Advanced Card Systems Ltd.
Card & Reader Technologies

ACR122L

带液晶显示屏的 NFC 读写器（串口）

通信协议 V1.03

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

目录

图目录

表目录

1.0. 简介

ACR122L 串行协议定义了 PC 与读写器之间的接口，以及 PC 与支持的非接触式标签/卡—ISO 14443-4 A 类和 B 类卡、Mifare 卡、ISO 18092（NFC 卡）和 FeliCa 之间的通信通道。主要支持以下应用：

- **访问控制、识别：**读取天线场内所有卡片的序列号
- **数据存储：**执行加密的读写操作。
- **票务：**加密环境中执行读、写、增量和减量操作。
- **多应用：**在卡的各个扇区执行读、写、增量和减量操作。

2.0. 特性

- RS-232 串行接口：波特率 = 115200 bps，8-N-1
- 电源：7 V 直流适配器
- 仿 CCID 架构（二进制格式）
- 智能卡读写器：
  - 读/写速率高达 424 Kbps
  - 内置天线用于读写非接触式标签，读取智能卡的距离可达 50 mm（视标签的类型而定）
  - 支持 ISO 14443 第 4 部分 A 类和 B 类卡、MIFARE 卡、FeliCa 卡和全部 4 种 NFC（ISO/IEC 18092）标签
  - 内建防冲突特性（任何时候都只能访问 1 张标签）
  - 3 个符合 ISO 7816 标准的 SAM 卡槽
- 内置外围设备：
  - 2 行图形液晶显示屏，可进行交互操作（如上下、左右滚动）并支持多种语言（中文、英语、日语和一些欧洲语言）
  - 4 个用户可控的 LED 指示灯
  - 1 个用户可控的蜂鸣器
- 符合下列标准：
  - ISO 14443
  - CE
  - FCC
  - VCCI
  - RoHS

2.1. 串行接口

ACR122L 通过 RS232C 串行接口连接主机，波特率为 115200 bps，8-N-1。

引脚	信号	功能
1	VCC	为读写器提供 +7V 的电源（最大 350 mA，常规 200 mA）
2	TXD	读写器向主机发送的信号
3	RXD	主机向读写器发送的信号
4	GND	参考电压等级

表1：引脚配置

2.2. LCD

提供了一个用户可控的 LCD。

- 2 行 x 16 个字符的点阵式 LCD，每个字符 5 x 8 点，STN 黄绿色
- 黄绿色背光
- 6 点钟视角

2.3. LED 指示灯

提供了四个用户可控的单色 LED 指示灯。

- 可选择由用户或固件控制 LED 指示灯。
- 从左至右，LED 指示灯的颜色分别是绿色、蓝色、橙色和红色。

2.4. 蜂鸣器

提供了一个用户可控的单音蜂鸣器，默认处于 OFF 状态。

2.5. SAM 接口

提供了 3 个 SAM 卡槽，支持 ISO 7816-1/2/3 T=0 卡。

2.6. 内置天线

提供了一个中间抽头的 3 圈对称环形天线。

- 预计尺寸 = 46 mm x 64 mm.
- 回路电感大概在 1.6 μH 到 2.5 μH 之间。
- 标签的工作距离可达到 50 mm（视标签类型而定）。
- 每次只能访问一张标签。

3.0. 主机和非接触接口、SAM 及外设之间的通讯

主机通过私有 APDU 访问非接触接口和外设。

另外，主机通过标准 APDU 访问 SAM 接口。

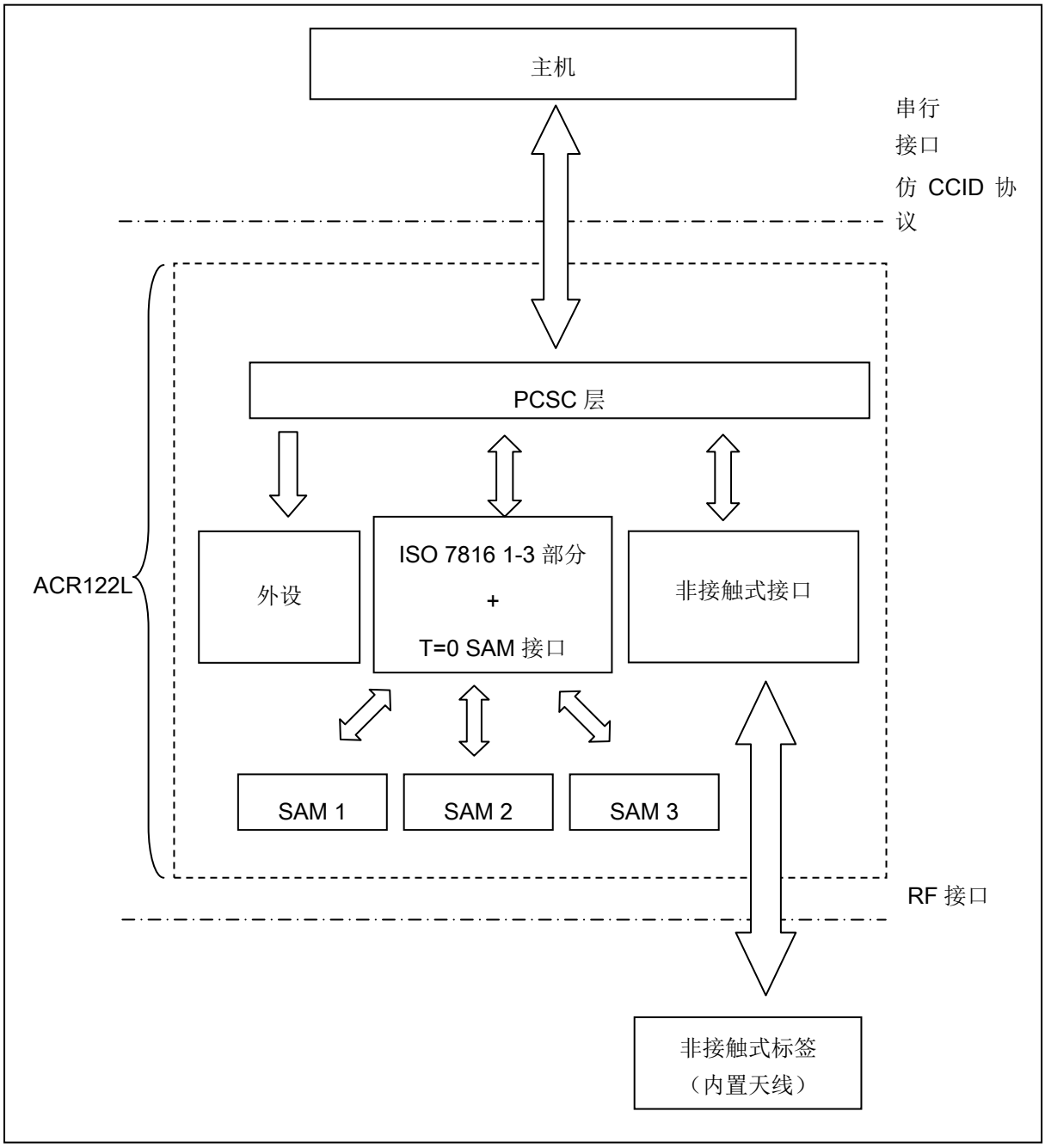


图1：ACR122L 通讯流程图

4.0. 串行接口（仿 CCID 架构）

正常操作中，ACR122L 在计算机和读写器通讯过程中充当从设备。通讯过程包括一连串的的命令-应答：计算机发送一条命令给读卡器，并在命令执行后收到读卡器的应答。计算机只有在收到前一条命令的回应后才能给 ACR122L 发送下一条命令。没有收到计算机命令时，读写器能直接传输的数据只有读写器复位消息和卡片状态消息。

注：*通讯设置为 115200 bps，8-N-1.*

主机与 ACR122L 之间的通信协议非常类似于 CCID 协议。

ACR122L 的命令帧结构

STX (02h)	Bulk-OUT 头	APDU 命令或参数	校验和	ETX (03h)
1 个字节	10 个字节	M 个字节（如适用）	1 个字节	1 个字节

ACR122L 的状态帧结构

STX (02h)	状态	校验和	ETX (03h)
1 个字节	1 个字节	1 个字节	1 个字节

ACR122L 的应答帧结构

STX (02h)	Bulk-IN 头	APDU 响应或 abData	校验和	ETX (03h)
1 个字节	10 个字节	N 个字节 （如有）	1 个字节	1 个字节

校验和 = XOR {Bulk-OUT 头，APDU 命令或参数}

校验和 = XOR {Bulk-IN 头，APDU 应答或 abData}

如需控制 SAM 卡槽 1，STX = 02h，同时 ETX = 03h。

如需控制 SAM 卡槽 2，STX = 12h，同时 ETX = 13h。

如需控制 SAM 卡槽 3，STX = 22h，同时 ETX = 23h。

如需控制对非接触接口和外设（例如：LED，LCD 和蜂鸣器）的访问，则同控制 SAM 卡槽 1 的情况相同，STX 必须等于 02h，而 ETX 必须等于 03h。

总体而言，会利用三种 Bulk-OUT 头：

- HOST_to_RDR_IccPowerOn：激活 SAM 接口。如果有，将会返回 SAM 的 ATR。
- HOST_to_RDR_IccPowerOff：取消激活 SAM 接口。
- HOST_to_RDR_XfrBlock：主机和 ACR122L 交换 APDU。

要使用非接触接口和外设，必须先激活 SAM1 接口。总之，交换 APDU 必须通过 SAM 接口。

相应的，会用到两种 Bulk-IN 头：

- RDR_to_HOST_DataBlock：回应 HOST_to_RDR_IccPowerOn 帧和 HOST_to_RDR_XfrBlock 帧。
- RDR_to_HOST_SlotStatus：回应 HOST_to_RDR_IccPowerOff 帧。

RDR = ACR122L；HOST = 主机控制器。

HOST_to_RDR = 主机控制器 -> ACR122L

RDR_to_HOST = ACR122L -> 主机控制器

4.1. 协议流举例

（以 SAM 接口 1 为例）

A. 激活一个 SAM。

	主机		读写器
1.	主机发送一个帧。	→	02 62 00 00 00 00 00 01 01 00 00 [校验和] 03
2.	读写器立即返回一个肯定确认帧。	02 00 00 03 (肯定确认帧)	←
		.. 经过一些处理延迟…	
3.	读写器返回对命令的应答。	02 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [校验和] 03	←

B. 激活一个 SAM（不正确的校验和，主机）

	主机		读写器
1.	主机发送一个已经损坏的帧。	→	02 62 00 00 00 00 00 01 01 00 00 [不正确的校验和] 03
2.	读写器立即返回一个否定确认帧。	02 FF FF 03 (否定确认帧)	←
3.	主机重新发送帧。	→	02 62 00 00 00 00 00 01 01 00 00 [校验和] 03
4.	读写器立即返回一个肯定确认帧。	02 00 00 03 (肯定确认帧)	←
		.... 经过一些处理延迟 ..	
5.	读写器返回对命令的应答。	02 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [校验和] 03	←

C. 激活一个 SAM（不正确的校验和，读写器）

	主机		读写器
1. 主机发送一个帧。	→	02 62 00 00 00 00 00 01 01 00 00 [校验和] 03	
2. 读写器立即返回一个肯定确认帧。		02 00 00 03 (肯定确认帧)	←
		.. 经过一些处理延迟...	
3. 读写器返回对命令的应答（已经损坏）。	→	02 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [不正确的校验和] 03	←
4. 主机发送一个 NAK 帧来重新获取响应。		02 00 00 00 00 00 00 00 00 00 00 00 03 (NAK)	
5. 读写器返回对命令的应答。		02 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [校验和] 03	←

注：*读写器正确接收到主机发送的帧后，会立即发送一个肯定确认帧 = {02 00 00 03}给主机，通知主机该帧已经被成功接收。主机必须等待命令的响应，而读写器在命令处理期间不会接受其它的帧。*

