Advanced Card Systems Ltd.
Card & Reader Technologies

# ACR1555U


参考手册 V1.06



info@acs.com.hk
**www.acs.com.hk**

# 目录

# 图目录



# 表目录

# 1.0. 简介

ACR1555U NFC 蓝牙®读写器符合 ISO 14443 第 1-4 部分和 ISO 18092 标准，支持非接触式卡、MIFARE®卡、FeliCa™ 卡、NFC 标签以及 ISO 7816 A、B 和 C 类（5 V、3 V 和 1.8 V）SAM 卡。

该读写器支持蓝牙®5.2 标准单模操作，同时还是一款 USB Type-C 设备，兼容各种操作系统，如 iOS、Android™、Linux®和 Windows®平台。

ACR1555U 采用 450mAh 3.7V 锂离子电池供电，配有 3 个 LED 指示灯和 1 个蜂鸣器用于显示设备运行状态，另外还有 2 个按钮分别控制电池、设备状态以及蓝牙®状态。

## 1.1. 符号和缩写

| 缩写 | 说明 |
|---|---|
| ATR | 属性请求和属性响应（Attribute Request and Attribute Response） |
| DEP | 数据交换协议请求及数据交换协议响应（Data Exchange Protocol Request and Data Exchange Protocol Response） |
| DSL | 取消选择请求和取消选择响应（Deselect Request and Deselect Response） |
| PSL | 参数选择请求和参数选择响应（Parameter Selection Request and Parameter Selection Response） |
| RLS | 释放请求和释放响应（Release Request and Release Response） |
| WUP | 唤醒请求和唤醒响应（Wakeup Request and Wakeup Response） |
| DID | 设备 ID（Device ID） |
| BS | 发送比特周期（Sending bit duration） |
| BR | 接收比特周期（Receiving bit duration） |
| PP | 协议参数（Protocol Parameters） |

**表1：** 符号和缩写

# 2.0. 特性

- USB全速接口
- 蓝牙接口
- 符合CCID标准
- 智能卡读写器：
  - 非接触接口：
    - 读写速率达 26 kbps（ISO 15693 卡）以及 848 kbps（ISO 14443 卡）
    - 内置天线用于读写非接触式标签，智能卡读取距离可达 50 mm（视标签的类型而定）
    - 支持 ISO 15693 卡
    - 支持 ISO 14443 第 4 部分 A 类和 B 类卡、MIFARE®系列卡、FeliCa 卡和全部五种 NFC（ISO/IEC 18092）标签
    - 内建防冲突特性
    - 支持扩展的 APDU（最大 64 KB）
  - SAM接口：
    - 1个SAM卡槽
    - 支持ISO 7816 A类SAM卡
- 应用程序编程接口：
  - 支持 PC/SC
  - 支持 CT-API（通过 PC/SC 上一层的封装）
- 内置外设：
  - 3 个用户可控的 LED 指示灯（蓝色、黄色、红&绿双色 LED）
  - 用户可控的蜂鸣器
  - 用于控制电池、设备和蓝牙®状态的按钮
- 支持USB固件升级[1]
- 支持Android™ 4.3及以上版本[2]
- 支持iOS和iPadOS 12或更高版本[3]
- 符合下列标准：
  - IEC/EN 62368
  - CE
  - UKCA
  - FCC
  - VCCI
  - RoHS
  - REACH
  - Bluetooth® BQB
  - TELEC (日本)
  - Microsoft® WHQL
  - KCC (韩国)
  - ISO 14443

[1] *适用于 PC 连接模式*

[2] *使用 ACS 定义的安卓库*

[3] *使用 ACS 定义的 iOS 或 iPadOS 库*

- ISO 15693
- ISO 7816
- PC/SC
- CCID
- WEEE

# 3.0. ACR1555U 架构

## 3.1. 读写器功能框图

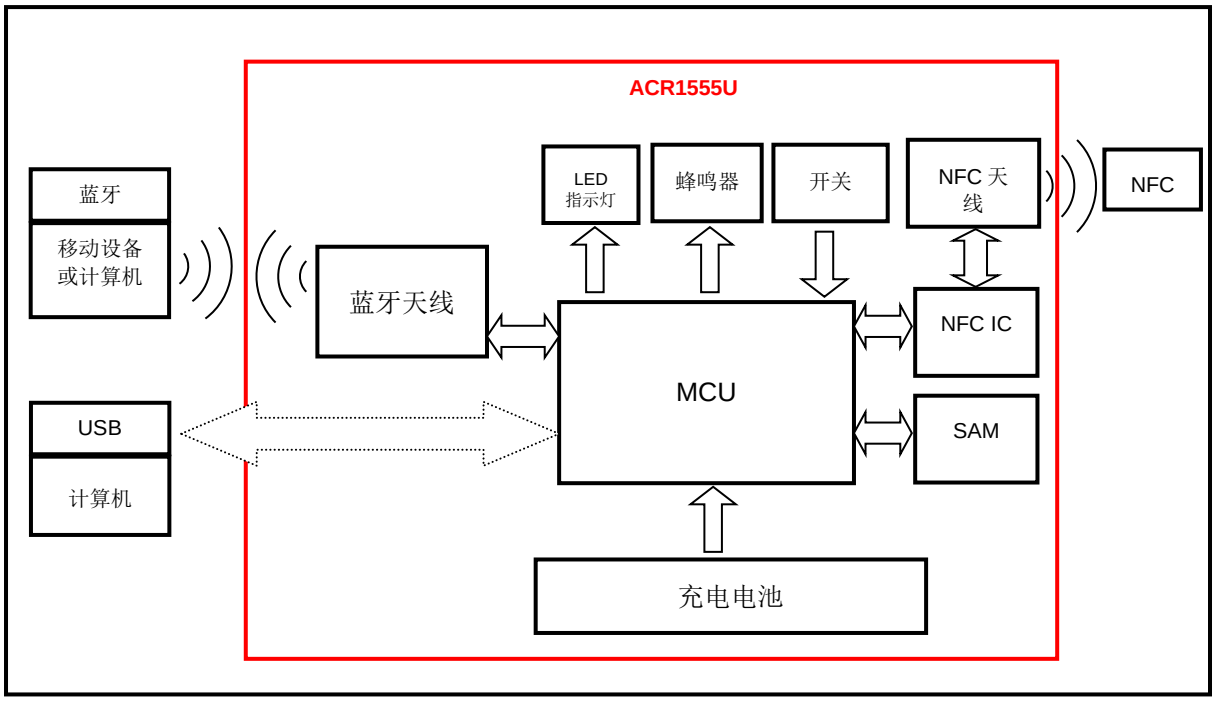



**图1：** ACR1555U 读写器功能框图

## 3.2. PC/SC 驱动与 PICC 和 SAM 间的通信

ACR1555U 与计算机之间使用 CCID 协议进行通信。PICC 和 SAM 间的通信则完全符合 PC/SC 标准。

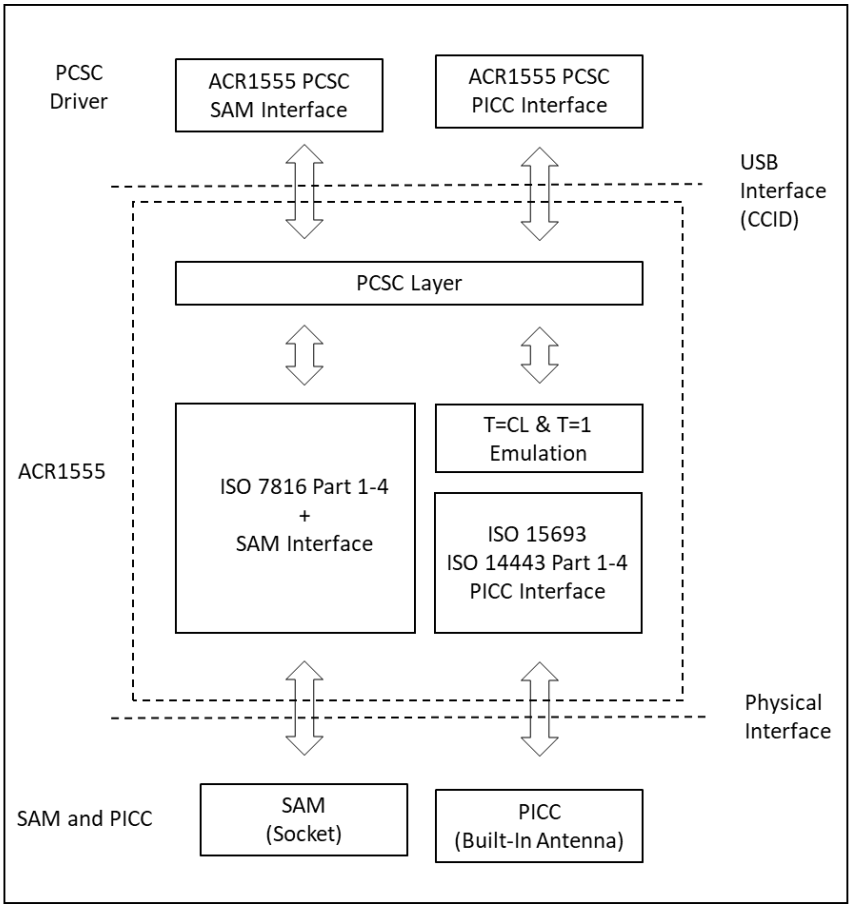



**图2：** ACR1555U 架构

蓝牙低功耗协议栈架构如下所示：

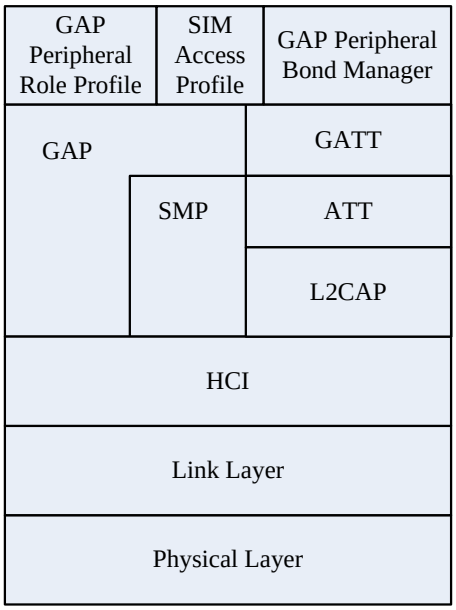



**图3：** BLE 协议栈

# 4.0. 硬件设计

## 4.1. 电池

ACR1555U 使用容量为 450 mAh 的锂离子充电电池。

### 4.1.1. 电池充电

ACR1555U 电池电量耗尽时，可以在下列任意模式下充电：连接电源插座，可以在**关机**、**USB** 或者**蓝牙**模式下充电。

### 4.1.2. 电池续航时间

电池的续航时间取决于设备使用情况。以下是根据各种工作条件预估的电池续航时间：

| 模式 | 预计电池续航时间 |
|---|---|
| 工作模式 | 5.5 小时* (1) |
| 待机模式 | 12 小时* (2) |
| 关机模式 | 48 天 |

**表2**：预计电池续航时间

***注：****结果可能因采用不同的智能卡而发生变化。*

*(1) 在蓝牙模式下，关闭睡眠模式并连续操作.。*

*(2) 在蓝牙模式下，休眠时间设为 60 秒，每天唤醒 10 次操作，每次操作一分钟。*

## 4.2. 蓝牙

ACR1555U 采用蓝牙方式将设备与电脑和移动设备配对。

## 4.3. USB

ACR1555U 按照 USB 标准通过 USB 与计算机连接。

### 4.3.1. 通信参数

ACR1555U 按照 USB 规范 2.0 通过 USB 接口与计算机建立连接，支持 USB 全速模式，速率为 12 Mbps。

| 引脚 | 信号 | 功能 |
|---|---|---|
| 1 | $V_{BUS}$ | 为读写器提供+5 V 的电源 |
| 2 | D- | ACR1555U 和 PC 间以差分信号传输数据 |
| 3 | D+ | ACR1555U 和 PC 间以差分信号传输数据 |
| 4 | GND | 参考电压等级 |

**表3**：USB 接口配线

注 - 为了使 ACR1555U 能够通过 USB 接口正常工作，必须先安装 **ACS 专有设备驱动**或 **Microsoft CCID 驱动**。详情请参考《设备驱动安装指南》。

### 4.3.2. 端点

ACR1555U 通过下列端点与主计算机进行通信：

**Control Endpoint** 用于设置和控制

**PICC：**

**EP1 Bulk OUT** 用于从主计算机发送至 ACR1555U PICC 接口的命令（数据包大小为 64 字节）

**EP1 Bulk IN** 用于从 ACR1555U PICC 接口发送至主计算机的响应（数据包大小为 64 字节）

**EP2 Interrupt IN** 用于从 ACR1555U PICC 接口发送至主计算机的卡片状态报文（数据包大小为 8 字节）

**SAM：**

**EP3 Bulk OUT** 用于从主计算机发送至 ACR1555U SAM 接口的命令（数据包大小为 64 字节）

**EP3 Bulk IN** 用于从 ACR1555U SAM 接口发送至主计算机的响应（数据包大小为 64 字节）

## 4.4. 非接触智能卡接口

ACR1555U 与非接触卡之间的接口遵循 ISO 14443 标准，并进行了某些限制或提升来增强 ACR1555U 的实用功能。

### 4.4.1. 载波频率

ACR1555U 的载波频率为 13.56MHz。

### 4.4.2. 卡片轮询

ACR1555U 会自动检测进入工作场的非接触卡。此功能支持 ISO 14443-4 的 A 类卡和 B 类卡、ISO 15693 卡、FeliCa 卡、Topaz 卡、MIFARE 系列卡和 NFC 标签。

## 4.5. 用户接口

### 4.5.1. LED 指示灯

LED 指示灯用于显示蓝牙模式、电源和 USB 模式状态。蓝色 LED 用于显示蓝牙模式状态，绿色 LED 用于显示设备电源状态，红色 LED 用于显示电池状态，黄色 LED 则用于显示 PICC 状态。

| 模式 | 颜色 | LED 操作 | 状态 |
|---|---|---|---|
| **蓝牙模式** | 蓝色 //(LED1) | 关闭 | 读写器电源关闭<br>未配对蓝牙设备<br>读写器处于 USB 模式 |
| | | 快速闪烁 (4Hz) | 等待用户确认配对（要求按一次**模式按钮**） |
| | | 缓慢闪烁 (0.5Hz) | 讯号广播<br>等待设备配对 |
| | | 长亮 | 蓝牙设备已连接 |
| | 红色 //(LED2) | 关闭 | 电池已充满电<br>读写器仅由 USB 供电 /<br>电池电压高于 3.5 V，且没有 USB 供电 |
| | | 缓慢闪烁(0.2Hz) | 电池电量不足（低于 30%） |
| | | 长亮 | 电池正在充电 |
| | 橙色 [4] (LED2) | 长亮 | 设备已接通电源并正在充电 |
| | 绿色 //(LED2) | 关闭 | 设备电源已关闭 |
| | | 长亮 | 设备电源已接通 |
| | 黄色 //(LED3) | 关闭 | RF 已关闭 |
| | | 快速闪烁 | 智能卡和读写器之间有读写操作 |
| | | 长亮 | 卡片已存在，读写器正在等待指令 |

**表4：** 蓝牙模式下的 LED 显示

[4]绿色和红色同时亮

| 模式 | 颜色 | LED 操作 | 状态 |
|---|---|---|---|
| USB 模式 | 蓝色 //(LED1) | 关闭 | 读写器处于 USB 模式 |
| | 红色 //(LED2) | 关闭 | 电池已充满电<br>读写器仅由 USB 供电 /<br>电池电压高于 3.5 V，且没有 USB 供电 |
| | | 缓慢闪烁(0.2Hz) | 电池电量不足（低于 30%） |
| | | 长亮 | 电池正在充电 |
| | 橙色 [5] (LED2) | 长亮 | 设备已接通电源并正在充电 |
| | 绿色 (LED2) | 关闭 | 设备电源已关闭 |
| | | 长亮 | 设备电源已接通 |
| | 黄色 //(LED3) | 关闭 | 不存在卡片，RF 已关闭 |
| | 黄色 //(LED3) | 快速闪烁 | 智能卡和读写器之间有读写操作 |
| | | 长亮 | 卡片已存在，读写器正在等待指令 |
| | | 关闭 | 不存在卡片，RF 已关闭 |

**表5：**USB 模式下的 LED 显示

### 4.5.2. 蜂鸣器

蜂鸣器用于显示卡片轮询、蓝牙连接、休眠和电量不足状态。

| 蜂鸣器操作 | 事件 |
|---|---|
| “哔”一下 | 1.读写器已接通电源<br>2.检测到或者移出卡片<br>3.从 USB 模式切换到蓝牙模式（长“哔”声） |
| “哔”两下 | 读写器已接通电源，且电池电量不足 |
| “哔”三下 | 电源已断开 |

[5] 绿色和红色同时亮

### 4.5.3. 按钮

ACR1555U 有 2 个按钮：BLE 按钮和开关按钮。

| 按钮 | 条件 | 模式 | 按钮状态 | 说明 |
|---|---|---|---|---|
| 模式按钮 | - | BLE 模式（配对） | 短按 | 确认蓝牙绑定 |
| | 插入 PC USB | USB 模式 | 长按 | 切换到 BLE 模式 |
| 开关按钮 | 读写器关 | 任意 | 长按 | 开机/激活 NFC |
| | 读写器开 | | 长按 | 关机 |

# 5.0. 软件设计

## 5.1. 读写器模式选择

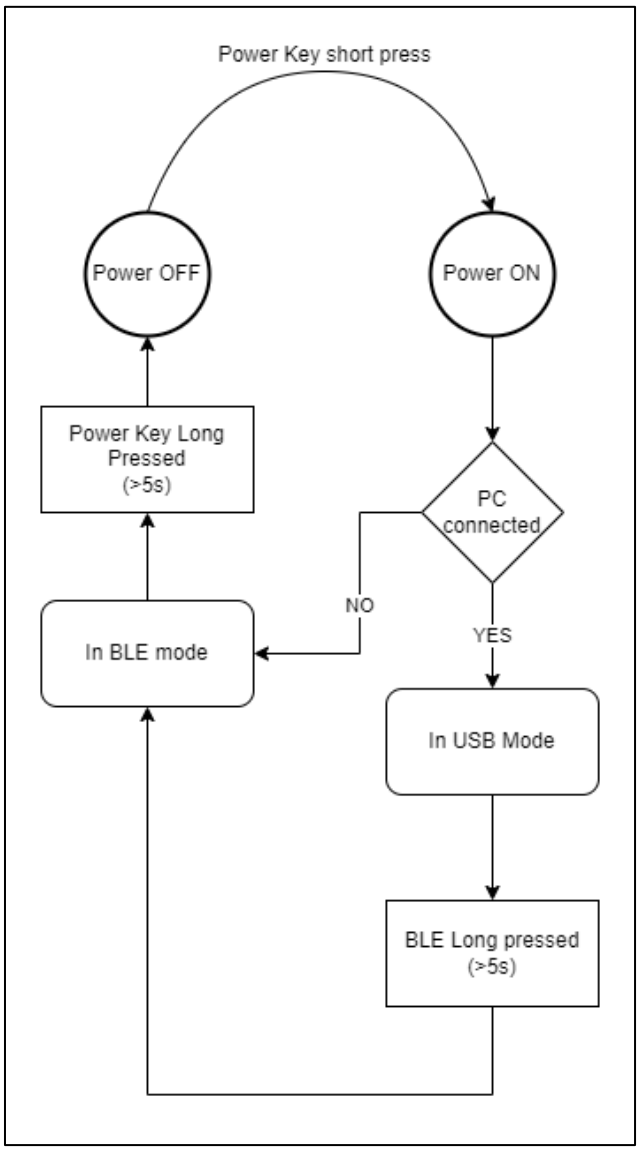



**图4：**读写器模式

读写器开机后，会先检测与 USB 端口的连接情况。如果检测到端口有 5V 电压，读写器将启动 USB 模式，并且成功启动后会保持在 USB 模式。如果未能启动 USB 模式，读写器将开启蓝牙广播模式。如果需要从蓝牙模式切换回 USB 模式，用户需要关闭设备电源，然后再次开机，重新确认 USB 设备。

## 5.2. 蓝牙通信

### 5.2.1. 蓝牙连接流程

蓝牙连接的程序流程如下所示：

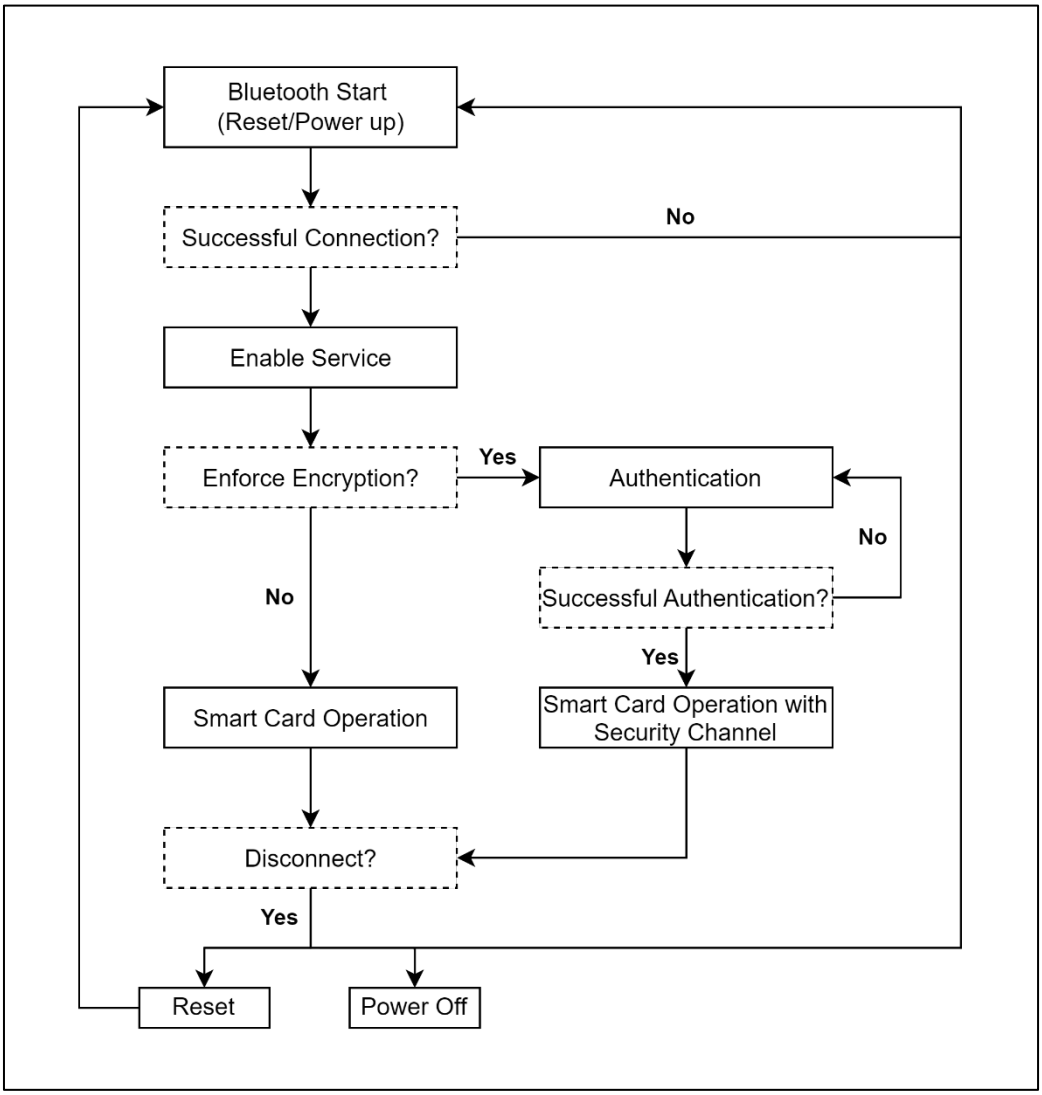



**图5：**蓝牙连接流程

### 5.2.2. 配置文件选择

ACR1555U 是一款使用蓝牙技术作为接口传输数据的智能卡读写器。它采用了能通过三条通道进行命令通信的定制服务：第一条通道用于命令请求，第二条通道用于命令响应，第三条通道用于卡片通知。

此外，当读写器处于蓝牙模式时，其当前耗电量会显著增加，因此采用标准电池服务来通知配对设备当前的电池状态。当电池状态发生变化时，读写器将通过特定通道通知配对设备。为简化操作，电池电量分为三组：电量充足（≥ 3.78 V）、电量不足（< 3.78 V 和 ≥ 3.68 V）和无电（< 3.68 V）。

最后，为了向用户提供更多的读写器信息，还添加了定制的设备信息服务。该信息只能手动读取，或通过应用程序请求读取，其中包括型号、序列号、固件版本和设备商名称。

| 服务 | UUID | 通道 |
|---|---|---|
| 智能卡 | 00003971-817C-48DF-8DB2-476A8134EDE0 | 命令请求 |
| | 00003972-817C-48DF-8DB2-476A8134EDE0 | 命令响应 |
| | 00003973-817C-48DF-8DB2-476A8134EDE0 | 卡片通知 |
| 电池 | 2A19 | 电池电量 |
| 设备信息 | 2A23 | 系统 ID |
| | 2A24 | 型号 |
| | 2A25 | 序列号 |
| | 2A26 | 固件版本 |
| | 2A27 | 硬件版本 |
| | 2A29 | 生产商名称 |

