Advanced Card Systems Ltd.
Card & Reader Technologies

ACR89U-A2
手持式
智能卡读写器

参考手册 V1.01

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

目录

图目录

表目录

1.0. 简介

ACR89U-A2 手持式智能卡读写器强大而有效，属于双界面智能卡读写器，支持 NFC 标签，带便携式键盘，支持符合 ISO 7816 标准的 MCU 卡，符合 ISO 14443 标准的 A 类和 B 类非接触卡，MIFARE 卡，FeliCa 卡和符合 ISO 18092 的 NFC 标签。连机模式下，适用于办公环境；脱机模式下，适用于现场环境。

连机模式下，ACR89U-A2 是计算机和卡片之间的中间设备。读写器会执行计算机发送的命令，专门与非接触式标签、MCU 卡、SAM 卡及外围设备进行通信。

本文描述了如何用 ACR89 软件编程接口控制 ACR89 多功能智能卡读写器的内置附件。内置附件包括嵌入 ACR89 的键盘，LCD 屏幕，LED，蜂鸣器和实时时钟。这类组件不受智能卡读写器库控制。此外，本文描述了如何用 PC/SC APDU 命令操作设备的非接触式标签。

2.0. 硬件设计

2.1. 体系结构

ACR89U-A2 的库架构如下图所示：

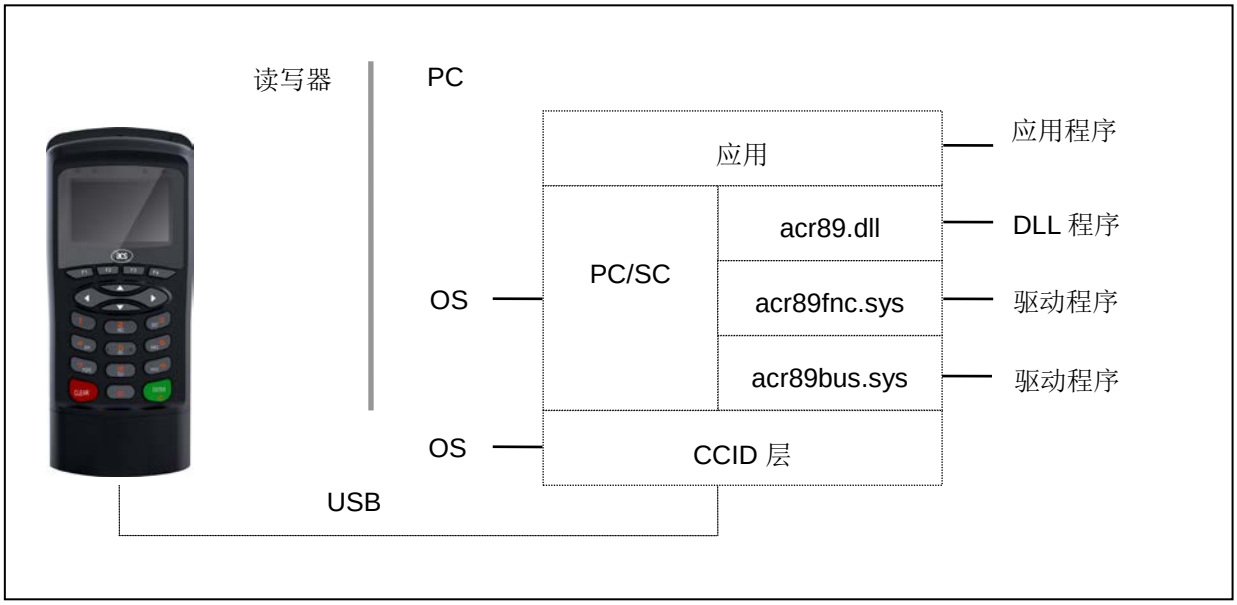

图1：ACR89U-A2 架构

2.2. USB 接口

ACR89U-A2 通过标准 USB 接口与计算机建立连接。

2.3. 通信参数

ACR89U-A2 通过 USB（如《USB 规格书 2.0》所述）连接计算机，支持 USB 全速模式（12 Mbps）。

引脚	信号	功能
1	V_{BUS}	为读写器提供+5 V 的电源
2	D-	ACR89U-A2 和 PC 间以微分信号传输数据
3	D+	ACR89U-A2 和 PC 间以微分信号传输数据
4	GND	参考电压等级

表1：USB 接口配线

注：*为了使 ACR89U-A2 通过 USB 接口正常运行，应该先安装设备驱动程序。*

2.4. 端点

ACR89U-A2 通过以下端点与主机进行通信：

Control Endpoint	设置和控制
Bulk OUT	主机发送至 ACR89U-A2 的命令（数据包为 64 字节）
Bulk IN	ACR89U-A2 发送至主机的命令（数据包为 64 字节）
Interrupt IN	ACR89U-A2 发送至主机的卡片状态报文（数据包为 8 字节）

2.5. 接触式智能卡接口

ACR89U-A2 与智能卡之间的界面符合 ISO 7816-3 标准协议，并进行了某些限制或提升来增强 ACR89U-A2 的实用功能。

2.5.1. 智能卡电源 VCC（C1）

智能卡电流消耗不得大于 50 mA。

2.5.2. 卡片类型选择

激活插入的卡片之前，主控计算机需要向 ACR89U-A2 发送适当的命令来选择卡片类型。

如果 MCU 卡同时支持 T=0 和 T=1，读写器可通过协议与参数选择（PPS）为 MCU 卡选择 T=0 或 T=1 中作为首选协议。如果 MCU 卡仅支持 T=0 或 T=1，则读卡器会自动采用该协议类型，而不管应用程序选择哪一种。

2.5.3. CPU 卡接口

CPU 卡只使用触点 C1（VCC）、C2（RST)、C3（CLK）、C5（GND）和 C7（I/O）。时钟信号（C3）的频率为 4 MHz。

2.6. 非接触式智能卡接口

ACR89U-A2 与非接触卡之间的界面遵循 ISO14443 标准，并进行了某些限制或提升来增强 ACR89U-A2 的实用功能。

2.6.1. 载波频率

ACR89U-A2 的载波频率为 13.56 MHz。

2.6.2. 卡片轮询

ACR89U-A2 会自动检测进入工作场内的非接触标签。此功能支持 ISO 14443-4 的 A 类卡和 B 类卡，Mifare 卡，FeliCa 卡以及 NFC 标签。

3.0. ACR89 的 USB 通信协议

ACR89 通过 USB 连接主机（连机模式下）。现在的行业内规范 — CCID 标准，已经为 USB 芯片-智能卡接口设备定义了与此相关的协议。CCID 涵盖了操作智能卡和 PIN 所需的全部协议。然而，它并没有定义操作 ACR89 其他外设特性的协议。ACR89 的通信协议须遵循 CCID 标准并可扩展支持读写器其他特性。

3.1. 设备配置

ACR89 的 USB 端点配置和使用符合 CCID 标准第 3 部分的规定。概述如下：

1. 控制命令通过控制通道（缺省通道）发送，其中包括类特定请求和 USB 标准请求。由缺省通道发送的命令会通过缺省通道向主机反馈信息。
2. CCID 事件通过中断通道发送。
3. CCID 命令经由 BULK-OUT 端点发出。发送至 ACR89 的每个命令都有一个相关的最终响应，一些命令也有过程响应。
4. CCID 响应经由 BULK-IN 端点发出。所有发送至 ACR89 的命令都必须同步发送（即：对于 ACR89 来说，*bMaxCCIDBusySlots* 等于 1）。

