXXX | 命令字，如寻卡命令字为0x0201 |
| Status/Flag | 0xXX | 状态字节或/和标志字节，主要在响应帧中出现 |
| VData | … | 可变字节的负载，发送时可以携带参数，接收时为响应数据 |
| FCS | 0xXX | 校验和，从LENGTH（不包含）开始到FCS（不包含）结束的所有字节的异或值 |

注：请求和响应帧结构一致，最后校验和是字节，VDATA也是按照字节计算，其余均为字，低位在前，高位在后。

1. REQUEST\_A**命令**

REQUEST\_A命令字是0x0201，当13.56MHz模块通过串口接收到REQUEST\_A命令后执行ISO 14443A协议下的寻卡命令。13.56MHz模块可以寻找到处在天线辐射范围内的标签。REQUEST\_A命令帧结构如表2.2所示。

**表2.2 REQUEST\_A命令请求帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | **FCS** |
| AA BB | 05 00 | 00 00 | 01 02 | XX | 10 |

注：VData为请求参数，0x52为寻所有卡，0x26为寻未休眠的卡

13.56MHz模块接收到请求命令并寻卡后会将寻到的卡号和执行的结果状态结果反馈给请求方（如PC），响应帧结构如表2.3所示。

**表2.3 REQUEST\_A命令响应帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status** | **VData** | **FCS** |
| AA BB | 08 00 | 00 00 | 01 02 | XX | XX XX | fcs |

表2.3中，Status字节表示该命令执行是否成功，如果成功为0x00，否则返回错误码。VData为2个字节，为卡片类型代码，类型代码和卡片类型如表2.4所示。

**表2.4 卡片类型**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **代码（hex）** | 4400 | 0400 | 0200 | 0800 | 0403 | 4403 |
| **卡片类型（Mifare\_）** | UltraLight | One（S50） | One（S70） | Pro | ProX | DESFire |

在表2.4中可以看到，REQUEST\_A命令返回的是卡的类型，而不是卡号，命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧 |
| Send | AA BB 06 00 00 00 01 02 52 51 |
| Recv | AA BB 08 00 00 00 01 02 00 04 00 07 |

1. **ANTICOLL 命令**

ANTICOLL命令字是0x0202，当13.56MHz模块通过串口接收到ANTICOLL命令后执行ISO 14443A协议下的A卡防冲撞命令。13.56MHz模块接收到ANTICOLL命令帧结构如表2.5所示。

**表2.5**  ANTICOLL**命令请求帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | **FCS** |
| AA BB | 06 00 | 00 00 | 02 02 | 04 | 04 |

13.56MHz模块接收到请求命令并执行后会将执行结果状态反馈给请求方（如PC），响应帧结构如表2.6所示。

**表2.6** ANTICOLL**命令响应帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status** | VData | **FCS** |
| AA BB | 0A 00 | 00 00 | 02 10 | XX | UID | fcs |

A卡防冲撞命令需紧接着寻卡命令操作，单独使用不能获取卡号（单独使用寻卡命令也不能获取卡号）。在实际使用中REQUEST\_A、ANTICOLL、SELECT命令结合使用，REQUEST\_A获取卡的类型，紧接着发送防冲撞命令，读取卡号并选定该卡。ANTICOLL命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧 |
| Send | AA BB 06 00 00 00 02 02 04 04 |
| Recv | AA BB 0A 00 00 00 02 02 00 30 8F 6C 54 87 |

1. **SELECT**

SELECT命令字是0x0203，当13.56MHz模块通过串口接收到SELECT命令后执行ISO 14443A协议下的选择命令。13.56MHz模块接收到SELECT命令帧结构如表2.7所示。

**表2.7 SELECT命令请求帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | **FCS** |
| AA BB | 09 00 | 00 00 | 03 02 | UID | fcs |

13.56MHz模块接收到请求命令并执行后会将执行结果状态反馈给请求方（如PC），响应帧结构如表2.8所示。

**表2.8 SELECT命令响应帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status** | **VData** | **FCS** |
| AA BB | 07 00 | 00 00 | 03 02 | XX | 08 | fcs |

SELECT命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧 |
| Send | AA BB 09 00 00 00 03 02 30 8F 6C 54 86 |
| Recv | AA BB 07 00 00 00 03 02 00 08 09 |