出现差错时，读写器会向主机发送一个否定确认帧，表示帧已经损坏或者格式错误。

校验和错误帧 = {02 FF FF 03}。

长度错误帧 = {02 FE FE 03}。“dDwLength”的长度大于 0105h 个字节。

ETX 错误帧 = {02 FD FD 03}。最后一个字节不等于 ETX “03h”。

超时错误帧 = {02 FC FC 03}。没有接收到完整的数据包。

NAK 帧仅由主机使用来获取最后一个应答。

{02 00 00 00 00 00 00 00 00 00 00 00 03} // 11 个零

5.0. SAM 接口

ACR122L 有三个 SAM 接口。

激活 SAM 接口

5.1. ACR122L 的命令帧结构

STX	Bulk-OUT 头 (HOST_to_RDR_IccPowerOn)	参数	校验和	ETX
1 个字节	10 个字节	0 个字节	1 个字节	1 个字节

如需控制 SAM 接口 1，STX = 02h，同时 ETX = 03h。

如需控制 SAM 接口 2，STX = 12h，同时 ETX = 13h。

如需控制 SAM 接口 3，STX = 22h，同时 ETX = 23h。

HOST_to_RDR_IccPowerOn 的结构

偏移	字段	大小	值	说明
0	*bMessageType*	1	62h	
1	*dDwLength <LSB ..MSB>*	4	00000000h	消息特定的数据长度。
5	*bSlot*	1	00-FFh	命令的插槽号。默认为 00h.
6	*bSeq*	1	00-FFh	命令的序列号。
7	*bPowerSelect*	1	00h 01h 02h 03h	ICC 上的电压值： 00h – 自动电压选择 01h – 5.0 V 02h – 3.0 V 03h – 1.8 V
8	*abRFU*	2		保留为将来使用

ACR122L 的应答帧结构

STX	Bulk-IN 头 (RDR_to_HOST_DataBlock)	abData	校验和	ETX
1 个字节	10 个字节	N 个字节（ATR）	1 个字节	1 个字节

如需控制 SAM 接口 1，STX = 02h，同时 ETX = 03h。

如需控制 SAM 接口 2，STX = 12h，同时 ETX = 13h。

如需控制 SAM 接口 3，STX = 22h，同时 ETX = 23h。

RDR_to_HOST_DataBlock 的结构

偏移	字段	大小	值	说明
0	*bMessageType*	1	80h	表示正在从 ACR122L 发送一个数据块。
1	*dwLength* *<LSB ..MSB>*	4	N	abData 字段的大小（N 个字节）。
5	*bSlot*	1	与 Bulk-OUT 相同	标识命令的插槽号。
6	*bSeq*	1	与 Bulk-OUT 相同	相应命令的序列号
7	*bStatus*	1		
8	*bError*	1		
9	*bChainParameter*	1		

例 1：激活 SAM 接口 1 槽位 0（默认），序列号为 1，5V 卡。

HOST -> 02 62 00 00 00 00 00 01 01 00 00 [校验和] 03

RDR -> 02 00 00 03

RDR -> 02 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [校验和] 03

ATR = 3B 2A 00 80 65 24 B0 00 02 00 82；SW1 SW2 = 90 00

例 2：激活 SAM 接口 2 槽位 0（默认），序列号为 1，5V 卡。

HOST -> 12 62 00 00 00 00 00 01 01 00 00 [校验和] 13

RDR -> 12 00 00 13

RDR -> 12 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [校验和] 13

ATR = 3B 2A 00 80 65 24 B0 00 02 00 82；SW1 SW2 = 90 00

例 3：激活 SAM 接口 3 槽位 0 （默认），序列号为 1，5V 卡。

HOST -> 22 62 00 00 00 00 00 01 01 00 00 [校验和] 23

RDR -> 22 00 00 23

RDR -> 22 80 0D 00 00 00 00 01 00 00 00 3B 2A 00 80 65 24 B0 00 02 00 82 90 00 [校验和] 23

ATR = 3B 2A 00 80 65 24 B0 00 02 00 82；SW1 SW2 = 90 00

5.2. 取消激活 SAM 接口

ACR122L 的命令帧结构

STX	Bulk-OUT 头 (HOST_to_RDR_IccPowerOff)	参数	校验和	ETX
1 个字节	10 个字节	0 个字节	1 个字节	1 个字节

如需控制 SAM 接口 1，STX = 02h，同时 ETX = 03h。
如需控制 SAM 接口 2，STX = 12h，同时 ETX = 13h。
如需控制 SAM 接口 3，STX = 22h，同时 ETX = 23h。

HOST_to_RDR_IccPowerOff 的结构

偏移	字段	大小	值	说明
0	*bMessageType*	1	63h	
1	*dDwLength <LSB ..MSB>*	4	00000000h	消息特定的数据长度。
5	*bSlot*	1	00-FFh	标识命令的插槽号。默认为 00h.
6	*bSeq*	1	00-FFh	命令的序列号。
7	*abRFU*	3		保留为将来使用

ACR122L 的应答帧结构

STX	Bulk-IN 头 (RDR_to_HOST_SlotStatus)	abData	校验和	ETX
1 个字节	10 个字节	0 个字节	1 个字节	1 个字节

如需控制 SAM 接口 1，STX = 02h，同时 ETX = 03h。
如需控制 SAM 接口 2，STX = 12h，同时 ETX = 13h。
如需控制 SAM 接口 3，STX = 22h，同时 ETX = 23h。

RDR_to_HOST_DataBlock 的结构

偏移	字段	大小	值	说明
0	*bMessageType*	1	81h	表示正在从 ACR122L 发送一个数据块。
1	*dwLength <LSB ..MSB>*	4	0	abData 字段的大小（0 个字节）。
5	*bSlot*	1	与 Bulk-OUT 相同	标识命令的插槽号。
6	*bSeq*	1	与 Bulk-OUT 相同	相应命令的序列号
7	*bStatus*	1		

偏移	字段	大小	值	说明
8	*bError*	1		
9	*bClockStatus*	1		

例 1：取消激活 SAM 接口 1 槽位 0（默认），序列号为 2。

HOST -> 02 63 00 00 00 00 00 02 00 00 00 [校验和] 03

RDR -> 02 00 00 03

RDR -> 02 81 00 00 00 00 00 02 00 00 00 [校验和] 03

例 2：取消激活 SAM 接口 2 槽位 0（默认），序列号为 2。

HOST -> 12 63 00 00 00 00 00 02 00 00 00 [校验和] 13

RDR -> 12 00 00 13

RDR -> 12 81 00 00 00 00 00 02 00 00 00 [校验和] 13

例 3：取消激活 SAM 接口 3 槽位 0（默认），序列号为 2。

HOST -> 22 63 00 00 00 00 00 02 00 00 00 [校验和] 23

RDR -> 22 00 00 23

RDR -> 22 81 00 00 00 00 00 02 00 00 00 [校验和] 23

5.3. 通过 SAM 接口交换数据

ACR122L 的命令帧结构

STX	Bulk-OUT 头 (HOST_to_RDR_XfrBlock)	参数	校验和	ETX
1 个字节	10 个字节	M 个字节	1 个字节	1 个字节

如需控制 SAM 接口 1，STX = 02h，同时 ETX = 03h。

如需控制 SAM 接口 2，STX = 12h，同时 ETX = 13h。

如需控制 SAM 接口 3，STX = 22h，同时 ETX = 23h。

HOST_to_RDR_XfrBlock 的结构

偏移	字段	大小	值	说明
0	*bMessageType*	1	6Fh	
1	*dDwLength* *<LSB ..MSB>*	4	M	消息特定的数据长度。
5	*bSlot*	1	00-FFh	标识命令的插槽号。默认为 00h.
6	*bSeq*	1	00-FFh	命令的序列号。

偏移	字段	大小	值	说明
7	*bBWI*	1	00-FFh	用于延长块的超时等待时间
8	*wLevelParameter*	2	0000h	

ACR122L 的应答帧结构

STX	Bulk-IN 头 (RDR_to_HOST_DataBlock)	abData	校验和	ETX
1 个字节	10 个字节	N 个字节（ATR）	1 个字节	1 个字节

如需控制 SAM 接口 1，STX = 02h，同时 ETX = 03。

如需控制 SAM 接口 2，STX = 12h，同时 ETX = 13h。

如需控制 SAM 接口 3，STX = 22h，同时 ETX = 23h。

RDR_to_HOST_DataBlock 的结构

偏移	字段	大小	值	说明
0	*bMessageType*	1	80h	表示正在从 ACR122L 发送一个数据块。
1	*dwLength <LSB ..MSB>*	4	N	abData 字段的大小（N 个字节）。
5	*bSlot*	1	与 Bulk-OUT 相同	标识命令的插槽号。
6	*bSeq*	1	与 Bulk-OUT 相同	相应命令的序列号
7	*bStatus*	1		
8	*bError*	1		
9	*bChainParameter*	1		

例 1：发送 APDU“80 84 00 00 08”给 SAM 接口 1 槽位 0（默认），序列号为 3。

HOST -> 02 6F 05 00 00 00 00 03 00 00 00 80 84 00 00 08 [校验和] 03

RDR -> 02 00 00 03

RDR -> 02 80 0A 00 00 00 00 03 00 00 00 E3 51 B0 FC 88 AA 2D 18 90 00 [校验和] 03

应答 = E3 51 B0 FC 88 AA 2D 18; SW1 SW2 = 90 00

例 2：发送 APDU“80 84 00 00 08”给 SAM 接口 2 槽位 0（默认），序列号为 3。

HOST -> 12 6F 05 00 00 00 00 03 00 00 00 80 84 00 00 08 [校验和] 13

RDR -> 12 00 00 13

RDR -> 12 80 0A 00 00 00 00 03 00 00 00 E3 51 B0 FC 88 AA 2D 18 90 00 [校验和] 13

应答 = E3 51 B0 FC 88 AA 2D 18; SW1 SW2 = 90 00

例 3：发送一个 APDU“80 84 00 00 08”给 SAM 接口 3 槽位 0（默认），序列号为 3。

HOST -> 22 6F 05 00 00 00 00 03 00 00 00 80 84 00 00 08 [校验和] 23

RDR -> 22 00 00 23

RDR -> 22 80 0A 00 00 00 00 03 00 00 00 E3 51 B0 FC 88 AA 2D 18 90 00 [校验和] 23

应答 = E3 51 B0 FC 88 AA 2D 18; SW1 SW2 = 90 00

6.0. 用以控制非接触接口和外设的私有 APDU

提供了两个基础命令以控制非接触接口和外设<Class FFh>。

注：对于下面的所有私有 APDU（除了 6.2—更改通讯速度和 6.3—获取固件版本），STX 必须等于 02h，而 ETX 必须等于 03h。

6.1. 直接传输（Direct Transmit）

此命令用于发送私有 APDU（标签命令），并返回响应数据的长度。

Direct Transmit 的命令结构（RC531_TAG 命令的长度 + 5 个字节）

命令	CLA	INS	P1	P2	Lc	命令数据域	
Direct Transmit	FFh	00h	00h	00h	待发送的字节数	标签命令	数据

其中：

Lc 待发送的字节数（1 个字节）。

最大值为 255 个字节。

命令数据域 标签命令。

待发送给标签的数据。

Direct Transmit 的响应结构（标签响应 + 数据 + 2 个字节）

项	命令	数据			含义
1	D4 40	Tg		[DataOut[]]	标签交换数据
2	D4 4A	MaxTg	BrTy	[InitiatorData[]]	标签轮询

其中：

Tg 一个字节，包含相关目标的逻辑号。此字节也包含 MI 位（More Information 更多信息），即 bit 6。如果 MI 位设为 1，表示主机控制器需发送更多数据，即 DataOUT[]数组包含的所有数据。MI 位只对 TPE 目标有效。

DataOut 一个原始数据数组，由非接触芯片发送，0-262 个字节。

MaxTg 非接触芯片初始化目标的最大数量。芯片最多处理两个目标，因此该字段的值不得超过 02h。

Brty 初始化时使用的波特率和调制方式。

00h：106 kbps A 类 (ISO/IEC14443 A 类)，

01h：212 kbps (FeliCa 轮询)，

02h：424 kbps (FeliCa 轮询)，

03h106 kbps B 类 (ISO/IEC 14443-3B)，

04h：106 kbps Innovision Jewel 标签.