ACR89 支持的 CCID 功能由其类别描述符定义：

偏移	字段	大小	值	说明
0	*bLength*	1	36h	描述符的字节数
1	*bDescriptorType*	1	21h	CCID 功能描述符的类别
2	*bcdCCID*	2	0100h	CCID 以二进制编码的十进制指定版本号码
4	*bMaxSlotIndex*	1	04h	5 个插槽
5	*bVoltageSupport*	1	07h	支持 1.8V、3.0V 和 5.0V 的槽位电压
6	*dwProtocols*	4	00000003h	支持 T=0 和 T=1 协议
10	*dwDefaultClock*	4	000012C0h	默认 ICC 时钟频率为 4.8 MHz
14	*dwMaximumClock*	4	000012C0h	ICC 支持的最大时钟频率为 4.8 MHz
18	*bNumClockSupported*	1	00h	不支持手动设置时钟频率
19	*dwDataRate*	4	003267h	默认 ICC I/O 数据传输速率为 12,903 bps
23	*dwMaxDataRate*	4	00032673h	ICC I/O 的最大数据传输速率为 206,451 bps
27	*bNumDataRatesSupported*	1	00h	不支持手动设置数据传输速率
28	*dwMaxIFSD*	4	00000FEh	T=1 协议下，最大 IFSD 是 254
32	*dwSynchProtocols*	4	00000000h	不支持同步卡
36	*dwMechanical*	4	00000000h	不支持特殊机械特性

偏移	字段	大小	值	说明
40	*dwFeatures*	4	000204B2h	ACR89 有以下特性： - 根据 ATR 数据自动配置参数 -根据参数自动改变 ICC 时钟频率 -根据频率和 FI、DI 参数自动改变波特率 -根据当前参数自动进行 PPS - 自动 IFSD - 与 ACR89 进行短 APDU 级交换
44	*dwMaxCCIDMessageLength*	4	00000110h	最大信息长度为 272 字节
48	*bClassGetResponse*	1	FFh	Get Response 命令中 APDU 回应级别
49	*bClassEnvelope*	1	FFh	无意义（短 APDU 级交换）
50	*wLCDLayout*	2	0815h	LCD，8 行 x 21 个字符
52	*bPINSupport*	1	03h	支持 PIN 码验证和修改
53	*bMaxCCIDBusySlots*	1	01h	同一时间内只能有 1 个槽位处于工作状态

表2：ACR89 支持 CCID 功能类别描述符

注：*CCID 标准采用小端格式。*

3.2. CCID 类特定请求

基于 ACR89 的命令消息结构标准,ACR89 通过 USB 与计算机进行通讯。ACR89 支持通过控制通道发送的 CCID 类特定请求。

3.2.1. 命令汇总

停止当前运行的任意命令，使 ACR89 可执行新命令：

bmRequestType	bRequest	wValue	wIndex	wLength	数据
00100001b	ABORT (01h)	bSeq, bSlot	接口	0000h	无

3.3. CCID 命令通道 Bulk-OUT 消息

ACR89 遵循 CCID 协议第四部分有关 Bulk-OUT 消息的规定。CCID 还定义了操作附加功能的扩展命令。此节列举了 ACR89 支持的 CCID 类 Bulk-OUT 消息。**小节 3.5** 将介绍扩展命令。

3.3.1. 命令汇总

3.3.1.1. PC_to_RDR_IccPowerOn

激活插槽并返回卡片的 ATR。

偏移	字段	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	-	标识命令的插槽号。
5	*bSeq*	1	-	命令的序号。
6	*bPowerSelect*	1	-	ICC 上的电压值： 00h = 自动电压选择 01h = 5 V 02h = 3 V 03h = 1.8 V
7	*abRFU*	2	-	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）。

3.3.1.2. PC_to_RDR_IccPowerOff

取消激活插槽。

偏移	字段	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	-	标识命令的插槽号
6	*bSeq*	1	-	命令的序号
7	*abRFU*	3	-	保留为将来使用

此消息的响应是 RDR_to_PC_SlotStatus 消息。

3.3.1.3. PC_to_RDR_GetSlotStatus

获取当前的插槽状态。

偏移	字段	大小	值	说明
0	*bMessageType*	1	65h	-
1	*dwLength*	4	00000000h	此消息的额外字节的大小

偏移	字段	大小	值	说明
5	*bSlot*	1	-	标识命令的插槽号
6	*bSeq*	1	-	命令的序号
7	*abRFU*	3	-	保留为将来使用

此消息的响应是 RDR_to_PC_SlotStatus 消息。

3.3.1.4. PC_to_RDR_XfrBlock

向 ICC 传输数据块。

偏移	字段	大小	值	说明
0	*bMessageType*	1	6Fh	-
1	*dwLength*	4	-	此消息的 abData 字段的大小
5	*bSlot*	1	-	标识命令的插槽号
6	*bSeq*	1	-	命令的序号
7	*bBWI*	1	-	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。
8	*wLevelParameter*	2	0000h	RFU（TPDU 交换级别）
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。

3.3.1.5. PC_to_RDR_GetParameters

获取插槽参数。

偏移	字段	大小	值	说明
0	*bMessageType*	1	6Ch	-
1	*dwLength*	4	00000000h	此消息的额外字节的大小
5	*bSlot*	1	-	标识命令的插槽号
6	*bSeq*	1	-	命令的序号
7	*abRFU*	3	-	保留为将来使用

此消息的响应是 RDR_to_PC_Parameters 消息。

3.3.1.6. PC_to_RDR_ResetParameters

重置插槽参数为默认值。

偏移	字段	大小	值	说明
0	bMessageType	1	6Dh	-
1	dwLength	4	00000000h	此消息的额外字节的大小
5	bSlot	1	-	标识命令的插槽号
6	bSeq	1	-	命令的序号
7	abRFU	3	-	保留为将来使用

此消息的响应是 RDR_to_PC_Parameters 消息。

3.3.1.7. PC_to_RDR_SetParameters

设置插槽参数。

偏移	字段	大小	值	说明
0	*bMessageType*	1	61h	-
1	*dwLength*	4	-	此消息的额外字节的大小
5	*bSlot*	1	-	标识命令的插槽号
6	*bSeq*	1	-	命令的序号
7	*bProtocolNum*	1	-	指定采用的协议数据结构。 00h = T=0 协议结构 01h = T=1 协议结构 以下值保留为将来使用： 80h：2 线协议结构 81h：3 线协议结构 82h：I2C 协议结构
8	*abRFU*	2	-	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	字段	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 B3-0 – DI - ISO/IEC 7816-3:1997 中表 8 的索引，选择一个波特率转换因子

偏移	字段	大小	值	说明
11	*bmTCCKST0*	1	-	B0 – 0b, B7-2 – 000000b 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	-	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 等同于 00h。
13	*bWaitingIntegerT0*	1	-	T=0 时 WI 用于定义 WWT
14	*bClockStop*	1	-	支持 ICC 时钟停止 00h = 不允许停止时钟 01h = 时钟信号为低时停止 02h = 时钟信号为高时停止 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	字段	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 B3-0 – DI - ISO/IEC 7816-3:1997 中表 8 的索引，选择一个波特率转换因子
11	*BmTCCKST1*	1	-	B7-2 – 000100b B0 – 校验和的类型（b0=0：LRC； b0=1：CRC） 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	-	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	-	B7-4 = BWI 值 0-9 有效 B3-0 = CWI 值 0-Fh 有效
14	*bClockStop*	1	-	支持 ICC 时钟停止 00h = 不允许停止时钟 01h = 时钟信号为低时停止 02h = 时钟信号为高时停止 03h = 时钟信号为高或为低时停止
15	*bIFSC*	1	-	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。

3.3.1.8. PC_to_RDR_Escape

使能小节 **3.5** 定义的 ACR89 的扩展特性。

偏移	字段	大小	值	说明
0	*bMessageType*	1	6Bh	-
1	*DwLength*	4	-	此消息的 abData 字段的大小
5	*Bslot*	1	-	标识命令的插槽号
6	*Bseq*	1	-	命令的序号
7	*AbRFU*	3	-	保留为将来使用
10	*AbData*	字节型数组	-	小节 **3.5.2 定义的命令**

此消息的响应是 RDR_to_PC_Escape 消息。

此消息能返回以下 ACR89 特有的错误。错误的更多信息包含在扩展响应里。

bmICCStatus	bmCommand Status	bError	说明
3	1	ACR89_ERROR	ACR89 特有错误。参见 ACR89 响应的 wReturnCode。
3	1	INVALID_MODE	ACR89 当前运行的模式不支持该命令。
3	1	DEVICE_VOID	ACR89 未初始化。