1. **SLEEP命令**

SLEEP命令字是0x0204，当13.56MHz模块通过串口接收到SLEEP命令后执行ISO 14443A协议下的休眠命令。命令帧结构如表2.9所示。

**表2.9 SLEEP命令请求帧数据**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **FCS** |
| AA BB | 05 00 | 00 00 | 04 02 | 06 |

响应帧结构如表2.10所示。

**表2.10 SLEEP命令响应帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status** | **FCS** |
| AA BB | 06 00 | 00 00 | 04 02 | XX | fcs |

SLEEP命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧 |
| Send | AA BB 05 00 00 00 04 02 06 |
| Recv | AA BB 06 00 00 00 04 02 00 09 |

1. **AUTHENTICATION命令**

AUTHENTICATION命令字是0x0207，当13.56MHz模块通过串口接收到AUTHENTICATION命令进行密钥验证，命令帧结构如表2.11所示。

**表2.11 AUTHENTICATION命令请求帧数据**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | | | **FCS** |
| AA BB | 0D 00 | 00 00 | 07 02 | Mode | Addr | Key | fcs |
| XX | Blockaddr | passwd |

在表2.11请求帧中，VData分为三个部分，第一部分为验证模式（Mode），如果使用密钥A，model = 0x60，如果使用密钥B，model = 0x61；第二部分为某扇区的起始块地址（一字节）；第三部分是密钥（六字节）。对14443A标签读写、一卡通或者钱包功能的充值、扣款、查询余额等功能均需要密钥验证后才能再次操作。13.56MHz模块执行请求命令后会将执行结果状态反馈给请求方（如PC），响应帧结构如表2.12所示。

**表2.12 AUTHENTICATION命令响应帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status** | **FCS** |
| AA BB | 06 00 | 00 00 | 07 02 | XX | fcs |

AUTHENTICATION命令会在读写标签的时候会首先验证，当验证成功才能进行读写操作。AUTHENTICATION命令示例：

|  |  |  |
| --- | --- | --- |
| 操作 | 交互帧 | |
| Send | AA BB 0D 00 00 00 07 02 60 00 FF FF FF FF FF FF 65 | 密钥A，0扇区0块 |
| Recv | AA BB 06 00 00 00 07 02 00 05 | |

1. **M1READ命令**

M1READ命令字是0x0208，用于读取标签某个扇区某个块的内存数据。命令帧结构如表2.13所示。

**表2.13 M1READ命令请求帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | **FCS** |
| AA BB | 13 00 | 00 00 | 08 02 | XX | fcs |

在表2.13请求帧中，VData部分为读取的块绝对地址，块绝对地址 = 扇区\*4 + 该扇区的块地址。响应帧结构如表2.14所示。

**表2.14 M1READ命令响应帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status/Flag** | VData | **FCS** |
| AA BB | 16 00 | 00 00 | 08 02 | XX | Data（16bit） | fcs |

注：每次读写的时候均需要进行密码验证，因此需要先执行AUTHENTICATION命令，再执行M1READ命令。

M1READ命令响应帧的VData部分为读取到的数据，长度固定为16个字节（有标签分区属性决定），M1READ命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧（读取扇区0块0的数据） |
| Send | AA BB 06 00 00 00 08 02 00 0A |
| Recv | AA BB 16 00 00 00 08 02 00 2B 00 00 00 D4 FF FF FF 2B 00 00 00 01 FE 01 FE 21 |

对于钱包数据在内部存储上有一定的规则，将16字节分成4部分，每部分四个字节，第一部分和第三部分一致，第二部分是第一部分按位取反，第四部分可再分成两个相同的部分，第一个字节是第二个字节按位取反。

1. **M1WRITE命令**

M1WRITE命令字是0x0209，用于将数据写入到指定的扇区和块地址处。命令帧结构如表2.15所示。

**表2.15 M1WRITE命令请求帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | | **FCS** |
| AA BB | 0F 00 | 00 00 | 09 02 | Addr | Data | fcs |
| addr | New Data |

写入命令帧的VData部分分为两个部分，第一个部分为写入的地址（块的绝对地址），addr = 扇区号\*4 + Block地址；第二个部分为写入的数据。响应帧结构如表2.16所示。

**表2.16 M1WRITE命令响应帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status/Flag** | **FCS** |
| AA BB | 06 00 | 00 00 | 09 02 | XX | fcs |