InitiatorData[] 目标初始化时使用的一个数据数组。字段内容因波特率而异。

106 kbps A 类 字段可选，只有主机控制器需初始化一个 UID 未知的对象时才会出现。

这时，InitiatorData[]含有卡的完整（或部分）UID。如果级联级别为 2 级或 3 级，UID 必须包括级联标签 CT。

级联级别 1

UID1	UID2	UID3	UID4

级联级别 2

UID1	UID2	UID3	UID4	UID5	UID6	UID7

级联级别 3

UID1	UID2	UID3	UID4	UID5	UID6	UID7	UID8	UID9	UID10

106 kbps B 类

InitiatorData[]的结构如下：

AFI (1 个字节)	[轮询方法]

AFI AFI（Application Family Identifier，应用族识别符）表示设备 IC 的目标应用的类型，用在 ATQB 之前预选 PICC。

此字段必选。

轮询方法 此字段可选。表示 ISO/IEC 14443-3B 初始化中用到的方法：

若 bit 0 = 1：ISO/IEC 14443-3B 初始化中的随机方法（选项 1），

若 bit 0 = 0： ISO/IEC 14443-3B 初始化中的时间槽方法（选项 2），

如果没有该字段，将使用时间槽法。

212/424 kbps 此字段必选，包含轮询请求命令（5 个字节，长度字节除外）须用到的完整有效载荷信息。

106 kbps InnoVision Jewel 标签。本字段无用途。

Data Out 读写器返回的响应。

Direct Transmit 的响应结构

响应	响应数据域				
结果	D5 41	状态	[DataIn[]]		SW1 SW2
	D5 4B	NbTg	[TargetData1[]]	[TargetData2[]]	

其中：

状态	一个字节，表示进程是否已成功终止。DEP 或 ISO/IEC 14443-4 PCD 模式下，该字节还表示 NAD（节点地址）是否在用以及用 MI 位传输的数据是否完整。
DataIn	一个原始数据数组，由非接触芯片接收，0-262 个字节。
NbTg	初始化目标的数量（最少 0 个，最多 2 个）。
TargetDatai[]	TargetDatai[]所含字母“i”是指 “1”或“2”。根据选定的波特率，该字段提供关于检测到的目标信息。以下仅列出一个目标的信息，其他初始化目标的信息与此相同 （NbTg 次）。

106 kbps A 类

Tg	SENS_RES10 （2 个字节）	SEL_RES （1 个字节）	NFCIDLength （1 个字节）	NFCID1[] (NFCIDLength bytes)	ATS (ATSLength bytes11))

106 kbps B 类

Tg	ATQB 响应 （12 个字节）	ATTRIB_RES 的长度 （1 个字节）	ATTTRIB_RES[] （ATTRIB_RES 的长度）

212/424 kbps

Tg	POL_RES 的长度	01h （响应码）	NFCID2t	Pad	SYST_CODE （可选）
1 个字节	1 个字节	1 个字节	8 个字节	8 个字节	2 个字节
	POL_RES （18 或 20 个字节）				

106 kbps Innovision Jewel 标签

Tg	SENS_RES （2 个字节）	JEWELID[] （4 个字节）

响应数据域 SW1 SW2。读卡器返回的状态码。

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

结果	SW1	SW2	含义
超时错误	63	01h	标签未响应。
校验和错误	63	27h	响应的校验和错误。
参数错误	63	7Fh	标签命令错误。

6.2. 更改通讯速度（Baud Rate Control）

此命令用于更改通信速度（波特率）。

注：STX = 32h 并且 ETX = 33h

Baud Rate Control 的命令结构（9 个字节）

命令	CLA	INS	P1	P2	Lc
Baud Rate Control	FFh	00h	44h	新波特率	00h

其中：

P2 新波特率

00h = 新波特率设为 9600 bps。

00h = 新波特率设为 115200 bps。

响应数据域 SW1 SW2。

状态码

结果	SW1	SW2	含义
成功	90	当前的波特率	操作成功完成。
错误	63	00h	操作失败。

其中：

SW2 当前的波特率。

00h = 当前的波特率是 9600 bps。

01h = 当前的波特率是 115200 bps。

注：成功更改通信速度后，程序必须对通信速度进行调整，以便继续进行剩余的数据交换。

初始通讯速度取决于 R12（0 ohm）是否存在。

- 带有 R12：初始通讯速度 = 115200 bps
- 不带 R12：初始通讯速度 = 9600 bps（默认）

例 1：初始化一个 FeliCa 标签（标签轮询）

步骤 1. 发送“Direct Transmit”APDU。

APDU 命令为“FF 00 00 00 09 D4 4A 01 01 00 FF FF 01 00”

其中，

“Direct Transmit” APDU = “FF 00 00 00”

标签命令的长度 = “09”

标签命令（InListPassiveTarget 212Kbps） = “D4 4A 01 01”

标签命令（System Code Request）= “00 FF FF 01 00”

发送 APDU 到槽位 0(默认)，序列号为 1。

HOST -> 02 6F 0E 00 00 00 00 01 00 00 00
FF 00 00 00 09 D4 4A 01 01 00 FF FF 01 00 [校验和] 03

RDR -> 02 00 00 03

RDR -> 02 81 1A 00 00 00 00 01 00 00 00
D5 4B 01 01 14 01 01 01 05 01 86 04 02 02 03 00
4B 02 4F 49 8A 8A 80 08 90 00 [校验和] 03

APDU 响应为“D5 4B 01 01 14 01 01 01 05 01 86 04 02 02 03 00 4B 02 4F 49 8A 8A 80 08 90 00”

其中，

非接触芯片的响应 = “D5 4B 01 01 14 01 01 01 05 01 86 04 02 02 03 00 4B 02 4F 49 8A 8A 80 08”

FeliCa 标签的 NFCID2t = “01 01 05 01 86 04 02 02”

读写器返回的状态码= “90 00”

例 2：向 FeliCa 标签写入 16 个字节（写标签）。

步骤 1. 发送一个“Direct Transmit”APDU。

APDU 命令为“FF 00 00 00 23 D4 40 01 20 08 01 01 05 01 86 04 02 02 01 09 01 01 80 00 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA”。

其中，

“Direct Transmit” APDU = “FF 00 00 00”

标签命令的长度 = “23”

标签命令（InDataExchange） = “D4 40 01”

标签命令（写数据）= “20 08 01 01 05 01 86 04 02 02 01 09 01 01 80 00 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA”

发送一个 APDU 到槽位 0(默认)，序列号为 2。

HOST -> 02 6F 26 00 00 00 00 02 00 00 00

FF 00 00 00 21 D4 40 01 20 08 01 01 05 01 86
04 02 02 01 09 01 01 80 00 00 AA 55 AA 55 AA 55
AA 55 AA 55 AA 55 AA
[校验和] 03

RDR -> 02 00 00 03

RDR -> 02 81 11 00 00 00 00 02 00 00 00
D5 41 00 0C 09 01 01 05 01 86 04 02 02 00 00 90 00
[校验和] 03

APDU 响应为“D5 41 00 0C 09 01 01 05 01 86 04 02 02 00 00 90 00”。

其中，

非接触芯片返回的响应 = “D5 41”

FeliCa 标签返回的响应 = “00 0C 09 01 01 05 01 86 04 02 02 00 00”

读写器返回的状态码= “90 00”

例 3：从 FeliCa 标签读取 16 个字节的数据（读标签）。

步骤 1. 发布“Direct Transmit”APDU。

APDU 命令为“FF 00 00 00 13 D4 40 01 10 06 01 01 05 01 86 04 02 02 01 09 01 01 80 00”。

其中，

“Direct Transmit” APDU = “FF 00 00 00”

标签命令的长度 = “13”

标签命令（InDataExchange） = “D4 40 01”

标签命令（读数据） = “10 06 01 01 05 01 86 04 02 02 01 09 01 01 80 00”

发送 APDU 到槽位 0(默认)，序列号为 3。

HOST -> 02 6F 18 00 00 00 00 03 00 00 00
FF 00 00 00 13 D4 40 01 10 06 01 01 05 01 86 04
02 02 01 09 01 01 80 00
[校验和] 03

RDR -> 02 00 00 03

RDR -> 02 81 22 00 00 00 00 03 00 00 00
D5 41 00 1D 07 01 01 05 01 86 04 02 02 00 00 01 00
AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 90 00
[校验和] 03

APDU 响应：

“D5 41 00 1D 07 01 01 05 01 86 04 02 02 00 00 01 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 90 00”

其中，

非接触芯片返回的响应 = “D5 41”

FeliCa 标签返回的响应 =

“00 1D 07 01 01 05 01 86 04 02 02 00 00 01 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA”

读写器返回的状态码= “90 00”

例 4：初始化一个 ISO 14443-4 B 类标签（标签轮询）。

步骤 1. 发布一个“Direct Transmit”APDU。

APDU 命令为“FF 00 00 00 05 D4 4A 01 03 00”。

其中，

Direct Transmit APDU = “FF 00 00 00”

标签命令的长度 = “05”

标签命令(InListPassiveTarget B 类 106Kbps) = “D4 4A 01 03 00”

发送一个 APDU 到槽位 0(默认)，序列号为 4。

HOST -> 02 6F 0A 00 00 00 00 04 00 00 00
FF 00 00 00 05 D4 4A 01 03 00
[校验和] 03

RDR -> 02 00 00 03

RDR -> 02 81 14 00 00 00 00 04 00 00 00
D5 41 01 01 50 00 01 32 F4 00 00 00 00 33 81 81 01 21
90 00 [校验和] 03

APDU 响应为

“D5 4B 01 01 50 00 01 32 F4 00 00 00 00 33 81 81 01 21 90 00”

其中，

非接触芯片返回的响应 = “D5 4B 01 01”

B 类标签的 ATQB = “50 00 01 32 F4 00 00 00 00 33 81 81”

CRC-B = “01 21”

读写器返回的状态码= “90 00”

例 5：发送 APDU 到 ISO 14443-4 B 类标签（数据交换）。

步骤 1. 发送一个“Direct Transmit”APDU。

USER APDU 命令为“00 84 00 00 08”。

APDU 命令为“FF 00 00 00 08 D4 4A 40 01 00 FF FF 00 08”

其中，

“Direct Transmit” APDU = “FF 00 00 00”

标签命令的长度 = “08”

标签命令（InDataExchange） = “D4 40 01”

标签命令（Get Challenge） = “00 84 00 00 08”

发送一个 APDU 到槽位 0(默认)，序列号为 5。

HOST -> 02 6F 0D 00 00 00 00 05 00 00 00
FF 00 00 00 08 D4 40 01 00 84 00 00 08
[校验和] 03

RDR -> 02 00 00 03

RDR -> 02 81 0F 00 00 00 00 05 00 00 00
D5 41 00 01 02 03 04 05 06 07 08 90 00 90 00
[校验和] 03

APDU 响应为“D5 41 00 0B 01 02 03 04 05 06 07 08 90 00”

其中，

非接触芯片返回的响应 = “D5 41 00”

B 类标签的响应 = “01 02 03 04 05 06 07 08 90 00”

读写器返回的状态码= “90 00”

6.3. 获取固件版本号（Get firmware version）

此命令用于获取读写器的固件版本号。

对于 SAM 接口 1 的控制器，STX = 02h，同时 ETX = 03h。

对于 SAM 接口 2 的控制器，STX = 12h，同时 ETX = 13h。

对于 SAM 接口 3 的控制器，STX = 22h，同时 ETX = 23h。

Get Firmware Version 的命令结构（5 个字节）

命令	CLA	INS	P1	P2	Le
Get Response	FFh	00h	48h	00h	00h

其中：

Le 待获取的字节数（1 个字节）。
最大值为 255 个字节。

对于 SAM 接口 1 的控制器，反馈的 STX = 02h，同时 ETX = 03h。
对于 SAM 接口 2 的控制器，反馈的 STX = 12h，同时 ETX = 13h。
对于 SAM 接口 3 的控制器，反馈的 STX = 22h，同时 ETX = 23h。