3.3.1.9. PC_to_RDR_Secure (RFU)

保留为将来所用。

用户可用此命令在智能卡上直接验证或修改 PIN 码。

偏移	字段	大小	值	说明
0	*bMessageType*	1	69h	-
1	*DwLength*	4	-	此消息的额外字节的大小
5	*BSlot*	1	-	标识命令的插槽号
6	*BSeq*	1	-	命令的序号
7	*BBWI*	1	-	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。此参数仅用于字符级转换。
8	*wLevelParameter*	2	0000h	RFU（TPDU 交换级别）

偏移	字段	大小	值	说明
10	*bPINOperation*	1	-	用于表示 PIN 操作： 00h：PIN 码校验 01h：PIN 码更改 02h:从安全的 CCID 设备传输 PIN 码缓存 03h：等待 ICC 响应 04h:取消 PIN 功能 05h:重新发送上一个 I-Block，只有 T=1 协议在用时有意义。 06h:发送 APDU 的下一部分，只有 T=1 协议在用时有意义。
11	*abPINDataStructure*	字节型数组	-	PIN 码验证数据结构或 PIN 码更改数据结构

此消息的响应是 RDR_to_PC_DataBlock 消息。

注：*关于 PIN 码验证/更改数据结构的更多详情，请参考 CCID 协议的 4.1.11 部分。*

3.3.1.10. PC_to_RDR_Abort

与控制通道中止的请求联用，通知 CCID 停止在特定插槽进行的传输，使插槽回到可以接受新命令通道 Bulk-OUT 消息的状态。

偏移	字段	大小	值	说明
0	*bMessageType*	1	72h	-
1	*DwLength*	4	00000000h	此消息的额外字节的大小
5	*BSlot*	1	-	标识命令的插槽号
6	*BSeq*	1	-	命令的序号
7	*AbRFU*	3	000000h	RTF

此消息的响应是 RDR_to_PC_SlotStatus 消息。

3.4. CCID 命令通道 Bulk-IN 消息

Bulk-IN 消息用于响应 Bulk-OUT 消息。ACR89 遵循 CCID 协议第 4 部分有关 Bulk-IN 消息的规定。此节列举了 ACR89 支持的 CCID 类 Bulk-IN 消息。

3.4.1. 消息汇总

3.4.1.1. RDR_to_PC_DataBlock

此消息是 ACR89 对 PC_to_RDR_IccPowerOn、PC_to_RDR_XfrBlock 和 PC_to_RDR_Secure 消息的响应。

偏移	字段	大小	值	说明
0	*bMessageType*	1	80h	表示 CCID 正在发送一个数据块.
1	*dwLength*	4	-	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	-	小节 **3.7** 定义了插槽状态和错误寄存器
8	*bError*	1	-	小节 **3.7** 定义了插槽状态和错误寄存器
9	*bChainParameter*	1	00h	RFU（TPDU 交换级别）
10	*AbData*	字节型数组	-	本数据域包含由 CCID 返还的数据

3.4.1.2. RDR_to_PC_SlotStatus

此消息是 ACR89 对 PC_to_RDR_IccPowerOff、PC_to_RDR_GetSlotStatus 和 PC_to_RDR_Abort 消息，以及类特定 ABORT 请求的响应。

偏移	字段	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	-	小节 **3.7** 定义了插槽状态和错误寄存器
8	*bError*	1	-	小节 **3.7** 定义了插槽状态和错误寄存器
9	*bClockStatus*	1	-	值： 00h = 时钟运行 01h = 时钟停于 L 状态 02h = 时钟停于 H 状态 03h = 时钟停于未知状态 所有其他值保留为将来使用

3.4.1.3. RDR_to_PC_Parameters

此消息是 ACR89 对 PC_to_RDR_GetParameters、PC_to_RDR_ResetParameters 和 PC_to_RDR_SetParameters 消息的响应。

偏移	字段	大小	值	说明
0	*bMessageType*	1	82h	-
1	*dwLength*	4	-	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	-	小节 **3.7** 定义了插槽状态和错误寄存器
8	*bError*	1	-	小节 **3.7** 定义了插槽状态和错误寄存器
9	*bProtocolNum*	1	-	指定采用的协议数据结构。 00h：T=0 协议的结构 01h：T=1 协议的结构 以下值保留为将来使用： 80h：2 线协议结构 81h：3 线协议结构 82h：I2C 协议结构
10	*abProtocolDataStructure*	字节型数组	-	协议数据结构如 CCID 5.2.3 节汇总

3.4.1.4. RDR_to_PC_Escape

此消息是 ACR89 对 PC_to_RDR_Escape 消息的响应。

偏移	字段	大小	值	说明
0	*bMessageType*	1	83h	-
1	*dwLength*	4	-	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	-	小节 **3.7** 定义了插槽状态和错误寄存器
8	*bError*	1	-	小节 **3.7** 定义了插槽状态和错误寄存器
9	*bRFU*	1	00h	RFU
10	*abData*	字节型数组	-	ACR89 对不同的扩展命令有不同的响应数据，详情如**小节 3.5.4** 所述

3.5. ACR89 兼容的扩展命令通道消息

本节描述了 CCID 不涵盖，但 ACR89 兼容的扩展命令，用于操作附加功能。这些命令总是在 PC_to_RDR_Escape Bulk-OUT 消息下执行，并响应 RDR_to_PC_Escape Bulk-IN 消息。

PC 请求消息	代码	ACR89 响应消息	代码
PC_to_ACR89_InputKey	12h	ACR89_to_PC_DataBlock	81h
PC_to_ACR89_SetCursor	18h	ACR89_to_PC_DisplayStatus	83h
PC_to_ACR89_SetBacklight	19h	ACR89_to_PC_DisplayStatus	83h
PC_to_ACR89_DisplayMessage	1bh	ACR89_to_PC_DisplayStatus	83h
PC_to_ACR89_DisplayRowGraphic	23h	ACR89_to_PC_DisplayStatus	83h
PC_to_ACR89_SetContrast	1ch	ACR89_to_PC_DisplayStatus	83h
PC_to_ACR89_ClearDisplay	1dh	ACR89_to_PC_DisplayStatus	83h
PC_to_ACR89_ReadRTC	08h	ACR89_to_PC_TimeStamp	84h
PC_to_ACR89_SetRTC	09h	ACR89_to_PC_TimeStamp	84h
PC_to_ACR89_Buzzer	0ah	ACR89_to_PC_Echo	90h
PC_to_ACR89_AccessEeprom	21h	ACR89_to_PC_Datablock	81h
PC_to_ACR89_SetLED	22h	ACR89_to_PC_Echo	90h
PC_to_ACR89_EraseSPIFlash	30h	ACR89_to_PC_ExMemStatus	b0h
PC_to_ACR89_ProgramSPIFlash	33h	ACR89_to_PC_MemoryStatus	b0h
PC_to_ACR89GetSPIFlash	34h	ACR89_to_PC_MemoryPage	b1h
PC_to_ACR89_GetVersion	36h	ACR89_to_PC_VersionInfo	b2h
PC_to_ACR89_AuthoInfo	38h	ACR89_to_PC_AuthInfo	b4h

