**UP-RFID-RT型综合教学平台**

**900MHz模块串口协议**

**(840MHz-960MHz)**

**博创智联科技有限公司**

**2016-08-18**

**900MHz模块串口通讯协议**

## **指令集合**

这部分主要介绍900MHz模块协议下的指令汇总。按照900MHz模块的工作端分成两个部分介绍，ISO 18000-6C/6B协议指令是900MHz模块和Gen2标签之间的通信协议指令，串口协议指令是900MHz模块与串口终端（如PC端、ARM端）之间的通信协议指令。

## **ISO 18000-6C/6B协议**

ISO 18000-6C/6B标准定义了应用于物品管理等方面的RFID技术标准。工作在860MHz到960MHz频段之间，适用于工业、科学、医疗等领域。ISO 18000-6B标准特点：标准成熟、产品稳定、应用广泛；ID好全球唯一；先读取ID号，后读取数据区；1024bits或2048bits的大容量；98Bytes或216Bytes的大用户数据区；支持多标签同时读取（最多同时读取10张标签）；数据读取速度为40Kbps。ISO 18000-6C标准特点：读取速度快，数据率可以达到40Kbps~640Kbps；可以同时读取标签的数量相对于6B标准更多，理论上可以同时读取1000多张；读取时首先读取EPC号码，标签ID需要用读取数据的方式读取；具有多种写保护方式，安全性更强。

ISO 18000-6B/6C协议的区别主要如下：6B一般用在闭环的领域，如烟草等资产管理领域；6C一般用在开环的领域，比如物流行业的资源管理。6B和6C同时读取标签数量的差异，因此6B用户区数据较长，6C用户区数据相对较短。在标签成本上来说6C相对于6B的标签来说更便宜。

EPC C1G2标签采用ISO 18000-6C标准，基本可以说ISO 18000-6C是EPC C1G2的子集，自2005年EPC C1G2申请成为ISO 18000-6标准到2006年正式加入到ISO 18000-6（ISO 18000-6C）标准后，ISO/EPC 18000-6标准得到了全面的更新与完善（EPC协议没有AFI，有EPC编号）。

UP-TECH RFID 900MHz模块采用的ISO 18000-6C标准，可以手动调整模块的功率、工作频率，工作频率在840MHz到960MHz之间均可调，因此可支持中国标准、欧洲标准、定频915MHz。在实际工作过程中，对模块功率进行调整后可能会修改其工作频率，由于修改工作功率，对标签的识别距离有影响。

## **串口协议指令**

**表1.1 串口协议指令**

|  |  |  |  |
| --- | --- | --- | --- |
| **指令标识** | **指令代码（Hex）** | **功能** | **响应等待时间（ms）** |
| UHFCMD\_GET\_STATUS | 00 | 询问状态 | 200 |
| UHFCMD\_GET\_POWER | 01 | 读取功率 | 200 |
| UHFCMD\_SET\_POWER | 02 | 设置功率 | 200 |
| UHFCMD\_GET\_FRE | 05 | 读取频率 | 200 |
| UHFCMD\_SET\_FRE | 06 | 设置频率 | 200 |
| UHFCMD\_GET\_VERSION | 07 | 读取版本信息 | 200 |
| UHFCMD\_INVENTORY | 10 | 识别标签（单标签识别） | 200 |
| UHFCMD\_INVENTORY\_ANTI | 11 | 识别标签（防碰撞识别） | 200 |
| UHFCMD\_STOP\_GET | 12 | 停止操作 | 200 |
| UHFCMD\_READ\_DATA | 13 | 读取标签数据 | 200 |
| UHFCMD\_WRITE\_DATA | 14 | 写入标签数据 | 200 |
| UHFCMD\_ERASE\_DATA | 15 | 擦除标签数据 | 200 |
| UHFCMD\_LOCK\_MEM | 16 | 锁定标签 | 200 |
| UHFCMD\_KILL\_TAG | 17 | 销毁标签 | 200 |
| UHFCMD\_INVENTORY\_SINGLE | 18 | 识别标签（单步识别） | 200 |
| UHFCMD\_WIEGAND\_INVENTORY | 19 | 韦根识别 | 200 |
| UHFCMD\_SINGLE\_READ\_DATA | 20 | 读取标签数据（不指定UII） | 200 |
| UHFCMD\_SINGLE\_WRITE\_DATA | 21 | 写入标签数据（不指定UII） | 200 |

## **指令说明**

## **ISO 18000-6C/6B协议指令说明**

ISO 18000-6C/6B协议使用的是国际标准，详细说明参照UHF-900M部分的文档。

## **串口协议指令说明**

900MHz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57600。此处详细说明900MHz模块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，具体如表2.1所示。

**表2.1 命令帧数据结构**

|  |  |  |
| --- | --- | --- |
| **字段** | **值** | **说明** |
| SOF | 0xAA | 主机（如PC）与900MHz模块通信的命令帧起始字节，固定为0xAA |
| Lenth | 0xXX | 一帧命令的长度，从Lenth（不包含）开始到EOF（包含）结束为止的总长度 |
| CMD | 0xXX | 命令字节，如寻卡命令字为0x18 |
| Status | 0xXX | 状态字节，主要在响应帧中出现，表示请求命令是否执行成功 |
| 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 |
| EOF | 0x55 | 帧结束标志，固定为0x55 |