Get Firmware Version 的响应结构（10 个字节）

响应	响应数据域
结果	固件版本号

例 1：SAM 接口 1 的控制器的响应
= 41 43 52 31 32 32 4C 31 30 31 53 41 4D 31(Hex) = ACR122L101SAM1 (ASCII)

例 2：SAM 接口 2 的控制器的响应
= 41 43 52 31 32 32 4C 31 30 31 53 41 4D 32(Hex) = ACR122L101SAM2 (ASCII)

例 3：SAM 接口 3 的控制器的响应
= 41 43 52 31 32 32 4C 31 30 31 53 41 4D 33(Hex) = ACR122L101SAM3 (ASCII)

注：*SAM 接口 1 的控制器的响应即是设备固件版本号。*

6.4. LCD 显示 - ASCII 模式（LCD Display - ASCII Mode）

此命令用于 ASCII 模式字符信息在 LCD 上的显示。

LCD Display 的命令结构（5 个字节 + 要在 LCD 上显示的消息的长度）

命令	CLA	INS	P1	P2	Lc	命令数据域（最大 16 个字节）
LCD Display	FFh	选项字节	68h	LCD 坐标	LCD 消息长度	LCD 消息

INS 选项字节（1 个字节）

CMD	项	说明
Bit 0	字符加粗	1 = 加粗；0 = 常规
Bit 1 - 3	保留	

CMD	项	说明
Bit 4 - 5	字库索引	00 = 字体 A 01 = 字体 B 10 = 字体 C
Bits 6 – 7	保留	

P2 LCD 坐标。字符在 LCD 上的显示位置，通过 DDRAM 地址进行指定。

请参考以下 DDRAM 地址表以了解 LCD 字符位置的表示。

行\列	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	显示位置
第 1 行	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	LCD 坐标
第 2 行	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	

表2： 字体 1 和 2 的 DDRAM 地址

行\列	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	显示位置
第 1 行	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	LCD 坐标
第 2 行	20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	
第 3 行	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	
第 4 行	60	61	62	63	64	65	66	67	68	69	6A	6B	6C	6D	6E	6F	

表3： 字体 3 的 DDRAM 地址

Lc LCD 消息长度

LCD 消息最大长度为 10h。如果消息长度超过 LCD 可以显示的字符数，多余字符将不会在 LCD 上显示。

命令数据域 LCD 消息。

待发送至 LCD 的数据，每行最多 16 个字符。

有关 LCD 上显示的字符索引，请参考以下字库（通过 INS 的 Bit 4-5 来选择）。

***注：** 字体 A 和字体 B 的字符大小为 8x16，而字体 C 的字符大小为 8x8。*

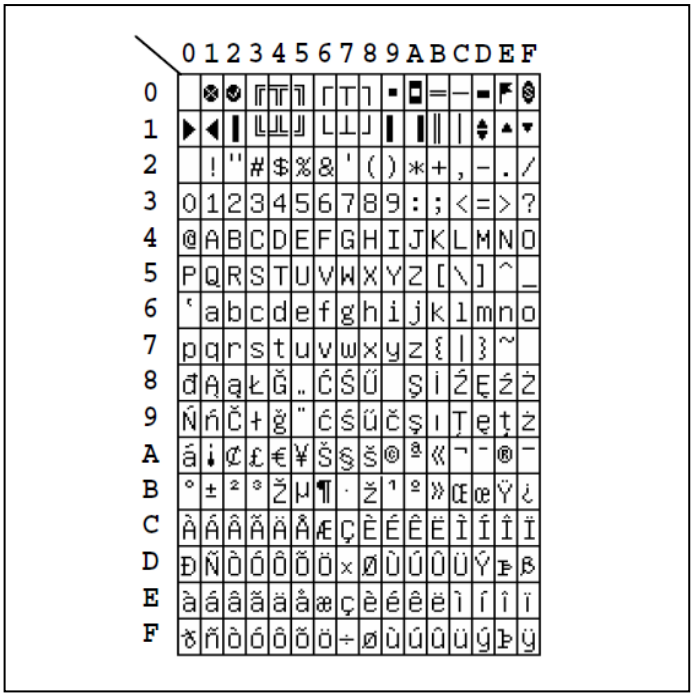


图2： 字符集 A

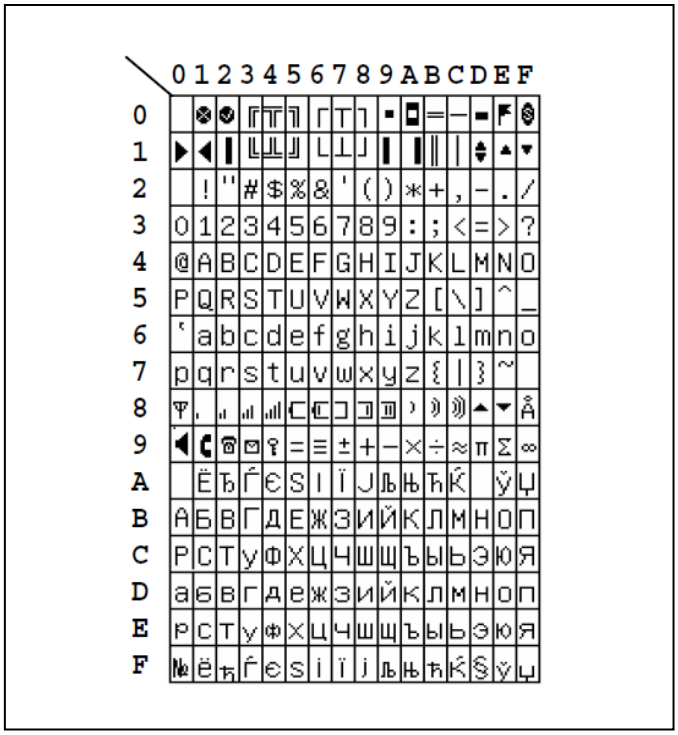


图3： 字符集 B

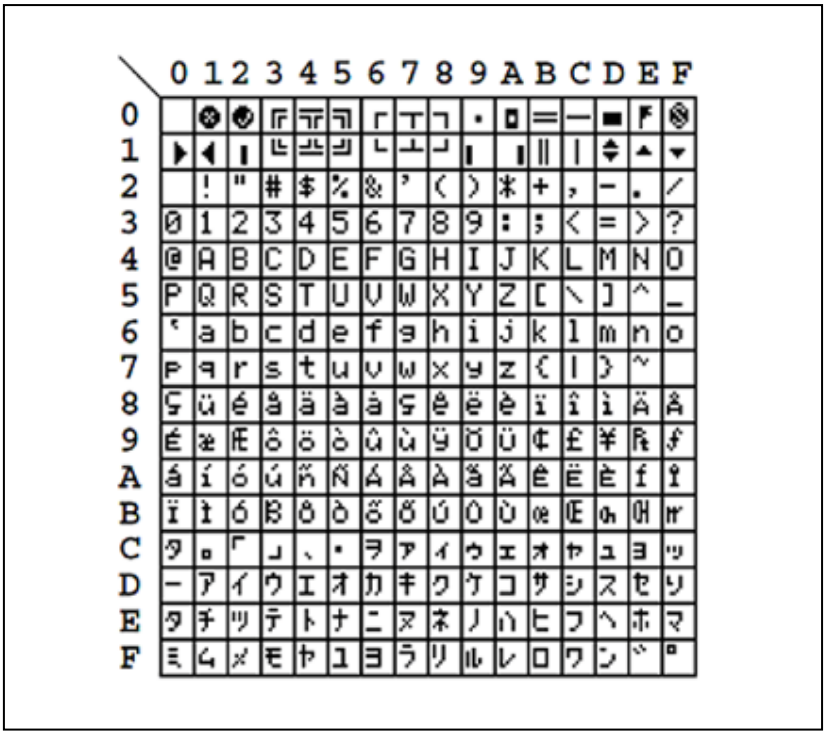


图4： 字符集 C

响应数据域　　SW1 SW2。

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。

6.5. LCD 显示 - GB 模式（LCD Display – GB Mode）

此命令用于 GB 模式字符信息在 LCD 上的显示。

LCD Display 的命令结构（5 个字节 + LCD 消息的长度）

命令	CLA	INS	P1	P2	Lc	命令数据域 (最多 16 个字节)
LCD Display	FFh	选项字节	69h	LCD 坐标	LCD 消息长度	LCD 消息

INS　　选项字节（1 个字节）

CMD	项	说明
Bit 0	字符加粗	1 = 加粗；0 = 常规
Bit 1 - 7	保留	

P2 LCD 坐标。

字符在 LCD 上的显示位置，通过 DDRAM 地址进行指定。

请参考下方的 DDRAM 地址表以了解 LCD 字符位置的表示。

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	显示位置
第 1 行	00		01		02		03		04		05		06		07		LCD 坐标
第 2 行	40		41		42		43		44		45		46		47		

表4： LCD 字符位置的表示

Lc LCD 消息长度

LCD 消息最大长度为 10h。如果消息的长度超过 LCD 可以显示的字符数，则多余的字符将不会在屏幕上显示。

由于每个中文字符（GB 码）包含两个字节，所以信息的长度为 2 的倍数。

命令数据域 要显示在 LCD 上的消息.

要发送至 LCD 的数据，每行最多 8 个字符（每个字符 2x8 位）。请参加下方 GB 编码的字库。

如需在此模式下发送 ASCII 码，字符数应为 2 的倍数；否则，在最后一个字符的后面添加 00h。

响应数据域 SW1 SW2。

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。

6.6. LCD 显示 - 图形模式（LCD Display - Graphic Mode）

此命令用于图形模式字符信息在 LCD 上的显示。

LCD Display 的命令结构（5 个字节 + LCD 消息的长度）

命令	CLA	INS	P1	P2	Lc	命令数据域 (最多 128 个字节)
LCD Display	FFh	00h	6Ah	行索引	像素数据长度	像素数据

其中：

P2 行索引。设置从哪一行开始更新 LCD 显示（参考下表的 LCD 显示位置）

Lc 像素数据长度。像素数据的长度（最大为 80h）。

命令数据域 像素数据。待发送到 LCD 进行显示的像素数据。

	字节 00h (X = 00h)								字节 01h (X = 01h)								…	字节 0Fh (X = 0Fh)							
	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	…	7	6	5	4	3	2	1	0
00h																									
01h																									
02h																									
03h																									
04h																									
05h																									
06h																									
07h																									
08h																									
09h																									
…	…																								
1Fh																									

表5：LCD 显示位置

LCD 总尺寸：128 x 32.

响应数据域　　SW1 SW2。

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。

6.7. 设置当前 LCD 字符滚动（Scroll Current LCD Display）

此命令用于设置当前 LCD 的字符滚动功能。

Scrolling LCD 的命令结构（5 个字节 + LCD 消息的长度）

命令	CLA	INS	P1	P2	Lc	命令数据域（6 个字节）
Scroll LCD	FFh	00h	6Dh	00h	06h	Scroll Ctrl

命令数据域　　Scroll Ctrl.