表3：ACR89 扩展命令通道消息

3.5.1. 扩展命令通道 Bulk-OUT 消息

本节定义的命令结构 abData 字段用于填充 PC_to_RDR_Escape 消息。

类似于 CCID 消息结构，该命令结构包括固定长度的命令头和长度可变的命令数据部分。命令头固定是 5 个字节。

与 CCID/USB 不同，扩展命令部分采用大端格式。

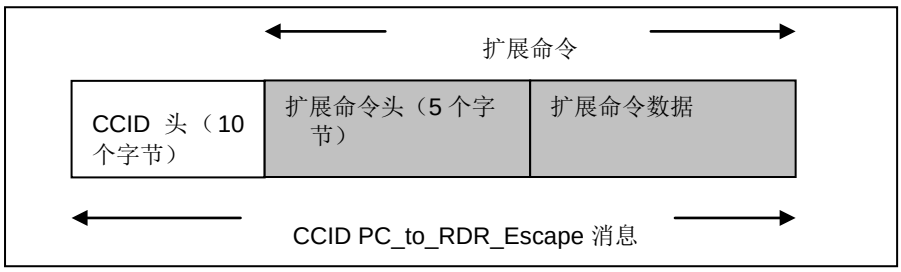

图2：CCID PC_to_RDR_Escape 消息

3.5.2. 命令详情

3.5.2.1. PC_to_ACR89_InputKey

接收用键盘输入数据。此命令的应用场景与插槽无关。

偏移	字段	类型	大小	值	说明
10	*BCmdCode*	16 进制	1	12h	-
11	*wCmdLength*	16 进制	2	0002h	命令数据大小（大端格式）
13	*AbRfu*	16 进制	2	0000h	-
15	*bKeyInputMode*	2 进制	1	-	B0 – 输入模式（b0=0 表示单按键输入, b0=1 表示按键串输入）。按键串输入模式下，按回车键表示完成输入。 B1 – 键盘模式（b1=0 表示数字输入, b1=1 表示字母数字输入） B3 到 b2 – 按键显示（b2=0 表示不显示按键，b2=1 表示显示按键。b2=1 时， b3=0 表示按键显示为明文，b3=1 表示按键显示为星号*。） B4 – 按键输入超时控制（b4=0 表示打开超时设置，b4=1 表示关闭超时设置） B5 – 安全按键传输（b5=0 表示明文传输，b5=1 表示密文传输）。此位保留为将来所用。 B6 – 0/1 – 关闭/开启控制按键 其它 – RFU
16	*bTimeoutValue*	16 进制	1	-	按键输入超时时间的单位是秒。仅在 bKeyInputMode 字段的按键输入超时控制位设置为 0 时有效。

此消息的响应是 ACR89_to_PC_DataBlock 消息。

3.5.2.2. PC_to_ACR89_SetCursor

设置 LCD 光标的新位置。此命令的应用场景与插槽无关。

偏移	字段	类型	大小	值	说明
10	*BcmdCode*	16 进制	1	18h	-
11	*wCmdLength*	16 进制	2	0002h	命令数据大小（大端格式）
13	*AbRfu*	16 进制	2	0000	保留为将来使用
15	*bRowPosition*	16 进制	1	00h 至 07h	新的光标行位置

偏移	字段	类型	大小	值	说明
16	*bColumnPosition*	16 进制	1	00h 至 7Fh	新的光标列位置

此消息的响应是 ACR89_to_PC_DisplayStatus 消息。

3.5.2.3. PC_to_ACR89_SetBacklight

配置 LCD 显示。此命令的应用场景与插槽无关。

偏移	字段	类型	大小	值	说明
10	*BCmdCode*	16 进制	1	19h	-
11	*wCmdLength*	16 进制	2	0001h	命令数据大小（大端格式）
13	*AbRfu*	16 进制	2	0000	保留为将来使用
15	*BBacklight*	16 进制	1	00h 或 01h	00h - 背光关 00h - 背光开 其他值保留为将来所用

此消息的响应是 ACR89_to_PC_DisplayStatus 消息。

3.5.2.4. PC_to_ACR89_DisplayMessage

用 ACR89 内置字体库里的字体从当前光标位置起横向显示字符串。ACR89 将根据字符位置和字符大小自动计算绝对坐标，相应移动光标。此命令的应用场景与插槽有关。

偏移	字段	类型	大小	值	说明
10	*BCmdCode*	16 进制	1	1Bh	-
11	*wCmdLength*	16 进制	2	可变	命令数据大小（大端格式）
13	*AbRfu*	16 进制	2	0000h	保留为将来使用
15	*bCharCoding*	16 进制	1	-	abData 字段的数据编码结构。字符大小取决于数据结构： 00h – ASCII (每个字符占 1 行 x 6 列) 所有其他值保留为将来使用
16	*AbData*	ASCII	字节型数组	-	bCharCoding 字段指出了字符串编码结构

此消息的响应是 ACR89_to_PC_DisplayStatus 消息。

3.5.2.5. PC_to_ACR89_DisplayRowGraphic

扫描 LCD 上待显示的一行图形。

偏移	字段	类型	大小	值	说明
10	*bCmdCode*	16 进制	1	23h	-
11	*wCmdLength*	16 进制	2	可变	命令数据大小（大端格式）

偏移	字段	类型	大小	值	说明
13	*abRfu*	16 进制	2	0000h	-
15	*bRowPosition*	16 进制	1	-	起始位置的行索引。行高 8 像素。
16	*bColumnPosition*	16 进制	1	-	起始位置的列索引。
17	*AbData*	16 进制	可变	-	待显示图形的行位图数据

wCmdLength 与 bColumnPosition 的总和不得超过 LCD 列数（128）。

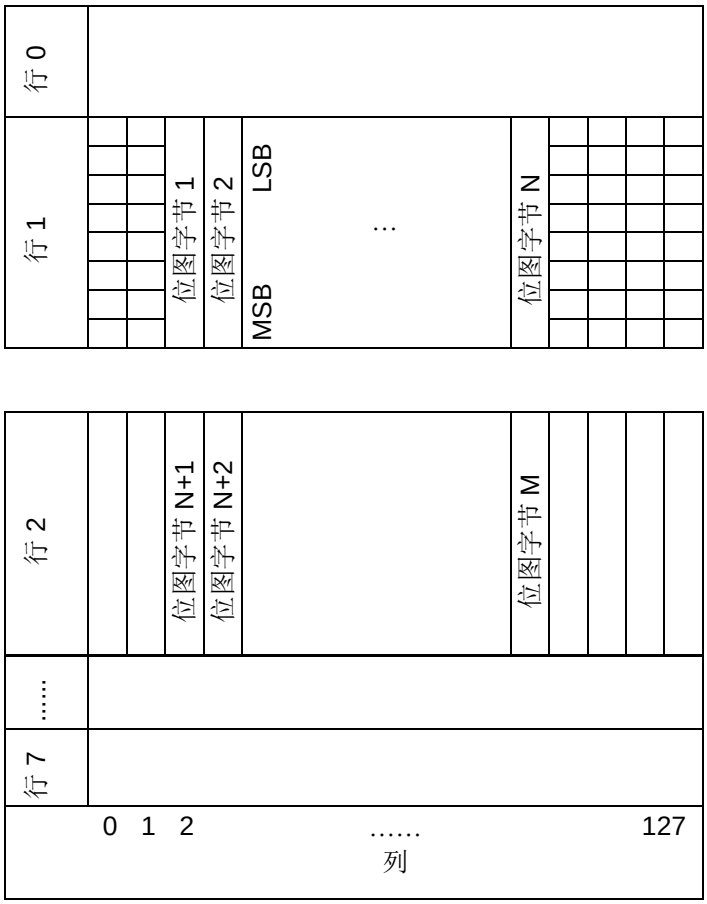

图3： PC_to_ACR89_DisplayGraphic – 位图结构

此消息的响应是 ACR89_to_PC_DisplayStatus 消息。

3.5.2.6. PC_to_ACR89_SetContrast

设置 LCD 对比度级别。此命令的应用场景与插槽无关。

偏移	字段	类型	大小	值	说明
10	*BCmdCode*	16 进制	1	1Ch	-
11	*wCmdLength*	16 进制	2	0001h	命令数据大小（大端格式）
13	*abRfu*	16 进制	2	0000	保留为将来使用
15	*bContrastLevel*	16 进制	1	00h 至 0x63h	LCD 的新对比度级别