其中CMD字段定义如表2.2所示。

**表2.2 CMD字段的定义**

|  |  |  |
| --- | --- | --- |
| **位** | Bit7 | Bit6~Bit0 |
| **描述** | 0：数据包中没有CRC-16  1：数据包中带有CRC-16 | 见表1.1 |

可以通过CMD的最高位设置该命令帧是否使用CRC-16进行验证，如果请求命令带有CRC-16验证功能，则响应帧中也带有CRC-16的验证功能（本协议中不使用CRC校验）。其中Status字段也有对CRC验证成功与否的标志，Status字段的定义如表2.3所示。

**表2.3 Status字段的定义**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **位** | Bit7 | Bit6 | Bit5 | Bit4 | Bit3~Bit0 |
| **描述** | 1：命令执行失败  0：命令执行成功 | 1：CRC验证失败  0：CRC验证成功 | 保留 | 保留 | 0000：其他错误  0011：存储空间溢出  0100：存储空间被锁定  1011：电力不足  1111：不明错误 |

为了避免数据中出现SOF、EOF字节，实际通信过程中利用插入字节保证SOF和EOF的唯一性。当发送数据包的SOF和EOF之间出现0xAA、0x55、0xFF字节时，发送方应在该字节前插入一个0XFF字节。接收方接收到包含插入字节的数据后应删除插入字节并提取有效数据，无论是发送方还是接受放插入字节不计入帧长度，示例如表2.4所示。

**表2.4 插入字节示例**

|  |  |
| --- | --- |
| 需要发送的数据包（hex） | AA 04 55 00 01 55 |
| 实际发送的数据包（hex） | AA 04 FF 55 00 01 55 |
| 需要发送的数据包（hex） | AA 05 00 00 01 AA 55 |
| 实际发送的数据包（hex） | AA 05 00 00 01 FF AA 55 |
| 需要发送的数据包（hex） | AA 06 00 00 01 AA FF 55 |
| 实际发送的数据包（hex） | AA 06 00 00 01 FF AA FF FF 55 |

1. **UHFCMD\_GET\_STATUS 命令**

UHFCMD\_GET\_STATUS命令字是0x00，用来连接的时候查询900MHz模块的状态，主要在串口连接900MHz模块的时候使用。当900MHz模块通过串口接收到UHFCMD\_GET\_STATUS命令，请求帧结构如表2.5所示。

**表2.5 UHFCMD\_GET\_STATUS命令请求帧结构**

|  |  |  |  |
| --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **EOF** |
| AA | 02 | 00 | 55 |

900MHz模块接收到请求命令并获取状态后会将状态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2.6所示。

**表2.6 UHFCMD\_GET\_STATUS命令响应帧结构**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | **EOF** |
| AA | 03 | 00 | XX | XX | 55 |

UHFCMD\_GET\_STATUS命令执行成功才有响应，否则无响应。UHFCMD\_GET\_STATUS命令示例如表2.7所示。

**表2.7 UHFCMD\_GET\_STATUS命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 02 00 55 | 请求连接模块 |
| Recv | AA 03 00 00 55 | 连接成 |

1. **UHFCMD\_GET\_POWER命令**

UHFCMD\_GET\_POWER命令字是0x01，用来读取900MHz模块的当前工作功率。当900MHz模块通过串口接收到UHFCMD\_GET\_POWER命令帧结构如表2.8所示。

**表2.8 UHFCMD\_GET\_POWER命令请求帧结构**

|  |  |  |  |
| --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **EOF** |
| AA | 02 | 01 | 55 |

900MHz模块接收到请求命令并读取当前功率后会将功率和执行的状态结果反馈给请求方（如PC），响应帧结构如表2.9所示。

**表2.9 UHFCMD\_GET\_POWER命令响应帧结构**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | **EOF** |
| AA | 04 | 01 | XX | XX | 55 |

UHFCMD\_GET\_POWER响应命令帧中Status用于表示成功与否，VData字节为模块当前的功率，实际功率为VData字节减0x80。UHFCMD\_GET\_POWER命令示例如表2.10所示。

**表2.10 UHFCMD\_GET\_POWER命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 02 01 55 | 请求读取功率 |
| Recv | AA 04 01 00 8A 55 | 成功，当前功率=0x8A – 0x80 = 0x0A = 10 dBm |

1. **UHFCMD\_SET\_POWER命令**

UHFCMD\_SET\_POWER命令字是0x02，用来设置900MHz模块的当前工作功率（如果没有设置则使用默认功率工作，修改功率可能修改工作频率范围并影响标签识别距离）。当900MHz模块通过串口接收到UHFCMD\_SET\_POWER命令帧结构如表2.11所示。

**表2.11 UHFCMD\_SET\_POWER命令请求帧结构**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **VData** | | **EOF** |
| AA | 04 | 02 | OPTION | POWER | 55 |

在请求帧VData中为2个字节，第一个字节为选择字段（OPTION），第二个字节为设置的实际功率值（例如设置为11dBm，则为0B）。OPTIONN字段的定义如表2.12所示。

**表2.12 OPTION字段的定义**

|  |  |  |
| --- | --- | --- |
| **位** | Bit7~Bit1 | Bit0 |
| **描述** | 保留 | 1：POWER的BIT6~0有效 |