Scrolling Control 的结构（6 个字节）

字节 0	字节 1	字节 2	字节 3	字节 4	字节 5
00h	00h	0Fh	1Fh	刷新速度控制	滚动方向

其中：

刷新速度控制　　Bit 0~Bit 3 — 滚动一下移动的像素数

Bit 4~Bit 7 — 滚动周期

Bit 7	Bit 6	Bit 5	Bit 4	滚动周期
0	0	0	0	1 个单位
0	0	0	1	3 个单位
0	0	1	0	5 个单位
0	0	1	1	7 个单位
0	1	0	0	17 个单位
0	1	0	1	19 个单位
0	1	1	0	21 个单位
0	1	1	1	23 个单位
1	0	0	0	129 个单位
1	0	0	1	131 个单位
1	0	1	0	133 个单位
1	0	1	1	135 个单位
1	1	0	0	145 个单位
1	1	0	1	147 个单位
1	1	1	0	149 个单位
1	1	1	1	151 个单位

表6：滚动周期

Bit 1	Bit 0	滚动方向
0	0	从左至右
0	1	从右至左
1	0	从上至下
1	1	从下至上

表7：滚动方向

响应数据域　　SW1 SW2。

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。

6.8. 暂停 LCD 字符滚动（Pause LCD Scrolling）

此命令用于暂停之前设置的 LCD 字符滚动功能。

需要重新开始滚动时，可再次发送 Scrolling LCD 命令（参见 6.7）。

Pause Scrolling 的命令结构（5 个字节）

命令	CLA	INS	P1	P2	Lc
Pause Scroll LCD	FFh	00h	6Eh	00h	00h

响应数据域　　SW1 SW2。

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。

6.9. 停止 LCD 字符滚动（Stop LCD Scrolling）

此命令用于停止之前设置的 LCD 字符滚动功能。LCD 显示将恢复到正常显示状态。

Stop Scrolling LCD 的命令结构（5 个字节）

命令	CLA	INS	P1	P2	Lc
Stop Scrol LCD	FFh	00h	6Fh	00h	00h

响应数据域　　SW1 SW2。

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。

6.10. 清除 LCD 上显示的全部内容（Clear LCD）

此命令用于清除 LCD 上显示的全部内容。

Clear LCD 命令的结构（5 个字节）

命令	CLA	INS	P1	P2	Lc
Clear LCD	FFh	00h	60h	00h	00h

响应数据域　　SW1 SW2。

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。

***注：**若 ACR122L 的固件版本为 307 及以上，在执行 Clear LCD 命令后立刻使用其他功能的情况下，应用需要处理额外 100 ms 的延迟时间。*

6.11. 控制 LCD 的背光（LCD Backlight Control）

此命令用于控制 LCD 的背光。

LCD Backlight Control 的命令结构（5 个字节）

命令	CLA	INS	P1	P2	Lc
LCD Backlight Control	FFh	00h	64h	背光控制	00h

P2　　背光控制。

Backlight Control 的结构（1 个字节）

CMD	说明
00h	LCD 背光关闭
FFh	LCD 背光开启

响应数据域　　SW1 SW2。

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。

6.12. 控制 LCD 的对比度（LCD Contrast Control）

此命令用于控制 LCD 的对比度。

LCD Contrast Control 的命令结构（5 个字节）

命令	CLA	INS	P1	P2	Lc
LCD Contrast Control	FFh	00h	6Ch	对比度控制	00h

其中：

P2 对比度控制。

取值范围：00h（最亮）— 0Fh（最暗）

响应数据域 SW1 SW2。

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。

6.13. 开启/关闭 LED（LED Enable/Disable）

此命令用于开启/关闭 LED。

注：*默认"关闭"。由固件控制 LED。*

LED Control Enable 的命令结构（5 个字节）

命令	CLA	INS	P1	P2	Lc
LED Control Enable	FFh	00h	43h	bLEDCtrl	00h

P2 bCtrlEable (1 个字节).

CMD	说明
00h	由用户关闭 LED 控制功能。
FFh	由用户开启 LED 控制功能。

响应数据域 SW1 SW2。

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。

6.14. 控制 LED（LED Control）

此命令用于对四个 LED 进行控制。

LEDs Control 的命令结构（5 个字节）

命令	CLA	INS	P1	P2	Lc
LED Control	FFh	00h	41h	bLEDsState	00h

P2 bLEDsState。

LED_0，LED_1，LED_2 and LED_3 Control 的结构（1 个字节）

CMD	项	说明
Bit 0	LED_0 状态	1 = 开；0 = 关
Bit 1	LED_1 状态	1 = 开；0 = 关
Bit 2	LED_2 状态	1 = 开；0 = 关
Bit 3	LED_3 状态	1 = 开；0 = 关
Bits 4 – 7	保留	

响应数据域 SW1 SW2。

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。

6.15. LED 和蜂鸣器控制（LED and Buzzer Control）

此命令用于控制 LED_0，LED_1 和蜂鸣器的状态。

LED_0，LED_1 and Buzzer Control 的命令结构（9 个字节）

命令	CLA	INS	P1	P2	Lc	命令数据域（4 个字节）
LEDs and Buzzer Control	FFh	00h	40h	LED 状态控制	04h	闪烁周期控制

P2 LED 状态控制

LED_0，LED_1 and Buzzer Control 的结构（1 个字节）

CMD	项	说明
Bit 0	最终的 LED_1 状态	1 = 开；0 = 关

CMD	项	说明
Bit 1	最终的 LED_1 状态	1 = 开；0 = 关
Bit 2	LED_1 状态掩码	1 = 更新其状态 0 = 不更改
Bit 3	LED_0 状态掩码	1 = 更新其状态 0 = 不更改
Bit 4	LED_1 初始闪烁状态	1 = 开；0 = 关
Bit 5	LED_1 初始闪烁状态	1 = 开；0 = 关
Bit 6	LED_1 闪烁掩码	1 = 闪烁 0 = 不闪烁
Bit 7	LED_0 闪烁掩码	1 = 闪烁 0 = 不闪烁

命令数据域　　闪烁周期控制。

LED_0，LED_1 Blinking Duration Control 的结构（4 个字节）

字节 0	字节 1	字节 2	字节 3
T1 周期 初始闪烁状态 （单位 = 100 ms）	T2 周期 切换闪烁状态 （单位 = 100 ms）	重复次数	蜂鸣器响应

其中：

字节 3　　蜂鸣器响应。在 LED 闪烁期间控制蜂鸣器的状态。

00h：蜂鸣器不开启。

01h：蜂鸣器在 T1 周期内开启。

02h：蜂鸣器在 T2 周期内开启。

03h：蜂鸣器在 T1 和 T2 周期内开启。

响应数据域　　SW1 SW2。读卡器返回的状态码。

状态码

结果	SW1	SW2	含义
成功	90	LED 当前状态	操作成功完成。
错误	63	00h	操作失败。

LED 当前状态（1 个字节）

状态	项	说明
Bit 0	当前的 LED_1	1 = 开；0 = 关
Bit 1	当前的 LED_0	1 = 开；0 = 关
Bits 2 – 7	保留	

注：

1. *LED 状态操作是在 LED 闪烁操作之后进行的。*
2. *如果相应的 LED 状态掩码未启用，则 LED 状态不会发生改变。*
3. *如果相应的 LED 闪烁掩码未启用，则 LED 不会闪烁。同时，重复次数的值必须大于 0。*
4. *T1 和 T2 周期参数主要用于控制 LED 闪烁的工作周期和蜂鸣器的鸣响时间。*
5. *比如说，如果 T1=1，T2=1，则工作周期 = 50%。工作周期 = T1/(T1 + T2)。*
6. *如果只想控制蜂鸣器，则将 P2 “LED 状态控制” 置为 0 即可。*
7. *要想使蜂鸣器工作，“重复次数”必须大于 0。*
8. *如果只想控制 LED，则将参数"蜂鸣器响应"置为 0 即可。*

例 1：读取当前 LED 的状态。

// 假设 LED_0 和 LED_1 最初都是关闭状态 //

// 无蜂鸣器响应 //

APDU = “FF 00 40 00 04 00 00 00 00”

响应 = “90 00”。LED_0 和 LED_1 均为关闭状态。

例 2：开启 LED_0 和 LED_1。

// 假设 LED_0 和 LED_1 最初都是关闭状态 //

// 无蜂鸣器响应 //

APDU = “FF 00 40 0F 04 00 00 00 00”

响应 = “90 03”。LED_0 和 LED_1 均为开启状态。

将 LED_0 和 LED_1 都关闭，APDU = “FF 00 40 0C 04 00 00 00 00”

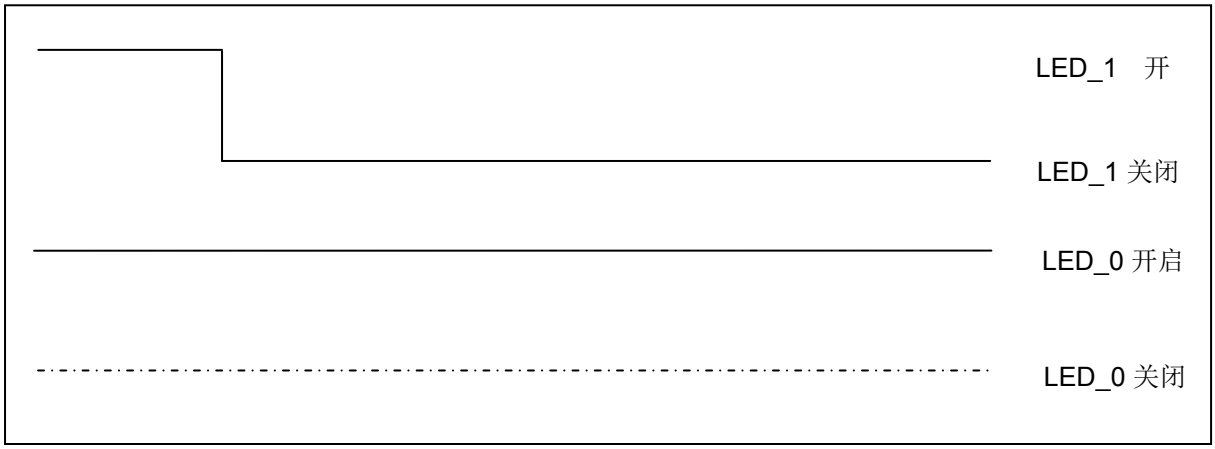
例 3：仅关闭 LED_1，保持 LED_0 现状。

// 假设 LED_0 和 LED_1 最初都是开启状态 //

// 无蜂鸣器响应 //

APDU = “FF 00 40 04 04 00 00 00 00”

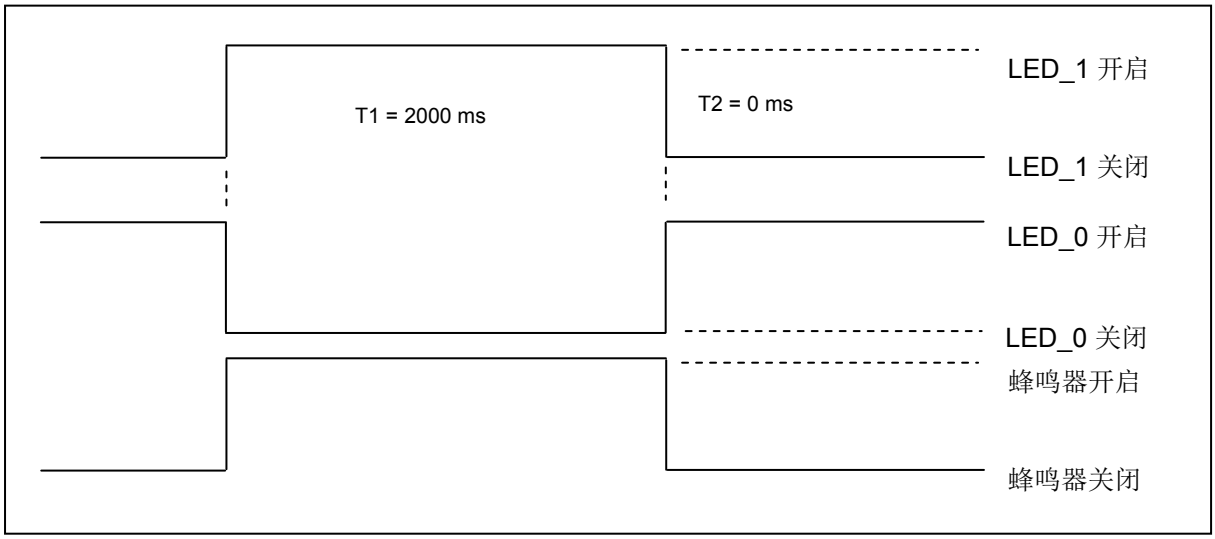
响应 = “90 02”。LED_0 保持不变（开启）；LED_1 关闭。



例 4：将 LED_1 开启 2 秒钟，之后返回到初始状态。

// 假设 LED_1 最初是关闭的，而 LED_0 最初是开启的。//

// 在 T1 周期内，LED_1 和蜂鸣器会开启，而 LED_0 会关闭。//



1 Hz = 1000 ms 时间间隔 = 500 ms 开启 + 500 ms 关闭

T1 周期 = 2000 ms = 14h

T2 周期 = 0 ms = 00h

重复次数 = 01h

蜂鸣器响应 = 01h

APDU = “FF 00 40 50 04 14 00 01 01”