M1WRITE命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧（0扇区块2写入16字节0） |
| Send | AA BB 16 00 00 00 09 02 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 09 |
| Recv | AA BB 06 00 00 00 09 02 00 0B |

1. **M1INITVAL命令**

M1INITVAL命令字是0x020A，用于模拟一卡通初始化操作。命令帧结构如表2.17所示。

**表2.17 M1INITVAL命令请求帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | | **FCS** |
| AA BB | 0A 00 | 00 00 | 0A 02 | Addr | Data | fcs |
| Block addr | Value |

在表2.17请求帧中，VData分为两个部分，第一部分为块绝对地址（Addr），addr = 扇区号\*4 + Block地址；第二部分为初始化的值，该值为整数的十六进制表示。响应帧结构如表2.18所示。

**表2.18 M1INITVAL命令响应帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status/Flag** | **FCS** |
| AA BB | 06 00 | 00 00 | 0A 02 | XX | fcs |

M1INITVAL命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧（0扇区1块初始化23元） |
| Send | AA BB 0A 00 00 00 0A 02 01 17 00 00 00 1E |
| Recv | AA BB 06 00 00 00 0A 02 00 08 |

1. **M1INCREMENT命令**

M1INCREMENT命令字是0x020D，用于给模拟的一卡通或者钱包充值。命令帧结构如表2.19所示。

**表2.19 M1INCREMENT命令请求帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | | **FCS** |
| AA BB | 0A 00 | 00 00 | 0D 02 | Addr | Data | fcs |
| Block addr | Value |

在表2.19请求帧中，VData分为两个部分，第一部分为块绝对地址（Addr），addr = 扇区号\*4 + Block地址；第二部分为充值的值，该值为整数的十六进制表示（四字节，低位在前，高位在后）。响应帧格式如表2.20所示。

**表2.20 M1INCREMENT命令响应帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status/Flag** | **FCS** |
| AA BB | 06 00 | 00 00 | 0D 02 | XX | fcs |

M1INCREMENT命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧（0扇区1块充值12元） |
| Send | AA BB 0A 00 00 00 0D 02 01 0C 00 00 00 1E |
| Recv | AA BB 06 00 00 00 0D 02 00 0F |

1. **M1DECREMENT命令**

M1DECREMENT命令字是0x020C，模拟一卡通或者钱包扣款或者消费。命令帧结构如表2.20所示。

**表2.21 M1DECREMENT命令请求帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | | **FCS** |
| AA BB | 0A 00 | 00 00 | 0C 02 | Addr | Data | fcs |
| BlockAddr | DecData |

在表2.21请求帧中，VData分为两部分，第一部分为块绝对地址（Addr），addr=扇区号\*4+Block地址；第二部分为扣款的值，该值为整数的十六进制表示（四字节，低位在前，高位在后）。响应帧格式如表2.22。

**表2.22 M1DECREMENT命令响应帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status/Flag** | **FCS** |
| **AA BB** | **06 00** | **00 00** | **0C 02** | **XX** | **fcs** |

M1DECREMENT命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧（0扇区1块扣款1元） |
| Send | AA BB 0A 00 00 00 0C 02 01 01 00 00 00 0E |
| Recv | AA BB 06 00 00 00 0C 02 00 0E |

1. **M1READVAL 命令**

M1READVAL 命令字是0x020B，用于查询模拟卡内的余额。命令帧结构如表2.23所示。

**表2.23 M1READVAL命令请求帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | **FCS** |
| AA BB | 0E 00 | 00 00 | 0B 02 | BlockAddr | fcs |

在表2.23请求帧中，VData为读取的绝对块地址，addr = 扇区号\*4 + Block地址。响应帧结构如表2.24所示。

**表2.24 M1READVAL命令响应帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status** | VData | **FCS** |
| AA BB | 06 00 | 00 00 | 0B 02 | XX | Data | fcs |

表2.24中Status为余额查询成功状态位，如果失败则为错误码，VData为余额值，十六进制的整数值。

M1READVAL 命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧（查询扇区0块1处的余额，查询值为32元） |
| Send | AA BB 06 00 00 00 0B 02 01 08 |
| Recv | AA BB 0A 00 00 00 0B 02 00 20 00 00 00 29 |