900MHz模块接收到请求命令并设置模块功率后会将功率设置的状态结果反馈给请求方（如PC），响应帧结构如表2.13所示。

**表2.13 UHFCMD\_SET\_POWER命令响应帧结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **EOF** |
| AA | 03 | 02 | XX | 55 |

UHFCMD\_GET\_POWER命令示例如表2.14所示。

**表2.14 UHFCMD\_SET\_POWER命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 04 02 01 0B 55 | 设置当前模块功率为11dBm |
| Recv | AA 03 02 00 55 | 设置成功 |

1. **UHFCMD\_GET\_FRE命令**

UHFCMD\_GET\_FRE命令字是0x05，用来获取900MHz模块的当前工作频率。当900MHz模块通过串口接收到UHFCMD\_GET\_FRE命令帧结构如表2.15所示。

**表2.15 UHFCMD\_GET\_FRE命令请求帧结构**

|  |  |  |  |
| --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **EOF** |
| AA | 02 | 05 | 55 |

900MHz模块接收到请求命令并读取模块频率后会将当前工作频率信息以及获取成功的状态标志反馈给请求方（如PC），响应帧结构如表2.16所示。

**表2.16 UHFCMD\_GET\_FRE命令响应帧结构**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | | | | | | **EOF** |
| AA | 0A | 05 | 00 | FREMODE | FREBASE | BF | CN | SPC | FREHOP | 55 |
| 1bit | 1bit | 2bit | 1bit | 1bit | 1bit |

在UHFCMD\_GET\_FRE命令的响应帧中，VData包含六部分信息，FREMODE为标准（读取频率的时候可能该字段不能正真区分），FREBASE为频率基数，BF为起始频率，CN为频道数，SPC为频道带宽，FREHOP为跳频书序方式，具体关系和意义可参照UHFCMD\_SET\_FRE部分，这里不详述。

UHFCMD\_GET\_FRE命令示例如表2.17所示。

**表2.17 UHFCMD\_GET\_FRE命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 02 05 55 | 请求读取频率 |
| Recv | AA 0A 05 00 00 01 73 01 0A 04 00 55 | 成功，工作频率为920.125MHz |

1. **UHFCMD\_SET\_FRE命令**

UHFCMD\_SET\_FRE命令字是0x06，用来设置900MHz模块的当前工作频率。当900MHz模块通过串口接收到UHFCMD\_SET\_FRE命令帧结构如表2.18所示。

**表2.18 UHFCMD\_SET\_FRE命令请求帧结构**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **VData** | | | | | | **EOF** |
| AA | 0A | 06 | FREMODE | FREBASE | BF | CN | SPC | FREHOP | 55 |
| 1bit | 1bit | 2bit | 1bit | 1bit | 1bit |

请求帧的VData部分和读取频率的响应帧类似，VData内包含该模块与频率相关的所有信息。频率设置有六个参数：频率工作模式（FREMODE）、频率基数（FREBASE）、起始频率（BF）、频道数（CN）、频道带宽（SPC）和跳频顺序方式（FREHOP）。其中频道数是模块在跳频时支持的最大频道个数，频道带宽是每一频道的信道带宽。

900MHz模块支持的频率范围为840MHZ~960MHZ，可以依据应用环境需求，自己定义频率范围。目前允许使用四种频率设置模式：

* “中国标准”模式

该模式为中国的标准，频率范围为840MHz~845MHz或者920MHz~925MHz。该模式下功率和频率的对应关系如表2.19所示。

**表2.19 中国标准下的功率、频率对应表**

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **频率范围（MHz）** | **功率（dBm）** | **功率值（W）** |
| 1 | 920.5~924.5 | 33 | 2 |
| 2 | 840.5~844.5 | 33 | 2 |
| 3 | 920.0~920.5,924.5~925 | 20 | 0.1 |
| 4 | 840.0~840.5，844.5~845 | 20 | 0.1 |

注：频率范围内各个频点之差为频率基数（例如频率基数是125K，则840.5~844.5之间的频点为840.625，840.75，840.875。。。）

* “ETSI标准”模式

该模式采用欧洲标准，有效频率范围为865MHz~868MHz，该模式下功率和频率的对应关系如表2.20所示。

**表2.20 ETSI标准下的功率、频率对应表**

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **频率范围（MHz）** | **功率（dBm）** | **功率值（W）** |
| 1 | 865.0~865.6 | 20 | 0.1 |
| 2 | 867.6~868.0 | 27 | 0.5 |
| 3 | 865.6~867.6, | 33 | 2 |

注：频率范围内各个频点之差为频率基数（例如频率基数是50K，则865~865.6之间的频点为865.05，865.1。。。）

* “定频915MHz”模式

该模式用于设置900M模块工作频率固定在915MHz，定频工作。

* “用户自定义”模式

用户通过设置六个参数进行设置所要的频率工作范围：频率工作模式（FREMODE）、频率基数（FREBASE）、起始频率（BF）、频道数（CN）、频道带宽（SPC）和跳频顺序方式（FREHOP）。各个参数存在如下关系：