**表6：**ACR1555U 蓝牙服务

| 属性名称 | UUID | 句柄 |
|---|---|---|
| DeviceName | 2A00 | 06h |
| 发送（读写器 → 配对设备） | 00003971-817C-48DF-8DB2-476A8134EDE0 | 28h |
| 接收（配对设备 → 读写器） | 00003972-817C-48DF-8DB2-476A8134EDE0 | 2Bh |
| 卡片通知（读写器 → 配对设备) | 00003973-817C-48DF-8DB2-476A8134EDE0 | 2Fh |
| ABatteryLevel | 2A19 | 20h |
| Manufacturer | 2A29 | 19h |
| SerialNumber | 2A25 | 11h |
| FW_Version | 2A26 | 13h |
| ModelNumber | 2A24 | 0Fh |

**表7：**ACR1555U 服务句柄和 UUID 消息列表

### 5.2.3. 通信配置文件

通信配置文件如下：

起始字节 + 槽位 + 长度 + *保留（1 字节）* + 数据块 + 校验和 + 结束字节

| 数据域 | 大小（字节） | 说明 |
|---|---|---|
| 起始字节 | 1 | 值：55h |
| 槽位 | 1 | 卡槽 00h：PICC，卡槽 01h SAM |
| 长度 | 2 | 长度表示 Datablock 数据域中的字节数 |
| 保留 | 1 | 保留 |
| 主机序列号 | 1 | 主机发送新帧时增加 1 |
| 读写器序列号 | 1 | 读写器发送新帧时增加 1 |
| 数据块 | N | 数据（符合 CCID 的消息体） |
| 校验和 | 1 | 槽位、长度、帧类型、主机&读写器序列号和数据域的异或（XOR）值 |
| 结束字节 | 1 | 值：AAh |

### 5.2.4. 蓝牙通信协议

ACR1555U 采用预定义协议的蓝牙接口与配对设备通信。该协议与 CCID 命令通道和响应通道的格式相似。

| 命令 | 支持模式 | 发送方 | 说明 |
|---|---|---|---|
| 62h | 明文, 已认证 | 配对设备 | PICC 上电 |
| 63h | 明文, 已认证 | 配对设备 | PICC 下电 |
| 65h | 明文, 已认证 | 配对设备 | 获取卡状态 |
| 6Fh | 明文, 已认证 | 配对设备 | 交换 APDU |
| 6Bh | 明文, 已认证 | 配对设备 | Escape 指令 |
| 61h | 明文, 已认证 | 配对设备 | 设置参数 |

**表8：**命令代码摘要

| 命令 | 支持模式 | 发送方 | 说明 |
|---|---|---|---|
| 80h | 明文, 已认证 | 读写器 | 对数据块的响应 |
| 81h | 明文, 已认证 | 读写器 | 对卡槽状态的响应 |
| 82h | 明文, 已认证 | 读写器 | 对参数的响应 |
| 83h | 明文, 已认证 | 读写器 | 对 Escape 指令的响应 |
| 53h | 明文, 已认证 | 读写器 | 对错误的响应 |

**表9：**响应代码摘要

| 命令 | 支持模式 | 发送方 | 说明 |
|---|---|---|---|
| 50h | 明文, 已认证 | 读写器 | 通知卡片状态 |
| 52h | 明文, 已认证 | 读写器 | 通知进入睡眠模式 |

**表10**：卡片通知代码摘要

#### 5.2.4.1. PC_to_RDR_IccPowerOn

激活卡槽并返回卡片的 ATR。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 62h | - |
| 1 | dwLength | 4 | 00000000h | 此消息的额外字节的大小 |
| 2 | bSlot | 1 | 00-01h | 标识命令的卡槽号<br>00h：PICC<br>01h：SAM |
| 5 | bSeq | 1 | 00-FFh | 命令的序号 |
| 6 | bPowerSelect | 1 | 00h-02h | 用于 ICC 的电压<br>00h – 自动电压选择<br>01h – 5 伏<br>02h – 3 伏 |
| 7 | abRFU | 2 | 0000h | 保留为将来使用 |

此消息的响应是 RDR_to_PC_DataBlock 消息，返回的数据是复位应答（ATR）。

#### 5.2.4.2. PC_to_RDR_IccPowerOff

取消激活卡槽。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 63h | - |
| 1 | dwLength | 4 | 00000000h | 此消息的额外字节的大小 |
| 5 | bSlot | 1 | 00-01h | 标识命令的卡槽号<br>00h：PICC<br>01h：SAM |
| 6 | bSeq | 1 | 00-FFh | 命令的序号 |
| 7 | abRFU | 3 | 000000h | 保留为将来使用 |

此消息的响应是 RDR_to_PC_SlotStatus 消息。

### 5.2.4.3. PC_to_RDR_GetSlotStatus

获取卡槽的当前状态。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 65h | - |
| 1 | dwLength | 4 | 00000000h | 此消息的额外字节的大小 |
| 5 | bSlot | 1 | 00-01h | 标识命令的卡槽号<br>00h：PICC<br>01h：SAM |
| 6 | bSeq | 1 | 00-FFh | 命令的序号 |
| 7 | abRFU | 3 | 000000h | 保留为将来使用 |

此消息的响应是 RDR_to_PC_SlotStatus 消息。

### 5.2.4.4. PC_to_RDR_XfrBlock

向 ICC 传输数据块。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 6Fh | - |
| 1 | dwLength | 4 | 00000000-000001E7h | 此消息的 abData 数据域的大小<br>数据域以小端格式存储 |
| 5 | bSlot | 1 | 00-01h | 标识命令的卡槽号<br>00h：PICC<br>01h：SAM |
| 6 | bSeq | 1 | 00-FFh | 命令的序号 |
| 7 | bBWI | 1 | 00-FFh | 用于延长当前传输的 CCID 块超时等待时间。“该数值乘以块等待时间”的时间段过去后，CCID 将设置该块超时。 |

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 8 | wLevelParameter | 2 | - | 字段以小端格式存储<br>TPDU 级，RFU，= 0000h<br>短 APDU 级，RFU，= 0000h<br>扩展 APDU 级：<br>标识 APDU 是否在该命令中开始或结束：<br>0000h<br>命令 APDU 在此命令中开始和结束，<br>0001h<br>命令 APDU 在此命令中开始，并在下一个 PC_to_RDR_XfrBlock 中继续，<br>0002h<br>abData 数据域继续传递命令 APDU 并结束该 APDU 命令，<br>0003h<br>abData 数据域继续传递命令 APDU，后面还跟随另外一个数据块，<br>0010h<br>空的 abData 数据域，下一个 RDR_to_PC_DataBlock 会继续传递响应 APDU |
| 10 | abData | 字节型数组 | - | 发送给 CCID 的数据块。信息“按原样”发送至 ICC（TPDU 交换级别） |

此消息的响应是 RDR_to_PC_DataBlock 消息。

### 5.2.4.5. PC_to_RDR_Escape

访问扩展功能。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 6Bh | - |
| 1 | dwLength | 4 | 00000000-000000FFh | 此消息的 abData 数据域的大小。<br>数据域以小端格式存储 |
| 5 | bSlot | 1 | 00h-01h | 标识命令的卡槽号<br>00h：PICC<br>01h：SAM |
| 6 | bSeq | 1 | 00-FFh | 命令的序号 |

| | | | | |
|---|---|---|---|---|
| 7 | abRFU | 3 | 000000h | 保留为将来使用 |
| 10 | abData | 字节型数组 | - | 发送至 CCID 的数据块 |

此命令消息的响应是 RDR_to_PC_Escape 消息。

### 5.2.4.6. PC_to_RDR_SetParameters

设置卡槽参数。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 61h | - |
| 1 | dwLength | 4 | 00000005h 或 00000007h | 此消息的 abProtocolDataStructure 数据域的大小。<br>数据域以小端格式存储 |
| 5 | bSlot | 1 | 00-01h | 标识命令的卡槽号<br>00h：PICC<br>01h：SAM |
| 6 | bSeq | 1 | 00-FFh | 命令的序号 |
| 7 | bProtocolNum | 1 | 00-01h | 指定所遵循的协议数据结构。<br>00h = T=0 协议结构<br>01h = T=1 协议结构<br>以下值保留为将来使用：<br>80h = 2 线协议结构<br>81h = 3 线协议结构<br>82h = I2C 协议结构 |
| 8 | abRFU | 2 | 0000h | 保留为将来使用 |
| 10 | abProtocolDataStructure | 字节型数组 | - | 协议数据结构 |

T=0 的协议数据结构（dwLength=00000005h）

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 10 | bmFindexDindex | 1 | | B7-4 – FI – ISO/IEC 7816-3:1997 中表 7 的索引，选择一个时钟速率转换因子<br>B3-0 – DI - ISO/IEC 7816-3:1997<br>中表 8 的索引，选择一个波特率转换因子 |
| 11 | bmTCCKST0 | 1 | 00h, 02h | B0 – 0b, B7-2 – 000000b<br>B1 – 使用的约定（b1=0：正向约定；b1=1：反向约定）注：CCID 忽略该位。 |

| | | | | |
|---|---|---|---|---|
| 12 | bGuardTimeT0 | 1 | 00-FFh | 两个字符间的额外保护时间。在通常的保护时间（12etu）基础上增加 0-254 个 etu。FFh 与 00h 相同。 |
| 13 | bWaitingIntegerT0 | 1 | 00-FFh | T=0 时 WI 用于定义 WWT |
| 14 | bClockStop | 1 | 00-03h | 支持 ICC 时钟停止<br>00h = 不允许停止时钟<br>01h = 时钟信号为低时停止<br>02h = 时钟信号为高时停止<br>03h = 时钟信号为高或为低时停止 |

T=1 的协议数据结构（dwLength=00000007h）

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 10 | bmFindexDindex | 1 | | B7-4 – FI – ISO/IEC 7816-3:1997 中表 7 的索引，选择一个时钟速率转换因子<br>B3-0 – DI - ISO/IEC 7816-3:1997<br>中表 8 的索引，选择一个波特率转换因子 |
| 11 | BmTCCKST1 | 1 | 10h, 11h, 12h, 13h | B7-2 – 000100b<br>B0 – 校验和的类型（b0=0：LRC；b0=1：CRC）<br>B1 – 使用的约定（b1=0：正向约定；b1=1：反向约定） 注：CCID 忽略该位。 |
| 12 | BGuardTimeT1 | 1 | 00-FFh | 额外保护时间（两个字符间为 0-254 个 etu）。若值为 FFh，则保护时间减少 1 个 etu。 |
| 13 | BwaitingIntegerT1 | 1 | 00-9Fh | B7-4 = BWI 值，0-9 有效<br>B3-0 = CWI 值，0-Fh 有效 |
| 14 | bClockStop | 1 | 00-03h | 支持 ICC 时钟停止<br>00h = 不允许停止时钟<br>01h = 时钟信号为低时停止<br>02h = 时钟信号为高时停止<br>03h = 时钟信号为高或为低时停止 |
| 15 | bIFSC | 1 | 00-FEh | 商定的 IFSC 的大小 |
| 16 | bNadValue | 1 | 00h | 只支持 NAD = 00h |

此消息的响应是 RDR_to_PC_Parameters 消息

### 5.2.4.7. RDR_to_PC_DataBlock

此消息由 ACR1555U 发出，是对 PC_to_RDR_IccPowerOn 和 PC_to_RDR_XfrBlock 消息的响应。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 80h | 表示 CCID 正在发送一个数据块. |
| 1 | dwLength | 4 | 00000000-000001E7h | 此消息的 abData 数据域的大小。数据域以小端格式存储 |
| 5 | bSlot | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 6 | bSeq | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 7 | bStatus | 1 | - | ***附录 B*** 定义的槽位状态寄存器 |
| 8 | bError | 1 | - | ***附录 B*** 定义的槽位错误寄存器 |
| 9 | bChainParameter | 1 | - | 短 APDU 级，RFU = 00h<br><br>扩展 APDU 级：<br>00h – 响应 APDU 在此命令中开始和结束。<br>01h – 响应 APDU 在此命令中开始，并会继续。<br>02h – 此 abData 数据域继续传递响应 APDU 并结束该响应 APDU。<br>03h – 此 abData 数据域继续传递响应 APDU，后面跟随另外一个数据块。<br>10h – 空的 abData 数据域，下一个 PC_to_RDR_XfrBlock 命令会继续传递命令 APDU |
| 10 | abData | 字节型数组 | - | 本数据域包含由 CCID 返还的数据 |

### 5.2.4.8. RDR_to_PC_SlotStatus

此消息由 ACR1555U 发出，是对 PC_to_RDR_IccPowerOff 和 PC_to_RDR_GetSlotStatus 的响应。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 81h | - |
| 1 | dwLength | 4 | 00000000h | 此消息的额外字节的大小 |
| 5 | bSlot | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 6 | bSeq | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 7 | bStatus | 1 | - | ***附录 B*** 定义的插槽状态寄存器 |
| 8 | bError | 1 | - | ***附录 B*** 定义的插槽错误寄存器 |
| 9 | bClockStatus | 1 | 00-03h | 值 =<br>00h 时钟运行<br>01h 时钟停于 L 状态<br>02h 时钟停于 H 状态<br>03h 时钟停止于未知状态<br>所有其他值保留为将来使用。 |

### 5.2.4.9. RDR_to_PC_Parameters

此消息由 ACR1555U 发出，是对 PC_to_RDR_GetParameters、PC_to_RDR_ResetParameters 和 PC_to_RDR_SetParameters 消息的响应。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 82h | - |
| 1 | dwLength | 4 | 00000005h<br>或<br>00000007h | 此消息的 abProtocolDataStructure 数据域的大小。<br>数据域以小端格式存储 |
| 5 | bSlot | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 6 | bSeq | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 7 | bStatus | 1 | - | ***附录 B*** 定义的槽位状态寄存器 |
| 8 | bError | 1 | - | ***附录 B*** 定义的槽位错误寄存器 |
| 9 | bProtocolNum | 1 | 00-01h | 指定所遵循的协议数据结构。<br>00h：T=0 协议的结构<br>01h：T=1 协议的结构<br>以下值保留为将来使用<br>80h = 2 线协议结构<br>81H = 3 线协议结构<br>82h = I2C 协议结构 |

| | | | | |
|---|---|---|---|---|
| 10 | abProtocolDataStructure | 字节型数组 | - | 协议数据结构见 ***5.2.4.6*** 概述。 |

### 5.2.4.10. RDR_to_PC_Escape

此消息由 ACR1555U 发出，是对 PC_to_RDR_Escape 消息的响应。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 83h | - |
| 1 | dwLength | 4 | 00000000-000000FFh | 此消息的 abData 数据域的大小。数据域以小端格式存储 |
| 5 | bSlot | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 6 | bSeq | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 7 | bStatus | 1 | - | ***附录 B*** 定义的槽位状态寄存器 |
| 8 | bError | 1 | - | ***附录 B*** 定义的槽位错误寄存器 |
| 9 | bChainParameter | 1 | 00h | RFU |
| 10 | abData | 字节型数组 | - | 从 CCID 发送的数据 |

### 5.2.4.11. RDR_to_PC_Error

如果设备接收到了错误的命令，此消息会返回一个错误信息。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 53h | - |
| 1 | dwLength | 4 | 00000000h | 此消息的额外字节的大小 |
| 5 | bSlot | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 6 | bSeq | 1 | - | 与 Bulk-OUT 消息中的值相同 |
| 7 | bStatus | 1 | - | ***附录 B*** 定义的插槽状态寄存器 |
| 8 | bErrorCode | 1 | - | 01h = 校验和错误<br>02h = 超时<br>03h = 命令错误<br>04h = 未经过授权<br>05h = 未定义错误<br>06h = 接收的数据错误<br>07h = 接收的数据长度错误<br>08h = 超出认证重试次数错误 |
| 9 | abRFU | 1 | 00h | RFU |

### 5.2.4.12. RDR_to_PC_NotifySlotChange

此消息由读写器发出，用于通知卡片的状态。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 50h | - |
| 1 | bmSlotICCState | 1 | - | 状态：<br>02h = 无 PICC 卡<br>03h = 有 PICC 卡 |

### 5.2.4.13. RDR_to_PC_Sleep

此消息由读写器发出，用于通知休眠的状态。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bMessageType | 1 | 52h | - |
| 1 | bParam | 1 | 00h | RFU |

### 5.2.5. 认证

如果启用了强制加密，向 ACR1555U 加载敏感数据之前，数据处理服务器必须通过 ACR1555U 的认证，才有权修改读写器内部的安全数据。ACR1555U 采用相互认证的方式。

为了更好说明，请参考下图（图中省去桥接设备，以便更简单明了地进行说明）：

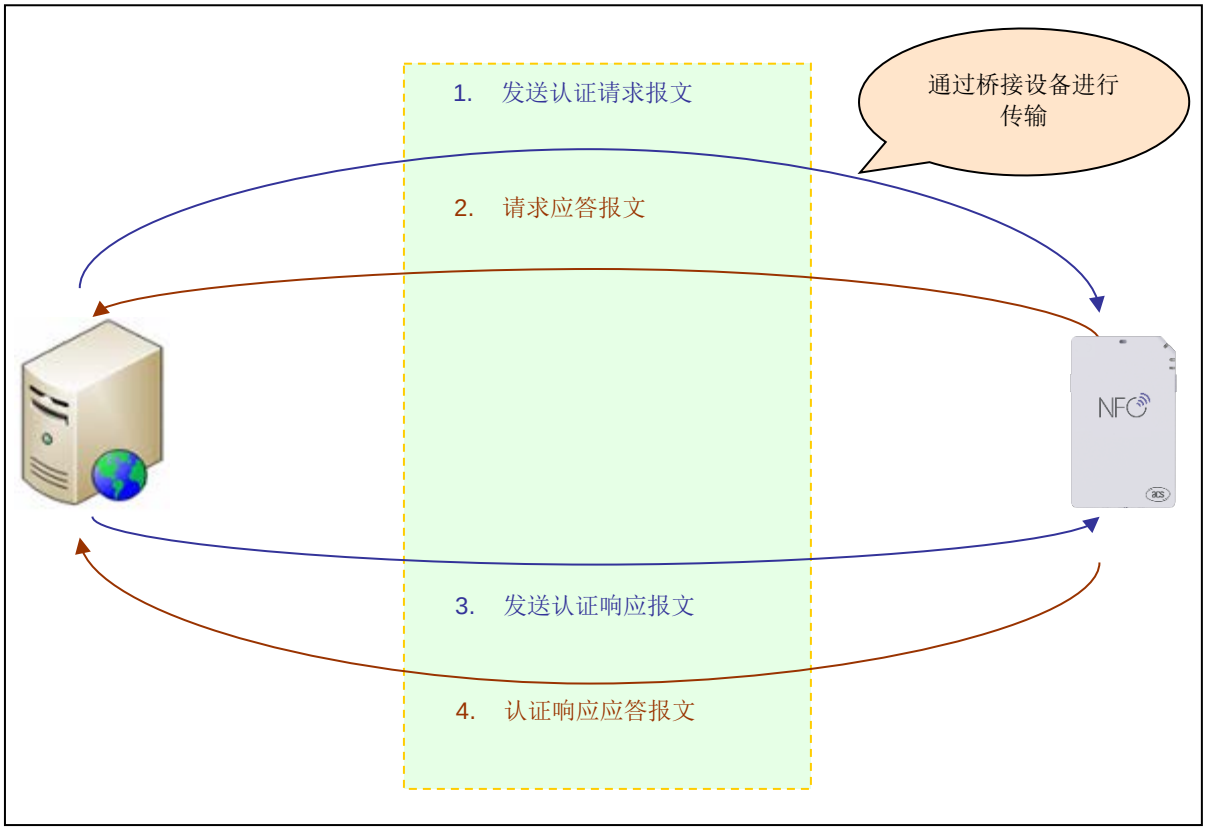



**图6**：认证步骤

认证成功后，ACR1555U 和数据服务器中都会生成一个 16 字节的过程密钥。

默认客户主密钥（十六进制）： **41 43 52 31 35 35 35 55 2D 41 31 20 41 75 74 68**

***注意：****认证密钥输入错误超过十（10）次后，读写器将锁定或无法使用。*

*如需了解详细情况，可联系 ACS 销售代表。*

### 5.2.6. 相互认证和加密协议

在蓝牙模式下，相互认证成功后将对 **17** 进行加密和传输。

#### 5.2.6.1. 蓝牙认证程序流程

如**认证**所示，相互认证用于避免中间人攻击，涉及的命令汇总见下表：

| 序号 | 命令 | 支持模式 | 发送方 | 说明 |
|---|---|---|---|---|
| 1 | 6Bh | 明文 | 配对设备 | SPH_to_RDR_ReqAuth |
| 2 | 83h | 明文 | 读写器 | RDR_to_SPH_AuthRsp1 |

| 序号 | 命令 | 支持模式 | 发送方 | 说明 |
|---|---|---|---|---|
| 3 | 6Bh | 明文 | 配对设备 | SPH_to_RDR_AuthRsp |
| 4 | 83h | 明文 | 读写器 | RDR_to_SPH_AuthRsp2 |

**表11：**相互认证命令汇总

### 5.2.6.2. SPH_to_RDR_ReqAuth

此命令请求 ACR1555U 与已配对的密钥生成设备进行身份认证。

关于认证流程的更多信息，请参考**认证**。

| 偏移 | 数据域 | 大小 | 值 | 说明 | 是否加密 |
|---|---|---|---|---|---|
| 0 | bMessageType | 1 | 6Bh | - | 否 |
| 1 | wLength | 4 | 00000005h | 此消息的 abData 数据域的大小。数据域以小端格式存储 | |
| 5 | bSlot | 1 | 00h | 标识命令的卡槽号<br>00h：PICC | |
| 6 | bSeq | 1 | 00h | 命令的序号 | |
| 7 | abRFU | 3 | 000000h | 保留为将来使用 | |
| 10 | abData | 5 | E0 00 00 73 00h | - | |

如果接收的命令消息无误，此消息的响应是 RDR_to_SPH_AuthRsp1。否则，响应是 RDR_to_SPH_ACK，提供错误信息。

### 5.2.6.3. RDR_to_SPH_AuthRsp1

此命令由 ACR1555U 发出，是对 SPH_to_RDR_ReqAuth 的响应。

| 偏移 | 数据域 | 大小 | 值 | 说明 | 是否加密 |
|---|---|---|---|---|---|
| 0 | bMessageType | 1 | **83h** | - | 否 |
| 1 | dwLength | 4 | 00000015h | 此消息的 abRndNum 数据域的大小。数据域以小端格式存储 | |
| 3 | bSlot | 1 | 00h | 标识命令的卡槽号<br>00h：PICC | |
| 4 | bSeq | 1 | 00h | 命令的序号 | |
| 5 | bStatus | 1 | - | ***附录 B*** 定义的槽位状态寄存器 | |
| 6 | bError | 1 | - | ***附录 B*** 定义的槽位错误寄存器 | |

| 偏移 | 数据域 | 大小 | 值 | 说明 | 是否加密 |
|---|---|---|---|---|---|
| 7 | abRndNum | 21 | E1 00 00 73 10 + 16 字节随机数 | abRndNum[0:15] – 16 字节的随机数<br>所有 16 字节数据必须使用当前存储在 ACR1555U 内的客户主密钥进行加密。<br>“E1 00 00 00 10”不需要加密。 | 是 |

### 5.2.6.4. SPH_to_RDR_AuthRsp

此命令属于认证流程的第二阶段。设备发送 SPH_to_RDR_ReqAuth 命令给 ACR1555U 之后，如果命令无误，读写器将返回 RDR_to_SPH_AuthRsp1 消息。

RDR_to_SPH_AuthRsp1 包含一组通过客户主密钥加密的 16 字节随机数序列。配对的密钥生成设备使用正确的客户主密钥进行解密，并将其添加到 16 字节随机数的末尾，然后使用客户主密钥解密全部 32 字节的随机数，通过此命令将结果返回给 ACR1555U，成功完成认证。

| 偏移 | 数据域 | 大小 | 值 | 说明 | 是否加密 |
|---|---|---|---|---|---|
| 0 | bMessageType | 1 | 6Bh | - | 否 |
| 1 | LEN1 LEN2 (wLength) | 2 | 00000025h | 此消息的 abAuthData 数据域的大小。数据域以小端格式存储 | |
| 3 | bSlot | 1 | 00h | 标识命令的卡槽号<br>00h：PICC | |
| 4 | bSeq | 1 | 00h | 命令的序号 | |
| 5 | bStatus | 1 | - | ***附录 B*** 定义的槽位状态寄存器 | |
| 6 | bError | 1 | - | ***附录 B*** 定义的槽位错误寄存器 | |
| 7 | abAuthData | 37 | E0 00 00 74 20 + 32 字节随机数 | abAuthData[0:15] – 数据处理服务器生成的 16 字节随机数。<br>abAuthData[16:31] – 接收自 ACR1555U 的 16 字节解密随机数。<br>所有 32 字节数据在 AES128 CBC 加密模式下通过客户主密钥进行解密处理。<br>“E0 00 00 74 20”不需要解密。 | 是 |