响应 = “90 02”

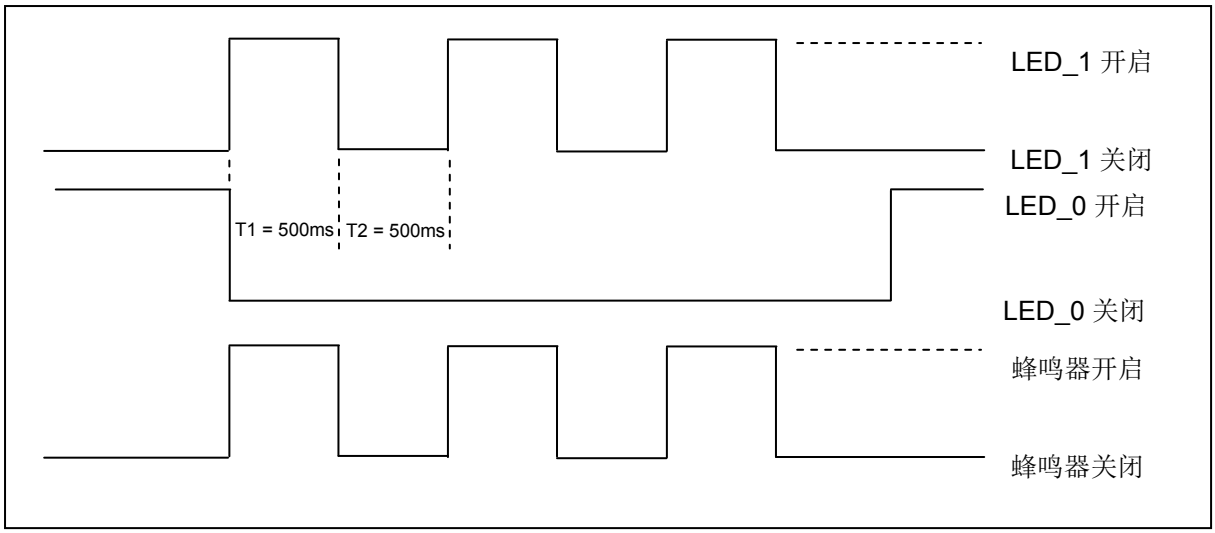
例 5：使 LED_1 闪烁 3 次，每次 1 Hz，之后返回到初始状态。

// 假设 LED_1 最初是关闭的，而 LED_0 最初是开启的。//

// LED_1 最初的闪烁状态是开启的。只有 LED_1 会闪烁。

// 蜂鸣器会在 T1 周期内开启；而 LED_0 会在 T1 和 T2 周期内关闭。

// 闪烁过后，LED_0 会开启。LED_1 会在闪烁后回到初始状态 //



1 Hz = 1000 ms 时间间隔 = 500 ms 开启 + 500 ms 关闭

T1 周期 = 500 ms = 05h

T2 周期 = 500 ms = 05h

重复次数 = 03h

蜂鸣器响应 = 01h

APDU = “FF 00 40 50 04 05 05 03 01”

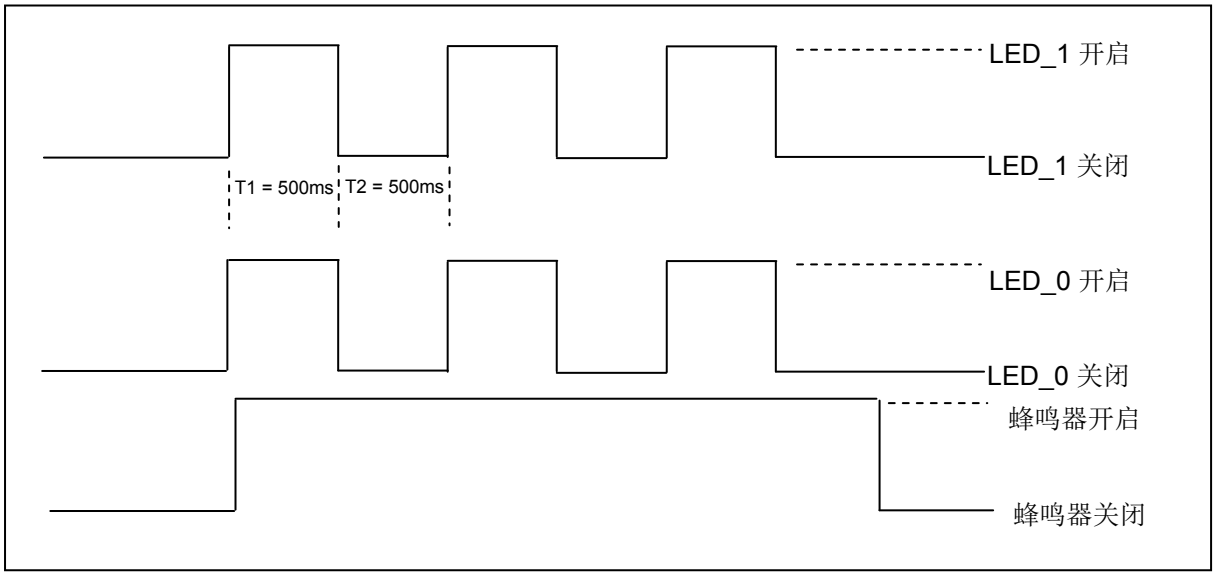
响应 = “90 02”

例 6：使 LED_0 和 LED_1 闪烁 3 次，每次 1 Hz。

// 假设 LED_0 和 LED_1 最初都是关闭状态 ////

// LED_0 和 LED_1 的初始闪烁状态都是开启的 //

// 蜂鸣器在 T1 和 T2 周期内都是开启的//



1 Hz = 1000 ms 时间间隔 = 500 ms 开启 + 500 ms 关闭

T1 周期 = 500 ms = 05

T2 周期 = 500 ms = 05

重复次数 = 03

蜂鸣器响应 = 03

APDU = “FF 00 40 F0 04 05 05 03 03”

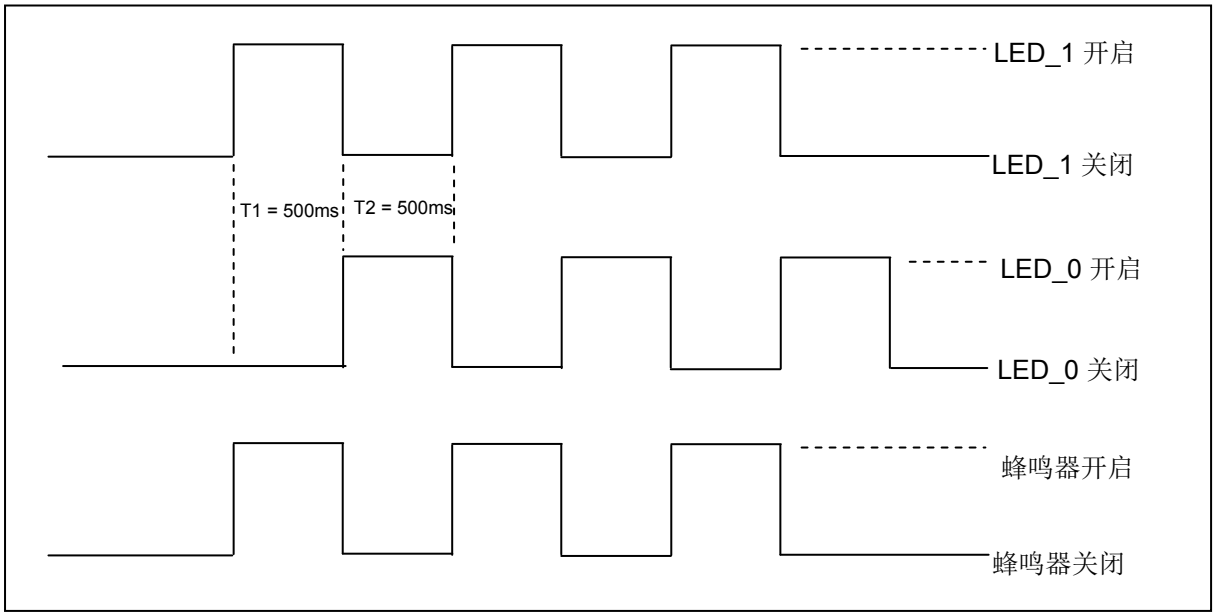
响应 = “90 00”

例 7：使 LED_0 和 LED_1 依次闪烁 3 次，每次 1 Hz。

// 假设 LED_0 和 LED_1 最初都是关闭的。//

// LED_1 的初始闪烁状态是开启的；LED_0 的初始闪烁状态是关闭的 //

// 蜂鸣器会在 T1 周期内开启//



1 Hz = 1000 ms 时间间隔 = 500 ms 开启 + 500 ms 关闭

T1 周期 = 500 ms = 05h

T2 周期 = 500 ms = 05h

重复次数 = 03h

蜂鸣器响应 = 01h

APDU = “FF 00 40 00 04 05 05 03 01”

响应 = “90 00”

6.16. 蜂鸣器控制（Buzzer Control）

此命令用于控制蜂鸣器。

Buzzer Control 的命令结构（5 个字节）

命令	CLA	INS	P1	P2	Lc	命令数据域（3 个字节）
Buzzer Control	FFh	00h	42h	00h	03h	蜂鸣器控制

命令数据域　　蜂鸣器控制。

蜂鸣器开启/关闭周期控制的结构（4 个字节）

字节 0	字节 1	字节 2
T1 周期 开启状态 （单位 = 100 ms）	T2 周期 关闭状态 （单位 = 100 ms）	重复次数

响应数据域　　SW1 SW2。

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。

6.17. ISO 14443-4 A 类和 B 类标签的基本流程

典型的操作顺序为：

1. 用正确的参数（A 类或 B 类）扫描天线场内的标签（轮询）。
2. 更改波特率（此选项仅对 A 类标签可选）。
3. 执行任意 T=CL 命令。
4. 取消选择标签。

步骤 1. ISO 14443-4 A 类标签的轮询，106 kbps

HOST -> 02 6F 09 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock 结构)

HOST -> FF 00 00 00 04 D4 4A 01 00 [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 15 00 00 00 00 01 01 00 00

RDR -> D5 4B 01 01 00 08 28 04 85 82 2F A0 07 77 F7 80 02 47 65 90 00　　[校验和] 03

其中，查找到的标签数量 = [01];　　目标编号 = 01

SENS_RES = 00 08;　　SEL_RES = 28，

UID 长度 = 4;　　UID = 85 82 2F A0

ATS = 07 77 F7 80 02 47 65

操作完成 = 90 00

OR

步骤 2. ISO 14443-4 B 类标签的轮询，106 kbps

HOST -> 02 6F 0A 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock 结构)

HOST -> FF 00 00 00 05 D4 4A 01 03 00 [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 14 00 00 00 00 01 01 00 00

RDR -> D5 4B 01 01 50 00 01 32 F4 00 00 00 00 33 81 81 01 21 90 00 [校验和] 03

其中，查找到的标签数量 = [01]; 目标编号 = 01

ATQB = 50 00 01 32 3B 00 00 00 00 33 81 81。

ATTRIB_RES 长度 = 01; ATTRIB_RES = 21

操作完成 = 90 00

步骤 3. 将波特率由默认值该为其他值（可选）。

HOST -> 02 6F 0A 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock 结构)

HOST -> FF 00 00 00 05 D4 4E 01 02 02 [校验和] 03 // 波特率更改成 424 kbps

OR

HOST -> FF 00 00 00 05 D4 4E 01 01 01 [校验和] 03 // 波特率更改成 212 kbps

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 4F [00] 90 00 [校验和] 03

注：*请检查标签支持的最大波特率。仅支持 A 类标签。*

步骤 3. 执行 T=CL 命令， APDU 随机取数 = 00 84 00 00 08。

HOST -> 02 6F 0D 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock 结构)