(1)、起始频率（BF）=【起始频率（整数部分）】＋【频率基数】×【起 始频率尾数积数】

如：起始频率＝840MHZ ＋125KHZ ×5 ＝840.625MHZ

(2)、频道带宽（SPC）＝【频道带宽积数】×【频率基数】

如：频道带宽（SPC）＝2 ×125KHZ ＝250KHZ

(3)、最终频率＝起始频率（BF）＋（频道数（CN）－1）×频道带宽（SPC）

如：最终频率＝840.625MHZ ＋（16－1）×250KHZ ＝844.375MHZ

(4)、带宽＝最终频率－起始频率（BF）

如：带宽＝844.375MHZ －840.625MHZ ＝3.75MHZ

注：【频率基数】×【频道带宽积数】不能超过1000KHZ；当【频率基数】为50KHZ时，【带宽】不能大于12MHZ，当【频率基数】为125KHZ时，【带宽】不能大于32MHZ。

请求帧的VData部分各个字段定义如表2.21到表2.25所示，FREMODE字段的定义如表2.21所示。

**表2.21 FREMODE字段的定义**

|  |  |  |
| --- | --- | --- |
| **位** | Bit7~Bit4 | Bit3~Bit0 |
| **功能** | 保留 | 0000：中国标准（920-925MHz）  0001：中国标准（840-845MHz）  0010：ETSI 标准  0011：定频模式（915MHz）  0100：用户自定义  其他：中国标准（920-925MHz） |

FREBASE字段的定义如表2.22所示。

**表2.22 FREBASE字段的定义**

|  |  |  |
| --- | --- | --- |
| **位** | Bit7~Bit1 | Bit0 |
| **描述** | 保留 | 频率基数 |
| **功能** | 0：50 KHz  1：125 KHz |

BF字段的定义如表2.23所示。

**表2.23 BF字段的定义**

|  |  |  |  |
| --- | --- | --- | --- |
| **位** | Bit15 | Bit14~Bit5 | Bit4~Bit0 |
| **功能** | 保留 | 起始频率（整数部分） | 起始频率尾数部分积数 |

CN、SPC字段定义如表2.24所示。

**表2.24 CN、SPC字段的定义**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | CN | SPC | |
| 位 | Bit7 ~Bit0 | Bit7 ~Bit4 | Bit3~Bit0 |
| 功能 | 频道数 | 保留 | 频道带宽积数 |

FREHOP字段的定义如表2.25所示。

**表2.25 FREHOP字段的定义**

|  |  |  |  |
| --- | --- | --- | --- |
| 位 | Bit7 ~Bit2 | Bit7 ~Bit4 | Bit1~Bit0 |
| 功能 | 频道数 | 保留 | 00：随机跳频  01：从高往低顺序跳频  10：从低往高顺序跳频  其他：随机跳频 |

900MHz模块接收到请求命令并设置模块频率后会将执行状态标志反馈给请求方（如PC），响应帧结构如表2.26所示。

**表2.26 UHFCMD\_SET\_FRE命令响应帧结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **EOF** |
| AA | 03 | 06 | 00 | 55 |

UHFCMD\_GET\_FRE命令示例如表2.27所示。

**表2.27 UHFCMD\_ SET\_FRE命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 09 06 00 01 73 01 0A 04 00 55 | 请求设置频率为920.125MHz，频道数10 |
| Recv | AA 03 06 00 55 | 设置成功 |

表2.27所示为设置频率实例，在设置频率的请求帧中，VData为00 01 73 01 0A 04 00，根据表2.21到表2.25可以得到：FREMODE= 0x00，中国标准，920MHz~925MHz；FREBASE=0x01，频率基数为125K；BF=0x7301，转化为二进制：0111 0011 0000 0001，根据表2.23，红色部分为整数部分，重新写一下为:11 1001 1000，也就是0x398=920，尾数的积数为0x01，也就是1，而频率基数为125K，因此这里设置的工作频率为920.125MHz；频道数CN=0x0A=10；频道带宽 SPC =0x04=4；跳频方式为随机跳。

1. **UHFCMD\_GET\_VERSION命令**

UHFCMD\_GET\_VERSION命令字是0x07，用来获取900M模块的版本号和序列号。当900MHz模块通过串口接收到UHFCMD\_GET\_VERSION命令帧结构如表2.28所示。

**表2.28 UHFCMD\_GET\_VERSION命令请求帧结构**

|  |  |  |  |
| --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **EOF** |
| AA | 02 | 07 | 55 |

900MHz模块接收到请求命令并获取模块版本信息后会将版本信息反馈给请求方（如PC），响应帧结构如表2.29所示。

**表2.29 UHFCMD\_GET\_VERSION命令响应帧结构**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | | **EOF** |
| AA | 0A | 07 | XX | Serial（6字节） | Version（1字节） | 55 |

900M模块响应帧VData分为两个部分，前面表示序列号，后面表示版本号。UHFCMD\_GET\_VERSION命令示例如表2.30所示。

**表2.30 UHFCMD\_GET\_VERSION命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 02 07 55 | 请求获取版本号信息 |
| Recv | AA 0A 07 00 00 00 00 00 00 00 58 55 | 序列号为000000000000，版本号为5.8 |

1. **UHFCMD\_INVENTORY命令**

UHFCMD\_INVENTORY命令字是0x10，用于单步循环识别标签ID，对单张卡识别一般用该命令。当900MHz模块通过串口接收到UHFCMD\_INVENTORY命令帧结构如表2.31所示。

**表2.31 UHFCMD\_INVENTORY命令请求帧结构**

|  |  |  |  |
| --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **EOF** |
| AA | 02 | 10 | 55 |