如果接收到的命令消息正确，并且配对设备返回的随机数也是正确的，此消息的响应是 RDR_to_SPH_AuthRsp2。否则，会收到提供错误信息的响应 RDR_to_SPH_ACK。

### 5.2.6.5. RDR_to_SPH_AuthRsp2

此命令由 ACR1555U 发出，是对 SPH_to_RDR_AuthRsp 的响应。

| 偏移 | 数据域 | 大小 | 值 | 说明 | 是否加密 |
|---|---|---|---|---|---|
| 0 | bMessageType | 1 | **83h** | - | 否 |
| 1 | dwLength | 2 | 00000015h | 此消息的 abRndNum 数据域。数据域以小端格式存储 | 否 |
| 3 | bSlot | 1 | 00h | 标识命令的卡槽号<br>00h：PICC | 否 |
| 4 | bSeq | 1 | 00h | 命令的序号 | 否 |
| 5 | bStatus | 1 | - | ***附录 B*** 定义的槽位状态寄存器 | 否 |
| 6 | bError | 1 | - | ***附录 B*** 定义的槽位错误寄存器 | 否 |
| 20 | abRndNum | 21 | E1 00 00 74 10 + 16 字节随机数 | abRndNum[0:15] – 从数据处理服务器获得的 16 字节随机数<br>所有 16 字节数据必须使用当前存储在 ACR1555U 内的客户主密钥进行加密<br>“E1 00 00 74 10”不需要加密。 | 是 |

### 5.2.6.6. RDR_to_SPH_ACK（错误处理）

此消息是由 ACR1555U 发送给配对设备的错误处理确认消息，用于确认已接受某些命令消息。在通讯过程中，所有出现的错误消息都通过 RDR_to_SPH_ACK 进行传输。此命令没有进行加密处理。

| 命令 | 支持模式 | 发送方 | 说明 |
|---|---|---|---|
| 51h | 明文, 已认证 | 读写器 | RDR_to_SPH_ACK（错误处理） |

必要时消息中会包含错误代码。

| 偏移 | 数据域 | 大小 | 值 | 说明 | 是否加密 |
|---|---|---|---|---|---|
| 0 | bMessageType | 1 | 53h | 错误处理响应头 | No |
| 1 | dwLength | 4 | 00000000h | 此消息中额外数据的大小 | |
| 5 | bSlot | 1 | - | Bulk-OUT 消息中的值相同 | |
| 6 | bSeq | 1 | - | Bulk-OUT 消息中的值相同 | |
| 7 | bStatus | 1 | - | ***附录 B*** 定义的槽位状态寄存器 | |
| 8 | bErrorCode | 1 | - | 指出先前处理的命令报文的错误代码。可能出现的错误代码见下表 | |
| 9 | abRFU | 1 | - | RFU | |

| 数据域 | 值 | 说明 |
|---|---|---|
| bErrorCode | 01h | 校验和错误 |
| | 02h | 超时 |
| | 03h | 命令错误 |
| | 04h | 未经授权 |
| | 05h | 未定义错误 |
| | 06h | 接收的数据错误 |
| | 07h | 接收的数据长度错误 |
| | 08h | 超出认证重试次数错误 |

**表12：**相互认证错误代码

## 5.3. PCSC API

本节介绍一些用于应用程序编程的 PCSC API。关于这些 API 的更多细节，请参考 Microsoft MSDN 库或 PCSC 工作组规格网站。

### 5.3.1. SCardEstablishContext

SCardEstablishContext 函数用于建立进行设备数据库操作的资源管理器上下文。

请参考：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379479%28v=vs.85%29.aspx

在执行任何其他 PCSC 操作前，应当先执行此函数。

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT hContext;
int retCode;
void main ()
{
   // To establish the resource manager context and assign it to “hContext”
   retCode = SCardEstablishContext(SCARD_SCOPE_USER,
                  NULL,
                  NULL,
                  &hContext);
   if (retCode != SCARD_S_SUCCESS)
   {
      // Establishing resource manager context failed
   }
   else
   {
      // Establishing resource manager context successful
      // Further PCSC operation can be performed
   }
}
```

例如：

### 5.3.2. SCardListReaders

SCardListReaders 函数用来获取系统中在指定读卡器组集合中的读卡器名字列表（消除重复项）。

调用方提供一个读卡器组列表，函数返回这些指定组里面的读卡器名字列表。无法识别的组名会被忽略。这个函数只会返回当前系统中可供使用的组里面的读卡器。

请参考：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379793%28v=vs.85%29.aspx

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT hContext; // Resource manager context
int retCode;
char readerName [256]; // List reader name

void main ()
{
   // To establish the resource manager context and assign to “hContext”
   retCode = SCardEstablishContext(SCARD_SCOPE_USER,
                  NULL,
                  NULL,
                  &hContext);
   if (retCode != SCARD_S_SUCCESS)
   {
      // Establishing resource manager context failed
   }
   else
   {
      // Establishing resource manager context successful
      // List the available reader which can be used in the system
      retCode = SCardListReaders (hContext,
                  NULL,
                  readerName,
                  &size);
      if (retCode != SCARD_S_SUCCESS)
      {
         // Listing reader fail
      }
      if (readerName == NULL)
      {
         // No reader available
      }
      else
      {
         // Reader listed
      }
   }
}
```

例如：

### 5.3.3. SCardConnect

SCardConnect 函数利用特定资源管理器上下文，在应用程序与特定读卡器包含的智能卡之间建立连接。如果特定读卡器中没有卡片，会返回一条错误信息。

请参考：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379473%28v=vs.85%29.aspx

例如：

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT      hContext;         // Resource manager context
SCARDHANDLE       hCard;            // Card context handle
unsigned long     dwActProtocol;    // Establish active protocol
int               retCode;
char              readerName [256]; // List reader name
char              rName [256];      // Reader name for connection

void main ()
{
   …
      if (readerName == NULL)
      {
         // No reader available
      }
      else
      {
         // Reader listed
      rName = "ACS ACR1555 1S CL Reader PICC 0"; // Depends on what
                                                 reader be used
                                                 // Should connect to
                                                 PICC interface
         retCode = SCardConnect(hContext,
               rName,
               SCARD_SHARE_SHARED,
               SCARD_PROTOCOL_T0,
               &hCard,
               &dwActProtocol);
         if (retCode != SCARD_S_SUCCESS)
         {
            // Connection failed (May be because of incorrect reader
         name, or no card was detected)
         }
         else
         {
            // Connection successful
         }
      }
   }
```

### 5.3.4. SCardControl

SCardControl 函数提供对读卡器的直接控制，可以在成功调用 SCardConnect 函数后，并且尚未成功调用 SCardDisconnect 函数前随时调用此函数。它对读卡器状态的影响取决于控制码。

请参考：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379474%28v=vs.85%29.aspx

***注：直接命令**中介绍的的命令要使用此 API 进行发送。*

例如：

```
#define SCARD_SCOPE_USER    0

#define EscapeCommand 0x310000 + 3500*4
   SCARDCONTEXT          hContext;          // Resource manager context
   SCARDHANDLE           hCard;             // Card context handle
   unsigned long         dwActProtocol;     // Established active protocol
   int                   retCode;
   char                  readerName [256]; // Lists reader name
   char                  rName [256];       // Reader name for connection
   BYTE                  SendBuff[262],     // APDU command buffer
                         RecvBuff[262];     // APDU response buffer
   BYTE                  FWVersion [20],    // For storing firmware
                                               version message
   BYTE                  ResponseData[50]; // For storing card response
   DWORD                 SendLen,           // APDU command length
                         RecvLen;           // APDU response length
void main ()
   {
      …
      rName = "ACS ACR1555 1S CL Reader PICC 0";    // Depends on what
                                                    reader will be used
                                                    // Should connect to
                                                    PICC interface

         retCode = SCardConnect(hContext,
            rName,
            SCARD_SHARE_DIRECT,
            SCARD_PROTOCOL_T0| SCARD_PROTOCOL_T1,
            &hCard,
            &dwActProtocol);
         if (retCode != SCARD_S_SUCCESS)
         {
            // Connection failed (may be because of incorrect reader
         name, or no card was detected)
         }
         else
         {
            // Connection successful
            RecvLen = 262;
            // Get firmware version
            SendBuff[0] = 0xE0;
            SendBuff[1] = 0x00;
            SendBuff[2] = 0x00;
            SendBuff[3] = 0x18;
            SendBuff[4] = 0x00;
```

```
            SendLen = 5;
            retCode = SCardControl ( hCard,
                     EscapeCommand,
                     SendBuff,
                     SendLen,
                     RecvBuff,
                     RecvLen,
                     &RecvLen);
            if (retCode != SCARD_S_SUCCESS)
            {
               // APDU sending failed
               return;
            }
            else
            {
               // APDU sending successful
               // The RecvBuff stores the firmware version message.
               for (int i=0;i< RecvLen-5;i++)
               {
                  FWVersion[i] = RecvBuff [5+i];
               }
            }
            // Connection successful
            RecvLen = 262;

            // Turn Green LED on, turn Red LED off
            SendBuff[0] = 0xE0;
            SendBuff[1] = 0x00;
            SendBuff[2] = 0x00;
            SendBuff[3] = 0x29;
            SendBuff[4] = 0x01;
            SendBuff[5] = 0x02;  // Green LED On, Red LED off
            SendLen = 6;
            retCode = SCardControl ( hCard,
                     EscapeCommand,
                     SendBuff,
                     SendLen,
                     RecvBuff,
                     RecvLen,
                     &RecvLen);
            if (retCode != SCARD_S_SUCCESS)
            {
               // APDU sending failed
               return;
            }
            else
            {
               // APDU sending success
            }
```

### 5.3.5. SCardTransmit

SCardTransmit 函数用来发送服务请求给智能卡，并接收从智能卡返回的数据。

请参考：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379804%28v=vs.85%29.aspx

***注：***使用此 API 发送 APDU 命令（即：发送给已建立连接的卡片的命令、***PICC 的 PCSC 私有 APDU（带专有扩展）***和 ***PICC 的专属私有 APDU***）。

例如：

```
#define SCARD_SCOPE_USER      0

SCARDCONTEXT      hContext;          // Resource manager context
SCARDHANDLE       hCard;             // Card context handle
unsigned long     dwActProtocol;     // Established active protocol
int               retCode;
char              readerName [256]; // List reader name
char              rName [256];       // Reader name for connect
BYTE              SendBuff[262],     // APDU command buffer
                  RecvBuff[262];     // APDU response buffer
BYTE              CardID [8],        // For storing the FeliCa IDM/
                                     MIFARE UID
BYTE              ResponseData[50]; // For storing card response
DWORD             SendLen,           // APDU command length
                  RecvLen;           // APDU response length
SCARD_IO_REQUEST  ioRequest;

void main ()
{
   …
   rName = "ACS ACR1555 1S CL Reader PICC 0"; // Depends on what
                                              reader should be used
                                              // Should connect to PICC
                                              interface

      retCode = SCardConnect(hContext,
                  rName,
                  SCARD_SHARE_SHARED,
                  SCARD_PROTOCOL_T0,
                  &hCard,
                  &dwActProtocol);
      if (retCode != SCARD_S_SUCCESS)
      {
         // Connection failed (May be because of incorrect reader
         name, or no card was detected)
      }
      else
      {
         // Connection successful
         ioRequest.dwProtocol = dwActProtocol;
         ioRequest.cbPciLength = sizeof(SCARD_IO_REQUEST);
         RecvLen = 262;
```

```
        // Get MIFARE UID/ FeliCa IDM
        SendBuff[0] = 0xFF;
        SendBuff[1] = 0xCA;
        SendBuff[2] = 0x00;
        SendBuff[3] = 0x00;
        SendBuff[4] = 0x00;
        SendLen = 5;
           retCode = SCardTransmit( hCard,
                           &ioRequest,
                           SendBuff,
                           SendLen,
                           NULL,
                           RecvBuff,
                           &RecvLen);

        if (retCode != SCARD_S_SUCCESS)
        {
           // APDU sending failed
           return;
        }
        else
        {
           // APDU sending successful
           // The RecvBuff stores the IDM for FeliCa / the UID for
           MIFARE.
           // Copy the content for further FeliCa access
           for (int i=0;i< RecvLen-2;i++)
           {
              CardID [i] = RecvBuff[i];
           }
        }
```

### 5.3.6. SCardDisconnect

**SCardDisconnect** 函数用来断开先前在应用程序和目标读卡器中的智能卡之间建立的连接。

请参考：http://msdn.microsoft.com/en-us/library/windows/desktop/aa379475%28v=vs.85%29.aspx

此函数用于结束 PCSC 操作。

例如：

```
#define SCARD_SCOPE_USER 0

SCARDCONTEXT      hContext;         // Resource manager context
SCARDHANDLE       hCard;            // Card context handle
unsigned long     dwActProtocol;    // Established active protocol
int               retCode;