对比度的区间范围是 00h-63h。值越大，明暗对比越强烈。值越小，明暗对比越柔和。整个 LCD 显示屏和显示的图像也影响对比度级别。

3.5.2.7. PC_to_ACR89_ClearDisplay

清除一行或多行 LCD 显示内容。执行该命令后，光标将移动到所清除块的起始位置。此命令的应用场景与插槽无关。

偏移	字段	类型	大小	值	说明
10	*BcmdCode*	16 进制	1	1Dh	-
11	*wCmdLength*	16 进制	2	0002h	命令数据大小（大端格式）
13	*AbRfu*	16 进制	2	0000h	保留为将来使用
15	*bClearMode*	16 进制	1	00h 或 01h 或 02h	00h = 清除屏幕上的所有内容 01h = 清除光标当前所在的行 02h = 清除光标当前所在行中光标后面的列 所有其他值保留为将来所用
16	*bNumber*	-	1	-	对于 bClearMode = 01h – 待清除的行数 对于 bClearMode = 02h – 待清除的列数 其他情况下无意义

此消息的响应是 ACR89_to_PC_DisplayStatus 消息。

3.5.2.8. PC_to_ACR89_ReadRTC

读取内置实时时钟当前的值。实时时钟每隔半秒钟更新一下时间值。此命令的应用场景与插槽无关。

偏移	字段	类型	大小	值	说明
10	*BCmdCode*	16 进制	1	08h	-
11	*wCmdLength*	16 进制	2	0000h	命令数据大小（大端格式）
13	*AbRFU*	16 进制	2	0000h	-

此消息的响应是 ACR89_to_PC_TimeStamp 消息。

3.5.2.9. PC_to_ACR89_SetRTC

给内置实时时钟设置特定时间值。此命令的应用场景与插槽无关。

偏移	字段	类型	大小	值	说明
10	*BCmdCode*	16 进制	1	09h	-
11	*wCmdLength*	16 进制	2	0006h	命令数据大小（大端格式）
13	*AbRFU*	16 进制	2	0000	-

偏移	字段	类型	大小	值	说明
15	*bRTCValue*	BCD	6	-	实时时钟的新值格式 YY，MM，DD，HH，MI 和 SS。

此消息的响应是 ACR89_to_PC_TimeStamp 消息。

3.5.2.10. PC_to_ACR89_Buzzer

偏移	字段	类型	大小	值	说明
10	*BCmdCode*	16 进制	1	0Ah	-
11	*wCmdLength*	16 进制	2	0002h	命令数据大小（大端格式）
13	*abRfu*	16 进制	2	0000h	-
15	*bBuzzerState*	16 进制	1	--	01h – 蜂鸣器开 00h - 蜂鸣器关
16	*BbuzzerOnDuration*	16 进制	1	-	蜂鸣器开启时间以百分之一毫秒为单位。仅字段 bBuzzerState 为 01h 时有效。 00h – 激活蜂鸣器并且不关闭 其他值 - 激活蜂鸣器，待其运行所设置数目的百分之一毫秒后，关闭蜂鸣器

此命令消息的响应是 ACR89_to_PC_Echo 消息。

3.5.2.11. PC_to_ACR89_AccessEeprom

允许用户从 EEPROM 读写数据。最大数据长度是 249 字节。

偏移	字段	类型	大小	值	说明
10	*bCmdCode*	16 进制	1	21h	-
11	*wCmdLength*	16 进制	2	可变	命令数据大小（大端格式）
13	*AbRFU*	16 进制	2	0000h	-
15	*bAccessMode*	ASCII	1	-	‘W’ – 写 EEPROM ‘R’ – 读 EEPROM
16	*BDeviceNumber*	16 进制	1	-	00 – 从 EEPROM 01- 中文字体 EEPROM (Rfu)
17	*AbAddress*	16 进制	4	-	EEPROM 地址 （大端格式）
21	*wDataLength*	16 进制	2	可变	数据长度（读/写）（大端格式）
23	*bEeprom Data*	16 进制	可变	-	EEPROM 数据

此消息的响应是 ACR89_to_PC_DataBlock 消息。

3.5.2.12. PC_to_ACR89_SetLED

允许用户打开和关闭智能卡读写器的电源，插槽 1，和插槽 2，开关状态用红/绿色的 LED 灯表示。

偏移	字段	类型	大小	值	说明
10	*BcmdCode*	16 进制	1	22h	-
11	*WcmdLength*	16 进制	2	0003h	命令数据大小（大端格式）
13	*AbRFU*	16 进制	2	0000h	-
15	*Power LED*	16 进制	1	-	Bit0 :1- 选择红色 Bit1 :1- 选择绿色 Bit2 :1- 选择黄色 Bit7 :0-关/1-开 例：开启红色 10000001b 关闭绿色 00000010b 忽略 xxxx0000b
16	*插槽 1 LED*	16 进制	1	-	Bit0 :1- 选择红色 Bit1 :1- 选择绿色 Bit2 :1- 选择黄色 Bit7 :0-关/1-开
17	*插槽 2 LED*	16 进制	1	-	Bit0 :1- 选择红色 Bit1 :1- 选择绿色 Bit2 :1- 选择黄色 Bit7 :0-关/1-开

此命令消息的响应是 ACR89_to_PC_Echo 消息。

3.5.2.13. PC_to_ACR89_EraseSPIFlash

擦除闪存块。

偏移	字段	类型	大小	Value.	说明
10	*bCmdCode*	16 进制	1	30h	命令代码
11	*bFlashType*	16 进制	1	02h	SPI 闪存
12	*bRFU*	16 进制	1	00h	-
13	*bStartBlockNum*	16 进制	1	-	除 0 之外的任意小于 08h 的数，例如 01h （如果串行闪存采用默认值）
14	*bEndBlockNum*	16 进制	1	-	不小于 bStartBlockNum

此消息的响应是 ACR89_to_PC_ExMemStatus 消息。

注：*闪存块的当前是 64k 字节。*

3.5.2.14. PC_to_ACR89_ProgramSPIFlash

将 256 字节的数据写入 SPI 闪存页。

偏移	字段	类型	大小	值	说明
10	*bCmdCode*	16 进制	1	33h	命令代码
11	*AbAddress*	16 进制	4	xxxxxx00h	闪存页的起始地址 （小端格式）
15	*AbData*	16 进制	256	-	写入闪存页的数据
271	*bCheckSum*	16 进制	1	-	AbData 的校验和

此消息的响应是 ACR89_to_PC_ExMemStatus 消息。

3.5.2.15. PC_to_ACR89_GetSPIFlashPage

从 SPI 闪存页读取 256 字节的数据。

偏移	字段	类型	大小	值	说明
10	*bCmdCode*	16 进制	1	34h	命令代码
11	*AbAddress*	16 进制	4	xxxxxx00h	闪存页的起始地址 （小端格式）

此消息的响应是 ACR89_to_PC_MemoryPage 消息。

3.5.2.16. PC_to_ACR89_GetVersion

读取启动载入器或应用的固件版本信息。

偏移	字段	类型	大小	值	说明
10	*bCmdCode*	16 进制	1	36h	命令代码
11	*bVersionType*	16 进制	1	-	01h = 启动载入器的版本 02h = 应用版本
12	*AbRFU*	16 进制	3	000000h	-

此消息的响应是 ACR89_to_PC_VersionInfo 消息。

3.5.2.17. PC_to_ACR89_ AuthInfo

读取 RomID 和 RomData。

偏移	字段	类型	大小	值	说明
10	*bCmdCode*	16 进制	1	38h	命令代码
11	*AbRFU*	16 进制	16	00…00h	-

此消息的响应是 ACR89_to_PC_AuthInfo 消息。

3.5.3. 扩展命令通道 Bulk-IN 消息

本节描述了 CCID 不涵盖，但 ACR89 兼容的扩展命令（用于操作附加功能）的响应消息。对这些消息的响应总是遵循 CCID 协议 4.2.2.4 部分的 RDR_to_PC_Escape Bulk-IN 消息。

本节定义的响应结构是 abData 字段，用于填充 RDR_to_PC_Escape 消息。类似于 CCID 消息结构，该响应结构包括固定长度的响应头和长度可变的响应数据部分。响应头固定是 5 个字节。