HOST -> FF 00 00 00 08 D4 40 01 00 84 00 00 08 [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 0F 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 62 89 99 ED C0 57 69 2B 90 00 90 00 [校验和] 03

其中，响应数据 = 62 89 99 ED C0 57 69 2B 90 00

步骤 4. 取消选择标签。

HOST -> 02 6F 08 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock 结构)

HOST -> FF 00 00 00 03 D4 44 01 [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 90 00 [校验和] 03

步骤 5. 关闭天线电源（可选）。

HOST -> 02 6F 09 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock 结构)

HOST -> FF 00 00 00 04 D4 32 01 00

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 04 00 00 00 00 01 01 00 00

RDR -> D5 33 90 00 [校验和] 03

注：*更多详情请参阅标签标准。*

6.18. MIFARE 应用的基本流程

典型的操作顺序为：

1. 扫描天线场内的标签（轮询）。
2. 认证。
3. 读/写标签的存储内容。
4. 终止标签（可选）。

步骤 1. 轮询 MIFARE 1K/4K 标签，106 kbps

<< 02 6F 09 00 00 00 00 01 00 00 00

FF 00 00 00 04 D4 4A 01 00 [校验和] 03

>> 02 00 00 03

>> 02 80 0E 00 00 00 00 01 01 00 00

D5 4B 01 01 00 02 18 04 F6 8E 2A 99 90 00 [校验和] 03

其中，查找到的标签数量 = [01]; 目标编号 = 01

SENS_RES = 00 02; SEL_RES = 18，

UID 长度 = 4; UID = F6 8E 2A 99

操作完成 = 90 00

注：*可通过识别 SEL_RES 来鉴定标签类型。*

几种常见标签类型的 SEL_RES

00 = MIFARE Ultralight

08 = MIFARE 1K

09 = MIFARE Mini

18 = MIFARE 4K

20 = MIFARE DESFire

28 = JCOP30

98 = Gemplus MPCOS

步骤 2. 认证密钥 A，Block **04**，KEY = FF FF FF FF FF FF，UID = F6 8E 2A 99。

<< 02 6F 14 00 00 00 00 00 01 00 00 00

FF 00 00 00 0F D4 40 01 **60** **04** FF FF FF FF FF FF F6 8E 2A 99 [校验和] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 01 00 00

D5 41 [00] 90 00 [校验和] 03

***注：**如果认证失败，显示错误码 [XX]。*

[00] = 有效，其他 = 错误。更多详情请参阅错误码表。

认证密钥 B 时：

<< 02 6F 14 00 00 00 00 00 01 00 00 00

FF 00 00 00 0F D4 40 01 **61** **04** FF FF FF FF FF FF F6 8E 2A 99 [校验和] 03

步骤 3. 读取 Block **04** 的内容。

<< 02 6F 0A 00 00 00 00 01 00 00 00

FF 00 00 00 05 D4 40 01 **30** **04** [校验和] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 01 00 00

D5 41 [00] 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 90 00

[校验和] 03

其中，值块数据 = 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16

步骤 4. 更新 Block **04** 的内容。

<< 02 6F 1A 00 00 00 00 01 00 00 00

FF 00 00 00 15 D4 40 01 **A0** **04** 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D

0E 0F 10 [校验和] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 00 00 00

D5 41 [00] 90 00 [校验和] 03

步骤 5. 终止标签（可选）。

<< 02 6F 08 00 00 00 00 01 00 00 00

FF 00 00 00 03 D4 44 01 [校验和] 03

>> 02 00 00 03

>> 02 80 05 00 00 00 00 01 01 00 00

D5 45 [00] 90 00 [校验和] 03

6.18.1. 处理 MIFARE 1K/4K 标签的值块

值块有电子钱包的功能，例如增值，减值，恢复，传输等。值块的固定数据结构使值块可进行差错检验，差错校正和备份管理。

字节号	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
说明	值				值				值				地址	地址	地址	地址

其中：

值 表示一个有符号的 4 字节值。其中值的最低有效字节存储在最低地址字节内。取反的字节以标准 2 的补码格式保存。

地址 表示一个 1 字节地址，可用于保存一个块的存储地址。（可选）

例如：

值 100（十进制） = 64 (十六进制)，假设 Block = 05h

格式化的值块 = 64 00 00 00 9B FF FF FF 64 00 00 00 05 FA 05 FA

步骤 1. **用值 100（十进制）更新值块 05** 的内容。

<< 02 6F 1A 00 00 00 00 01 00 00 00
FF 00 00 00 15 D4 40 01 **A0 05** 64 00 00 00 9B FF FF FF 64 00 00 00 05
FA 05 FA [校验和] 03
>> 02 00 00 03
>> 02 80 05 00 00 00 00 01 00 00 00
D5 41 [00] 90 00 [校验和] 03

步骤 2. // 使值块 **05** 的值增加 **1（十进制）。**

<< 02 6F 0E 00 00 00 00 01 00 00 00
FF 00 00 00 09 D4 40 01 **C1 05** 01 00 00 00 [校验和] 03
>> 02 00 00 03
>> 02 80 05 00 00 00 00 01 00 00 00
D5 41 [00] 90 00 [校验和] 03

注：** // 使值块 **05** 的值减少 **1（十进制）。

<< 02 6F 0E 00 00 00 00 01 00 00 00
FF 00 00 00 09 D4 40 01 **C0 05** 01 00 00 00 [校验和] 03

步骤 3. 传输值块 **05 先前计算的值（十进制）**.

<< 02 6F 0A 00 00 00 00 01 00 00 00
FF 00 00 00 05 D4 40 01 **B0 05** [校验和] 03
>> 02 00 00 03
>> 02 80 05 00 00 00 00 01 00 00 00
D5 41 [00] 90 00 [校验和] 03

***注：** 恢复值块 05 的值（取消先前的增值或减值操作）。*

<< 02 6F 0A 00 00 00 00 01 00 00 00
FF 00 00 00 05 D4 40 01 **C2 05** [校验和] 03

步骤 4. 读取值块 05 的内容。
<< 02 6F 0A 00 00 00 00 01 00 00 00
FF 00 00 00 05 D4 40 01 **30 05** [校验和] 03
>> 02 00 00 03
>> 02 80 15 00 00 00 00 01 00 00 00
D5 41 [00] 65 00 00 00 9A FF FF FF 65 00 00 00 05 FA 05 FA 90 00 [校验和] 03
其中，值 = 101 （十进制）

步骤 5. 复制值块 **05** 的值到值块 **06**（十进制）。
<< 02 6F 0A 00 00 00 00 01 00 00 00
FF 00 00 00 05 D4 40 01 **C2 05** [校验和] 03
>> 02 00 00 03
>> 02 80 05 00 00 00 00 01 00 00 00
D5 41 [00] 90 00 [校验和] 03
<< 02 6F 0A 00 00 00 00 01 00 00 00
FF 00 00 00 05 D4 40 01 **B0 06** [校验和] 03
>> 02 00 00 03
>> 02 80 05 00 00 00 00 01 00 00 00
D5 41 [00] 90 00 [校验和] 03

步骤 6. 读取值块 **06** 的内容。
<< 02 6F 0A 00 00 00 00 01 00 00 00
FF 00 00 00 05 D4 40 01 **30 06** [校验和] 03
>> 02 00 00 03
>> 02 80 15 00 00 00 00 01 00 00 00
D5 41 [00] 65 00 00 00 9A FF FF FF 65 00 00 00 05 FA 05 FA 90 00 [校验和] 03
其中，值 = 101 （十进制）地址“05 FA 05 FA”表明该值是从值块 05 复制的。

注：*更多详情请参阅 MIFARE 标准。*

6.18.2. 访问 MIFARE Ultralight 标签

典型的操作顺序为：

1. 扫描工作场内的标签（轮询）。
2. 读/写标签的存储内容。
3. 终止标签（可选）。

步骤 1. MIFARE Ultralight 标签的轮询，106 kbps。
HOST -> 02 6F 09 00 00 00 00 01 00 00 00
HOST -> FF 00 00 00 04 D4 4A 01 00 [校验和] 03
RDR -> 02 00 00 03 (等待标签)
RDR -> 02 80 11 00 00 00 00 01 01 00 00
RDR -> D5 4B 01 01 00 44 00 07 04 6E 0C A1 BF 02 84 90 00 [校验和] 03

其中，查找到的标签数量 = [01]; 目标编号 = 01

SENS_RES = 00 44;　　SEL_RES = 00，
UID 长度 = 7;　　UID = 04 6E 0C A1 BF 02 84
操作完成 = 90 00

步骤 2. 读取页 04 的内容。
HOST -> 02 6F 0A 00 00 00 00 01 00 00 00
HOST -> FF 00 00 00 05 D4 40 01 **30** **04** [校验和] 03
RDR -> 02 00 00 03 (等待标签)
RDR -> 02 80 15 00 00 00 00 01 01 00 00
RDR -> D5 41 [00] 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 90 00 [校验和] 03

其中，值块数据 = 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16

注：*检索 4 个连续页。//检索页 4、5、6 和 7。每个数据页包括 4 个字节。*

步骤 3. 用数据"AA BB CC DD"更新页 **04** 的内容。
HOST -> 02 6F 0E 00 00 00 00 01 00 00 00
HOST -> FF 00 00 00 09 D4 40 01 **A2** **04** AA BB CC DD [校验和] 03
RDR -> 02 00 00 03 (等待标签)
RDR -> 02 80 05 00 00 00 00 01 01 00 00
RDR -> D5 41 [00] 90 00 [校验和] 03

OR

步骤 3. 用数据"AA BB CC DD"写（MIFARE 兼容的写功能）页 **04 的内容。**
HOST -> 02 6F 1A 00 00 00 00 01 00 00 00
HOST -> FF 00 00 00 15 D4 40 01 **A0** **04** AA BB CC DD 00 00 00 00 00 00 00 00 00 00 00 00 [校验和] 03
RDR -> 02 00 00 03 (等待标签)
RDR -> 02 80 05 00 00 00 00 01 01 00 00
RDR -> D5 41 [00] 90 00 [校验和] 03

注：*此命令用于适应已建立的 MIFARE 1K/4K 卡的基础结构。数据须组装成一个 16 字节的帧。前 4 个字节是数据，其余字节（12 个 0）是填充。虽然发送给读写器的是 16 个字节，但是只更新页 4（4 个字节）。*

步骤 4. 再次读取页 04 的内容。
HOST -> 02 6F 0A 00 00 00 00 01 00 00 00
HOST -> FF 00 00 00 05 D4 40 01 **30** **04** [校验和] 03
RDR -> 02 00 00 03 (等待标签)
RDR -> 02 80 15 00 00 00 00 01 01 00 00
RDR -> D5 41 [00] AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16 90 00 [校验和] 03

其中，值块数据 = AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16

注：*只更新页 4。页 5，6，7 保持不变。*

步骤 5. 终止标签（可选）。
HOST -> 02 6F 08 00 00 00 00 01 00 00 00
HOST -> FF 00 00 00 03 D4 44 01 [校验和] 03
RDR -> 02 00 00 03 (等待标签)
RDR -> 02 80 05 00 00 00 00 01 01 00 00
RDR -> D5 45 [00] 90 00 [校验和] 03

注：*更多详情请参阅 MIFARE Ultralight 标准。*

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8： MIFARE Ultralight 卡的内存结构

6.18.3. 访问 MIFARE Ultralight C 标签

典型的操作顺序为：

1. 扫描工作场内的标签（轮询）。
2. 认证。
3. 读/写标签的存储内容。
4. 终止标签（可选）。

步骤 1. 轮询 MIFARE Ultralight C 标签，106 kbps

HOST -> 02 6F 09 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 04 D4 4A 01 00 [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 11 00 00 00 00 01 01 00 00