900MHz模块接收到请求命令后进行循环识别标签，该命令需与UHFCMD\_STOP\_GET命令结合使用，用于停止循环识别标签。模块识别到标签后将标签的ID获取并反馈给请求方（如PC），响应帧结构如表2.32所示。

**表2.32 UHFCMD\_INVENTORY命令响应帧结构**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | **EOF** |
| AA | XX | 10 | XX | UID | 55 |

900M模块响应帧VData分为两个部分，PCBits和UII。PCBits为协议控制字节，后面紧接着是UII，PCBits的值决定了UII的长度，UHFCMD\_INVENTORY命令示例如表2.33所示。

**表2.33 UHFCMD\_INVENTORY命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 02 10 55 | 请求单步循环识别标签 |
| Recv | AA 05 10 00 04 00 55 | 成功，卡号为 00 04 |

表2.33所示为UHFCMD\_INVENTORY的一次交互示例，当不使用UHFCMD\_STOP\_GET命令，将一直上报Recv帧。UHFCMD\_INVENTORY和后面的UHFCMD\_INVENTORY\_SINGLE命令类似，UHFCMD\_INVENTORY\_SINGLE命令每次只识别一次标签。响应帧的VData部分为UII（卡号），卡号由PCBits + UII 构成，在对指定UII操作的时候也需要携带PCBits，因此在识别卡号处理卡号的时候默认卡号包含PCBits和UII。PCBits详细格式如表2.34所示。

**表2.34 PCBits数据格式的定义**

|  |  |  |  |
| --- | --- | --- | --- |
| **位** | Bits0~Bit4 | Bit5~Bit6 | Bit7~Bit15 |
| **描述** | 以word（两个字节）为单位的PC和UII的总体长度 | 未定义 | NSI(未使用) |

其中UII从低位开始传输，前五位表示PCBits和UII的总长度，卡号长度为协议控制位的值乘以二，PCBits和卡号的长度关系如表2.35所示。

**表2.35 PCBits和卡号长度对应关系**

|  |  |
| --- | --- |
| PCBit0~PCBit4 | PC+UII长度（字节） |
| 00000 | 2 |
| 00001 | 4 |
| 00010 | 6 |
| …. | …. |

1. **UHFCMD\_INVENTORY\_ANTI命令**

UHFCMD\_INVENTORY\_ANT命令字是0x11，用于防碰撞识别标签ID，对多张卡识别采用该命令。当900MHz模块通过串口接收到UHFCMD\_INVENTORY\_ANT命令帧结构如表2.36所示。

**表2.36 UHFCMD\_INVENTORY\_ANT命令请求帧结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | VData | **EOF** |
| AA | 02 | 11 | Q（见附录） | 55 |

900MHz模块接收到请求命令后进行循环识别标签，该命令需与UHFCMD\_STOP\_GET命令结合使用，用于停止循环识别标签。模块识别到标签后将标签的ID获取并反馈给请求方（如PC），响应帧结构如表2.37所示。

**表2.37 UHFCMD\_INVENTORY\_ANT命令响应帧结构**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | **EOF** |
| AA | XX | 11 | XX | UID | 55 |

900M模块响应帧的Length随卡号的长度变化而变化，VData为卡号，格式和UHFCMD\_INVENTORY命令部分一致。

UHFCMD\_INVENTORY\_ANT命令示例如表2.38所示。

**表2.38 UHFCMD\_INVENTORY\_ANT命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 02 11 55 | 请求防碰撞识别标签 |
| Recv | AA 05 11 00 04 00 55 | 成功，卡号为 00 04 |

示例命令帧中各个字段的含义和UHFCMD\_INVENTORY命令的示例含义一致。

1. **UHFCMD\_STOP\_GET命令**

UHFCMD\_STOP\_GET命令字是0x12，用于停止900M模块当前所进行的任何操作，终结当前任何操作后模块处于空闲状态。当900MHz模块通过串口接收到UHFCMD\_STOP\_GET命令帧结构如表2.39所示。

**表2.39 UHFCMD\_STOP\_GET命令请求帧结构**

|  |  |  |  |
| --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **EOF** |
| AA | 02 | 12 | 55 |

900MHz模块接收到请求命令后停止当前的任何操作，然后将处理结果反馈给请求方（如PC），响应帧结构如表2.40所示。

**表2.40 UHFCMD\_STOP\_GET命令响应帧结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **EOF** |
| AA | 03 | 12 | XX | 55 |

UHFCMD\_STOP\_GET命令示例如表2.41所示。

**表2.41 UHFCMD\_STOP\_GET命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 02 12 55 | 请求停止当前所有操作 |
| Recv | AA 03 12 00 55 | 停止操作成功 |

1. **UHFCMD\_READ\_DATA（指定UII）命令**

UHFCMD\_READ\_DATA命令字是0x13，必须由用户指定特定的UII信息才能读取该标签的内部数据。读取标签、写入标签、擦除标签、锁定标签操作的命令帧中需含有存取密码，当存取密码不全为0时，模块利用ACCESS命令确保标签处在SECURED状态后进行相应操作。当900MHz模块通过串口接收到UHFCMD\_READ\_DATA命令帧结构如表2.42所示。