与 CCID/USB 不同，扩展响应部分采用大端格式。

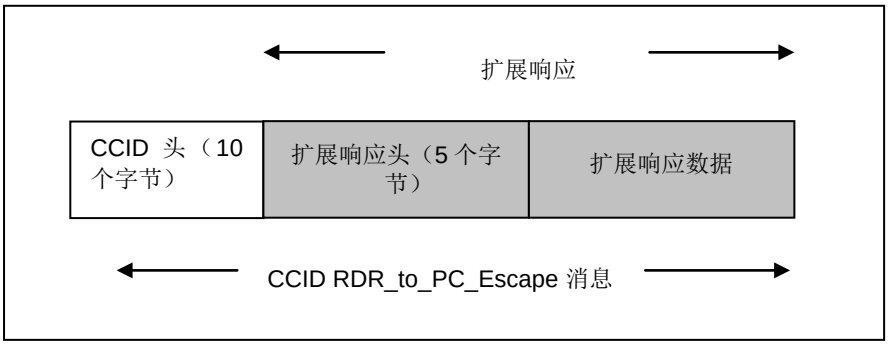

图4：CCID RDR_to_PC_Escape 消息

3.5.4. 消息详情

3.5.4.1. ACR89_to_PC_DataBlock

此消息是 ACR89 对 PC_to_ACR89_InputKey 命令的响应。

对于 PC_to_ACR89_InputKey 命令，将根据选择的按键输入模式返回从键盘获取的单个按键或按键串。

偏移	字段	大小	值	说明
10	*BrespType*	1	81h	-
11	*WReturnCode*	2	-	命令响应码（大端格式）
13	*WRespLength*	2	可变	响应数据大小（大端格式）
15	*Bdata*	可变	-	本字段包含由 ACR89 返还的数据

3.5.4.2. ACR89_to_PC_DisplayStatus

此消息是 ACR89 对以下命令的响应：

PC_to_ACR89_DisplaySetCursor

PC_to_ACR89_DisplayMessage

PC_to_ACR89_DisplayRowGraphic

PC_to_ACR89_ClearDisplay

偏移	字段	大小	值	说明
10	*BrespType*	1	83h	-
11	*wReturnCode*	2	-	命令响应码（大端格式）
13	*wRespLength*	2	0002h	响应数据大小（大端格式）
15	*bRowPosition*	1	00h 至 07h	光标的行位置
16	*bColumnPosition*	1	00h 至 7Fh	光标的列位置

3.5.4.3. ACR89_to_PC_TimeStamp

此消息是 ACR89 对 PC_to_ACR89_ReadRTC 和 PC_to_ACR89_SetRTC 命令的响应。

偏移	字段	大小	值	说明
10	*BRespType*	1	84h	-
11	*wReturnCode*	2	-	命令响应码（大端格式）
13	*wRespLength*	2	0006h	响应数据大小（大端格式）
15	*bTimeStamp*	6	-	实时时钟当前的值格式 YY，MM，DD，HH，MI 和 SS

3.5.4.4. ACR89_to_PC_Echo

此消息是 ACR89 对以下命令的响应：

PC_to_ACR89_Buzzer

PC_to_ACR89_SetLED

PC_to_ACR89_ExitScriptMode

偏移	字段	大小	值	说明
10	*bRespType*	1	90h	-
11	*wReturnCode*	2	90 00h	命令响应码，如果命令运行成功，会返回 90 00h （大端格式）
13	*wRespLength*	2	0000	响应数据大小（大端格式）

3.5.4.5. ACR89_to_PC_ExMemStatus

此消息是 ACR89 对以下命令的响应：

PC_to_ACR89_EraseSPIFlash

PC_to_ACR89_ProgramSPIFlash

偏移	字段	大小	值	说明
10	*bRespType*	1	B0h	-
11	*bReturnState*	1	-	命令返还状态（请参考后面的小节）
12	*bErrorCode*	1	-	错误码（请参考后面的小节）
13	*AbRFU*	2	0000h	-

3.5.4.6. ACR89_to_PC_MemoryPage

此消息是 ACR89 对 PC_to_ACR89_GetSPIFlashPage 命令的响应。

偏移	字段	大小	值	说明
10	*bRespType*	1	B1h	-
11	*bReturnState*	1	-	命令返还状态（请参考后面的小节）
12	*bErrorCode*	1	-	错误码（请参考后面的小节）
13	*AbRFU*	2	0000h	-
15	*AbData*	256	-	从闪存页读取的数据
271	*bCheckSum*	16 进制	1	AbData 的校验和

注：命令运行失败时，没有 AbData 和 bCheckSum。

3.5.4.7. ACR89_to_PC_VersionInfo

此消息是 ACR89 对 PC_to_ACR89_GetVersion 命令的响应。

偏移	字段	大小	值	说明
10	*bRespType*	1	B2h	-
11	*bReturnState*	1	-	命令返还状态（请参考后面的小节）
12	*bErrorCode*	1	-	错误码（请参考后面的小节）
13	*wInfoLength*	2	可变	bInfoData 的大小（小端格式）
15	*bInfoData*	可变	-	固件版本信息（ASCII）

***注**：无有效版本信息时，wInfoLength 的长度为 0。*

3.5.4.8. ACR89_to_PC_AuthInfo

此消息是 ACR89 对 PC_to_ACR89_AuthInfo 命令的响应。

偏移	字段	大小	值	说明
10	*bRespType*	1	B4h	-
11	*bReturnState*	1	-	命令返还状态（请参考后面的小节）
12	*bErrorCode*	1	-	错误码（请参考后面的小节）
13	*AbRFU*	2	0000h	-
15	*AbRomID*	8	-	唯一标识符
23	*AbRFU*	48	-	-

注：*命令运行失败时，没有偏移 15 的部分。*

3.5.5. 扩展命令响应码和 返还状态

下表汇总了 ACR89 CCID 扩展命令的响应码和返还状态。

响应码	值	说明
CMD_OKAY	9000h	命令执行成功
INVALID_PARAMETERS	FFFFh	扩展命令的参数错误
INVALID_COMMAND_CODE	FFFEh	扩展命令的命令码（偏移 10）无效
INVALID_COMMAND_LENGTH	FFFDh	扩展命令的长度错误
CANNOT_EXECUTE_COMMAND	FFFCh	扩展命令不能执行
TIMEOUT	FFFBh	执行扩展命令超时
SCRIPT_ERROR	FFFAh	脚本不能执行

表4： 扩展命令的响应码

返还状态	值	说明
CMD_OK	00h	命令执行成功
CMD_FAIL	01h	命令执行失败

表5： 扩展命令的返还状态

错误码	值	说明
COMMAND_NOT_SUPPORT	00h	不支持扩展命令的命令码（偏移 10）
HARDWARE_ERROR	01h	硬件错误
ACCESS_DENIED	02h	当前配置不允许的函数
ADDRESS_ERROR	03h	地址参数错误
FRAME_ERROR	04h	命令帧格式错误
CHECKSUM_ERROR	05h	数据部分的校验和错误

表6： 扩展命令的错误码

3.6. CCID Interrupt-IN 消息

Interrupt-IN 端点用于通知主机可能异步发生并且处于主机和 ACR89 的命令-响应交换之外的事件。ACR89 遵循 CCID 协议第四部分有关 Interrupt-IN 消息的规定。此节列举了 ACR89 支持的 CCID 类 Interrupt-IN 消息。

3.6.1. 消息汇总

3.6.1.1. RDR_to_PC_NotifySlotChange

ACR89 检测到 ICC 插槽状态变化时都会发送此消息。

偏移	字段	大小	值	说明
0	bMessageType	1	50h	-
1	bmSlotICCState	-	-	本字段报告字节粒度。大小(2 比特 * 插槽数）被向上舍入到最近的字节。每个插槽 2 比特。最低有效位表示插槽的当前状态（0b =无 ICC；1b = 有 ICC）最高有效位表示上一条 RDR_to_PC_NotifySlotChange 消息发出后，插槽的状态是否发生了变化（0b =未变化，1b =变化）。如果指定位置没有插槽，则该字段的这 2 个位返回 00b。 **例**：一个 3 插槽的 CCID 报告了格式如下的单字节： Bit 0 = 插槽 0 的当前状态 Bit 1 = 插槽 0 的变化状况 Bit 2 = 插槽 1 的当前状态 Bit 3 = 插槽 1 的变化状况 Bit 4 = 插槽 2 的当前状态 Bit 5 = 插槽 2 的变化状况 Bit 6 = 0b Bit 7 = 0b