void main ()
{
   …
         // Connection successful
      …
      retCode = SCardDisconnect(hCard,SCARD_RESET_CARD);
      if (retCode != SCARD_S_SUCCESS)
      {
         // Disconnection failed
      }
      else
      {
         // Disconnection successful
      }
   }
}
```

### 5.3.7. APDU 流程图

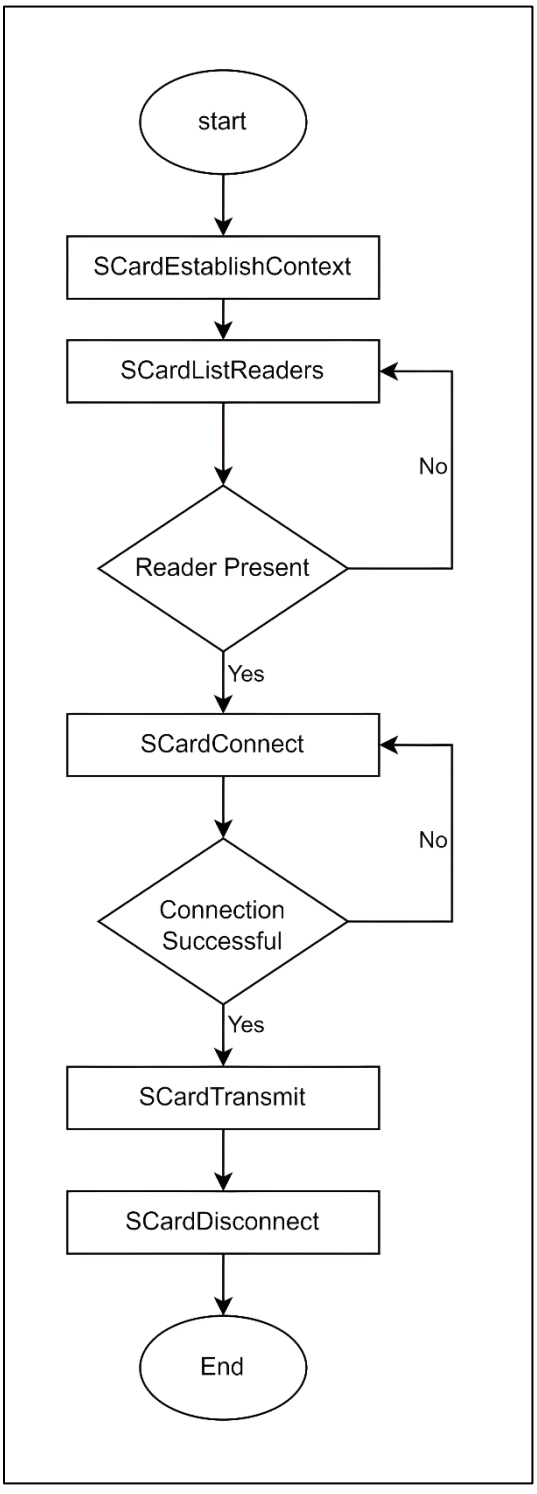



**图7：**ACR1555U APDU 流程图

### 5.3.8. 直接命令流程图

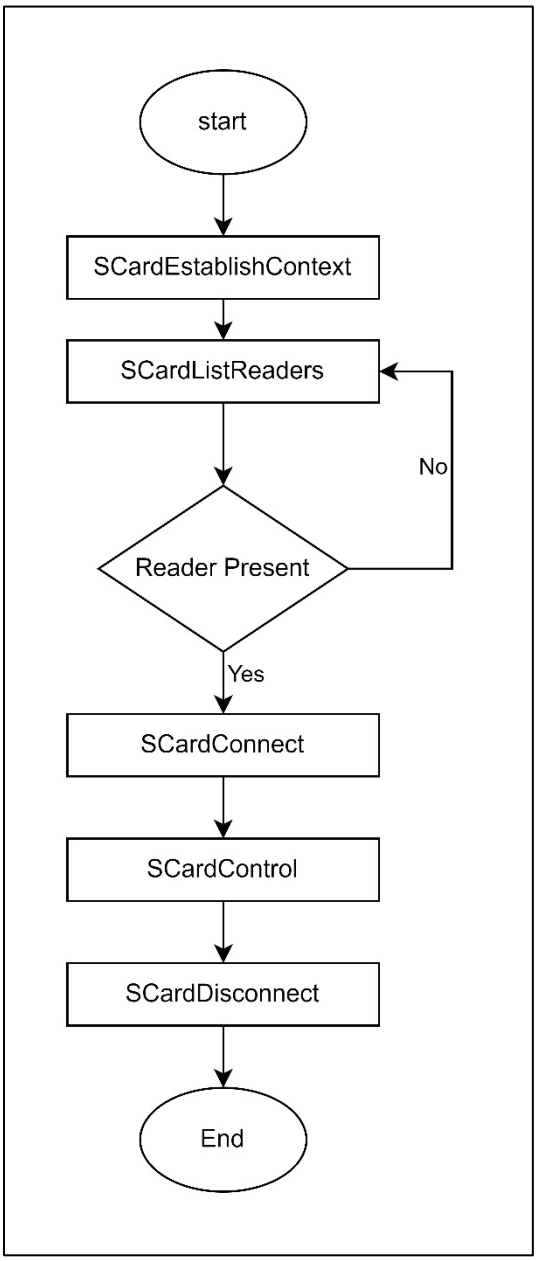



**图8：** ACR1555U 直接命令流程图

## 5.4. 接触式智能卡协议

### 5.4.1. ACOS6-SAM 卡命令

本节介绍 SAM 专用命令。CCID 主机可以使用 CCID 报文 PC_to_RDR_XfrBlock（对应于 PCSC API 中的 SCardTransmit()）向读卡器发送卡片专有命令或 APDU。

***注：**如需了解 ACOS6-SAM 命令的所有信息和应用场景，请联系 ACS 销售代表索取 ACOS6-SAM 参考手册。*

#### 5.4.1.1. 密钥生成（Generate Key）

该命令利用客户卡序列号等偏差数据生成分散密钥，并导入 ACOS3/6 或其它卡片中，用于满足客户发卡的目的。

| APDU | 说明 | |
|---|---|---|
| CLA | 80h | |
| INS | 88h | |
| P1 | 00h | 生成 8 字节密钥 |
| | 01h | 生成 16 字节密钥 |
| | 02h | 生成 24 字节密钥 |
| P2 | 用于生成分散密钥的主密钥的索引 | |
| P3 | 08h | |
| 数据 | 输入数据 | |

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 86h | 未选择 DF |
| 6A 86h | P1 或 P2 无效 |
| 67 00h | P3 不正确，必须是 08h |
| 6A 83h | 在 EF2 中找不到指定的密钥记录 |
| 69 81h | EF2 无效（记录大小、文件类型等） |
| 6A 88h | 找不到 EF2 |
| 62 83h | 当前 DF 被锁定；EF2 被锁定 |
| 69 83h | 使用计数器为 0 |
| 69 82h | 不满足安全条件 |
| 6A 87h | 指定的主密钥不支持 3DES 加密 |
| 61 08h | 命令完成，发送 GET RESPONSE 取结果 |

### 5.4.1.2. 密钥数据分散（或载入）（Diversify (or load) Key Data））

该命令通过密钥分散和密钥载入使 SAM 卡准备好执行加密操作。它将序列号和 CBC 初始向量作为命令数据输入。

| APDU | 说明 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| CLA | 80h | | | | | | | | |
| INS | 72h | | | | | | | | |
| P1 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | 说明 |
| | - | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 密码(Sc) |
| | - | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 帐户密钥($K_{ACCT}$) |
| | - | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 终端密钥 |
| | - | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 卡片密钥 |
| | - | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 批量加密密钥(非分散) |
| | - | 0 | 0 | 0 | 0 | 1 | 1 | 0 | 初始向量 |
| | 0 | - | - | - | - | - | - | - | 16 字节密钥 |
| | 1 | - | - | - | - | - | - | - | 24 字节密钥 |
| P2 | 主密钥的索引：<br>Bit7: 1 = 当前 EF2 中的局部密钥；<br>0 = 全局密钥 EF2<br>Bit6-Bit5: 00b - RFU<br>Bit4-Bit0: 密钥索引 | | | | | | | | |
| P3 | 若 P1 = 1-4, 则 P3 = 8/16,(如果算法为 AES，则 P3 = 8/16)<br>若 P1 = 5, 则 P3 = 0<br>若 P1 = 6,<br>P3 = 8 (主密钥的算法为 DES/ 3DES/ 3KDES)<br>P3 = 16 (主密钥的算法为 AES) | | | | | | | | |
| 数据 | 如果 P1 = 1-4，客户卡的序列号，(若算法为 AES，数据是客户卡的序列号，或者客户卡的序列号后面再加上“0000000000000000”)<br>如果 P1 = 5, 无命令数据.<br>如果 P1 = 6, DES/3DES/3KDES/AES CBC 初始向量。 | | | | | | | | |

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 86h | 未选择 DF |
| 6A 86h | P1 错误，P1 必须为 1-6 |
| 67 00h | P3 错误，P3 必须为 8（或 0） |
| 62 83h | 当前 DF 被锁定，或者 EF2 被锁定 |
| 69 82h | 不满足安全条件 |
| 6A 88h | 找不到 EF2 |

| SW1 SW2 | 说明 |
|---|---|
| 6A 83h | EF2 中找不到指定的主密钥 |
| 69 81h | EF2 无效（FDB、MRL 等不一致） |
| 6A 87h | 指定的密钥不支持认证 |
| 69 83h | 指定的密钥被锁定 |
| 90 00h | 已生成目标密钥，存在 SAM 存储器中 |

#### 5.4.1.3. 加密（Encrypt）

该命令使用 DES 或 3DES 算法来加密数据，它会使用：

1. 与 ACOS3/6、DESFire®、DESFire® EV1 或 MIFARE Plus 卡片相互认证生成的过程密钥。
2. 分散密钥（密码）。
3. 批量加密密钥。
4. 使用过程密钥对分散密码进行加密。
5. 给定一个非安全报文命令，准备 ACOS3 安全报文命令。

| APDU | 说明 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| CLA | 80h | | | | | | | | |
| INS | 74h | | | | | | | | |
| P1 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | 说明 |
| | - | 0 | 0 | 0 | 0 | 0 | 0 | - | ECB 模式 |
| | - | 0 | 0 | 0 | 0 | 0 | 1 | - | CBC 模式 |
| | - | 0 | 0 | 0 | 0 | 1 | 0 | - | 零售 MAC 模式 |
| | - | 0 | 0 | 0 | 0 | 1 | 1 | - | MAC 模式 |
| | - | 0 | 0 | 0 | 1 | 0 | 0 | - | 准备 ACOS3 SM 命令 |
| | - | 1 | 0 | 0 | 1 | 0 | 1 | - | MIFARE DESFire 加密 |
| | - | 1 | 0 | 0 | 1 | 1 | 0 | - | MIFARE DESFire EV1 加密 |
| | - | 0 | 0 | 0 | 1 | 1 | 1 | - | CMAC |
| | - | 0 | 1 | 0 | 0 | 0 | 0 | | MIFARE Plus 命令 |
| | - | 0 | 1 | 0 | 0 | 0 | 1 | | MIFARE Plus 响应 |
| | 0 | - | - | - | - | - | - | 0 | 3DES |
| | 0 | - | - | - | - | - | - | 1 | DES |
| | 1 | - | - | - | - | - | - | 0 | 3K DES |
| | 1 | - | - | - | - | - | - | 1 | AES |
| | - | - | - | - | - | - | - | - | 所有其他值 - RFU |

| APDU | 说明 |
|---|---|
| P2 | P2 代表使用 Load Key 功能在 SAM 集中分散出的密钥：<br>1 – 使用过程密钥 *Ks* 对数据进行加密<br>2 – 使用分散密钥 *Sc* 对数据进行加密<br>3 – 使用批量加密密钥对数据进行加密<br>0 – 返回 ENC (*Sc*, *Ks*)<br>如果 P1.b3 = 1 或 b5=1, P2 必须为 1<br>如果 P2 = 0h, P1 可以是 0 或 1 |
| P3 | P3 < 128<br>如果 P1 的 bit 3 不等于 1，并且 P1 的 bit 5 也不等于 1<br>- 如果 P2 = 1-3, 8(DES/3DES/3KDES)或 16(AES)的倍数，最高 128 字节<br>- 如果 P2 = 0, 0 |
| 数据 | 明文<br>如果 P2 b6 = 1, 数据格式应该是：<br>• 明文数据的长度<br>• DESFire 卡片的命令和卡片头的长度<br>• DESFire 卡片的命令和卡片头<br>• 明文<br>若 P1 = A1h，该加密用于 MIFARE Plus 命令<br>• 如果 MFP 命令是一个值操作命令，数据的格式应该是：Command Code(1 个字节)+BlockNum(2/4 个字节)+Value(4 个字节)。<br>• 如果 MFP 命令是接近度检测，数据的格式应该是：Command Code(1 个字节)+ PPS1(1 个字节)。<br>• 如果 MFP 命令是读，数据的格式应该是：Command Code(1 个字节)+ BlockNum(2 个字节)。<br>• 如果 MFP 命令是写，数据的格式应该是 Command Code(1 个字节)+ BlockNum(2 个字节)+plaintext<br>P1=A3h,<br>• 由 ICC 返回的数据（不包括 SC 码也不包括 RMAC（如果存在 RMAC)） |

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 86h | 未选择 DF |
| 6A 86h | P1 或 P2 无效 |
| 67 00h | P3 不正确 |
| 6A 83h | ACOS 目标密钥未准备就绪（使用 Diversify 命令生成密钥） |
| 61 XX | 加密完成，使用 GET RESPONSE 获取结果 |

### 5.4.1.4. 解密（Decrypt）

该命令用于通过 DES、3DES 或 AES 算法来解密数据，它会使用：

1. 与ACOS3/6、MIFARE DESFire、MIFARE DESFire EV1或MIFARE Plus卡片相互认证生成的过程密钥。
2. 分散密钥（密码）。
3. 批量加密密钥。
4. 使用过程密钥对分散密码进行解密。
5. 查验并解密ACOS3安全报文响应数据

查验并解密 ACOS3 安全报文响应数据

| APDU | 说明 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| CLA | 80h | | | | | | | | |
| INS | 76h | | | | | | | | |
| P1 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | 说明 |
| | - | 0 | 0 | 0 | 0 | 0 | 0 | - | ECB 模式 |
| | - | 0 | 0 | 0 | 0 | 0 | 1 | - | CBC 模式 |
| | - | 0 | 0 | 0 | 1 | 0 | 0 | - | 查验并解密 ACOS3 安全报文应答 |
| | - | 1 | 0 | 0 | 1 | 0 | 1 | - | MIFARE DESFire 解密 |
| | - | 1 | 0 | 0 | 1 | 1 | 0 | - | MIFARE DESFire EV1 解密 |
| | - | 0 | 1 | 0 | 0 | 1 | 0 | - | MIFARE Plus 解密 |
| | 0 | - | - | - | - | - | - | 0 | 3DES |
| | 0 | - | - | - | - | - | - | 1 | DES |
| | 1 | - | - | - | - | - | - | 0 | 3K DES |
| | 1 | - | - | - | - | - | - | 1 | AES |
| | 0 | 0 | 0 | 0 | - | - | - | - | 所有其他值 - RFU |
| P2 | P2 代表使用 Load Key 功能在 SAM 集中分散出的密钥：<br>1 – 使用过程密钥 *Ks* 对数据进行解密<br>2 – 使用分散密钥 *Sc* 对数据进行解密<br>3 – 使用批量加密密钥对数据进行解密<br>0 – 返回 DEC(*Sc*, *Ks*) | | | | | | | | |
| P3 | P3 < 128<br>如果 P1 = A5h, P3=16/32/48<br>如果 P1 的 bit 3 不等于 1<br>- 如果 P2 = 1-3, 8(DES/3DES/3KDES)或 16(AES)的倍数，最高 128 字节<br>- 如果 P2 = 0, 0 | | | | | | | | |

| APDU | 说明 |
|---|---|
| 数据 | 密文<br>如果 P1 = A5h，数据是加密的文本<br>如果 P2 b6 = 1，数据格式应该是：<br>• 明文数据的长度，如果未知，使用 00<br>• DESFire 卡片的命令和卡片头的长度<br>• DESFire 卡片的命令和卡片头<br>• 加密的文本 |

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 86h | 未选择 DF |
| 6A 86h | P1 或 P2 无效 |
| 67 00h | P3 不正确 |
| 6A 83h | ACOS 目标密钥未准备就绪（使用 Diversify 命令生成密钥） |
| 61 XX | 解密完成，使用 GET RESPONSE 获取结果 |

### 5.4.1.5. 认证准备（Prepare Authentication)

该命令用于验证 SAM 卡（作为终端）对于 ACOS 3/6、MIFARE Ultralight C/MIFARE DESFire 卡/MIFARE Plus 卡的合法性。

| APDU | 说明 |
|---|---|
| CLA | 80h |
| INS | 78h |
| P1 | 00h – 3DES<br>01h – DES<br>02h – 3KDES (MIFARE DESFire EV1/ACOS3)<br>03h – AES (MIFARE DESFire EV1/MIFARE Plus/ACOS3)<br>80h – 3DES (仅限 MIFARE DESFire 验证)<br>81h – DES (仅限 MIFARE DESFire 验证)<br>其它 – RFU |
| P2 | 0h – 查验 ACOS3/6 验证返回信息<br>01h – MIFARE Ultralight C/DESFire 验证，通过（分散的）终端密钥<br>05h – MIFARE Ultralight C/DESFire 验证，通过批量加密密钥<br>02h – MIFARE Plus 认证。从 SL1 到 SL3 的首次认证<br>03h – MIFARE Plus 认证。从 SL1 到 SL2 中的认证<br>04h – MIFARE Plus 认证。从 SL2 到 SL3 的跟随认证 |
| P3 | 8 – (P1 = 00h, 01h, 02h, 80h, 81h)<br>16 – (P1 = 03h) |
| 数据 | 卡片随机数据 |

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 86h | 未选择 DF |
| 6A 86h | P1 或 P2 无效 |
| 67 00h | P3 不正确，必须是 08h |
| 6A 83h | ACOS 密钥（KT 或 KC）未准备就绪，（使用 Diversify 生成该密钥） |
| 69 82h | 不满足安全条件 |
| 61 10h | 命令完成，发送 GET RESPONSE 取结果 |

### 5.4.1.6. 校验认证（Verify Authentication）

此命令用于校验 ACOS 3/6、MIFARE Ultralight C、MIFARE DESFire/MIFARE DESFire EV1 或 MIFARE Plus 卡对于终端的合法性，也会在内部生成过程密钥 Ks。

| APDU | 说明 |
|---|---|
| CLA | 80h |
| INS | 7Ah |
| P1 | 00h – 3DES (P2 = 0)<br>01h – DES (P2 = 0)<br>02h – 3KDES (P2 = 0，ACOS3)<br>03h – AES (P2 = 0，ACOS3)<br>其它 – RFU |
| P2 | 00h – 查验 ACOS3/6 认证返回信息<br>01h – 查验 MIFARE Ultralight C®/ DESFire®/ DESFire® EV1 认证返回信息<br>02h – 查验 MIFARE Plus 认证返回信息 |
| P3 | 08h – (P2 = 0, P2 = 1，且过程密钥采用 DES/3DES)<br>16h – (P2 = 1，且过程密钥采用 3KDES/AES)<br>16h – (P2=02，且 MIFARE Plus 返回数据 ek(RndA’))<br>32h – (P2=02，且 MIFARE Plus 返回数据 ek(TI+PICCcap2+PCDcap2)) |
| 数据 | ACOS 3/6：DES ($K_S$, $RND_T$)<br>MIFARE DESFire/ DESFire EV1 返回数据：ek(RndA’)<br>MIFARE Plus 返回数据 ek(RndA’)或 ek(TI+PICCcap2+PCDcap2) |

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 86h | 未选择 DF |
| 6A 86h | P1 或 P2 无效 |
| 67 00h | P3 不正确，必须是 08h |
| 6A 83h | ACOS-SAM 过程密钥或 $RND_T$ 没有准备就绪。使用 PREPARE AUTHENTICATION 命令来生成这些密钥。 |
| 69 82h | 数据不正确 |
| 90 00h | 数据正确，ACOS 相互认证成功 |

### 5.4.1.7. ACOS 查询帐户校验（Verify ACOS Inquire Account）

该命令用于检查 ACOS3/6 卡片的查询帐户钱包命令。它会使用 SAM 的分散密钥验证 ACOS3/6 返回的 MAC 校验和是否正确。

| APDU | 说明 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| CLA | 80h | | | | | | | | |
| INS | 7Ch | | | | | | | | |
| P1 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | 说明 |
| | - | 0 | 0 | 0 | 0 | - | 0 | - | ACOS INQ_AUT 未启用 |
| | - | 0 | 0 | 0 | 0 | - | 1 | - | ACOS INQ_AUT 启用 |
| | - | 0 | 0 | 0 | 0 | 0 | - | - | ACOS INQ_ACC_MAC 未启用 |
| | - | 0 | 0 | 0 | 0 | 1 | - | - | ACOS INQ_ACC_MAC 启用 |
| | 0 | - | - | - | - | - | - | 0 | 3DES |
| | 0 | - | - | - | - | - | - | 1 | DES |
| | 1 | - | - | - | - | - | - | 0 | 3K DES（仅 ACOS3） |
| | 1 | - | - | - | - | - | - | 1 | AES（仅 ACOS3） |
| P2 | 0h | | | | | | | | |
| P3 | 1Dh | | | | | | | | |
| 数据 | 客户 ACOS 卡片的 INQUIRE ACCOUNT 命令返回的数据块，见下文。 | | | | | | | | |

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 86h | 未选择 DF |
| 6A 86h | P1 或 P2 无效 |
| 67 00h | P3 不正确 |
| 6A 83h | ACOS 密钥 $K_S$ 或 $K_{ACCT}$ 未准备就绪；使用 DIVERSIFY 命令生成 $K_{ACCT}$；如适用，通过“Prepare Authentication”生成 $K_S$。 |
| 6F 00h | 数据块的 MAC 不正确 |
| 90 00h | 数据块的 MAC 正确 |

### 5.4.1.8. ACOS 账户交易准备（Prepare ACOS Account Transaction）

为了生成 ACOS3/6 充值(Credit)/扣款(Debit)命令，必须计算 MAC 供 ACOS3/6 进行校验。

| APDU | 说明 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| CLA | 80h | | | | | | | | |
| INS | 7Eh | | | | | | | | |
| P1 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | 说明 |
| | - | 0 | 0 | 0 | 0 | 0 | 0 | - | ACOS TRNS_AUT 未启用 |
| | - | 0 | 0 | 0 | 0 | 0 | 1 | - | ACOS TRNS_AUT 启用 |
| | 0 | - | - | - | - | - | - | 0 | 3DES |
| | 0 | - | - | - | - | - | - | 1 | DES |
| | 1 | - | - | - | - | - | - | 0 | 3K DES（仅 ACOS3） |
| | 1 | - | - | - | - | - | - | 1 | AES（仅 ACOS3） |
| P2 | E2h: 充值<br>E6h: 扣款 | | | | | | | | |
| P3 | 0Dh | | | | | | | | |
| 数据 | 数据块 | | | | | | | | |

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 86h | 未选择 DF |
| 6A 86h | P1 或 P2 无效 |
| 67 00h | P3 不正确，必须是 0Dh |
| 6A 83h | ACOS 密钥 $K_S$ 或 $K_{ACCT}$ 未准备就绪；使用 DIVERSIFY 命令生成 $K_{ACCT}$；如适用，通过“Prepare Authentication”生成 $K_S$。 |
| 61 0Bh | 命令完成，发送 GET RESPONSE 取结果 |

### 5.4.1.9. 扣款证书查验（Verify Debit Certificate）

对于 ACOS3/6，若 DEBIT 命令中的 P1=1，会返回一个扣款证书。可以通过比较此命令的结果和 ACOS3 的响应报文对该扣款证书进行检查。

| APDU | 说明 | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| CLA | 80h | | | | | | | | |
| INS | 70h | | | | | | | | |
| P1 | b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 | 说明 |
| | - | 0 | 0 | 0 | 0 | 0 | 0 | - | ACOS TRNS_AUT 未启用 |
| | - | 0 | 0 | 0 | 0 | 0 | 1 | - | ACOS TRNS_AUT 启用 |
| | 0 | - | - | - | - | - | - | 0 | 3DES |

| APDU | 说明 |
|---|---|
| | 0 - - - - - - 1 DES |
| | 1 - - - - - - 0 3K DES（仅 ACOS3） |
| | 1 - - - - - - 1 AES（仅 ACOS3） |
| P2 | 0h |
| P3 | 14h |
| 数据 | 数据块 |

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 86h | 未选择 DF |
| 6A 86h | P1 或 P2 无效 |
| 67 00h | P3 不正确，必须是 14h |
| 6A 83h | ACOS 密钥 $K_S$ 或 $K_{ACCT}$ 未准备就绪；使用 DIVERSIFY 命令生成 $K_{ACCT}$；如适用，运行 PREPARE AUTHENTICATION 生成 $K_S$。 |
| 69 82h | 不满足安全条件 |
| 6F 00h | DEBIT CERTIFICATE 无效 |
| 90 00h | 成功，DEBIT CERTIFICATE 有效 |

### 5.4.1.10. 取密钥（Get Key)

取密钥命令使密钥从当前 SAM 的密钥文件（SFI=02h）安全地注入另外一张 ACOS6/ACOS6-SAM 卡片，这一过程可以通过也可以不通过密钥分散来实现。这样做可以确保待导入的密钥受到加密和消息验证代码的保护。

此外，该命令还可以通过密钥分散，使密钥安全地从当前SAM的密钥文件（SFI=02h）注入ACOS7/10、MIFARE DESFire、MIFARE DESFire EV1 或 MIFARE Plus 卡。这样做可以确保待导入的密钥受到加密和消息验证代码的保护。

若**卡片头模块**（见 ACOS6-SAM 参考手册第 3.2 节）设置了特殊功能标志 bit7（仅密钥注入标志），且密钥文件已被激活，必须使用 Get Key 才可以载入或变更卡片内的密钥。Bit7 设置后，密钥文件一旦激活，在任何情况下都禁用对其使用 Read Record 命令。

在取密钥命令执行之前，已经通过**相互认证**（ACOS6-SAM 参考手册第 5.3 节）中讲述的相互认证过程，或者是 MIFARE Plus/MIFARE DESFire 的相互认证过程在目标卡片中建立了过程密钥。

***注：****GET KEY 命令只能获取密钥数据。*

| APDU | 说明 |
|---|---|
| CLA | 80h |
| INS | CAh |
| P1 | 取密钥，供 ACOS 卡写/重装密钥 |

<table>
<tr><th>APDU</th><th>说明</th></tr>
<tr><td></td><td>00h 响应数据是 MSAM 中的密钥<br>01h 响应数据是 16 个字节的分散密钥<br>02h 响应数据是 24 个字节的分散密钥<br>03h 响应数据是 MIFARE Plus 卡的 Change Key 命令</td></tr>
<tr><td></td><td>取密钥，供 DESFire 卡更改密钥，响应数据供 DESFire/DESFire EV1 更改密钥。
<table>
<tr><th></th><th>卡片类型</th><th>验证密钥号和修改密钥号*</th><th>密钥长度</th></tr>
<tr><td>80h</td><td>MIFARE DESFire</td><td>在 MIFARE DESFire 卡片中是不同的</td><td>16 字节</td></tr>
<tr><td>81h</td><td>MIFARE DESFire EV1</td><td>在 MIFARE DESFire EV1 卡片中是不同的</td><td>16 字节</td></tr>
<tr><td>82h</td><td>MIFARE DESFire EV1</td><td>在 MIFARE DESFire EV1 卡片中是不同的</td><td>24 字节</td></tr>
<tr><td>88h</td><td>MIFARE DESFire</td><td>在 MIFARE DESFire 卡片中是相同的</td><td>16 字节</td></tr>
<tr><td>89h</td><td>MIFARE DESFire EV1</td><td>在 MIFARE DESFire EV1 卡片中是相同的</td><td>16 字节</td></tr>
<tr><td>8Ah</td><td>MIFARE DESFire EV1</td><td>在 MIFARE DESFire EV1 卡片中是相同的</td><td>24 字节</td></tr>
</table></td></tr>
<tr><td>P2</td><td>SAM 中的 Key ID（用于变更的新密钥）</td></tr>
<tr><td>P3</td><td>若 P1 = 00h，P3 = 08h<br>若 P1 = 01/02h，P3 = 10h<br>若 P1 = 03h，P3 = 0Bh<br>若 P1 = 80/81/82/88/89/8Ah：P3 = 0Bh</td></tr>
<tr><td>数据</td><td>若 P1 = 00h, 命令数据为 $RND_{Target}$<br>若 P1 = 01/02h，命令数据为 $RND_{Target}$ + 目标卡片的序列号（或批号）<br>若 P1 = 03h<br>- 目标卡片的序列号（8 字节）<br>- 写命令（A0 或 A1）（1 个字节）<br>- BNr (2 个字节)<br>若 P1 = 80/81/82/88/89/8Ah：<br>- 目标卡片的序列号（8 字节）<br>- 初始 Key ID（SAM 卡中的 Key 存储了初始 key，00=DESFire 卡的默认 Key)<br>- Key No.(DESFire 卡的 Key No.)<br>- Key Version（DESFire 卡的 Key 版本，如未使用，值=00）</td></tr>
</table>

* 此列表指出所列卡片是否具有不同的 Change Key 和 Authenticate Key，或者两个密钥是否使用相同的值。

**特定的响应报文状态字节：**

| SW1 SW2 | 说明 |
|---|---|
| 69 85h | SAM 过程密钥未准备就绪 |
| 62 83h | 当前 DF 被锁定，或目标 EF 被锁定 |
| 69 86h | 未选择 DF |
| 69 81h | KEY 文件的类型错误，应该是内部线性变长文件 |
| 69 82h | 目标文件头块的校验和错误，或者不满足安全条件 |
| 6A 86h | P1 或 P2 无效 |
| 67 00h | P3 不正确 |
| 6A 83h | 目标密钥未准备好或密钥长度小于 16 |
| 61 1Ch | 命令成功，使用 GET RESPONSE 获取结果 |

## 5.5. 非接触式智能卡协议

### 5.5.1. ATR 的生成

读写器检测到 PICC 后，一个 ATR 会被发送至 PCSC 驱动来识别 PICC。

#### 5.5.1.1. ATR 信息格式（适用于 ISO14443-3 PICC）

| 字节 | 值 | 标记 | 说明 |
|---|---|---|---|
| 0 | 3Bh | 初始字符 | |
| 1 | 8Nh | T0 | 高半字节8表示：后续不存在TA1、TB1和TC1，只存在TD1。<br>低半字节 N 表示历史字符的个数（HistByte 0 - HistByte N-1) |
| 2 | 80h | TD1 | 高半字节8表示：后续不存在TA2、TB2和TC2，只存在TD2。<br>低半字节 0 表示协议类型为 T=0 |
| 3 | 01h | TD2 | 高半字节0表示后续不存在TA3、TB3、TC3和TD3。<br>低半字节 1 表示协议类型为 T=1 |
| 4 ~ 3+N | 80h | T1 | 类别指示字节，80表示在可选的COMPACT-TLV数据对象中或许存在一个状态标识符 |
| | 4Fh | Tk | 应用标识符存在标识 |
| | 0Ch | | 长度 |
| | RID | | 注册的应用提供商标识(RID) # A0 00 00 03 06 |
| | SS | | 标准字节 |
| | C0 ..C1h | | 卡片名称字节 |
| | 00 00 00 00h | RFU | RFU # 00 00 00 00 |
| 4+N | UU | TCK | T0至Tk的所有字节按位异或 |

**例如：**

MIFARE Classic 1K卡的ATR = {3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6Ah}

其中：

| | |
|---|---|
| **长度(YY)** | = 0Ch |
| **RID** | = {A0 00 00 03 06h} (PC/SC工作组) |
| 标准**(SS)** | = 03h (ISO 14443A，第3部分) |
| **卡片名称(C0 ..C1)** | = {00 01h} (MIFARE Classic 1K) |
| **标准(SS)** | = 03h：ISO 14443A，第3部分<br>= 11h：FeliCa |

**卡片名称(C0 ..C1)：**

| | |
|---|---|
| 00 01：MIFARE Classic 1K | 00 38：MIFARE Plus® SL2 2K |