**表2.42 UHFCMD\_READ\_DATA命令请求帧结构**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **VData** | | | | | **EOF** |
| AA | xx | 13 | APWD | BANK | ADDR | CNT | UII | 55 |
| 4bit | 1bit | 1bit | 1bit | n bit |

注：APWD是读写标签等操作的存取密码；BANK是您选择的区编号，如UII为01；ADDR是该区的具体起始地址；CNT是读取等操作的长度（字）；UII为指定的标签ID。

900MHz模块接收到请求命令后处理请求并将结果反馈给请求方（如PC），响应帧结构如表2.43所示。

**表2.43 UHFCMD\_READ\_DATA命令响应帧结构**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | **EOF** |
| AA | XX | 13 | XX | UII | 55 |

UHFCMD\_READ\_DATA命令示例如表2.44所示。

**表2.44 UHFCMD\_READ\_DATA命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 0D 13 00 00 00 00 01 01 01 0C 00 12 34 55 | 请求读取UII的地址1处一个字的数据 |
| Recv | AA 05 13 00 0C 00 55 | 读取成功，数据0C 00 |

1. **UHFCMD\_WRITE\_DATA（指定UII）命令**

UHFCMD\_WRITE\_DATA命令字是0x14，必须由用户指定特定的UII信息才能将数据写入该标签。读取标签、写入标签、擦除标签、锁定标签操作的命令帧中需含有存取密码，当存取密码不全为0时，模块利用ACCESS命令确保标签处在SECURED状态后进行相应操作。当900MHz模块通过串口接收到UHFCMD\_WRITE\_DATA命令帧结构如表2.45所示。

**表2.45 UHFCMD\_WRITE\_DATA命令请求帧结构**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **VData** | | | | | | **EOF** |
| AA | xx | 14 | APWD | BANK | ADDR | CNT | DATA | UII | 55 |
| 4bit | 1bit | 1bit | 1bit | 2bit | n bit |

注：APWD是读写标签等操作的存取密码；BANK是您选择的区编号，如UII为01；ADDR是该区的具体起始地址；CNT是读取等操作的长度（字）；DATA为需写入的数据，目前支持写一个字；UII为指定的标签ID。

900MHz模块接收到请求命令后验证密码并写入数据到标签，但如果写入的UII区，很可能修改PCBits，如果修改了PCBits则后续操作如果需要指定UII，则需要重新识别标签获取UII才能操作。UHFCMD\_WRITE\_DATA命令与读取标签内容命令类似，处理完求命令后将结果反馈给请求方（如PC），响应帧结构如表2.46所示。

**表2.46 UHFCMD\_WRITE\_DATA命令响应帧结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **EOF** |
| AA | XX | 14 | XX | 55 |

UHFCMD\_WRITE\_DATA命令示例如表2.47所示。

**表2.47 UHFCMD\_WRITE\_DATA命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 0F 14 00 00 00 00 01 01 01 0B 00 0C 00 12 34 55 | 请求在UII的地址1处写入0x0B00 |
| Recv | AA 03 14 00 55 | 写入成功 |

1. **UHFCMD\_ERASE\_DATA（指定UII）命令**

UHFCMD\_ERASE\_DATA命令字是0x15，必须由用户指定特定的UII信息才能将该标签指定的数据段擦除。该命令只对支持BlockErase命令的标签有效。读取标签、写入标签、擦除标签、锁定标签操作的命令帧中需含有存取密码，当存取密码不全为0时，模块利用ACCESS命令确保标签处在SECURED状态后进行相应操作。当900MHz模块通过串口接收到UHFCMD\_ ERASE \_DATA命令帧结构如表2.48所示。

**表2.48 UHFCMD\_ERASE\_DATA命令请求帧结构**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **VData** | | | | | **EOF** |
| AA | xx | 15 | APWD | BANK | ADDR | CNT | UII | 55 |
| 4bit | 1bit | 1bit | 1bit | n bit |

注：APWD是擦除标签等操作的存取密码；BANK是您选择的区编号，如UII为01；ADDR是该区的具体起始地址；CNT是读取等操作的长度（字）； UII为指定的标签ID。

900MHz模块接收到擦除命令后验证密码并擦除数据，处理完求命令后将结果反馈给请求方（如PC），响应帧结构如表2.49所示。

**表2.49 UHFCMD\_ERASE \_DATA命令响应帧结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **EOF** |
| AA | 03 | 15 | XX | 55 |

UHFCMD\_ERASE\_DATA命令示例如表2.50所示。

**表2.50 UHFCMD\_ERASE\_DATA命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 0B 15 00 00 00 00 01 01 01 04 00 55 | 请求在UII的地址1处做擦除操作 |
| Recv | AA 03 15 00 55 | 擦除成功 |

1. **UHFCMD\_LOCK\_MEM命令**

UHFCMD\_LOCK\_MEM命令字是0x16，必须由用户指定特定的UII信息才能对该标签执行lock操作。读取标签、写入标签、擦除标签、锁定标签操作的命令帧中需含有存取密码，当存取密码不全为0时，模块利用ACCESS命令确保标签处在SECURED状态后进行相应操作。当900MHz模块通过串口接收到UHFCMD\_LOCK\_DATA命令帧结构如表2.51所示。

**表2.51 UHFCMD\_LOCK\_DATA命令请求帧结构**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **VData** | | | **EOF** |
| AA | xx | 16 | APWD | LOCKDATA | UII | 55 |
| 4bit | 3bit | n bit |