3.7. CCID 错误码和状态码

本节是对 CCID 标准第 12 部分的扩展，以表格形式列出了 Bulk-IN 消息中有可能与插槽错误寄存器同用的错误码。下表汇总了 CCID 定义的错误码以及为 ACR89 的扩展命令另外定义的错误码。

错误名	错误码	可能原因
CMD_ABORTED	FFh	主机中止了当前活动
ICC_MUTE	FEh	与 ICC 通讯时，CCID 超时
XFR_PARITY_ERROR	FDh	与 ICC 通讯时，奇偶校验错误
XFR_OVERRUN	FCh	与 ICC 通讯时，超限错误
HW_ERROR	FBh	总括性的硬件错误
BAD_ATR_TS	F8h	
BAD_ATR_TCK	F7h	
ICC_PROTOCOL_NOT_SUPPORTED	F6h	
ICC_CLASS_NOT_SUPPORTED	F5h	
PROCEDURE_BYTE_CONFLICT	F4h	
DEACTIVATED_PROTOCOL	F3h	
BUSY_WITH_AUTO_SEQUENCE	F2h	自动序列进行中
PIN_TIMEOUT	F0h	
PIN_CANCELLED	EFh	
CMD_SLOT_BUSY	E0h	向正在处理命令的插槽发送另外一个命令
ACR89_ERROR	10h	定义在 ACR89 响应头而不是错误寄存器里的错误码
DEVICE_VOID	11h	ACR89 没有初始化。出厂模式，等待卖主个人化设置或者设备已被篡改。
INVALID_SECRET_KEY	12h	私钥错误
INVALID_MODE	13h	尝试执行命令，当前模式不允许
保留为将来使用		（剩余所有未提到的值）

表7： CCID 错误码和状态码

4.0. 软件设计

4.1. 非接触式智能卡协议

4.1.1. ATR 的生成

读写器检测到 PICC 后，会发送 ATR 至 PC/SC 驱动来识别 PICC。

4.1.1.1. ATR 信息格式（适用于 ISO 14443-3 PICC）

字节	值（十六进制）	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。 低半字节 N 指出历史字符的个数（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。 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。 低半字节 1 表示协议类型为 T=1
4 至 3+N	80h	T1	类别指示字节，80 表示在可选的 COMPACT-TLV 数据对象中可能有状态标识符
	4Fh	Tk	应用标识符存在标识
	0Ch		长度
	RID		注册应用供应商标识 (RID) # A0 00 00 03 06h
	SSH		标准字节
	C0h..C1h		卡片名称字节
	00 00 00 00h	RFU	RFU # 00 00 00 00h
4+N	UUh	TCK	T0 至 Tk 的所有字节按位异或

表8：ISO 14443 第 3 部分规定的 ATR 格式

例：

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

ATR											
初始字符	T0	TD1	TD2	T1	Tk	长度	RID	标准	卡片名称	RFU	TCK
3Bh	8Fh	80h	01h	80h	4Fh	0Ch	A0 00 00 03 06h	03h	00 01h	00 00 00 00h	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 1K）
	[00 02h] (MIFARE 4K)
	[00 03h] (MIFARE Ultralight)
	[00 26h] (MIFARE Mini)
	[F0 04h] Topaz 和 Jewel
	[F0 11h] FeliCa 212K
	[F0 12h] FeliCa 424K
	[FF 28h] JCOP 30
	FF SAK 未定义标签

4.1.1.2. ATR 信息格式（适用于 ISO 14443-4 PICC）

字节	值（十六进制）	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。 低半字节 N 指出历史字符的个数（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。 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。 低半字节 1 表示协议类型为 T=1
4 至 3 + N	XXh	T1	历史字节： ISO 14443A： 来自 ATS 应答的历史字节。参考 ISO 14443-4 标准。 ISO 14443B： 来自 ATTRIB 应答（ATQB）的上层响应。参考 ISO 14443-3 标准。
	XX XX XXh	Tk	
4+N	UUh	TCK	T0 至 Tk 的所有字节按位异或

表9：ISO 14443 第 4 部分规定的 ATR 格式

例 1：考虑来自 MIFARE DESFire 的 ATR 如下：

MIFARE DESFire (ATR) = 3B 81 80 01 80 80h (6 个字节的 ATR)

注：使用 APDU“FF CA 01 00 00h”判断 PICC 是否符合 ISO 14443A-4 或 ISO 14443B-4，如果有 ATS 的话，取回完整的 ATS。符合 ISO 14443A-3 或 ISO 14443B-3/4 类的 PICC 会返回 ATS。

APDU 命令 = FF CA 01 00 00h

APDU 响应 = 06 75 77 81 02 90 00h

ATS = {06 75 77 81 02 80h}

例 2：考虑来自 ST19XRC8E 的 ATR 如下：

ST19XRC8E (ATR) = 3B 88 80 01 12 53 54 4E 33 81 C3 00 23h

ATQB 的应用数据 = 12 53 54 4Eh

ATQB 的协议信息 = 33 81 C3h

4.1.2. 非接触接口的私有 APDU 指令

4.1.2.1. 采用 ACR89U-A2 结构的直接传输

发送一个私有 APDU（非接触芯片和标签命令），并且会返回响应数据。

Direct Transmit 的命令结构（非接触芯片的长度和标签命令 + 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	非接触芯片和标签命令	数据

其中：

Lc 1 个字节待发送的字节数

- 最大值为 255

Data In 非接触芯片或标签命令。

要发送给非接触芯片和标签的数据。

Direct Transmit 的响应结构（非接触芯片和标签响应 + 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 1 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 0-262 字节。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/IEC 14443 A 类)，

01h:212 Kbps (FeliCa 轮询),

02h:424 Kbps (FeliCa 轮询),

03h:106 Kbps B 类 (ISO/IEC 14443-3B)，

04h:106kbps Innovision Jewel 标签

InitiatorData[] 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 type A 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 type B InitiatorData[]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。

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

- 若没有该字段，将使用时间槽法。

212/424 Kbps 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 tag 暂未使用

Data Out 非接触芯片和标签响应。

读写器返回的非接触芯片和标签响应。

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：

Status 1 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 0-262 字节；非接触芯片接收的原始数据数组。

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”。根据选定的波特率提供关于检测到的目标的信息。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	0x01h（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。
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。

表10 ：DIRECT TRANSMIT 的响应码

4.1.2.2. 获取数据（Get Data）

返回“已建立连接的 PICC”的序列号或 ATS。

Get UID 的 APDU 结构（5 个字节）

命令	CLA	INS	P1	P2	Le
Get Data	FFh	CAh	00h 01h	00h	00h (全长)

P1 = 00h 时，Get UID 的响应结构（UID + 2 个字节）

响应	响应数据域					
结果	UID (LSB)			UID (MSB)	SW1	SW2

P1 = 01h 时，获取 ISO 14443 A 卡的 ATS（ATS + 2 个字节）

响应	响应数据域		
结果	ATS	SW1	SW2

Get Data 的响应码

结果	SW1 SW2	含义
成功	90 00h	操作成功完成。
警告	62 82h	UID/ATS 的末尾先于 Le 字节到达（Le 大于 UID 的长度）。
错误	6C XXh	长度错误（错误的 Le：‘XX’表示确切的数字），如果 Le 小于 UID 的长度。
错误	63 00h	操作失败。
错误	6A 81h	不支持此功能

例 1：获取“已经建立连接的 PICC”的序列号

```
UINT8 GET_UID[5]={FFh, CAh, 00h, 00h, 00h};
```

例 2：获取“已经建立连接的 ISO 14443-A PICC”的 ATS

```
UINT8 GET_ATS[5]={FFh, CAh, 01h, 00h, 00h};
```