| 00 02：MIFARE Classic 4K | 00 39：MIFARE Plus® SL2 4K |
| 00 03：MIFARE Ultralight® | 00 30：Topaz和Jewel |
| 00 26：MIFARE Mini® | 00 3B：FeliCa |
| 00 3A：MIFARE Ultralight® C | FF 28：JCOP 30 |
| 00 36：MIFARE Plus® SL1 2K | FF [SAK]：尚未定义的标签 |
| 00 37：MIFARE Plus® SL1 4K | |

### 5.5.1.2. ATR 信息格式（适用于 ISO14443-4 PICC）

<table>
<tr><th>字节</th><th>值</th><th>标记</th><th>说明</th></tr>
<tr><td>0</td><td>3Bh</td><td>初始字符</td><td></td></tr>
<tr><td>1</td><td>8Nh</td><td>T0</td><td>高半字节8表示：后续不存在TA1、TB1和TC1，只存在TD1。<br>低半字节 N 表示历史字符的个数（HistByte 0 - HistByte N-1)</td></tr>
<tr><td>2</td><td>80h</td><td>TD1</td><td>高半字节8表示：后续不存在TA2、TB2和TC2，只存在TD2。<br>低半字节 0 表示协议类型为 T=0</td></tr>
<tr><td>3</td><td>01h</td><td>TD2</td><td>高半字节0表示后续不存在TA3、TB3、TC3和TD3。<br>低半字节 1 表示协议类型为 T=1</td></tr>
<tr><td rowspan="2">4 ~ 3+N</td><td>XX</td><td>T1</td><td rowspan="2">历史字节：<br><br>ISO 14443-A：<br>来自ATS响应的历史字节。参考ISO 14443-4标准。<br><br>ISO 14443-B：
<table>
<tr><th>字节1~4</th><th>字节5~7</th><th>字节8</th></tr>
<tr><td>ATQB的应用数据</td><td>ATQB的协议信息字符</td><td>高半字节=ATTRIB命令的MBLI；低半字节(RFU)=0</td></tr>
</table></td></tr>
<tr><td>XX</td><td>Tk</td></tr>
<tr><td>4+N</td><td>UU</td><td>TCK</td><td>T0至Tk的所有字节按位异或</td></tr>
</table>

**例1：**
MIFARE® DESFire®的ATR = {3B 81 80 01 80 80h} // 6个字节的ATR

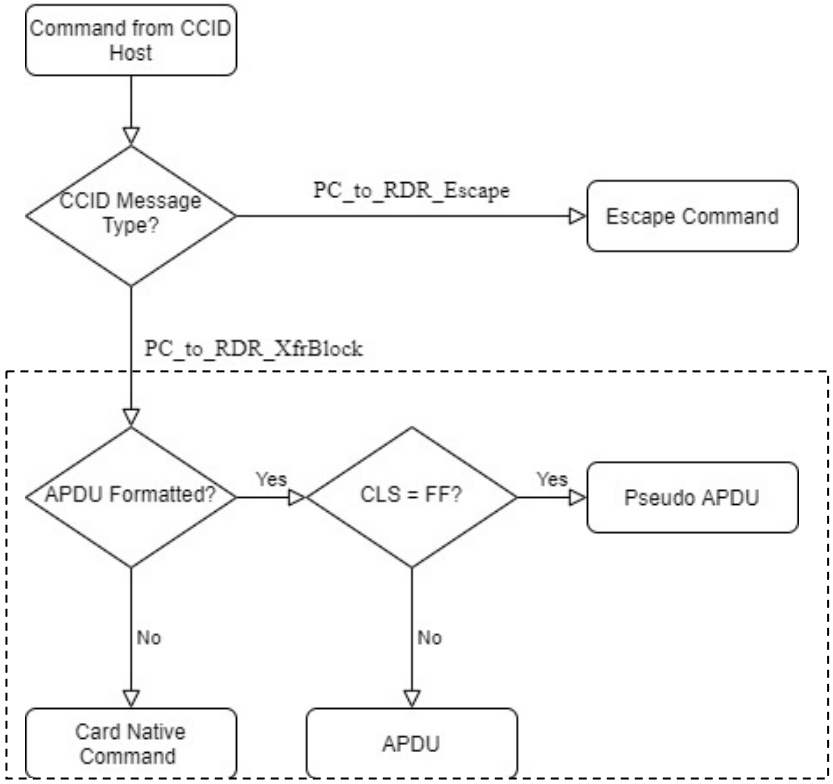
***注：****使用APDU“FF CA 01 00 00h”来区分是符合ISO 14443A-4的PICC还是符合ISO 14443B-4的PICC，并且如果有的话，取回完整的ATS。符合ISO 14443A-3或ISO 14443B-3/4的PICC会返回ATS。*

APDU命令 = FF CA 01 00 00h
APDU响应 = 06 75 77 81 02 80 90 00h
ATS = {06 75 77 81 02 80h}

**例2：**
EZ-Link的ATR = {3B 88 80 01 1C 2D 94 11 F7 71 85 00 BEh}
ATQB的应用数据 = 1C 2D 94 11h
ATQB的协议信息 = F7 71 85h
ATTRIB 的 MBLI = 00h

### 5.5.2. APDU、私有 APDU 和卡片专有命令

用户可以通过 PC_to_RDR_XfrBlock 消息向读写器发送 APDU、私有 APDU（Pseudo APDU）和卡片专有命令（Card Native Command）。处理完命令后，读写器会通过 RDR_to_PC_DataBlock 消息返回响应。




CCID 主机可以使用 CCID 报文 PC_to_RDR_XfrBlock（对应于 PCSC API 中的 SCardTransmit()）向读卡器发送卡片专有命令或 APDU。对于 PICC，如果卡片支持 ISO14443 第 4 部分协议或 Innovation 协议，读写器会将命令/APDU 打包到协议帧中直接发送给卡片，不会对命令/APDU 进行解析。如果卡片不支持这两种协议，则会向 CCID 主机返回消息“6A 81”。

注：由于 Microsoft Window 支持智能卡即插即用，Microsoft Window 可能在卡片出示时向卡片发送 APDU 指令。该操作会使 DESFire 卡进入 ISO APDU 模式，使得卡片无法接收专有命令，除非重置卡片。通常情况下，Microsoft Window 会在卡片处于无反应状态 10 秒后重置卡片（通过 PC_to_RDR_IccPowerOff）。

### 5.5.3. PICC 的 PCSC 私有 APDU（带专有扩展）

下列私有（Pseudo）APDU 用于间接访问非接触卡。CCID 主机可以使用 CCID 报文 PC_to_RDR_XfrBlock（对应于 PCSC API 中的 SCardTransmit()）向读卡器发送这些 APDU。收到私有 APDU 后，读写器会解读生成低级别的卡片命令，然后发送给卡片。卡片处理完这些低级别命令后，读写器收集卡片响应并创建响应发回 CCID 主机。

### 5.5.3.1. 获取数据（Get Data）[FF CA …]

此命令用来读取激活过程中获得的数据，例如序列号、协议参数等。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Data | FFh | CAh | 见下表 | | 00h<br>(全长) |

命令参数

| P1 | P2 | 含义 |
|---|---|---|
| 00h | 00h | 获取卡片的 UID/PUPI/SN |
| 01h | 00h | 获取 A 类第 4 部分的 ATS |
| 02h | 00h | 获取以下卡片类型相关数据，传输顺序：<br>A 类：2 字节 ATQA/ATVA + 4/7/10 字节 UID + 1 字节最后一个 SAK。<br><br>B 类：12 字节 ATQB |
| 80h | 00h | 获取以下卡片类型相关数据，传输顺序：<br>A 类：2 字节 ATQA/ATVA + 4/7/10 字节 UID + 1/2/3 字节 SAK。<br><br>B 类：12 字节 ATQB<br><br>FeliCa：17 字节 ATQ (+ 6 字节 ATTR，如已激活)<br><br>SRI：8 字节 UID + 1 字节芯片 ID。<br><br>ISO15693：1 字节 DSFID + 8 字节 UID<br><br>CTS：4 字节 SN + 2 字节 ATQT<br><br>Innovatron：4 字节 SN + 1 字节标签地址。 |

响应

| 响应 | 响应数据域 | | |
|---|---|---|---|
| 结果 | 数据 | SW1 | SW2 |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 6X XXh | 失败。 |

**例如：**

获取“已经建立连接的 PICC”的序列号：

UINT8 GET_UID[5] = {FF, CA, 00, 00, 00};

获取“已经建立连接的 ISO 14443-A PICC”的 ATS：

UINT8 GET_ATS[5] = {FF, CA, 01, 00, 00};

### 5.5.3.2. 加载密钥（Load Key）[FF 82 …]

此命令用于向密钥缓冲区号指定的内部密钥缓冲区加载密钥数据。密钥缓冲区属于易失存储区，里面的内容将用于认证。此命令不会产生卡片数据传输。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| Load Authentication Keys | FFh | 82h | 00h | 密钥缓冲区号（0-1） | 密钥长度 | 密钥数据 |

密钥长度/数据

| 卡片类型 | 密钥长度 (Lc) | 密钥数据 (按传输/存储顺序) |
|---|---|---|
| MIFARE Standard<br>MIFARE Plus SL1 | 06h | 6 字节 Crypto1 Key A/B。 |
| MIFARE Plus SL1<br>MIFARE Plus SL2 | 16h | 6 字节 Crypto1 Key A/B + 16 字节 AES Key。 |
| MIFARE Plus SL2 | 06h | 6 字节加密 Crypto1 Key A/B。 |
| MIFARE UltraLightC<br>MIFARE DESFire | 10h | 16 字节 2K3DES Key。 |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 6X XXh | 失败。 |

**例如：**

// 向易失性存储器位置 00h 加载密钥 {FF FF FF FF FF FFh}。

APDU = {FF 82 00 00 06 FF FF FF FF FF FFh}

### 5.5.3.3. 认证（Authenticate）[FF 86 00 00 05 …]

此命令用于向卡片执行认证过程。认证成功后，用户可以访问受保护的块/页。命令发送前，用户需通过 Load Key 命令将正确的密钥数据加载到**密钥缓冲区号**指定的缓冲区。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| Authenticate | FFh | 86h | 00h | 00h | 05h | 见下表 |

命令数据

| 字节 0 | 字节 1 | 字节 2 | 字节 3 | 字节 4 |
|---|---|---|---|---|
| 01h | 00h (RFU) | 地址 | 密钥类型 | 密钥缓冲区号 |

地址和密钥类型

| 卡类型 | 地址 | 密钥类型 |
|---|---|---|
| MIFARE Standard<br>MIFARE Plus SL1<br>MIFARE Plus SL2 | 00h~FFh：块 0~255 | 60h：Crypto1 Key A<br>61h：Crypto1 Key B |
| MIFARE UltraLightC | 00h (RFU) | 80h：2K3DES |
| MIFARE DESFire | 00h~0Eh：DESFire 密钥号 0~14 | 0Ah：2K3DES |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 6X XXh | 失败。 |

| 扇区<br>(共 16 个扇区，每个扇区包含 4 个连续的块) | 数据块<br>(3 个块，每块 16 字节) | 尾部块<br>(1 个块，16 字节) |
|---|---|---|
| 扇区 0 | 00h – 02h | 03h |
| 扇区 1 | 04h – 06h | 07h |
| .. | .. | .. |
| .. | .. | .. |
| 扇区 14 | 38h – 0Ah | 3Bh |
| 扇区 15 | 3Ch – 3Eh | 3Fh |

1 KB

**表13：** MIFARE Classic 1K 卡的内存结构

| 扇区<br>(共 32 个扇区，每个扇区包含 4 个连续的块) | 数据块<br>(3 个块，每块 16 字节) | 尾部块<br>(1 个块，16 字节) |
|---|---|---|
| 扇区 0 | 00h ~ 02h | 03h |
| 扇区 1 | 04h ~ 06h | 07h |
| .. | | |
| .. | | |
| 扇区 30 | 78h ~ 7Ah | 7Bh |
| 扇区 31 | 7Ch ~ 7Eh | 7Fh |

| 扇区<br>(共 8 个扇区，每个扇区包含 16 个连续的块) | 数据块<br>(15 个块，每块 16 字节) | 尾部块<br>(1 个块，16 字节) |
|---|---|---|
| 扇区 32 | 80h ~ 8Eh | 8Fh |
| 扇区 33 | 90h ~ 9Eh | 9Fh |
| .. | | |
| .. | | |
| 扇区 38 | E0h ~ EEh | EFh |
| 扇区 39 | F0h ~ FEh | FFh |

2 KB

**表14：** MIFARE Classic 4K 卡的内存结构

| 字节号 | 0 | 1 | 2 | 3 | 页 |
|---|---|---|---|---|---|
| 序列号 | SN0 | SN1 | SN2 | BCC0 | 0 |
| 序列号 | SN3 | SN4 | SN5 | SN6 | 1 |
| 内部/锁 | BCC1 | Internal | Lock0 | Lock1 | 2 |
| OTP | OPT0 | OPT1 | OTP2 | OTP3 | 3 |
| 数据读/写 | Data0 | Data1 | Data2 | Data3 | 4 |
| 数据读/写 | Data4 | Data5 | Data6 | Data7 | 5 |
| 数据读/写 | Data8 | Data9 | Data10 | Data11 | 6 |
| 数据读/写 | Data12 | Data13 | Data14 | Data15 | 7 |
| 数据读/写 | Data16 | Data17 | Data18 | Data19 | 8 |
| 数据读/写 | Data20 | Data21 | Data22 | Data23 | 9 |
| 数据读/写 | Data24 | Data25 | Data26 | Data27 | 10 |
| 数据读/写 | Data28 | Data29 | Data30 | Data31 | 11 |
| 数据读/写 | Data32 | Data33 | Data34 | Data35 | 12 |
| 数据读/写 | Data36 | Data37 | Data38 | Data39 | 13 |
| 数据读/写 | Data40 | Data41 | Data42 | Data43 | 14 |
| 数据读/写 | Data44 | Data45 | Data46 | Data47 | 15 |

512 位
或
64 字节

**表15**：MIFARE Ultralight 卡的内存结构

**例如：**

// 要使用{TYPE A，密钥号 00h}验证块 04h。PC/SC V2.01，弃用

APDU = {FF 88 00 04 60 00h};

// 要使用{TYPE A，密钥号 00h}验证块 04h。PC/SC V2.07

APDU = {FF 86 00 00 05 01 00 04 60 00h}

***注：MIFARE Ultralight 不需要进行验证，其内存可以自由访问。***

### 5.5.3.4. 读取二进制块（Read Binary Blocks）[FF B0 …]

此命令用于从指定块/页地址的位置开始从 PICC 读取指定字节的数据。根据卡片类型的不同，调用此命令前，用户可能需要先进行认证并获得这些块/页的访问权限。

命令：

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Read Binary Blocks | FFh | B0h | 模式和地址 | | 待读取的字节数 |

P1/P2（模式和地址）

| 卡类型 | P1[7:4]<br>模式 | P1[3:0] + P2[7:0]<br>起始地址（MSB 在前） |
|---|---|---|
| MIFARE Standard<br>MIFARE Plus SL1<br>MIFARE Plus SL2 | 00h：跳过尾部块<br>08h：包含尾部块 | 000h~0FFh：块 0~255 |
| MIFARE UltraLight<br>MIFARE UltraLightC | 00h（保留） | 000h~02Fh：页 0~47 |
| SRIX4K/SRT512 | 00h（保留） | 000h~07Fh：块 0~127<br>0FFh：系统区域 |
| PicoPass | 00h（保留） | 000h~0FFh：块 0~255 |
| ISO15693 | 00h（保留） | 000h~0FFh：块 0~255 |
| Topaz/NFC Type-1 标签 | 00h（保留） | 000h~7FFh：字节地址 |

Le（待读取的字节数）

| 类型 | 字节 0 | 字节 1 | 字节 2 |
|---|---|---|---|
| 短 | 00h：读取 256 字节<br>01h~FFh：读取 1~255 字节 | -- | |
| 长 | 00h | 0000h：读取 65536 字节<br>0001h~FFFFh：读取 1~65535 字节 | |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 6X XXh | 失败。 |

例如：

// 从二进制块 04h 中读取 16 字节（MIFARE Classic 1K 或 4K）

APDU = FF B0 00 04 10h

// 从二进制块 80h 开始读取 240 字节（MIFARE Classic 4K）

// 块 80h 至块 8Eh（15 个块）

APDU = FF B0 00 80 F0h

### 5.5.3.5. 更新二进制块（Update Binary Blocks）[FF D6 …]

此命令用于从指定块/页地址的位置开始向 PICC 写入指定字节（必须是块/页大小的倍数）的数据。根据卡片类型的不同，调用此命令前，用户可能需要先进行认证并获得这些块/页的访问权限。

向卡片内的块/页写入数据可能改变卡片的安全设置（例如 MIFARE 卡的尾部块），所以应当格外小心。如果写入错误的数据或者操作失败，可能会将卡锁死。为了减少卡片锁定的风险，不建议在涉及安全块/页时，通过一个 APDU 命令向多个块/页写入数据。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| Update Binary Blocks | FFh | D6h | 模式和地址 | | 待写入的字节数量 | 数据字节 |

P1/P2（模式和地址）和 Write Size 一致（块/页大小）

| 卡类型 | P1[7:4]<br>模式 | P1[3:0] + P2[7:0]<br>起始地址（MSB 在前） | 块/页大小（字节） |
|---|---|---|---|
| MIFARE Standard<br>MIFARE Plus SL1<br>MIFARE Plus SL2 | 0x0：跳过尾部块<br>0x8：包含尾部块 | 000h~0FFh：块 0~255 | 16 |
| MIFARE UltraLight<br>MIFARE UltraLightC | 0x0（保留） | 000h~02Fh：页 0~47 | 4 |
| SRIX4K/SRT512 | 0x0（保留） | SRIX4K/SRT512 | 4 |
| PicoPass | 0x0（保留） | PicoPass | 8 |
| ISO15693 | 0x0（保留） | ISO15693 | 1 ~ 32 |
| Topaz/NFC<br>Type-1 标签 | 0x0：包括擦除<br>0x8：不包括擦除 | 000h~7FFh：字节地址 | 1(地址 78h)或 8(其它) |

Lc（待写入的字节数量）

| 类型 | 字节 0 | 字节 1 | 字节 2 |
|---|---|---|---|
| 短 | 01h~FFh：写入 1~255 个字节 | -- | |

| 类型 | 字节 0 | 字节 1 | 字节 2 |
|---|---|---|---|
| 长 | 00h | 0001h~FFFFh：写入 1~65535 个字节 | |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 6X XXh | 失败。 |

**例如：**

// 将 MIFARE Classic 1K/4K 卡中的二进制块 04h 的数据更新为{00 01 ..0Fh}

APDU = {FF D6 00 04 10 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0Fh}

// 将 MIFARE Ultralight 卡中的二进制块 04h 的数据更新为{00 01 02 03h}

APDU = {FF D6 00 04 04 00 01 02 03h}

### 5.5.4. PCSC 2.0 第 3 部分支持的 APDU 指令（V2.02 及以上版本）

PCSC 2.0 第三部分规定的命令用于将数据从应用程序透明传递给非接触式标签，将接收到的数据透明返回给应用程序和协议，同时切换协议。

#### 5.5.4.1. PCSC 2.0 第 3 部分命令流程图

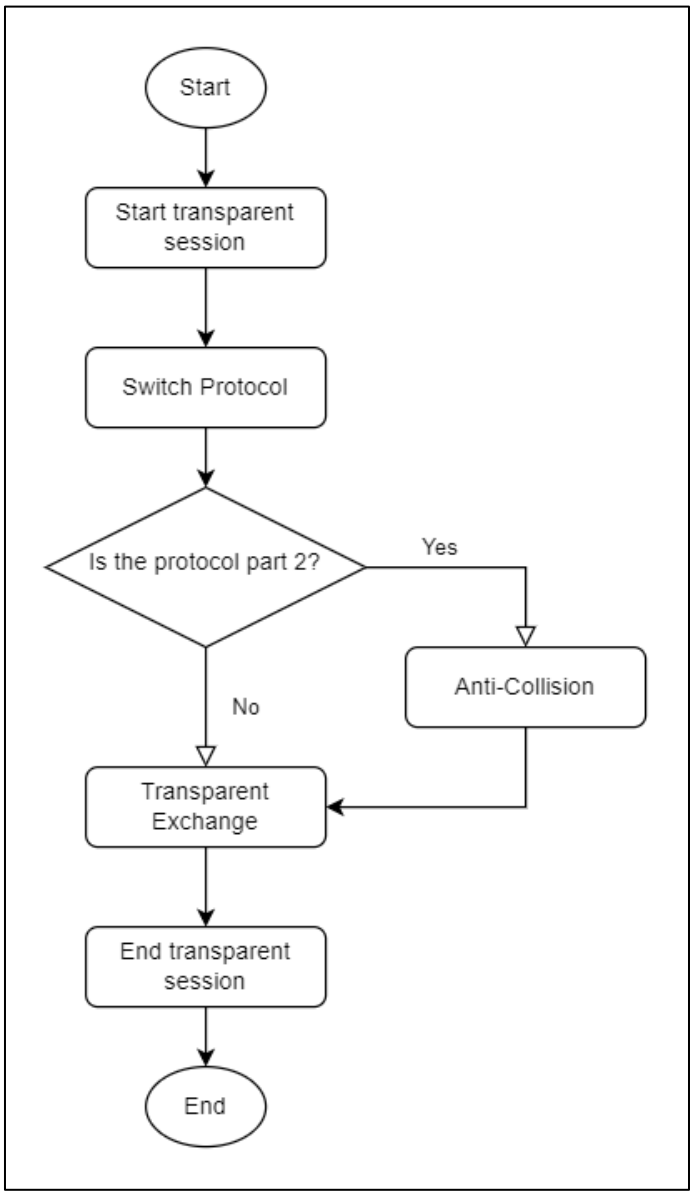



**图9：** 透明会话流程图

#### 5.5.4.2. 命令和响应的 APDU 格式

命令格式

| CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|
| FFh | C2h | 00h | 功能 | 数据长度 | 数据[数据长度] |

**其中功能**（1 个字节）：

00h = 会话管理
01h = 透明交互

02h = 切换协议
其它 = RFU

响应格式

| 响应数据域 | SW1 | SW2 |
|---|---|---|
| 编码的数据域 BER-TLV | | |

每个命令都会返回 SW1 和 SW2 加上响应数据域（如有）。SW1 和 SW2 符合 ISO 7816 的规定。也应使用以下 C0 数据对象的 SW1 SW2。

C0 数据元格式

| 标签 | 长度（1 字节） | SW2 |
|---|---|---|
| C0h | 03h | 错误状态 |

错误状态说明

| 错误状态 | 说明 |
|---|---|
| XX SW1 SW2 | XX = APDU 中不良数据对象的编号<br>00 = APDU 常见错误<br>01 = 第 1 个数据对象有错误<br>02 = 第 2 个数据对象有错误 |
| 00 90 00h | 未发生错误 |
| XX 62 82h | 数据对象 XX 告警，请求信息不存在 |
| XX 63 00h | 未有信息 |
| XX 63 01h | 由于其它数据对象失败，停止执行 |
| XX 6A 81h | 不支持数据对象 XX |
| XX 67 00h | 意外长度的数据对象 XX |
| XX 6A 80h | 意外值的数据对象 XX |
| XX 64 00h | 数据对象 XX 执行错误（IFD 无响应） |
| XX 64 01h | 数据对象 XX 执行错误（ICC 无响应) |
| XX 6F 00h | 数据对象 XX 失败，没有准确诊断 |

第一个字节的值表示错误数据对象 XX 的编号，最后两个字节是对错误的解释。允许使用 ISO 7816 规定的 SW1 SW2 值。

如果 C-APDU 数据域中存在多个数据对象，而且其中一个数据对象失败，那么在其它数据对象不依赖于失败的数据对象的情况下，IFD 可以处理接下来的数据对象。

### 5.5.4.3. 管理会话（Manage Session）[FF C2 00 00 …]

此命令允许用户开启会话并禁用轮询功能，以进行后续通信。通信完成后，用户应立即结束会话。

需要注意的是，如果不正确使用，此命令可能会使读写器无法检测到卡片是否存在，并且无法自动恢复，除非逻辑/物理断开读写器连接。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 | Le |
|---|---|---|---|---|---|---|---|
| Manage Session | FFh | C2h | 00h | 00h | Cmd 数据长度 | Cmd TLV | --/00h |

响应状态码

| 响应数据 | SW1 SW2 | 含义 |
|---|---|---|
| -- | 90 00h | 操作成功完成。 |
| Rsp TLV | 90 00h | Le = 0x00：Cmd TLV 之一失败。如需了解错误详情，请参考 Rsp TLV。 |
| -- | 6X XXh | Le = --：Cmd TLV 之一失败。 |

Cmd TLV

| Cmd | 含义 |
|---|---|
| Start Session:<br>81 00h | 开始会话并禁用轮询。 |
| RF Off:<br>83 00h | 关闭 RF。 |
| Timer:<br>5F 46 04h [TIME] | 设置下一个 RF On/Off TLV 前的休眠时间<br>[TIME]: 4 字节值（MSB 在前），范围为 1000 到 100000 us。实际休眠时间将四舍五入到最接近的 1000us。 |
| RF On:<br>84 00h | 打开 RF。 |
| End Session:<br>82 00h | 结束会话，重新启用轮询。 |

Rsp TLV

| Rsp | 含义 |
|---|---|
| TLV Error:<br>C0 03 NN 6X XXh | 第 NN 个命令 TLV 错误。 |

#### 5.5.4.3.1. 开始会话数据对象（Start Session Data Object）

此命令用于开启透明会话。会话开始后，自动轮询功能被禁用，直到会话结束。

开始会话数据对象

| 标签 | 长度（1 字节） | 值 |
|---|---|---|
| 81h | 00h | - |

#### 5.5.4.3.2. 结束会话数据对象（End Session Data Object）

此命令用于终止透明会话。在新的会话开始之前，重置为自动轮询状态。

结束会话数据对象

| 标签 | 长度（1 字节） | 值 |
|---|---|---|
| 82h | 00h | - |

#### 5.5.4.3.3. 关闭 RF 数据对象（Turn Off the RF Data Object）

此命令用于关闭天线场。

关闭 RF 场数据对象

| 标签 | 长度（1 字节） | 值 |
|---|---|---|
| 83h | 00h | - |

#### 5.5.4.3.4. 开启 RF 数据对象（Turn On the RF Data Object）

此命令用于开启天线场。

开启 RF 场数据对象

| 标签 | 长度（1 字节） | 值 |
|---|---|---|
| 84h | 00h | - |

#### 5.5.4.3.5. 计时器数据对象（Timer Data Object）

此命令用于创建一个 32 位计时器数据对象，以 1 µs 为单位。

**例如：**如果在关闭 RF 数据对象和开启 RF 数据对象之间有 5000 µs 的计时器数据对象，读写器会关闭 RF 场大约 5000µs，然后再重新开启 RF 场。

计时器数据对象

| 标签 | 长度（1 字节） | 值 |
|---|---|---|
| 5F 46h | 04h | 计时器（4 个字节） |

### 5.5.4.4. 透明交互（Transparent Exchange）[FF C2 00 01 …]

此命令允许用户向卡片发送/从卡片接收任意位或字节，并可以选择配置各种链路和传输层（例如 ISO14443 第 4 部分）以及一些链路层冗余（CRC 和奇偶校验）。用户可以将任何卡片特定的原始数据嵌入到这个私有 APDU 中，然后发送给卡片。

需要注意的是，此命令可能会干扰卡支持的内部处理过程，可能会在不通知驱动程序/固件的情况下更改卡片状态，并且可能需要重置和/或移除卡片才能使驱动程序/固件恢复正常。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 | Le |
|---|---|---|---|---|---|---|---|
| Transparent Exchange | FFh | C2h | 00h | 01h | Cmd 数据长度 | Cmd TLV | 00h |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 6X XXh | 失败。 |

Cmd TLV

| Cmd | 含义 |
|---|---|
| Transceive Flag:<br>90 02 [Flag] 00h | 设置下列 Transceive TLV 的 Flag<br>Flag[7:5]: RFU; 设为 0<br>Flag[4]: 设为禁用 ISO14443-4<br>Flag[3]: 设为禁止接收奇偶校验处理<br>Flag[2]: 设为禁止传输奇偶校验处理<br>Flag[1]: 设为禁止接收 CRC 处理<br>Flag[0]: 设为禁止传输 CRC 处理<br><br>如果此 TLV 缺失，则使用上一个命令中设置的 Flag 值。如果从未设置过 Flag 值，则使用当前协议值。 |
| Transmit Bit Frame:<br>91 01h [NumBit] | 设置下列 Transceive TLV 的 Bit Frame。如果此 TLV 缺失，则默认值为 0。<br><br>NumBit[7:3]: RFU; 设为 0<br>NumBit[2:0]: 最后一个字节中的有效位的数量（0 表示所有的位都有效） |
| Timer:<br>5F 46 04h [TIME] | 设置下列 Transceive TLV 的超时时间。<br>[TIME]: 4 字节值（MSB 在前），范围为 1 us 到 1000000 us。实际超时时间将四舍五入到最接近的 302.07 x 20~15 us。<br><br>如果此 TLV 缺失，则使用先前设置的 FWTI 值作为超时时间。 |
| Set FWTI: | 设置 Transceive 的 FWT/超时。若先前未通过“FF C2h ...”命令设置 FWTI，则默认值为 0. |

| Cmd | 含义 |
|---|---|
| FF 6E 03 03 01h<br>[FWTI] | FWTI: 0 ~ 15, FWT/超时 = 302.07 x 2FWTI us |
| Transceive:<br>95h [Size] [Data] | Size: BER-TLV 长度数据域中编码数据的大小<br>Data: 待传输的数据 |

Rsp TLV

| Rsp | 含义 |
|---|---|
| Receive Bit framing:<br>92 01h [NumBit] | NumBit[7:3]: RFU; 设为 0.<br>NumBit[2:0]: 最后一个字节中的有效位的数量（0 表示所有的位都有效） |
| Response:<br>97h [Size] [Data] | Size: BER-TLV 长度字段中编码数据的大小<br>Data: 接收的数据。 |
| Response Status:<br>96 02h [Status] 00h | Status [7:4]: RFU.<br>Status[3]: 成帧错误。<br>Status[2]: 奇偶校验错误。<br>Status[1]: RFU.<br>Status[0]: CRC 错误。 |

#### 5.5.4.4.1. 发送和接收标志数据对象（Transmission and Reception Flag Data Object）

此命令用于为下列传输定义成帧参数和 RF 参数。

发送和接收标志数据对象

| 标签 | 长度（1 字节） | 值 | | |
|---|---|---|---|---|
| | | 字节 0 | | 字节 1 |
| | | 位 | 说明 | |
| 90h | 02h | 0 | 0 – 在传输的数据后添加 CRC<br>1 – 不在传输的数据后添加 CRC | 00h |
| | | 1 | 0 – 对接收数据进行 CRC 检查<br>1 – 不对接收数据进行 CRC 检查 | |
| | | 2 | 0 – 在传输的数据中插入奇偶校验位<br>1 – 不插入奇偶校验位 | |
| | | 3 | 0 – 期望接收的数据中含有奇偶校验位<br>1 – 不期望接收的数据中含有奇偶校验位（即不进行奇偶校验） | |
| | | 4 | 0 – 在传输数据中添加协议头，或者从响应中丢弃<br>1 – 不添加或者丢弃协议头（如有）（例如 PCB、CID、NAD） | |
| | | 5-7 | RFU | |

#### 5.5.4.4.2. 发送位成帧数据对象（Transmission Bit Framing Data Object）

此命令用于定义待发送或待收发数据中最后一个字节的有效位数量。

发送位成帧数据对象

| 标签 | 长度（1 字节） | 值 | |
|---|---|---|---|
| | | 位 | 说明 |
| 91h | 01h | 0-2 | 最后一个字节中的有效位数量（0 表示所有的位都有效） |
| | | 3-7 | RFU |

发送位成帧数据对象只能和“发送”或“收发”数据对象一起使用。如果不存在此数据对象，则表明所有的位都有效。

#### 5.5.4.4.3. 收发数据对象（Transceive Data Object）

此命令用于发送和接收来自 ICC 的数据。数据发送完成后，读写器会保持等待状态，直到计时器数据对象规定的时间结束。

如果没有在数据域中定义计时器数据对象，读写器会保持等待状态直到设置参数 FWTI 数据对象规定的时间段结束。如果没有设置 FWTI，读写器会等待大约 302 µs。

收发数据对象

| 标签 | 长度（1 字节） | 值 |
|---|---|---|
| 95h | 数据长度 | 数据（N 个字节） |

#### 5.5.4.4.4. 计时器数据对象（Timer Data Object）

此命令用于创建一个 32 位计时器数据对象，以 1 µs 为单位。

**例如：**如果有 5000 µs 的计时器数据对象，读写器会等待后续的收发 TLV 大约 5000µs，然后再超时。

计时器数据对象

| 标签 | 长度（1 字节） | 值 |
|---|---|---|
| 5F 46h | 04h | 计时器（4 个字节） |

#### 5.5.4.4.5. 响应位成帧数据对象（Response Bit Framing Data Object）

此命令用于在响应中提示接收到的发送位成帧数据对象

| 标签 | 长度（1 字节） | 值 | |
|---|---|---|---|
| | | 位 | 说明 |
| 92h | 01h | 0-2 | 最后一个字节中的有效位数量（0 表示所有的位都有效） |
| | | 3-7 | RFU |