注：APWD是锁定标签等操作的存取密码；LOCKDATA数据段的高四位保留，低二十位是Lock-CommandPayload； UII为指定的标签ID。

LOCKDATA的数据格式如表2.52所示。

**表2.52 LOCKDATA的数据格式定义**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **Kill password** | | **Accesspassword** | | **UIImemory** | | **TIDmemory** | | **Usermemory** | |
| 23~20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 |
| 保留 | S/W | S/W | S/W | S/W | S/W | S/W | S/W | S/W | S/W | S/W |
| 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PRW | PL | PRW | PL | PW | PL | PW | PL | PW | PL |

注：S/W是Skip/Write的缩写，PRW为Pwdread/write的缩写，PL为Permalock的缩写，PW为Pwdwrite的缩写。位10~19为Mask，位0~9为Action，Action字段决定响应的数据段在OPEN或SECURED状态下是否可读写、锁定等操作。

LOCKDATA具体控制标签的哪个区和控制效果如表2.52所示，LOCKDATA主要控制五个部分，分别是Killpassword、Accesspassword、UIImemory、TIDmemory、Usermemory，每个区有四个控制位控制，高位是掩码，低位为具体操作。如Usermemory，由LOCKDATA的第0、1、10、11位控制，10、11为掩码，0、1为允许对Usermemory操作的具体动作（Lock Action）。Lock Action定义如表2.53所示。

**表2.53 Lock Action 的定义（Pwd-write/Permalock）**

|  |  |  |
| --- | --- | --- |
| **Pwd-write** | **Permalock** | **描述** |
| 0 | 0 | 相应数据段在OPEN或SECURED状态下可写入 |
| 0 | 1 | 相应数据段在OPEN或SECURED状态下永久可写入，不可锁定 |
| 1 | 0 | 相应数据段在SECURED状态下可写入，OPEN状态下不可写入 |
| 1 | 1 | 相应数据段在任何状态下不可写入 |

**表2.53 Lock Action 的定义（Pwd-read/write / Permalock）**

|  |  |  |
| --- | --- | --- |
| **Pwd-read/write** | **Permalock** | **描述** |
| 0 | 0 | 相应数据段在OPEN或SECURED状态下可读取和写入 |
| 0 | 1 | 相应数据段在OPEN或SECURED状态下永久可读取和写入，不可锁定 |
| 1 | 0 | 相应数据段在SECURED状态下可读写，OPEN状态下不可读写 |
| 1 | 1 | 相应数据段在任何状态下不可读取和写入 |

900MHz模块接收到请求命令，处理完命令后将结果反馈给请求方（如PC），响应帧结构如表2.54所示。

**表2.54 UHFCMD\_LOCK\_MEM命令响应帧结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **EOF** |
| AA | XX | 16 | XX | 55 |

UHFCMD\_LOCK\_MEM命令示例如表2.55所示。

**表2.55 UHFCMD\_LOCK\_MEM命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 0D 16 00 00 00 00 00 10 04 08 00 00 01 55 | 请求锁定标签（08 00 00 01） |
| Recv | AA 03 16 00 55 | 锁定成功 |

1. **UHFCMD\_KILL\_TAG命令**

UHFCMD\_KILL\_TAG命令字是0x17，必须由用户指定特定的UII信息才能对该标签执行销毁操作。当900MHz模块通过串口接收到UHFCMD\_KILL\_TAG命令帧结构如表2.56所示。

**表2.56 UHFCMD\_KILL\_TAG命令请求帧结构**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **VData** | | **EOF** |
| AA | xx | 17 | KILLPWD | UII | 55 |
| 4bit | n bit |

注：KILLPWD是销毁标签等操作的密码。

900MHz模块接收到请求命令，处理完命令后将结果反馈给请求方（如PC），响应帧结构如表2.57所示。

**表2.57 UHFCMD\_KILL\_TAG命令响应帧结构**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **EOF** |
| AA | XX | 17 | XX | 55 |

UHFCMD\_KILL\_TAG命令示例如表2.58所示。

**表2.58 UHFCMD\_KILL\_TAG命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 0A 17 00 00 00 00 08 00 00 01 55 | 请求销毁标签（08 00 00 01） |
| Recv | AA 03 17 00 55 | 销毁成功 |

1. **UHFCMD\_INVENTORY\_SINGLE 命令**

UHFCMD\_INVENTORY\_SINGLE命令字是0x18，用于单步识别标签ID，对单张卡非循环识别一般用该命令。当900MHz模块通过串口接收到UHFCMD\_INVENTORY\_SINGLE命令帧结构如表2.59所示。

**表2.59 UHFCMD\_INVENTORY\_SINGLE命令请求帧结构**

|  |  |  |  |
| --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **EOF** |
| AA | 02 | 18 | 55 |

模块接收到请求命令后进行单步识别标签并将标签的ID反馈给请求方，响应帧结构如表2.60所示。

**表2.60 UHFCMD\_INVENTORY\_SINGLE命令响应帧结构**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | **EOF** |
| AA | XX | 18 | XX | UID | 55 |

900M模块响应帧VData分为两个部分，PCBits和UII。PCBits为协议控制字节，后面紧接着是UII，PCBits的值决定了UII的长度，UHFCMD\_INVENTORY\_SIGLE命令示例如表2.61所示。