1. **ANT\_CONTROL 命令**

ANT\_CONTROL 命令字是0x010C，控制天线打开和关闭。命令帧结构如表2.25所示。

**表2.25 ANT\_CONTROL命令请求帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | **FCS** |
| AA BB | 06 00 | 00 00 | 0C 01 | XX | fcs |

在表2.25请求帧中，VData为打开天线或关闭天线的参数，00为关闭，否则为打开天线响应帧结构如表2.26所示。

**表2.26 ANT\_CONTROL命令响应帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status** | **FCS** |
| AA BB | 06 00 | 00 00 | 0C 01 | XX | fcs |

ANT\_CONTROL 命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧 |
| Send | AA BB 06 00 00 00 0C 01 00 08 |
| Recv | AA BB 06 00 00 00 0C 01 00 0D |

1. **GET\_HARDMODEL 命令**

GET\_HARDMODEL命令字是0x0104，该命令是对模块操作，因此不涉及到ISO 14443A相关的协议，有无标签在天线识别范围内对该命令无影响，当13.56MHz模块通过串口接收到GET\_HARDMODEL命令帧结构如表2.27所示。

**表2.27 GET\_HARDMODEL命令请求帧数据**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **FCS** |
| AA BB | 05 00 | 00 00 | 04 01 | fcs |

13.56MHz模块获取固件版本信息后将版本信息作为负载反馈给请求端，响应帧结构如表2.28所示。

**表2.28 GET\_HARDMODEL命令响应帧数据**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status/Flag** | **VData** | **FCS** |
| AA BB | 12 00 | 00 00 | 04 01 | XX | … | fcs |

注：VData部分为版本信息的字节数组，无需转换，直接显示即可。

GET\_HARDMODEL命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧 |
| Send | AA BB 05 00 00 00 04 01 05 |
| Recv | AA BB 12 00 00 00 04 01 00 53 4C 36 30 31 46 2D 30 35 31 32 00 40 |

1. **SET\_BAUDRATE 命令**

SET\_BAUDRATE命令字是0x0101，用于对模块操作，不涉及到ISO 14443A相关的协议，有无标签在天线识别范围内对该命令无影响。当模块通过串口接收到SET\_BAUDRATE命令帧结构如表2.29所示。

**表2.29 SET\_BAUDRATE命令请求帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **VData** | **FCS** |
| AA BB | 06 00 | 00 00 | 01 01 | XX | fcs |

在表2.32请求帧中，在VData部分用一个字节代表一个特定的波特率，13.56MHz模块支持的波特率如表2.30所示。

**表2.30 13.56MHz模块支持的波特率**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **编码** | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| **波特率** | 4800 | 9600 | 14400 | 19200 | 28800 | 38400 | 57600 | 115200 |

13.56MHz模块接收到请求帧，VData字节为表2.30中某一个编码，根据编码将模块波特率设置成对应的波特率，在不断电的情况下一只工作在该波特率下。同时用修改之前的波特率将设置结果响应给请求端，响应帧格式如表2.31所示。

**表2.31 SET\_BAUDRATE命令响应帧数据**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **Dev\_ID** | **CMD** | **Status/Flag** | **FCS** |
| AA BB | 06 00 | 00 00 | 01 10 | XX | fcs |

注：波特率的设置同时会影响15693协议，在不断电的情况下15693和14443A必须始终采用一致的波特率进行工作。SET\_BAUDRATE命令示例：

|  |  |
| --- | --- |
| 操作 | 交互帧（设置波特率为19200） |
| Send | AA BB 06 00 00 00 01 01 03 03 |
| Recv | AA BB 06 00 00 00 01 01 00 00 |

**附录**

FCS（Frame check sequence）：

在14443A协议下，每个命令帧最后会追加一个字节作为该帧的校验和，从LENGTH（不包含）开始到FCS（不包含）结束的所有字节的异或值。计算方法（C语言实现）：

//校验值计算

uint8 RC632\_UartCalcFCS( uint8 \*msg\_ptr, uint8 len )

{

uint8 x;

uint8 xorResult;

xorResult = 0;

for ( x = 0; x < len; x++, msg\_ptr++ )

xorResult = xorResult ^ \*msg\_ptr;

return ( xorResult );

}

说明：第一个参数是要计算的字节数组的起始地址，第二个参数为要计算的长度。