发送位成帧数据对象只能和“发送”或“收发”数据对象一起使用。如果不存在此数据对象，则表明所有的位都有效。

#### 5.5.4.4.6. 响应状态数据对象（Response Status Data Object）

此命令用于在响应中提示接收到的数据状态

响应状态数据对象

| 标签 | 长度（1 字节） | 值 | | |
|---|---|---|---|---|
| | | 字节 0 | | 字节 1 |
| | | 位 | 说明 | |
| 96h | 02h | 0 | 0 – CRC 正确，或未进行校验<br>1 – CRC 校验失败 | RFU |
| | | 1 | 0 – 无冲突<br>1 – 检测到冲突 | |
| | | 2 | 0 – 无奇偶校验位错误<br>1 – 检测到奇偶校验位错误 | |
| | | 3 | 0 – 无成帧错误<br>1 – 检测到成帧错误 | |
| | | 4 - 7 | RFU | |

#### 5.5.4.4.7. 响应数据对象（Response Data Object）

此命令用于在响应中提示接收到的数据状态

响应数据对象

| 标签 | 长度（1 字节） | 值 |
|---|---|---|
| 97h | 数据长度 | 响应数据（N 字节） |

### 5.5.4.5. 切换协议（Switch Protocol）[FF C2 00 02 …]

此命令允许用户切换并指定协议，以及选择协议层和参数。

需要注意的是，此命令可能会干扰卡支持的内部处理过程，可能会在不通知驱动程序/固件的情况下更改卡片状态，并且可能需要重置和/或移除卡片才能使驱动程序/固件恢复正常。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 | Le |
|---|---|---|---|---|---|---|---|
| Switch Protocol | FFh | C2h | 00h | 02h | Cmd 数据长度 | Cmd TLV | 00h |

响应状态码

| 响应数据 | SW1 SW2 | 含义 |
|---|---|---|
| Rsp TLV | 90 00h | 数据成功。 |
| -- | 90 00h | 成功 |

| 响应数据 | SW1 SW2 | 含义 |
|---|---|---|
| -- | 6X XXh | 失败。 |

Cmd TLV

| Cmd | 含义 |
|---|---|
| Set Baud:<br>FF 6E 03 05 01h [Baud] | 设置要在协议切换过程中使用的第 4 部分/层的波特率。如果尚未通过“FF C2h ...”命令设置[Baud]，则默认值为 98h (106 kbps)。<br><br>Baud[7:2]: RFU, 设为 100110b。<br>Baud[1:0]: 待设置的波特率，00b (106 kbps), 01b (212 kbps), 10b (424 kbps), 11b (848 kbps)。 |
| Switch Protocol:<br>8F 02h [RF] [Layer] | 将协议切换到指定的 RF 和/或层。<br><br>[RF]:<br>00h: ISO14443A, 01h: ISO14443B<br>02h: ISO15693, 03h: FeliCa, FFh: 当前 RF<br>其它：RFU<br><br>[Layer]:<br>02h: 第 2 层/部分 03h: 第 3 层/部分,<br>04h: 第 4 层/部分 (仅用于 A/B)<br>其它：RFU<br><br>注：如果切换到第 2 层/部分，则必须处于透明会话（禁用轮询）状态。 |

Rsp TLV

| Rsp | 含义 |
|---|---|
| Response:<br>8Fh [Size] [Data] | Size: BER-TLV 长度数据域中编码数据的大小<br>Data: ATR (如果是第 4 部分)、最终 SAK (如果是 A 类第 3 部分)、或者 ATQB 中的 PI (如果是 B 类第 3 部分)。 |

#### 5.5.4.5.1. 切换协议数据对象（Switch Protocol Data Object）

此命令用于指定协议和不同标准层。

切换协议数据对象

| 标签 | 长度（1 字节） | 值 | |
|---|---|---|---|
| | | 字节 0 | 字节 1 |
| 8Fh | 02h | 00h – ISO/IEC14443 A 类<br>01h – ISO/IEC14443 B 类<br>02h - ISO15693<br>03h – FeliCa<br>其它 – RFU | 02h – 切换到第二层<br>03h – 切换或激活到第三层<br>04h – 激活到第四层<br>其它 - RFU |

#### 5.5.4.5.2. 响应数据对象（Response Data Object）

此命令用于在响应中提示接收到的数据状态

响应数据对象

| 标签 | 长度（1 字节） | 值 |
|---|---|---|
| 5F 51h | 数据长度 | ATR |
| 8Fh | 数据长度 | 最终 SAK (如果是 A 类第 3 部分)、或者 ATQB 中的 PI (如果是 B 类第 3 部分)。 |

### 5.5.4.6. PCSC 2.0 第 3 部分示例

1. 开始透明会话

   命令：**FF C2 00 00 02 81 00**

   响应：**C0 03 00 90 00 90 00**

2. 关闭天线场.

   命令: **FF C2 00 00 02 83 00**

   响应: **C0 03 00 90 00 90 00**

3. 打开天线场

   命令：**FF C2 00 00 02 84 00**

   响应：**C0 03 00 90 00 90 00**

4. 激活 ISO 14443-4A

   命令： **FF C2 00 02 04 8F 02 00 04**

   响应： **C0 03 01 64 01 90 00** （如果不存在卡片）

   **C0 03 00 90 00 5F 51 [Len] [ATR] 90 00**

5. 将 PCB 设为 0Ah，并在传输数据中启用 CRC、奇偶校验和协议头。

   命令：**FF C2 00 01 0A 90 02 00 00 FF 6E 03 07 01 0A**

   响应：**C0 03 00 90 00 90 00**

6. 发送 APDU “80B2000008”至卡片并取响应。

   命令：**FF C2 00 01 0E 5F 46 04 40 42 0F 00 95 05 80 B2 00 00 08**

   响应：**C0 03 00 90 00 92 01 00 96 02 00 00 97 0C [卡片响应] 90 00**

7. 结束透明会话。

   命令：**FF C2 00 00 02 82 00**

   响应：**C0 03 00 90 00 90 00**

### 5.5.5. PICC 的专属私有 APDU

下列私有（Pseudo）APDU 用于间接访问非接触卡，是对 PCSC Pseudo APDU 的补充。这些 APDU 的内部处理过程与 PCSC Pseudo APDU 类似。

#### 5.5.5.1. 写入值块（Write Value Block）[FF D7 …]

此命令用于将 4 个字节的值写入兼容 MIFARE 标准的卡的块。调用此命令前，用户应当先成功进行认证并获得对该块的访问权限。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| Write Value Block | FFh | D7h | 00h | 块号 | 05h | 见下表 |

命令数据

| 字节 0 | 字节 1 | 字节 2 | 字节 3 | 字节 4 |
|---|---|---|---|---|
| 00h | 4 个字节的值（MSB 在前） | | | |

**例 1：** Decimal －4 = {FFh, FFh, FFh, FCh}

| VB_Value | | | |
|---|---|---|---|
| MSB | | | LSB |
| FFh | FFh | FFh | FCh |

**例 2：** Decimal 1 = {00h, 00h, 00h, 01h}

| VB_Value | | | |
|---|---|---|---|
| MSB | | | LSB |
| 00h | 00h | 00h | 01h |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 6X XXh | 失败。 |

### 5.5.5.2. 读取值块（Read Value Block） [FF B1 ···]

此命令用于从兼容 MIFARE 标准的卡的有效值块中读取 4 个字节的值。调用此命令前，用户应当先成功进行认证并获得对该块的访问权限。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Read Value Block | FFh | B1h | 00h | 块号 | 04h |

**例 1：** Decimal －4 = {FFh, FFh, FFh, FCh}

| 值 | | | |
|---|---|---|---|
| MSB | | | LSB |
| FFh | FFh | FFh | FCh |

**例 2：** Decimal 1 = {00h, 00h, 00h, 01h}

| 值 | | | |
|---|---|---|---|
| MSB | | | LSB |
| 00h | 00h | 00h | 01h |

响应

| 响应数据 | SW1 SW2 | 含义 |
|---|---|---|
| 4 个字节的值（MSB 在前） | 90 00h | 数据成功。 |
| -- | 6X XXh | 失败。 |

### 5.5.5.3. 减少/增加值（Decrement/Increment Value）[FF D7 ···]

此命令用于从源块减少/增加一个 4 字节的值，并将结果存入目标块（卡片需兼容 MIFARE 标准）。如果要将结果存入同一个源块，则可以将目标块的编号设为 0 或者源块号。调用此命令前，用户应当先成功进行认证并获得对源块和目标块的访问权限。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| Decrement/Increment Value | FFh | D7h | 目标块# | 源块# | 05h | 见下表 |

命令数据

| 字节 0 | 字节 1 | 字节 2 | 字节 3 | 字节 4 |
|---|---|---|---|---|
| 01h | 4 个字节的增加值（MSB 在前） | | | |
| 02h | 4 个字节的减少值（MSB 在前） | | | |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 6X XXh | 失败。 |

### 5.5.5.4. 复制值块（Copy Value Block）[FF D7 …]

此命令用于将值从源块复制到目标块（卡片需兼容 MIFARE 标准）。调用此命令前，用户应当先成功进行认证并获得对源块和目标块的访问权限。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| Copy Value Block | FFh | D7h | 00h | 源块# | 02h | 见下表 |

命令数据

| 字节 0 | 字节 1 |
|---|---|
| 03h | 目标块# |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 6X XXh | 失败。 |

### 5.5.6. 访问符合 PCSC 的标签（ISO14443-4）

所有符合 ISO 14443-4 标准的卡片（PICC）都可以理解 ISO 7816-4 规定的 APDU。ACR1555U 读写器与符合 ISO 14443-4 标准的卡片进行通信时，只需要交互 ISO 7816-4 规定的 APDU 和响应。ACR1555U 会在内部处理 ISO 14443 第 1-4 部分协议。

另外 MIFARE Classic (1K/4K)、MIFARE Mini 和 MIFARE Ultralight 标签是通过 T=CL 模拟进行支持的。只要将 MIFARE 标签视作标准的 ISO 14443-4 标签即可。更多信息请参阅 **PICC 的 PCSC 私有 APDU（带专有扩展）**。

ISO 7816-4 规定的 APDU 报文结构

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 | Le |
|---|---|---|---|---|---|---|---|
| ISO 7816 第 4 部分规定的命令 | | | | | 命令数据域的长度 | | 期望返回的响应数据的长度 |

ISO 7816-4 规定的响应报文的结构（数据 + 2 字节）

| 响应 | 响应数据域 | | |
|---|---|---|---|
| 结果 | 响应数据 | SW1 | SW2 |

常见的 ISO 7816-4 命令的响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 63 00h | 操作失败。 |

典型的操作顺序是：

1. 出示标签，与 PICC 接口建立连接。
2. 读取/更新标签的存储内容。

要实现这些：

1. 与标签建立连接。

   标签的 ATR 为 3B 88 80 01 00 00 00 00 33 81 81 00 3Ah。

   其中，

   ATQB 应用数据 = 00 00 00 00，ATQB 协议信息 = 33 81 81。这是一个 ISO 14443-4 Type B 标签。

2. 发送 APDU，取随机数。

   << 00 84 00 00 08h

   >> 1A F7 F3 1B CD 2B A9 58h [90 00h]

   ***注：****对于 ISO 14443-4 Type A 标签来说，可以通过 APDU “FF CA 01 00 00h” 获取 ATS。*

**例如：**

// 从 ISO 14443-4 Type B PICC（ST19XR08E）中读取 8 字节

APDU = {80 B2 80 00 08h}

CLA = 80h
INS = B2h
P1 = 80h
P2 = 00h
Lc = 无
命令数据域 = 无
Le = 08h

应答：00 01 02 03 04 05 06 07h [$9000h]

### 5.5.7. 访问 MIFARE DESFire 标签（ISO 14443-3）

MIFARE® DESFire®支持 ISO7816-4 APDU 包模式和本地模式。一旦 MIFARE® DESFire® 标签被激活，命令发送至 MIFARE® DESFire® 标签的第一个 APDU 就会确定“指令模式”。如果第一个 APDU 采用“本地模式”，则其余的 APDU 指令都必须是“本地模式”。同样，如果第一个 APDU 采用“ISO 7816-4 APDU 包模式”，则其余的 APUDU 都必须是“ISO 7816-4 APDU 包模式”。

**例 1：MIFARE® DESFire® ISO 7816-4 APDU 包**

//从 ISO 14443-4 Type A PICC（MIFARE® DESFire®）中读取 8 个字节的随机数

APDU = {90 0A 00 00 01 00 00}

CLA = 90h; INS = 0Ah (MIFARE DESFire 指令); P1 = 00h; P2 = 00h

Lc = 01h; 数据输入 = 00h; Le = 00h (Le = 00h 为最大长度)

应答：7B 18 92 9D 9A 25 05 21 [$91AF]

# 状态码{91 AF}由 MIFARE® DESFire®标准定义，详情请参阅 MIFARE® DESFire®标准。

**例 2：MIFARE® DESFire® 分页链接（ISO 7816 包模式）**

// 在本例中，应用涉及到"分页链接"。

// 要获取 MIFARE® DESFire®卡的版本号。

步骤 1：发送 APDU{90 60 00 00 00}来获取第一个数据页。INS=60h

应答：04 01 01 00 02 18 05 91 AF [$91AF]

步骤 2：发送 APDU {90 AF 00 00 00}来获取第二个数据页。INS=AFh

应答：04 01 01 00 06 18 05 91 AF [$91AF]

步骤 3：发送 APDU {90 AF 00 00 00} 来获取最后一个数据页。INS=AFh

应答：04 52 5A 19 B2 1B 80 8E 36 54 4D 40 26 04 91 00 [$9100]

### 5.5.8. 访问 FeliCa 标签

访问 FeliCa 标签的命令不同于访问 PCSC 和 MIFARE 标签的命令。此命令符合 FeliCa 规范，加了一个命令头。

FeliCa 命令结构

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| FeliCa 命令 | FFh | 00h | 00h | 00h | 命令数据域的长度 | FeliCa 命令（开始于长度字节） |

FeliCa 的响应结构（数据 + 2 字节）

| 响应 | 响应数据域 |
|---|---|
| 结果 | 响应数据 |

**以读取内存块为例：**

1. 与 FeliCa 建立连接。

   ATR = 3B 8F 80 01 80 4F 0C A0 00 00 03 06 11 00 3B 00 00 00 00 42h

   其中，11 00 3Bh = FeliCa

2. 读取 FeliCa IDM。

   命令 = FF CA 00 00 00h

   响应 = [IDM (8 字节)] 90 00h

   例如：FeliCa IDM = 01 01 06 01 CB 09 57 03h

3. FeliCa 命令访问。

   例如：“读取”内存块。

   命令 = FF 00 00 00 10 10 06 01 01 06 01 CB 09 57 03 01 09 01 01 80 00h

   其中：

   Felica 命令 = 10 06 01 01 06 01 CB 09 57 03 01 09 01 01 80 00h

   IDM = 01 01 06 01 CB 09 57 03h

   响应 = 内存块数据

### 5.5.8. 访问 ISO15693 标签

本节介绍 ISO15693 协议中的选项命令。

#### 5.5.8.1. 读单块（Read Single Block）

此命令用于从 ISO15693 标签取回一个数据块。

命令：

| 命令 | CLA | INS | P1 | P2 | LC | 数据 | | Le |
|---|---|---|---|---|---|---|---|---|
| Read Single Block | FFh | FBh | 00h | 00h | 02h | 20h | 块号 | --/00h |

其中：

**块号** 1 个字节。
数据块编号。

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 64 XXh | 失败。XX 是标签返回的错误代码 |

**例如：**

//读 NXP ICODE SLI 卡中块 10 的数据

命令： = { FF FB 00 00 02 20 10 }

响应： = { XX XX XX XX 90 00 }

#### 5.5.8.2. 写单块（Write Single Block）

此命令用于向 ISO15693 标签写入一个数据块。

命令：

| 命令 | CLA | INS | P1 | P2 | LC | 数据 | | | Le |
|---|---|---|---|---|---|---|---|---|---|
| Write Single Block | FFh | FBh | 00h | 00h | N+2h | 21h | 块号 | 块数据 | --/00h |

其中：

**块号** 1 个字节。
数据块编号。

**块数据** N 个字节。

| | |
|---|---|
| | 待向数据块写入的数据 |
| **LC** | 1 个字节。<br>基于数据块长度 + 2 |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 64 XXh | 失败。XX 是标签返回的错误代码 |

**例如：**

//向 NXP ICODE SLI 卡的块 10 写入数据

命令： = { FF FB 00 00 06 21 10 11 12 13 14}

响应： = { 90 00 }

### 5.5.8.3. 读多块（Read Multiple Blocks）

此命令用于从 ISO15693 标签取回多个数据块。

命令：

| 命令 | CLA | INS | P1 | P2 | LC | 数据 | | | Le |
|---|---|---|---|---|---|---|---|---|---|
| Read Multiple Blocks | FFh | FBh | 00h | 00h | 03h | 23h | 第一个块号 | 块数 | --/00h |

其中：

| | |
|---|---|
| **第一个块号** | 1 个字节。<br>起始数据块的编号。 |
| **块数** | 1 个字节。<br>请求中的块数比标签将在响应中返回的块安全状态数量少一个。<br>块数 = 请求中的块数 - 1 |

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 64 XXh | 失败。XX 是标签返回的错误代码 |

**例如：**

//读多块安全状态，从 0x10 至 0x12。NXP ICODE SLI 卡的 0X03 个连续块。

命令： = { FF FB 00 00 03 23 10 02 }

响应： = { XX XX XX XX XX XX XX XX XX XX XX XX 90 00 }

### 5.5.8.4. 写多块（Write Multiple Blocks）

此命令用于向 ISO15693 标签写多个数据块。

命令：

| 命令 | CLA | INS | P1 | P2 | LC | 数据 | | | | Le |
|---|---|---|---|---|---|---|---|---|---|---|
| Write Multiple Blocks | FFh | FBh | 00h | 00h | N+3h | 24h | 第一个块号 | 块数 | 块数据 | --/00h |

其中：

**第一个块号** 1 个字节。

起始数据块的编号。

**块数** 1 个字节。

请求中的块数比标签将在响应中返回的块安全状态数量少一个。

**块数** = 请求中的块数 – 1

**块数据** N 个字节。

待向数据块写入的数据

**LC** 1 个字节。

基于块数据的长度 + 3

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 64 XXh | 失败。XX 是标签返回的错误代码 |

### 5.5.8.5. 锁数据块（Lock Block）

LOCK BLOCK 命令用于将请求的块永久锁定，并在响应中报告操作的成功状态。

命令：

| 命令 | CLA | INS | P1 | P2 | LC | 数据 | | Le |
|---|---|---|---|---|---|---|---|---|
| Lock Blocks | FFh | FBh | 00h | 00h | 02h | 22h | 块号 | --/00h |

其中：

**块号** 1 个字节。
数据块编号。

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 64 XXh | 失败。XX 是标签返回的错误代码 |

**例如：**

命令： = { FF FB 00 00 02 22 10 }

响应： = { 90 00 }

### 5.5.8.6. 获取系统信息（Get System Information）

GET SYSTEM INFORMATION 命令用于获取标签系统的信息。

命令：

| 命令 | CLA | INS | P1 | P2 | LC | 数据 | Le |
|---|---|---|---|---|---|---|---|
| Get System Information | FFh | FBh | 00h | 00h | 01h | 2Bh | --/00h |

Get System Information 的响应格式

| 响应 | 响应数据域 | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 结果 | 信息标志 | UID | DSFID | AFI | 存储器大小 | IC 编号 | SW1 | SW2 |

其中：

信息标志 - 1 个字节

| 位 | 值 | 说明 |
|---|---|---|
| Bit 0 | 0 | 不存在 DSFID |
| | 1 | 存在 DSFID |
| Bit 1 | 0 | 不存在 AFI |
| | 1 | 存在 AFI |
| Bit 2 | 0 | 不存在存储器大小 |
| | 1 | 存在存储器大小 |
| Bit 3 | 0 | 不存在 IC 编号 |
| | 1 | 存在 IC 编号 |
| Bit 4 ~7 | 0 | RFU |

UID - 8 个字节

DSFID - 1 个字节

AFI - 1 个字节

存储器大小 - 2 个字节

| 字节 | 说明 |
|---|---|
| 0 | 块数 - 1<br>（实际的块数 = 块数 + 1 |
| 1 | 块大小(以字节为单位) - 1<br>（实际的块大小 = 块大小(以字节为单位) + 1） |

IC 编号 - 1 个字节

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 64 XXh | 失败。XX 是标签返回的错误代码 |

**例如：**

命令： = { FF FB 00 00 01 2B }

响应： = { XX XX XX XX XX XX XX XX XX XX XX XX XX XX 90 00}

### 5.5.8.7. 获取多块安全状态（Get Multiple Blocks Security Status）

GET MULTIPLE BLOCKS SECURITY STATUS 命令用于获取块的安全状态。

命令：

| 命令 | CLA | INS | P1 | P2 | LC | 数据 | | | Le |
|---|---|---|---|---|---|---|---|---|---|
| Get Multiple Blocks Security Status | FFh | FBh | 00h | 00h | 03h | 2Ch | 第一个块号 | 块数 | --/00h |

其中：

**第一个块号** 1 个字节。

起始数据块的编号。

**块数** 1 个字节。

将读取数据块安全状态的数量。请求中的块数比标签将在响应中返回的块安全状态数量少一个。

**块数** = 请求中的块数 - 1

Get System Information 的响应格式

| 响应 | 响应数据域 | | |
|---|---|---|---|
| 结果 | 块安全状态 | SW1 | SW2 |

其中：

**块安全状态** 每块一个字节。

00h：未锁定

01h：锁定

响应状态码

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 64 XXh | 失败。XX 是标签返回的错误代码 |

**例如：**

//读多块安全状态，从 0x10 至 0x12。0X03 个连续块。

命令： = { FF FB 00 00 03 2C 10 02 }

响应： = { XX XX XX 90 00 }

### 5.5.9. 支持的 PICC ATR

默认支持下列 PICC 类型/技术。在读写器上刷卡后，PC_to_RDR_IccPowerOn 命令会将下列 ATR 返回给 CCID 主机。

| 卡片类型/技术 | ATR |
|---|---|
| MIFARE Std 1k6 | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A |
| MIFARE Std 4k6 | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 02 00 00 00 00 69 |
| MIFARE UltraLight6 | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 03 00 00 00 00 68 |
| MIFARE Plus SL1 2k6 | 默认：与 MIFARE Std 1k 相同<br>备用：3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 36 00 00 00 00 5D |
| MIFARE Plus SL1 4k6 | 默认：与 MIFARE Std 4k 相同<br>备用：3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 37 00 00 00 00 5C |
| MIFARE Plus SL2 2k | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 38 00 00 00 00 53 |
| MIFARE Plus SL2 4k | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 39 00 00 00 00 52 |
| MIFARE UltraLight C6 | 默认：3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 3A 00 00 00 00 51<br>备用：与 MIFARE UltraLight 相同 |
| SmartMX，模拟 MIFARE Std 1k6 | 默认：与 MIFARE Std 1k 相同<br>备用：与 ISO14443-4, Type A 相同 |
| SmartMX，模拟 MIFARE Std 4k[6] | 默认：与 MIFARE Std 4k 相同<br>备用：与 ISO14443-4, Type A 相同 |
| ISO14443-4, Type A | 3B 8n 80 01 T1 ..Tn Tck<br><br>n = ATS 中历史字节的数量<br>T1 ..Tn = ATS 中的历史字节<br>Tck = 异或 8n 80 01 T1 ..Tn |
| ISO14443-4, Type B | 3B 88 80 01 T1 ..T8 Tck<br><br>T1 ..T4 = ATQB 中的应用数据<br>T5 ..T7 = ATQB 中的协议信息<br>T8 = ATA 中的 MBLI<br>Tck = 异或 88 80 01 T1 ..T8 |
| FeliCa | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 11 00 3B 00 00 00 00 42 |
| ISO15693-3 Generic | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 00 00 00 00 00 63 |
| Infineon My-D Vicinity (SRF55Vxxx) | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 0E 00 00 00 00 6D |
| ST LRI | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 13 00 00 00 00 70 |
| NXP I-Code SLI | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 14 00 00 00 00 77 |

*使用 ACS 定义的安卓库*

[2] *使用 ACS 定义的 iOS 或 iPadOS 库*

备用 ATR 定义的取消。

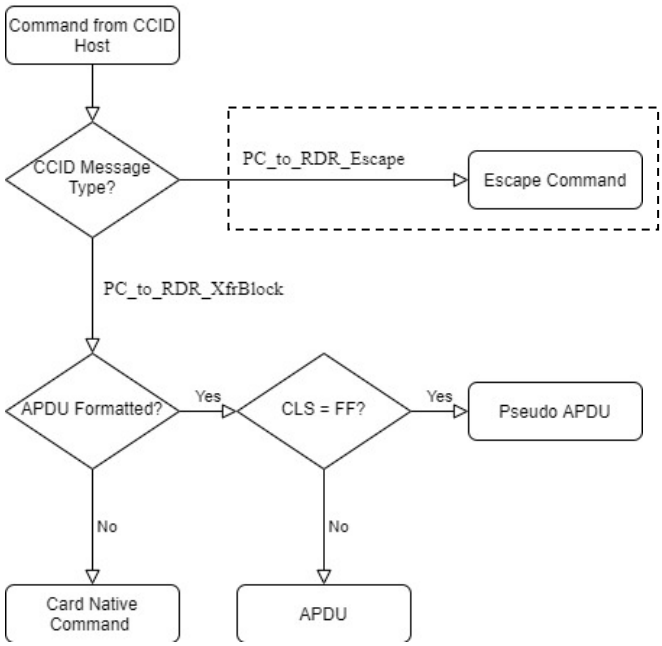
| 卡片类型/技术 | ATR |
|---|---|
| NXP I-Code SLIX/SLIX2 | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 35 00 00 00 00 56 |
| PicoPass 2K | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 17 00 00 00 00 79 |
| PicoPass 2KS | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 18 00 00 00 00 76 |
| PicoPass 16K | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 19 00 00 00 00 77 |
| PicoPass 16KS | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1A 00 00 00 00 74 |
| PicoPass 16K (8 x 2) | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1B 00 00 00 00 75 |
| PicoPass 16KS (8 x 2) | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1C 00 00 00 00 72 |
| PicoPass 32KS (16 + 16) | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1D 00 00 00 00 73 |
| PicoPass 32KS (16 + 8x2) | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1E 00 00 00 00 70 |