**表2.61 UHFCMD\_INVENTORY\_SINGLE命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 02 18 55 | 请求单步识别标签 |
| Recv | AA 05 18 00 04 00 55 | 成功，卡号为 00 04 |

PCBits等信息和前面所述UHFCMD\_INVENTORY命令一致，可参考UHFCMD\_INVENTORY命令。

1. **UHFCMD\_SINGLE\_READ\_DATA（不指定UII）命令**

UHFCMD\_SINGLE\_READ\_DATA命令字是0x20，不必由用户指定特定的UII信息来读取该标签的内部数据。读取标签、写入标签、擦除标签、锁定标签操作的命令帧中需含有存取密码，当存取密码不全为0时，模块利用ACCESS命令确保标签处在SECURED状态后进行相应操作。当900MHz模块通过串口接收到UHFCMD\_SINGLE\_READ\_DATA命令帧结构如表2.62所示。

**表2.62 UHFCMD\_SINGLE\_READ\_DATA命令请求帧结构**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **VData** | | | | **EOF** |
| AA | xx | 20 | APWD | BANK | ADDR | CNT | 55 |
| 4bit | 1bit | 1bit | 1bit |

注：APWD是读写标签等操作的存取密码；BANK是您选择的区编号，如UII为01；ADDR是该区的具体起始地址；CNT是读取等操作的长度（字）。

UHFCMD\_SINGLE\_READ\_DATA命令和前面UHFCMD\_READ\_DATA命令类似，各个字段的含义也一致，可以参照UHFCMD\_READ\_DATA命令。900MHz模块接收到请求命令后处理请求并将结果反馈给请求方（如PC），响应帧结构如表2.63所示。

**表2.63 UHFCMD\_SINGLE\_READ\_DATA命令响应帧结构**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | | **EOF** |
| AA | XX | 20 | XX | Data | UII | 55 |

UHFCMD\_SINGLE\_READ\_DATA命令示例如表2.64所示。

**表2.64 UHFCMD\_SINGLE\_READ\_DATA命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 09 20 00 00 00 00 03 01 01 55 | 请求读取UII的地址1处一个字的数据 |
| Recv | AA 07 20 00 00 00 04 00 55 | 读取成功，数据00 00 ，卡号04 00 |

在响应帧中和UHFCMD\_READ\_DATA命令略有不同，该命令的响应帧中包含了UII信息。可以理解为没有指定UII则需要知道读取的是哪张卡的数据，因此附带了卡号信息，而指定了UII进行读取，也就是提前知道该数据是哪张卡的数据，因此没必要在响应帧中携带是哪张标签的信息。

1. **UHFCMD\_SINGLE\_WRITE\_DATA（不指定UII）命令**

UHFCMD\_SINGLE\_WRITE\_DATA命令字是0x21，不必由用户指定特定的UII信息，直接将数据写入到标签。读取标签、写入标签、擦除标签、锁定标签操作的命令帧中需含有存取密码，当存取密码不全为0时，模块利用ACCESS命令确保标签处在SECURED状态后进行相应操作。当900MHz模块通过串口接收到UHFCMD\_SINGLE\_WRITE\_DATA命令帧结构如表2.65所示。

**表2.65 UHFCMD\_SINGLE\_WRITE\_DATA命令请求帧结构**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **VData** | | | | | **EOF** |
| AA | xx | 21 | APWD | BANK | ADDR | CNT | DATA | 55 |
| 4bit | 1bit | 1bit | 1bit | Data |

注：APWD是读写标签等操作的存取密码；BANK是您选择的区编号，如UII为01；ADDR是该区的具体起始地址；CNT是读取等操作的长度（字），目前只支持1个字；DATA为要写入的数据（长度为2字节）。

UHFCMD\_SINGLE\_WRITE\_DATA命令和前面UHFCMD\_WRITE\_DATA命令类似，各个字段的含义也一致，可以参照UHFCMD\_WRITE\_DATA命令。900MHz模块接收到请求命令后处理请求并将结果反馈给请求方（如PC），响应帧结构如表2.66所示。

**表2.66 UHFCMD\_SINGLE\_ WRITE \_DATA命令响应帧结构**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SOF** | **Lenth** | **CMD** | **Status** | **VData** | **EOF** |
| AA | XX | 21 | XX | UII | 55 |

UHFCMD\_SINGLE\_WRITE\_DATA命令和UHFCMD\_SINGLE\_READ\_DATA命令类似，反馈信息中包含UII信息，可以参照UHFCMD\_SINGLE\_READ\_DATA命令，示例如表2.67所示。

**表2.67 UHFCMD\_SINGLE\_WRITE\_DATA命令示例**

|  |  |  |
| --- | --- | --- |
| **操作** | **交互帧(hex)** | **说明** |
| Send | AA 0B 21 00 00 00 00 01 01 01 10 00 55 | 请求请求写入数据（10 00） |
| Recv | AA 05 21 00 04 00 55 | 写入成功，卡号04 00 |

**附录**

关于Q 值的选择，详细参考ISO18000-6C 协议中关于防碰撞的内容说明。主要用于电磁场内存在多张电子标签时，避免多张电子标签信号相互碰撞，导致读写器无法正常识别电子标签。Q值的选择依据为读卡器通讯范围内的电子标签的数目（假设为 N），当2Q 接近N 时，最佳。