RDR -> D5 4B 01 01 00 44 00 07 04 6E 0C A1 BF 02 84 90 00 [校验和] 03

其中，查找到的标签数量 = [01]; 目标编号 = 01

SENS_RES = 00 44; SEL_RES = 00，

UID 长度 = 7; UID = 04 6E 0C A1 BF 02 84

操作完成 = 90 00

步骤 2. 3DES 认证。

HOST -> 02 6F 09 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 04 D4 42 1A 00 10 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 0E 00 00 00 00 01 01 00 00

RDR -> D5 43 [00] 04 77 64 89 99 74 24 67 90 00 [校验和] 03

其中，卡片的 3DES 随机数 = [04 77 64 89 99 74 24 67];

操作完成 = 90 00

HOST -> 02 6F 18 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 13 D4 42 AF 88 68 45 07 65 86 99 67 00 53 77 56 98 65 49 67 [校验和] 03

其中，卡片接收的 3DES 应答 = [88 68 45 07 65 86 99 67 00 53 77 56 98 65 49 67];

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 0E 00 00 00 00 01 01 00 00

RDR -> D5 43 [00] 00 06 78 53 80 68 89 61 24 90 00 [校验和] 03

其中，卡片发送的 3DES 应答 = [06 78 53 80 68 89 61 24];

操作完成 = 90 00

注：*须检查卡片发送的 3DES 应答以保证卡片合法。*

步骤 3. 读取页 04 的内容。

HOST -> 02 6F 09 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 05 D4 40 01 **30 04** [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 15 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 90 00 [校验和] 03

其中，值块数据 = 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16

注：*4 个连续页将被检索。//检索页 4、5、6 和 7。每个数据页包括 4 个字节。*

步骤 4. 用数据"AA BB CC DD"更新页 **04** 的内容。

HOST -> 02 6F 0E 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 09 D4 40 01 **A2** **04** AA BB CC DD [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 90 00 [校验和] 03

OR

步骤 4. 用数据"AA BB CC DD"写（MIFARE 兼容的写功能）页 **04** 的内容。

HOST -> 02 6F 1A 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 15 D4 40 01 **A0** **04** AA BB CC DD 00 00 00 00 00 00 00 00 00 00 00 00 [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] 90 00 [校验和] 03

注：*此命令用于适应已建立的 MIFARE 1K/4K 卡的基础结构。数据须组装成一个 16 字节的帧。前 4 个字节是数据，其余字节（12 个 0）是填充。虽然发送给读写器的是 16 个字节，但是只更新页 4（4 个字节）。*

步骤 5. 再次读取页 **04** 的内容。

HOST -> 02 6F 0A 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 05 D4 40 01 **30** **04** [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 15 00 00 00 00 01 01 00 00

RDR -> D5 41 [00] AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16 90 00 [校验和] 03

其中，值块数据 = AA BB CC DD 05 06 07 08 09 10 11 12 13 14 15 16

注：*只更新页 4。页 5，6，7 保持不变。*

步骤 6. 终止标签（可选）。

HOST -> 02 6F 08 00 00 00 00 01 00 00 00

HOST -> FF 00 00 00 03 D4 44 01 [校验和] 03

RDR -> 02 00 00 03 (等待标签)

RDR -> 02 80 05 00 00 00 00 01 01 00 00

RDR -> D5 45 [00] 90 00 [校验和] 03

***注：**更多详情请参阅 MIFARE Ultralight C 标准。*

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	内部	锁	锁	2
OTP	OTP0	OTP1	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
数据读/写	Data48	Data49	Data50	Data51	16
数据读/写	Data52	Data53	Data54	Data55	17
数据读/写	Data56	Data57	Data58	Data59	18
数据读/写	Data60	Data61	Data62	Data63	19
数据读/写	Data64	Data65	Data66	Data67	20
数据读/写	Data68	Data69	Data70	Data71	21
数据读/写	Data72	Data73	Data74	Data75	22
数据读/写	Data76	Data77	Data78	Data79	23
数据读/写	Data80	Data81	Data82	Data83	24
数据读/写	Data84	Data85	Data86	Data87	25
数据读/写	Data88	Data89	Data90	Data91	26

字节号	0	1	2	3	页
数据读/写	Data92	Data93	Data94	Data95	27
数据读/写	Data96	Data97	Data98	Data99	28
数据读/写	Data100	Data101	Data102	Data103	29
数据读/写	Data104	Data105	Data106	Data107	30
数据读/写	Data108	Data109	Data110	Data111	31
数据读/写	Data112	Data113	Data114	Data115	32
数据读/写	Data116	Data117	Data118	Data119	33
数据读/写	Data120	Data121	Data122	Data123	34
数据读/写	Data124	Data125	Data126	Data127	35
数据读/写	Data128	Data129	Data130	Data131	36
数据读/写	Data132	Data133	Data134	Data135	37
数据读/写	Data136	Data137	Data138	Data139	38
数据读/写	Data140	Data141	Data142	Data143	39
锁	锁	锁	-	-	40
16 位计数器	16 位计数器	16 位计数器	-	-	41
认证配置	认证配置	认证配置	认证配置	认证配置	42
认证配置	认证配置	认证配置	认证配置	认证配置	43
认证密钥	认证密钥	认证密钥	认证密钥	认证密钥	44
认证密钥	认证密钥	认证密钥	认证密钥	认证密钥	45
认证密钥	认证密钥	认证密钥	认证密钥	认证密钥	46
认证密钥	认证密钥	认证密钥	认证密钥	认证密钥	47

表9：MIFARE Ultralight C 卡的内存结构

页总大小：198 字节的 792 位

6.19. FeliCa 应用的基本流程

步骤 0. 启动应用程序。首先要激活“SAM 接口”。返回 SAM 的 ATR（如果插入了 SAM）或者一个私有 ATR“3B 00”（加入没有插入 SAM）。换句话说，从应用的角度看，SAM 总是存在。

步骤 1. 修改 PN531 的操作参数。设置重试次数为 1。

步骤 2. 发送 “Direct Transmit”和“Get Response”APDUs，以轮询 FeliCa 标签（标签轮询）。

步骤 3. 如果没有发现标签，返回步骤 2，直到发现一个 FeliCa 标签。

步骤 4. 发送一个 APDU（读或写标签）访问 FeliCa 标签。

步骤 5. 如果不对 FeliCa 标签执行任何操作，则返回步骤 2 轮询其他 FeliCa 标签。

..

步骤 N. 取消激活“SAM 接口”。关闭应用程序。

注：

1. *标签命令"InListPassiveTarget" 的默认重试次数是无限次。发送 APDU"FF 00 00 00 06 D4 32 05 00 00 00"以修改重试次数为 1。*
2. *如果不用访问非接触标签，建议关闭天线。*
   用以开启天线电源的 APDU = APDU "FF 00 00 00 04 D4 32 01 03"
   用以关闭天线电源的 APDU = APDU "FF 00 00 00 04 D4 32 01 02"

6.20. NFC 论坛 Type 1 标签应用的基本流程

例如：Jewel 和 Topaz 标签

典型的操作顺序为：

1. 扫描工作场内的标签（轮询）。
2. 读取/更新标签的存储内容
3. 取消选择标签。

步骤 1. Jewel 或 Topaz 标签的轮询，106 kbps
HOST -> 02 6F 09 00 00 00 00 01 00 00 00 (HOST_to_RDR_XfrBlock 结构)
HOST -> FF 00 00 00 04 D4 4A 01 04 [校验和] 03
RDR -> 02 00 00 03 (等待标签)
RDR -> 02 80 0C 00 00 00 00 01 01 00 00
RDR -> D5 4B 01 01 0C 00 B5 3E 21 00 90 00 [校验和] 03

其中，查找到的标签数量 = [01]; 目标编号 = 01
ATQA_RES = 0C 00; UID = B5 3E 21 00
操作完成 = 90 00

步骤 2. 读存储地址 08（Block 1：Byte-0).
HOST -> 02 6F 0A 00 00 00 00 01 00 00 00 FF 00 00 00 05 D4 40 01 01 08 [校验和] 03
RDR -> 02 00 00 03 02 80 06 00 00 00 00 01 01 00 00 D5 41 [00] 18 90 00 [校验和] 03

其中，响应数据 = 18

注：*从存储地址 00 开始读取标签的所有存储内容。*

HOST -> 02 6F 09 00 00 00 00 01 00 00 00 FF 00 00 00 04 D4 40 01 00 [校验和] 03
RDR -> 02 00 00 03 02 80 7F 00 00 00 00 01 01 00 00 D5 41 00 11 48
RDR -> show all data … 90 00 [校验和] 03

步骤 3. 更新存储地址 08（Block 1：Byte-0）更新为数据 FF。
HOST -> 2 6F 0B 00 00 00 00 01 00 00 00 FF 00 00 00 06 D4 40 01 53 08 FF [校验和] 03
RDR -> 02 00 00 03 02 80 05 00 00 00 00 01 01 00 00 D5 41 [00] FF 90 00 [校验和] 03
其中，响应数据 = FF

注：*从存储地址 08 开始更新标签的一个以上的存储内容(Block 1：Byte-0)。*

HOST -> 02 6F 0D 00 00 00 00 01 00 00 00 FF 00 00 00 08 D4 40 01 58 08 02 AA BB [校验和] 03
RDR -> 02 00 00 03 02 80 06 00 00 00 00 01 01 00 00 D5 41 [00] 90 00 [校验和] 03

其中， 命令 = 58; 起始存储地址 = 08;
写内容的数量 = 02; 存储内容 = AA，BB;

步骤 4. 取消选择标签。
HOST -> 02 6F 08 00 00 00 00 01 00 00 00 FF 00 00 00 03 D4 44 01 [校验和] 03
RDR -> 02 00 00 03 02 80 05 00 00 00 00 01 01 00 00 D5 45 [00] 90 00 [校验和] 03

附录A. ACR122 错误代码

错误代码	错误
00h	没有错误。
01h	超时，目标无应答。
02h	非接触 UART 检测到 CRC 错误。
03h	非接触 UART 检测到奇偶校验错误。
04h	在 MIFARE 防冲突/选择操作中，检测到错误的位计数。
05h	MIFARE 卡操作过程中出现帧错误。
06h	以 106 kbps 速率进行逐位补防冲突的过程中检测到异常的位冲突。
07h	通信缓冲区的大小不足。
08h	非接触 UART 检测到 RF 缓冲区溢出（寄存器 CL_ERROR 的 BufferOvfl 位）。
0Ah	在主动通信模式下，对应方没有及时开启 RF 磁场（定义见 NFCIP-1 标准）。
0Bh	RF 协议错误（cf. 参考[4]，CL_ERROR 寄存器的说明）。
0Dh	温度错误：内部温度传感器检测到过热，因此自动关闭了天线的驱动。
0Eh	内部缓冲区溢出
10h	参数无效（范围，格式等）
12h	DEP 协议：在目标模式下配置的芯片不支持从发起者收到的命令（收到的命令不是下列之一：ATR_REQ，WUP_REQ，PSL_REQ，DEP_REQ，DSL_REQ，RLS_REQ，ref. [1]).
13h	DEP 协议/MIFARE/ISO/IEC 14443-4：数据格式不符合规范。根据采用的 RF 协议，可能是： RF 接收帧的长度错误 PCB 或 PFB 值不正确 无效的或意外的 RF 接收帧 NAD 或 DID 不一致。
14h	MIFARE：认证错误。
23h	ISO/IEC 14443-3：UID 检查字节错误。
25h	DEP 协议：无效的设备状态，系统所处的状态不允许执行该操作。
26h	在此配置下不允许执行操作（主机控制器接口）。
27h	当前的芯片状态导致命令不能被接收（发起方 vs.目标，未知的目标号，目标状态不佳，等等）。
29h	配置为目标的芯片是由其发起方发布的。
2Ah	仅限 ISO/IEC 14443-3B：卡片的 ID 号不匹配，意味着预期的卡片已经被调换。
2Bh	仅限 ISO/IEC 14443-3B：先前激活的卡片消失了。
2Ch	NFCID3 发起方和 NFCID3 目标方在 DEP 212/424 kbps 被动模式下不匹配。
2Dh	检测到过流事件。

错误代码	错误
2Eh	DEP 结构中缺少 NAD。

表10 ： ACR122 错误代码