| PicoPass 32KS (8x2 + 16) | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 1F 00 00 00 00 71 |
| PicoPass 32KS (8x2 + 8x2) | ISO14443B: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 20 00 00 00 00 4E |

为了缩短常见应用的响应时间，默认禁用对下列 PICC 类型/技术的支持。用户可以通过直接命令“Set operation Mode”来启用对各种类型/技术的支持。如果相应类型/技术已启用并且在读写器上刷卡，PC_to_RDR_IccPowerOn 命令会将下列 ATR 返回给 CCID 主机。

| 卡片类型/技术 | ATR |
|---|---|
| SRI (SRIX4K/SRT512) | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 06 00 07 00 00 00 00 69 |
| Topaz | 3B 8F 80 01 80 4F 0C A0 00 00 03 06 02 00 30 00 00 00 00 5A |
| PicoPass 2K | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 17 00 00 00 00 75 |
| PicoPass 2KS | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 18 00 00 00 00 7A |
| PicoPass 16K | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 19 00 00 00 00 7B |
| PicoPass 16KS | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1A 00 00 00 00 78 |
| PicoPass 16K (8 x 2) | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1B 00 00 00 00 79 |
| PicoPass 16KS (8 x 2) | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1C 00 00 00 00 7E |
| PicoPass 32KS (16 + 16) | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1D 00 00 00 00 7F |
| PicoPass 32KS (16 + 8x2) | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1E 00 00 00 00 7C |
| PicoPass 32KS (8x2 + 16) | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 1F 00 00 00 00 7D |
| PicoPass 32KS (8x2 + 8x2) | ISO15693: 3B 8F 80 01 80 4F 0C A0 00 00 03 06 0A 00 20 00 00 00 00 42 |
| Innovatron | 3B 88 80 01 80 4F 05 F0 49 4E 4E 4F 35 |
| CTS | 3B 87 80 01 80 4F 04 F0 43 54 53 79 |

# 6.0. 直接命令

直接（Escape）命令通过 PC_to_RDR_Escape（对应于 PCSC API 中 SCARD_CTL_CODE(3500)的 SCardControl()）来发送。处理完命令后，读写器通过 RDR_to_PC_Escape 消息返回响应。




下列命令用于配置 PCD/NFC，以及访问读写器的特殊功能。CCID 主机可以使用 CCID 报文 PC_to_RDR_Escape（对应于 PCSC API 中 SCARD_CTL_CODE(3500)的 SCardControl()）向读卡器发送这些命令。收到Escape命令后，读写器会解读命令并执行各项操作，然后生成响应并发送回CCID主机。

***注：***
*这些命令需通过正确的接口发送。例如E0 00 00 25 01 00 (6.1.1 节)应当通过PICC接口发送(6.0 节)。*

## 6.1. PICC 的 Escape 命令

### 6.1.1. RF 控制（RF Control）[E0 00 00 25 01 …]

此命令用于设置 RF 控制。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| RF Control | E0h | 00h | 00h | 25h | 01h | RF 状态 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | RF 状态 |

RF 状态：1 个字节

| RF 状态 | 说明 |
|---|---|
| 00h | RF 关闭 |
| 01h | RF 开启，轮询 |
| 02h | RF 开启，不轮询 |

*默认设置 – 01h (RF 开启，轮询)*

### 6.1.2. 获取 PCD/PICC 状态（Get PCD/PICC Status）[E0 00 00 25 00]

此命令用于获取 PCD/PICC 的状态。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get PCD/PICC Status | E0h | 00h | 00h | 25h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | Get PCD/PICC Status |

PCD/PICC 状态：1 个字节

| RF 状态 | 说明 |
|---|---|
| 00h | RF 关闭 |
| 01h | 无 PICC |
| 02h | PICC 已就绪 |
| 03h | PICC 已选定/已激活 |
| FFh | 错误 |

### 6.1.3. 获取轮询/ATR 选项（Get Polling/ATR Option）[E0 00 00 23 00]

此命令用于设置/获取轮询选项，无需其它命令即可保存设置。此命令仅用于最初的读写器配置。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Polling/ATR Option | E0h | 00h | 00h | 23h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 03h | 01h | PICC 轮询/ATR 选项 |

### 6.1.4. 设置轮询/ATR 选项（Set Polling/ATR Option）[E0 00 00 23 01 …]

此命令用于设置轮询选项。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Set Polling/ATR Option | E0h | 00h | 00h | 23h | 01h | PICC 轮询/ATR 选项 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | PICC 轮询/ATR 选项 |

PICC 轮询/ATR 选项 - 1 个字节

| 操作参数 | 参数 | 说明 | 选项 |
|---|---|---|---|
| Bit 0 | 启用轮询 | PICC轮询要检测的标签类型。 | 1 = 检测<br>0 = 跳过 |
| Bit 1 | 启用 RF 关闭间隔 | | |
| Bit 2 | RFU | | |
| Bit 3 | 启用第 3 部分卡片 ATR 对额外 MIFARE 类型的识别 | PICC 轮询要检测的标签类型。 | 1 = 检测<br>0 = 跳过 |
| Bit 4 ~ 5 | RF 关闭间隔 | | 见下表 |
| Bit 6 | RFU | | |
| Bit 7 | 启用第 4 部分 ATR 适用于 SmartMX/JCOS 卡模拟 MIFARE | PICC 轮询要检测的标签类型。 | 1 = 检测<br>0 = 跳过 |

RF 关闭间隔 – 2 Bit **情形 1:** 禁用 RF 关闭（Bit 1=0）

| 操作参数 | | USB 运行(D0) | USB 挂起(D2) |
|---|---|---|---|
| Bit 5 | Bit 4 | | |
| 0 | 0 | 无 RF 关闭 | 250 ms |
| 0 | 1 | | 500 ms |
| 1 | 0 | | 1000 ms |
| 1 | 1 | | 2500 ms |

**情形 2：** 启用 RF 关闭 (Bit 1 = 1)

| 操作参数 | | USB 运行(D0) | USB 挂起(D2) |
|---|---|---|---|
| Bit 5 | Bit 4 | | |
| 0 | 0 | 250 ms | 500 ms |
| 0 | 1 | 500 ms | 1000 ms |
| 1 | 0 | 1000 ms | 2500 ms |
| 1 | 1 | 2500 ms | 2500 ms |

*默认设置 – 8Bh (启用轮询，启用 RF 关闭, 启用第 3 部分卡片在 ATR 中对额外 MIFARE 类型的识别, Rf 关闭间隔[00], 启用第 4 部分 ATR 适用于 SmartMX/JCOS 卡模拟 MIFARE)*

### 6.1.5. 获取 PICC 轮询类型（Get PICC Polling Type）[E0 00 01 20 00]

此命令用于获取允许的技术/轮询类型，无需其它命令即可保存设置。仅用于最初的读写器配置。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get PICC Polling Type | E0h | 00h | 01h | 20h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | PICC 轮询类型 |

### 6.1.6. 设置 PICC 轮询类型（Set PICC Polling Type）[E0 00 01 20 02 …]

此命令用于设置 PICC 轮询类型。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 | |
|---|---|---|---|---|---|---|---|
| | | | | | | 字节 1 | 字节 0 |
| Set PICC Polling Type | E0h | 00h | 01h | 20h | 02h | PICC 轮询类型 | |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | |
|---|---|---|---|---|---|---|---|
| | | | | | | 字节 1 | 字节 0 |
| 结果 | E1h | 00h | 00h | 00h | 02h | PICC 轮询类型 | |

PICC 轮询类型 - 2 个字节，LSB 在前，位掩码如下

| 字节 | 操作参数 | 参数 | 说明 | 选项 |
|---|---|---|---|---|
| **字节 1** | Bit 0 | ISO 14443A | PICC轮询要检测的标签类型。RFU位应设置为 0。 | 1 = 检测<br>0 = 跳过 |
| | Bit 1 | ISO 14443B | | |
| | Bit 2 | FeliCa | | |
| | Bit 3 | RFU | | |
| | Bit 4 | Topaz | | |
| | Bit 5 | Innovatron | | |
| | Bit 6 | SRI/SRIX | | |
| | Bit 7 | RFU | | |
| **字节 0** | Bit 0 | Picopass (ISO14443B) | | |
| | Bit 1 | Picopass (ISO15693) | | |
| | Bit 2 | ISO15693 | | |
| | Bit 3 | CTS | | |
| | Bit 4-7 | RFU | | |

*默认设置 –* ***字节 1: 07h (ISO14443 A 类, ISO14443 B 类, FeliCa)***

***字节 0: 05h (Picopass (ISO14443B), ISO15693)***

例子:

**命令** : E0 00 01 20 02 07 05

**响应**: E1 00 00 00 02 07 05

轮询类型: **字节** 1 = 07h = 0000 0111b = ISO14443A, ISO14443B, FeliCa

**字节** 0 = 05h = 0000 0101b = Picopass (ISO14443B), ISO15693

### 6.1.7. 获取自动 PPS（Get Auto PPS）[E0 00 00 24 00]

每次识别出 PICC，读写器都会尝试按照最大连接速度的定义更改 PCD 和 PICC 间的通信速率。若卡片不支持建议的连接速度，读写器会尝试以较慢的速度与卡片建立连接。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Auto PPS | E0h | 00h | 00h | 24h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | |
|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | 最高速率 | 当前速率 |

### 6.1.8. 设置自动 PPS（Set Auto PPS）[E0 00 00 24 01 ...]

此命令用于设置自动 PPS。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Set Auto PPS | E0h | 00h | 00h | 24h | 01h | 最高速率 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | |
|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | 最高速率 | 当前速率 |

PPS的速率

| 速率 | 说明 |
|---|---|
| 00h | 106 kbps; 等同于没有设置自动PPS |
| 01h | 212 kbps |
| 02h | 424 kbps |
| 03h | 848 kbps |

*默认设置 – 02h (424 kbps)*

***注:***

*1.通常来说，应用程序应了解正在使用的PICC的最大连接速率，周围环境也会对最大可达速率有所影响。读写器只是以建议的通信速率来与PICC进行对话。如果PICC或周围环境不能满足建议的通信速率的要求，PICC将变得不能访问。*

*2.如果较高的速率设置影响到读写器运行，请切换回较低的速率设置。*

### 6.1.9. 读取 PICC 类型（Read PICC Type）[E0 00 00 35 00]

此命令用于读取 PICC 类型。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get PICC Type | E0h | 00h | 00h | 35h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | |
|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | 类型 | 状态 |

类型: 1 个字节

| 类型 | 说明 |
|---|---|
| CCh | 无 PICC |
| 04h | Topaz |
| 10h | MIFARE |
| 11h | FeliCa |
| 20h | Type A, Part 4 |
| 23h | Type B, Part 4 |
| 25h | Innovatron |
| 28h | SRIX |
| 30h | PicoPass |
| FFh | 其它 |

状态：1 个字节

| 状态 | 说明 |
|---|---|
| 00h | RF 关闭 |
| 01h | 无 PICC |
| 02h | PICC 已就绪 |
| 03h | PICC 已选定/已激活 |
| FFh | 错误 |

### 6.1.10. PICC – HID 键盘的 Escape 命令

#### 6.1.10.1. 获取输出格式（Get Output Format）[E0 00 00 90 00]

此命令用于获取输出格式。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Output Format | E0h | 00h | 00h | 90h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | |
|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | 输出格式 | 输出顺序 |

### 6.1.10.2. 设置输出格式（Set Output Format）[E0 00 00 90 02 …]

此命令用于设置输出格式。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 | |
|---|---|---|---|---|---|---|---|
| Set Output Format | E0h | 00h | 00h | 90h | 02h | 输出格式 | 输出顺序 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | |
|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | 输出格式 | 输出顺序 |

输出格式：1 个字节

| 操作参数 | 参数 | 说明 | 选项 |
|---|---|---|---|
| Bit 7 ~ 4 | 字母大小写 | PICC轮询要检测的标签类型。 | 1 = 检测<br>0 = 跳过 |
| Bit 3 ~ 0 | 显示模式 | | |

输出顺序：1 个字节

| 状态 | 说明 |
|---|---|
| 00h | 默认顺序(UID字节0, UID字节1 … UID字节N)<br>例如: aa cc bb dd (原始/实际UID顺序) |
| 01h | 逆序(UID字节N, UID字节N-1 … UID字节0)<br>例如: dd bb cc aa (UID顺序反转) |

字母大小写：高 4 位(Bit 7 到 Bit 4)

| 状态(从 bit 7 到 bit 4) | 说明(无需关注 x bit) |
|---|---|
| 1xxx | 保留 |
| 00x0 | 小写字母 |
| 00x1 | 大写字母 |
| 000x | 仅支持4字节UID |
| 001x | 支持4、7、8、10字节UID |

显示模式：低 4 位(Bit 3 到 Bit 0)

| 状态(从 bit 7 到 bit 4) | 说明(无需关注 x bit) |
|---|---|
| 0h | Hex |
| 1h | Dec (逐字节) |
| 2h | Dec |
| 3h | 6H-6H |
| 4h | 8H-8H |
| 5h | 10H-10H |
| 6h | 14H-14H |
| 7h | 20H-20H |
| 8h | 6H-8D |
| 9h | 6H-10D |
| Ah | 8H-10D |
| Bh | 10H-14D |
| Ch | 2H4H-8D |
| Dh | 14H-17D |

### 6.1.10.3. 获取 UID 起始、中间和结束位字符（Get Character at Start, Between, at End UID）[E0 00 00 91 00]

此命令用于获取 UID 起始、中间和结束位置的字符。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Character of UID | E0h | 00h | 00h | 91h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | | |
|---|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 03h | 中间 | 结束 | 开始 |

### 6.1.10.4. 设置 UID 起始、中间和结束位字符（Set Character at Start, Between, at End UID）[E0 00 00 91 03 …]

此命令用于设置 UID 起始、中间和结束位置的字符。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 | | |
|---|---|---|---|---|---|---|---|---|
| Set Character of UID | E0h | 00h | 00h | 91h | 03h | 中间 | 结束 | 开始 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | | |
|---|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 03h | 中间 | 结束 | 开始 |

中间: 1 个字节（每个 UID 之间的字符）

| 状态 | 说明 |
|---|---|
| FFh | 中间没有字符 |
| 其它 | 请参阅通用串行总线（USB）HID使用表 |

结束: 1 个字节（输出末尾的字符）

| 状态 | 说明 |
|---|---|
| FFh | 中间没有字符 |
| 其它 | 请参阅通用串行总线（USB）HID使用表 |

起始: 1 个字节（输出开始的字符）

| 状态 | 说明 |
|---|---|
| FFh | 中间没有字符 |
| 其它 | 请参阅通用串行总线（USB）HID使用表 |

*注：*

1. *AZERTY键盘布局仅支持“;” “,” “,” “,” “-“作为中间的字符，不支持零(0)和退格键。*

### 6.1.10.5. 获取键盘布局语言（Get Keyboard Layout Language）[E0 00 00 92 00]

此命令用于获取键盘布局语言。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Keyboard Layout Language | E0h | 00h | 00h | 92h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 键盘布局语言 |

### 6.1.10.6. 设置键盘布局语言（Set Keyboard Layout Language）[E0 00 00 92 01 …]

此命令用于设置键盘布局语言。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Set Keyboard Layout Language | E0h | 00h | 00h | 92h | 01h | 键盘布局语言 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 键盘布局语言 |

键盘布局语言：1 个字节

| 状态 | 说明 |
|---|---|
| 00h | 英语 |
| 01h | 法语 |
| 02h | 保留 |
| 03h | 立陶宛语 |

### 6.1.10.7. 获取主机接口（Get Host Interface） [E0 00 00 93 00]

此命令用于获取主机接口。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Host Interface | E0h | 00h | 00h | 93h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 主机接口 |

### 6.1.10.8. 设置主机接口（Set Host Interface）[E0 00 00 93 01 ...]

此命令用于设置主机接口。

**注意：阅读器将自动关机以应用设置。**

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Set Host Interface | E0h | 00h | 00h | 93h | 01h | 主机接口 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 主机接口 |

主机接口：1 个字节

| 状态 | 说明 |
|---|---|
| 00h | 仅HID键盘 |
| 01h | 仅CCID读写器 |

*默认设置 – 01h*

## 6.1.11. PICC – 卡模拟的 Escape 命令

### 6.1.11.1. 进入卡模拟模式（Enter Card Emulation Mode）[E0 00 00 40 03 ...]

此命令用于设置读写器进入卡模拟模式，以便模拟 MIFARE Ultralight 卡或 FeliCa 卡。

***注：****模拟 MIFARE Ultralight 卡时不支持 Lock 字节。UID 可由用户编写。*

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 | | |
|---|---|---|---|---|---|---|---|---|
| Enter Card Emulation Mode | E0h | 00h | 00h：暂时<br>01h：保存 | 40h | 03h | NFC 模式 | 00h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 03h | NFC 模式 |

NFC 设备模式：3 个字节

| 状态 | 说明 |
|---|---|
| 02h | NFC论坛类型2标签模式 |
| 03h | FeliCa |
| 其它 | 卡片读/写模式 |

***注：****在切换到不同的卡模拟模式之前，请先进入卡片读/写模式。卡模拟模式初始完成后将显示响应。*

| 字节号 | 0 | 1 | 2 | 3 | USB 访问字节地址 |
|---|---|---|---|---|---|
| 序列号 | SN0 | SN1 | SN2 | SN3 | Nil |
| 保留 | 保留 | 保留 | 保留 | 保留 | Nil |
| 内部/锁 | 保留 | 内部 | Lock0 | Lock1 | Nil |
| 数据读/写 | Data0 | Data1 | Data2 | Data3 | 0-3 |
| 数据读/写 | Data4 | Data5 | Data6 | Data7 | 4-7 |
| 数据读/写 | Data8 | Data9 | Data10 | Data11 | 8-11 |
| 数据读/写 | Data12 | Data13 | Data14 | Data15 | 12-15 |
| 数据读/写 | Data16 | Data17 | Data18 | Data19 | 16-19 |
| 数据读/写 | Data20 | Data21 | Data22 | Data23 | 20-23 |
| 数据读/写 | Data24 | Data25 | Data26 | Data27 | 24-27 |
| 数据读/写 | Data28 | Data29 | Data30 | Data31 | 28-31 |
| 数据读/写 | Data32 | Data33 | Data34 | Data35 | 32-35 |
| 数据读/写 | Data36 | Data37 | Data38 | Data39 | 36-39 |
| 数据读/写 | Data40 | Data41 | Data42 | Data43 | 40-43 |
| 数据读/写 | Data44 | Data45 | Data46 | Data47 | 44-47 |
| 数据读/写 | Data48 | Data49 | Data50 | Data51 | 48-51 |
| 数据读/写 | Data52 | Data53 | Data54 | Data55 | 52-55 |
| 数据读/写 | | | … | | … |
| 数据读/写 | Data1984 | Data1985 | Data1986 | Data1987 | 1984-1987 |

可访问区 (1988 字节)

**表16：**NFC 论坛类型 2 标签的内存结构（2000 字节）

| 内存 | 1 数据块（16 字节） | USB 访问字节地址 |
|---|---|---|
| 数据读/写 | Block 0 | 0-15 |
| 数据读/写 | Block 1 | 16-31 |
| 数据读/写 | Block 2 | 32-47 |
| 数据读/写 | Block 3 | 48-63 |
| 数据读/写 | Block 4 | 64-79 |
| 数据读/写 | Block 5 | 80-95 |
| 数据读/写 | Block 6 | 96-111 |
| 数据读/写 | Block 7 | 112-127 |
| 数据读/写 | Block 8 | 128-143 |
| 数据读/写 | Block 9 | 144-159 |

**表17**：FeliCa 卡的内存结构（160 字节）

其中：

**默认**：　块 0 数据：{10h, 01h, 01h, 00h, 09h, 00h, 00h, 00h, 00h, 00h, 01h, 00h, 00h, 00h, 00h, 1Ch}

**默认块 0 数据**　NFC 类型 3 标签属性信息块

***注：***

1. *FeliCa 卡模拟支持不带加密读/写。*
2. *FeliCa 卡片识别号码（IDm）可由用户定义，而生厂商编码固定为(03 88)。*

### 6.1.11.2. 读取卡模拟数据（Read Card Emulation Data）（NFC 论坛类型 2 标签） [E0 00 00 60 04 …]

此命令用于读取所模拟卡片的内容。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 | | | |
|---|---|---|---|---|---|---|---|---|---|
| Read Card Emulation Data | E0h | 00h | 00h | 60h | 04h | 00h | NFC 模式 | 起始偏移量 | 长度 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 长度 | 数据 |

起始偏移量：　1 字节 – **表 16** 中从 Data0 开始的地址

长度：　1 字节 – 字节数量

### 6.1.11.3. 写入卡模拟数据（Write Card Emulation Data）（NFC 论坛类型 2 标签）[E0 00 00 60 …]

此命令用于写入模拟的卡片内容。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Write Card Emulation Data | E0h | 00h | 00h | 60h | 长度 + 04h | 01h | NFC 模式 | 起始偏移量 | 长度 | 数据 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | | |
|---|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 03h | 长度 | 90h | 00h |

NFC 设备模式：1 个字节

| 状态 | 说明 |
|---|---|
| 02h | NFC论坛类型2标签模式 |
| 03h | FeliCa |
| 其它 | 卡片读/写模式 |

起始偏移量： 1 字节 – **表 16** 中从 Data0 开始的地址

长度： 1 字节 – 字节数量

### 6.1.11.4. 读取卡模拟数据（Read Card Emulation Data）（NFC 论坛类型 2 标签）（扩展）

此命令用于读取所模拟卡片的内容。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | | 命令数据域 | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Read Card Emulation Data | E0h | 00h | 01h | 60h | 05h | 00h | NFC 模式 | 起始偏移量 Bit[15:8] | 起始偏移量 Bit[7:0] | 长度 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 长度 | 数据 |

起始偏移量： 2 字节 – **表 16** 中从 SN0 起的开始读取地址

长度： 1 字节 – 待读取的字节数

### 6.1.11.5. 写入卡模拟数据（Write Card Emulation Data）（NFC 论坛类型 2 标签）（扩展）

此命令用于写入模拟的卡片内容。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | | 命令数据域 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Write Card Emulation Data | E0h | 00h | 01h | 60h | 长度 + 05h | 01h | NFC 模式 | 起始偏移量 Bit[15:8] | 起始偏移量 Bit[7:0] | 长度 | 数据 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | | |
|---|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 03h | 长度 | 90h | 00h |

NFC 设备模式：1 个字节

| 状态 | 说明 |
|---|---|
| 02h | NFC论坛类型2标签模式 |
| 其它 | 卡片读/写模式 |

起始偏移量： 2 字节 – **表 16** 中从 SN0 起的开始写入地址

长度： 1 个字节 – 要写入的字节数

### 6.1.11.6. 设置 NFC 论坛类型 2 标签卡模拟 ID（Set Card Emulation of NFC Forum Type 2 Tag ID）[E0 00 00 61 03 …]

此命令用于设置所模拟的 MIFARE Ultralight 卡的 UID。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| Set Card Emulation Lock Data | E0h | 00h | 00h | 61h | 03h | 3 字节 UID |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 | |
|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | 90h | 00h |

### 6.1.11.7. 设置 NFC 卡模拟锁定数据（Set Card Emulation Lock Data in NFC）[E0 00 00 65 01 …]

此命令用于设置 NFC 通信过程中卡片模拟数据的锁定。数据锁定后，不能再通过 NFC 进行重写。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| Set Card Emulation Lock Data | E0h | 00h | 00h | 65h | 01h | 锁 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 锁 |

锁：1 个字节 – 保护数据不能通过 NFC 通信进行重写

| 操作参数 | 参数 | 说明 | 选项 |
|---|---|---|---|
| Bit 7 ~ 2 | 保留 | 保留 | |
| Bit 1 | 启用 FeliCa 锁 | 数据不能通过 NFC 通信进行修改。数据仍然可以使用 USB 直接命令进行修改。 | 0：禁用锁<br>1：启用锁 |
| Bit 0 | 启用 NFC 论坛类型 2 标签 | | |

### 6.1.11.8. 设置卡模拟时 FeliCa 的 IDm（Set Card Emulation FeliCa IDm）[E0 00 00 64 06 …]

此命令用于在所模拟的 FeliCa 卡片上设置 6 字节 FeliCa 卡标识号。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 命令数据域 |
|---|---|---|---|---|---|---|
| Set Card Emulation FeliCa IDm | E0h | 00h | 00h | 64h | 06h | IDm |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 06h | IDm |

其中：

**IDm** 6 字节

### 6.1.11.9. 获取卡模拟状态（Get Card Emulation Status）[E0 00 00 69 00]

此命令用于获取 NFC 通信中卡片模拟数据的状态。

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Get Card Emulation Status | E0h | 00h | 00h | 69h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 状态 |

状态：1 个字节

| 操作参数 | 模式 | 说明 |
|---|---|---|
| Bit 7 ~ 6 | 保留 | 保留 |
| Bit 5 | 模拟卡已激活 | 1 = 已激活 |
| Bit 4 | 模拟卡已移出 | 1 = 卡片已移出 |
| Bit 3 | 模拟卡已全部读取 | 1 = 所有数据均已读取 |
| Bit 2 | 模拟卡已读取 | 1 = 数据已读取 |
| Bit 1 | 模拟卡已写入 | 1 = 数据已写入 |
| Bit 0 | 模拟卡已被检测到 | 1 = 卡片检测中 |

### 6.1.11.10. 模拟 NFC 论坛类型 2 标签模式的示例命令集

此命令集以 ACR1555U 模拟 NFC 论坛类型 2 标签模式，触发 ACS 网站 https://www.acs.com.hk。步骤如下：

1. 通过下面的命令进入卡模拟模式：
    - 发送进入卡模拟模式（Enter Card Emulation Mode）

      **E0 00 00 40 03 02 00 00**
2. 通过下面的命令写 NDEF 数据：
    - 发送写入卡模拟数据（Write Card Emulation Data）（NFC 论坛类型 2 标签）

      **E0 00 00 60 1A 01 02 00 16 E1 10 F4 00 03 0F D1 01 0B 55 02 61 63 73 2E 63 6F 6D 2E 68 6B FE**

该命令集将触发一个示例长 URL 网址

https://www.example.com/this/is/a/very/long/url/that/keeps/going/on/and/on/with/even/more/segments/added/to/make/sure/it/exceeds/the/typical/length/limit/of/260/bytes/which/is/surprisingly/easy/to/do/if/you/keep/adding/more/and/more/segments/like/this/one/and/even/more

使用 ACR1555U 模拟 NFC 论坛类型 2 标签模式。具体步骤如下：

1. 通过下面的命令进入卡模拟模式：
    - 发送进入卡模拟模式（Enter Card Emulation Mode）命令

      **E0 00 00 40 03 02 00 00**
2. 通过下面的命令写 NDEF 数据：
    - 发送写入卡模拟数据（Write Card Emulation Data）（NFC 论坛类型 2 标签）命令。由于 NDEF 报文长度超过 256 字节，因此需要分成两部分发送给 NFC 论坛类型 2 标签。

      **E0 00 00 60 AC 01 02 00 A8 E1 10 F4 00 03 FF 01 09 C1 01 00 00 01 02 55 02 65 78 61 6D 70 6C 65 2E 63 6F 6D 2F 74 68 69 73 2F 69 73 2F 61 2F 76 65 72 79 2F 6C 6F 6E 67 2F 75 72 6C 2F 74 68 61 74 2F 6B 65 65 70 73 2F 67 6F 69 6E 67 2F 6F 6E 2F 61 6E 64 2F 6F 6E 2F 77 69 74 68 2F 65 76 65 6E 2F 6D 6F 72 65 2F 73 65 67 6D 65 6E 74 73 2F 61 64 64 65 64 2F 74 6F 2F 6D 61 6B 65 2F 73 75 72 65 2F 69 74 2F 65 78 63 65 65 64 73 2F 74 68 65 2F 74 79 70 69 63 61 6C 2F 6C 65 6E 67 74 68 2F 6C 69 6D 69 74 2F 6F 66 2F 32 36 30 2F 62 79 74**

      **E0 00 00 60 6E 01 02 A8 6A 65 73 2F 77 68 69 63 68 2F 69 73 2F 73 75 72 70 72 69 73 69 6E 67 6C 79 2F 65 61 73 79 2F 74 6F 2F 64 6F 2F 69 66 2F 79 6F 75 2F 6B 65 65 70 2F 61 64 64 69 6E 67 2F 6D 6F 72 65 2F 61 6E 64 2F 6D 6F 72 65 2F 73 65 67 6D 65 6E 74 73 2F 6C 69 6B 65 2F 74 68 69 73 2F 6F 6E 65 2F 61 6E 64 2F 65 76 65 6E 2F 6D 6F 72 65 FE**

***注：***
*如需了解更多关于NDEF（NFC 数据交互格式）的信息和规定，建议参考NDEF 规范；该规范就NDEF 记录的结构和使用提供了全面指引和详细信息，且这些 NDEF 记录常用于 NFC 数据交互。NDEF 规范有助于深入了解如何在 ACR1555U 设备环境中解读和利用 NDEF 命令和数据。*

## 6.2. ICC 的 Escape 命令

### 6.2.1. 获取卡片电源配置（Get Card Power Configuration）[E0 00 00 0B 00]

此命令用于获取 ICC 卡的电源配置，仅用于最初的读写器配置。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Card Power Configuration | E0h | 00h | 00h | 0Bh | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 卡片电源配置 |

### 6.2.2. 设置卡片电源配置（Set Card Power Configuration）[E0 00 00 0B 01 …]

此命令用于设置和保存 ICC 卡的电源配置。仅用于最初的读写器配置。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Set Card Power Configuration | E0h | 00h | 00h | 0Bh | 01h | 配置 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 卡片电源配置 |

卡片电源配置（1 个字节）

| 卡片电源配置 | 说明 |
|---|---|
| 00h | 自动检测，1.8V -> 3V -> 5V |
| 01h | 仅 5V |
| 02h | 仅 3V |
| 03h | 仅 1.8V |
| 04h | 自动检测，5V -> 3V -> 1.8V |
| 其它 | RFU |

*默认设置 – 04h (自动检测，5V -> 3V -> 1.8V)*

## 6.3. 外设控制及其他的 Escape 命令

### 6.3.1. 获取固件版本（Get Firmware Version）[E0 00 00 18 00]

此命令用于获取读写器的固件信息。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Firmware Version | E0h | 00h | 00h | 18h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 固件版本的长度 | 固件版本 |

例如：
命令： E0 00 00 18 00
响应状态码： E1 00 00 00 12 41 43 52 31 35 35 35 20 46 57 20 31 2E 30 30 2E 30 30
十六进制固件版本：41 43 52 31 35 35 35 20 46 57 20 31 2E 30 30 2E 30 30
ASCII 固件版本： ACR1555 FW 1.00.00

### 6.3.2. 获取序列号（Get Serial Number）[E0 00 00 47 00]

此命令用于获取序列号。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get Serial Number | E0h | 00h | 00h | 47h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 序列号长度 | 序列号 |

### 6.3.3. 设置 USB 描述符中的 S/N（Set S/N in USB Descriptor）[E0 00 00 F0]

此命令用于设置 USB 描述符中的 S/N。

命令

| 命令 | CLA | INS | P1 | P2 | Le | 命令数据域 | |
|---|---|---|---|---|---|---|---|
| Set S/N in USB Descriptor | E0h | 00h | 00h | F0h | 02h | 00h | 启用 USB 描述符中的 SN |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 | | |
|---|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 03h | 启用 USB 描述符中的 SN | 90h | 00h |

启用 USB 描述符中的 SN（1 字节）

| 启用 USB 描述符中的 SN | 说明 |
|---|---|
| 00h | 禁用 USB 描述符中的 SN |
| 01h | 启用 USB 描述符中的 SN |

### 6.3.4. 设置蜂鸣器控制-单次（Set Buzzer Control - Single Time）[E0 00 00 28 01 …]

此命令用于设置单次蜂鸣器。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Buzzer Control | E0h | 00h | 00h | 28h | 01h | 蜂鸣器状态 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 蜂鸣器状态 |

蜂鸣器状态（1 个字节）

| 蜂鸣器状态 | 说明 |
|---|---|
| 00h | 关闭 |
| 01 ~ FFh | 开启，持续时间以 10ms 为单位 |

### 6.3.5. 设置蜂鸣器控制-重复（Set Buzzer Control - Repeatable）[E0 00 00 28 03 …]

此命令用于设置蜂鸣器的周期

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Buzzer Control | E0h | 00h | 00h | 28h | 03h | 蜂鸣器状态 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 03h | 蜂鸣器状态 |

蜂鸣器状态（3 个字节）

| 操作参数 | 蜂鸣器状态 | 说明 |
|---|---|---|
| 参数 1 – 字节 0 | 开启时间段 | 01 ~ FF: 开启的持续时间，以 10ms 为单位 |
| 参数 2 – 字节 1 | 关闭时间段 | 01 ~ FF: 关闭的持续时间，以 10ms 为单位 |
| 参数 3 – 字节 2 | 重复次数 | 01 ~ FF: 重复的次数 |

### 6.3.6. 获取 LED 状态（Get LED Status）[E0 00 00 29 00]

此命令用于获取当前 LED 的状态。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get LED Status | E0h | 00h | 00h | 29h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | LED 状态 |

### 6.3.7. 设置 LED 控制（Set LED Control）[E0 00 00 29 01 …]

此命令用于设置 LED 控制

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Set LED Control | E0h | 00h | 00h | 29h | 01h | LED 状态 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | LED 状态 |

LED 状态（1 字节）

| LED 状态 | 说明 |
|---|---|
| Bit 0：绿色 LED | 1 = 开；0 = 关 |
| Bit 2：蓝色 LED | 1 = 开；0 = 关 |
| Bit 3：黄色 LED | 1 = 开；0 = 关 |
| Bit 4-7：RFU | 其它 |

### 6.3.8. 获取 UI 操作（Get UI Behaviour）[E0 00 00 21 00]

此命令用于获取 PCD UI 的操作，无需其它命令即可保存设置。仅用于最初的读写器配置。

命令

| 命令 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| Get PICC UI Behaviour | E0h | 00h | 00h | 21h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | PICC UI 操作 |

### 6.3.9. 设置 UI 操作（Set UI Behaviour）[E0 00 00 21 01 …]

此命令用于设置 PICC UI 的操作。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Set PICC UI Behaviour | E0h | 00h | 00h | 21h | 01h | PICC UI 操作 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | PICC UI 操作 |

UI 操作 - 1 个字节，位掩码如下

| 操作参数 | 参数 | 说明 | 选项 |
|---|---|---|---|
| Bit 0 | 读写中（LED 快速闪烁） | 读写器的UI操作 | 1 = 启用<br>0 = 停用 |
| Bit 1 | 等待出示卡片（LED 长亮） | | |
| Bit 2 | 卡片存在/已激活（LED 长亮） | | |
| Bit 3 | 卡片进入天线区域事件（蜂鸣器短暂鸣响） | | |
| Bit 4 | 卡片移出天线区域事件（蜂鸣器短暂鸣响） | | |

*PICC 默认设置 – 1Fh*

***注：***

1. *获取/设置UI操作命令不涵盖SAM接口*

### 6.3.10. 获取 BLE UI 操作（Get BLE UI Behaviour）[E0 00 00 4B 01 05]

此命令用于读取 LED 的当前操作。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据域 |
|---|---|---|---|---|---|---|
| Get BLE UI Behaviours | E0h | 00h | 00h | 4Bh | 01h | 05h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | BLE 和充电 UI 操作 |

### 6.3.11. 设置 BLE UI 操作（Set BLE UI Behaviour）[E0 00 00 4B 02 05 …]

此命令用于设置蓝色蓝牙 LED 的操作。

命令

| 命令 | CLA | INS | P1 | P2 | Lc | 响应数据（第 1 个字节） | 响应数据域 |
|---|---|---|---|---|---|---|---|
| Set BLE UI Behaviour | E0h | 00h | 00h | 4Bh | 02h | 05h | BLE UI 操作 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | BLE UI 操作 |

BLE UI 操作 - 1 个字节，位掩码如下

| 操作参数 | 参数 | 说明 | 选项 |
|---|---|---|---|
| Bit 0 | 蓝色 BLE LED | LED由读写器控制。 | 1 = 启用<br>0 = 停用 |

*BLE 的默认设置 – 01h*

### 6.3.12. 获取休眠模式选项（Get Sleep Mode Option）[E0 00 00 50 00]

此命令用于查看休眠模式计时器。

***注：****仅适用于固件版本 1.02.04 及更高版本*

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Get Sleep Timer Option | E0h | 00h | 00h | 50h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 时间 |

其中：

**时间** 1 个字节：计时器

00h = 60s（默认）

01h = 90s

02h = 120s

03h = 180s

04h = 无休眠

### 6.3.13. 设置休眠模式选项（Set Sleep Mode Option）[E0 00 00 48 …]

默认情况下，如果 60 秒内没有操作，读卡器会进入休眠状态。此命令用于设置设备进入休眠模式前的时间间隔。

***注：****仅适用于固件版本 1.02.04 及更高版本*

命令

| 命令 | CLA | INS | P1 | P2 | 响应数据域 |
|---|---|---|---|---|---|
| Set Auto Power Off | E0h | 00h | 00h | 48h | 时间 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 时间 |

其中：

**时间** 1 个字节：时间

00h = 60s（默认）

01h = 90s

02h = 120s

03h = 180s

04h = 无休眠

### 6.3.14. 获取 Tx 功率值（Get Tx Power Value）[E0 00 00 51 00]

此命令用于读取蓝牙的 Tx 功率。

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Get Tx power Value | E0h | 00h | 00h | 51h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | Tx 功率 |

### 6.3.15. 设置 Tx 功率值（Set Tx Power Value）[E0 00 00 49 …]

此命令用于修改蓝牙的 Tx 功率。

命令

| 命令 | CLA | INS | P1 | P2 | 响应数据域 |
|---|---|---|---|---|---|
| Get Tx power Value | E0h | 00h | 00h | 49h | Tx 功率 |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | Tx 功率 |

其中：

**Tx 功率** 1 个字节

00h = -23 dBm，距离：~3 米

01h = -6 dBm（默认），距离：~7 米

02h = 0 dBm，距离：~17 米

03h = 4 dBm，距离：~25 米

*默认值 – 01h*

### 6.3.16. 获取 MAC 地址（Get MAC Address）[E0 00 00 43 00]

此命令用于读取 BLE 读写器的 MAC 地址。

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Get MAC Address | E0h | 00h | 00h | 43h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 06h | MAC 地址 |

其中：

MAC 地址 6 个字节

AA:BB:CC:DD:EE:FF（小端格式）

BLE MAC 地址：FF:EE:DD:CC:BB:AA

### 6.3.17. 获取 BLE 广播名称（Get BLE Advertising Name）[E0 00 00 44 00]

此命令用于读取 BLE 广播名称。

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Get BLE Advertising Name | E0h | 00h | 00h | 44h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | length | BLE 广播名称 |

### 6.3.18. 获取电量（Get Battery Level）[E0 00 00 52 00]

此命令用于查看当前的电池电量。

***注：****仅适用于读写器处于蓝牙模式状态。*

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Get Battery Level | E0h | 00h | 00h | 52h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 命令数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 电池电量 |

其中：

电池电量　1 个字节

64h = 100%电量

5Ah = 90%电量

50h = 80%电量

46h = 70%电量

3Ch = 60%电量

32h = 50%电量

28h = 40%电量

1Eh = 30%电量

14h = 20%电量

0Fh = 15%电量

### 6.3.19. 删除 BLE 绑定记录（Remove BLE Bonding Record）[E0 00 00 5B 00]

此命令用于删除 BLE 绑定记录。

***注：****仅适用于 USB 模式的读写器*

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Remove BLE Bonding Record | E0h | 00h | 00h | 5Bh | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 | |
|---|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | 90h | 00h |

### 6.3.20. 读取 BLE 通信模式（Read BLE Communication Mode）

此命令用于读取 BLE 通信模式。

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Read BLE Communication Mode | E0h | 00h | 00h | 77h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 通信模式 |

### 6.3.21. 设置 BLE 通信模式（Set BLE Communication Mode）

此命令用于设置 BLE 通信模式。

步骤（1）获取随机数：

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Get Random Number | E0h | 00h | 00h | 75h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 10h | 16 字节随机数 |

步骤（2）设置 BLE 通信模式：

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Set BLE Communication Mode | E0h | 00h | 00h | 77h | 加密的 16 字节随机数<br>+<br>加密的 16 字节模式值 |

其中：

| 模式值 | |
|---|---|
| 通信模式 | 15 个字节为任意值 |

通信模式　1 个字节　　00h：明文
01h：认证

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | SW1 SW2 |

| 结果 | SW1 SW2 | 含义 |
|---|---|---|
| 成功 | 90 00h | 操作成功完成。 |
| 错误 | 67 00h | 操作失败。 |

### 6.3.22. 重写客户主密钥（Customer Master Key Rewrite）

此命令用于设置客户主密钥。

步骤（1）获取随机数：

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Get Random Number | E0h | 00h | 00h | 75h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 10h | 16 字节随机数 |

步骤（2）重写客户主密钥：

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Customer Master Key Rewrite | E0h | 00h | 00h | 76h | 加密的 16 字节随机数<br>+<br>加密的 16 字节新主密钥 |

响应状态码（成功）

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 02h | 90 00h |

响应状态码（失败）

| 响应 | CLA | INS | P1 | P2 | Le |
|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 00h |

### 6.3.23. 读取认证错误计数器（Read Authentication Error Counter）

此命令用于读取认证错误计数器。

命令

| 命令 | CLA | INS | P1 | P2 | Lc |
|---|---|---|---|---|---|
| Read Authentication Error Counter | E0h | 00h | 00h | 72h | 00h |

响应状态码

| 响应 | CLA | INS | P1 | P2 | Le | 响应数据域 |
|---|---|---|---|---|---|---|
| 结果 | E1h | 00h | 00h | 00h | 01h | 错误计数器值 |

其中：

错误计数器值　1个字节

# 附录A. NDEF 消息

本节介绍如何使用 NDEF 消息将 URL 编码到 NTag 上。

如需了解数据结构，请参考“NFC Forum NFC Data Exchange Format (NDEF) Specifications 1.0”规范。

**例如：**

NDEF 消息 = { D1 01 0B 55 02 61 63 73 2E 63 6F 6D 2E 68 6Bh}

| 偏移 | 内容 | 长度 | 说明 |
|---|---|---|---|
| 0 | D1 | 1 | NDEF 头部。TNF = 01h, SR=1, MB=1, ME=1 |
| 1 | 01 | 1 | 记录名长度（1 字节） |
| 2 | 0B | 1 | URI 有效载荷的长度（11 字节） |
| 3 | 55 (“U”) | 1 | 记录类型：“U” |
| 4 | 02 | 1 | 缩写：“http://www.” |
| 5 | 61 63 73 2E 63 6F 6D 2E 68 6B | 10 | URL 本身。“acs.com.hk” |

编码到 Ntag = {03 0F D1 01 0B 55 01 61 63 73 2E 63 6F 6D 2E 68 6B FEh}

| 偏移 | 内容 | 长度 | 说明 |
|---|---|---|---|
| 0 | 03 | 1 | TLV 头部。03h = NDEF 消息 |
| 1 | 0F | 1 | NDEF 消息的长度（15 字节） |
| 2 | D1 01 0B 55 01 61 63 73 2E 63 6F 6D 2E 68 6B | 15 | NDEF 消息 |
| 17 | FE | 1 | TLV 头部。FEh = 记录结束 |

# 附录B. 槽位状态和槽位错误

每个 Builk-IN 消息都包含槽位错误和槽位状态寄存器的值。

| 偏移 | 数据域 | 大小 | 值 | 说明 |
|---|---|---|---|---|
| 0 | bmICCStatus | 2 位 | 0, 1, 2 | 0 - ICC 存在且处于激活状态（已开启稳定供电，RST 信号未激活）<br>1 - ICC 存在但未激活（由于硬件错误导致未激活或关闭)<br>2 - ICC 不存在<br>3 - RFU |
| 2 | bmRFU | 4 位 | RFU | 智能海报数据的长度（15 字节） |
| 6 | bmCommandStatus | 2 位 | 0, 1, 2 | 0 - 处理无误<br>1 - 失败（错误寄存器提供的错误代码）<br>2 - 请求延长时间<br>3 - RFU |

**表18：**槽位状态寄存器

| 错误代码 | 错误名称 | 可能的原因 |
|---|---|---|
| FFh | CMD_ABORTED | 主机中止了当前活动 |
| FEh | ICC_MUTE | 与 ICC 通讯时，CCID 超时 |
| FDh | XFR_PARITY_ERROR | 与 ICC 通讯时，奇偶校验错误 |
| FCh | XFR_OVERRUN | 与 ICC 通讯时，超限错误 |
| FBh | HW_ERROR | 发生综合性的硬件错误 |
| F8h | BAD_ATR_TS | |
| F7h | BAD_ATR_TCK | |
| F6h | ICC_PROTOCOL_NOT_SUPPORTED | |
| F5h | ICC_CLASS_NOT_SUPPORTED | |
| F4h | PROCEDURE_BYTE_CONFLICT | |
| F3h | DEACTIVATED_PROTOCOL | |
| F2h | BUSY_WITH_AUTO_SEQUENCE | 自动序列进行中 |
| E0h | CMD_SLOT_BUSY | 向已经在处理命令的插槽发送第二个命令 |
| C0h 至 81h | 用户定义 | |
| 80h 以及填补 | RFU | |

| 错误代码 | 错误名称 | 可能的原因 |
|---|---|---|
| 空缺的值 | | |
| 7Fh 至 01h | 不受支持的/错误的消息参数的索引 | 01h：dwLength 错误<br>05h：bSlot 不存在<br>07h：bPowerselect 错误（不支持）<br>08h：wLevelParameter 错误<br>0Ah：FI – DI 配对无效或不受支持<br>0Bh：TCCKTS 参数无效<br>0Ch：不支持保护时间<br>0Dh：T = 0<br>WI 无效或不受支持<br>T = 1<br>BWI 或 CWI 无效或不受支持<br>0Eh：请求的时钟停止支持无效或不受支持<br>0Fh：IFSC 大小无效或不受支持<br>10h：NAD 值无效或不受支持 |
| 00h | 不支持此命令 | |

**表19**：槽位错误寄存器（bmCommandStatus = 1）