4.1.2.3. MIFARE 1K/4K 存储卡的 PICC 命令（T=CL 模拟）

4.1.2.3.1. 加载认证密钥（Load Authentication Keys）

向读写器加载认证密钥。该认证密钥用于验证 MIFARE 1K/4K 存储卡的特定扇区。读写器提供了两种认证密钥位置：易失密钥位置和非易失密钥位置。

Load Authentication Keys 的 APDU 结构（11 个字节）

命令	CLA	INS	P1	P2	Lc	命令数据域
Load Authentication Keys	FFh	82h	密钥结构	密钥号	06h	密钥（6 个字节）

其中：

密钥结构 1 个字节。

00h = 密钥被载入读写器的易失存储器。

其它 = 保留

密钥号 1 个字节。

00h ~ 01h = 密钥位置。一旦读写器与电脑断开连接，密钥就会被删除。

密钥 6 个字节。载入读写器的密钥值。

例如：{FF FF FF FF FF FFh}。

Load Authentication Keys 的响应结构（2 个字节）

响应	响应数据域	
结果	SW1	SW2

Load Authentication Keys 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。

例：

向密钥位置 00h 加载密钥{FF FF FF FF FF FFh}。

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

4.1.2.3.2. MIFARE 1K/4K 卡认证（Authentication for MIFARE 1K/4K）

使用存储在读写器内的密钥来验证 MIFARE 1K/4K 卡（PICC）。其中会用到两种认证密钥：TYPE_A 和 TYPE_B。

Load Authentication Keys 的 APDU 结构（6 个字节）

命令	CLA	INS	P1	P2	P3	命令数据域
Authentication	FFh	88h	00h	块号	密钥类型	密钥号

Load Authentication Keys 的 APDU 结构（10 个字节）

命令	CLA	INS	P1	P2	Lc	命令数据域
Authentication	FFh	86h	00h	00h	05h	认证数据字节

认证数据字节（5 个字节）

字节 1	字节 2	字节 3	字节 4	字节 5
版本 01h	00h	块号	密钥类型	密钥号

其中：

块号 1 个字节。指出待验证的存储块。

密钥类型 1 个字节。

60h = 该密钥被用作 TYPE A 密钥进行验证。

61h = 该密钥被用作 TYPE B 密钥进行验证。

密钥号 1 个字节。

00h ~ 01h = 密钥位置。

注：*MIFARE 1K 卡的内存分为 16 个扇区，每个扇区包含 4 个连续的块。例如：扇区 00 包含块{00h、01h、02h 和 03h}；扇区 01h 包含块{04h、05h、06h 和 07h}；最后一个扇区 0Fh 包含块{3Ch、3Dh、3Eh 和 3Fh}。*

验证通过后，读取同一扇区内的其他块不需要再次进行验证。详情请参考 MIFARE 1K/4K 卡标准。

Load Authentication Keys 的响应结构（2 个字节）

响应	响应数据域	
结果	SW1	SW2

Load Authentication Keys 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。

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

1K 字节

表11 ：MIFARE 1K 卡的内存结构

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

2K 字节

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

2K 字节

表12 ：MIFARE 4K 卡的内存结构

例 1：
通过下列特征验证块 0x04h：A 类, 非易失, 密钥号 0x00h, 来自 PC/SC V2.01 (作废)。

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

例 2：

类似于前面的例子，通过下列特征验证块 04h：A 类, 非易失, 密钥号 00h, 来自 PC/SC V2.07。

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

注：*由于 MIFARE Ultralight 的用户数据区域可以自由访问，所以 MIFARE Ultralight 不需要通过验证。*

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 个字节

表13 ：MIFARE Ultralight 卡的内存结构

4.1.2.3.3. Read Binary Blocks

从 PICC 卡片中取回多个“数据块”。执行该命令前必须先对数据块/尾部块进行验证。

Read Binary 的 APDU 结构（5 个字节）

命令	CLA	INS	P1	P2	Le
Read Binary Blocks	FFh	B0h	00h	块号	待读取的字节数

其中：

块号 1 个字节。待访问的块。

待读取的字节数 1 个字节。最大值为 16 个字节。

Read Binary Block 的响应结构（N + 2 个字节）

响应	响应数据域		
结果	0 <= N <= 16	SW1	SW2

Read Binary Block 命令的响应码

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

例 1：从二进制块 04h 中读取 16 个字节（MIFARE 1K 或 4K）

APDU = {FF B0 00 04 10}

例 2：从二进制页 04h 中读取 4 个字节（MIFARE Ultralight）

APDU = {FF B0 00 04 04}

例 3：从二进制页 04h 开始读取 16 个字节（MIFARE Ultralight）（读取页 4, 5, 6 和 7）

APDU = {FF B0 00 04 10}

4.1.2.3.4. 更新二进制块（Update Binary Blocks）

向 PICC 写入多个“数据块”。执行该命令前必须先对数据块/尾部块进行验证。

Update Binary 的 APDU 结构（4 或 16 + 5 个字节）

命令	CLA	INS	P1	P2	Lc	命令数据域
Update Binary Blocks	FFh	D6h	00h	块号	待更新的字节数	块数据 MIFARE Ultralight 的待更新字节数为 4。或 MIFARE 1K/4K：16 个字节

其中：

块号 1 个字节。待更新的起始块。

待更新的字节数 1 个字节。

MIFARE 1K/4K 的待更新字节数为 16

MIFARE Ultralight 的待更新字节数为 4。

块数据 4 或 16 个字节

待写入二进制块的数据。

Update Binary Block 的响应码（2 个字节）

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

例 1：将 MIFARE 1K/4K 卡中二进制块 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}

例 2：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}

4.1.2.3.5. 值块操作（增量，减量，存储）-Value Block Operation (Increment, Decrement, Store)

操作数值（例如：增加值块的值等）。

Value Block Operation 的 APDU 结构（10 个字节）

命令	CLA	INS	P1	P2	Lc	命令数据域	
Value Block Operation	FFh	D7h	00h	块号	05h	VB_OP	VB_Value（4 个字节）{MSB ..LSB}

其中：

块号 1 个字节。待操作的值块

VB_OP 1 个字节。

00h = 将 VB_Value 存入该块。然后该块变为一个值块。

01h = 使值块的值增加 VB_Value，然后该块将变为值块。此命令仅适用于对值块的操作。

02h = 使值块的值减少 VB_Value。此命令仅适用于对值块的操作。

VB_Value 4 个字节。该数据的值是一个有符号长整数（4 个字节），用于数值操作。

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

VB_Value			
MSB			LSB
FFh	FFh	FF	FCh

例 2：Decimal 1 = {00h, 00h, 00h, 01h}

VB_Value			
MSB			LSB
00h	00h	00h	01h

Value Block Operation 的响应结构（2 个字节）

响应	响应数据域	
结果	SW1	SW2

Value Block Operation 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。

4.1.2.3.6. 读取数据块（Read Value Block）

获取值块中的数值，此命令仅适用于对数值块的操作。

Read Value Block 的 APDU 结构（5 个字节）

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

其中：

Block Number 1 个字节；要访问的值块。

Read Value Block 的响应结构（4 + 2 个字节）

响应	响应数据域		
结果	值 {MSB ..LSB}	SW1	SW2

其中：

值 4 个字节。卡片返回的值。是一个有符号长整数（4 个字节）。

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

Read Value Block 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。

4.1.2.3.7. 复制值块（Copy Value Block）

将一个值块中的数值复制到另外一个值块。

Copy Value Block 命令的 APDU 结构（7 个字节）

命令	CLA	INS	P1	P2	Lc	命令数据域	
Copy Value Block Operation	FFh	D7h	00h	源块号	02h	03h	目标块号

其中：

Source Block Number 1 个字节。源值块中的值会被复制到目标值块。

Target Block Number 1 个字节。待存储的值块。源值块和目标值块必须位于同一个扇区。

Copy Value Block 的响应报文结构（4+2 个字节）

响应	响应数据域	
结果	SW1	SW2

Copy Value Block 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。

例 1：将数值 “1” 存入块 05h

APDU = {FF D7 00 05 05 00 00 00 00 01h}

例 2：读取值块 05h

APDU = {FF B1 00 05 00h}

例 3：将值块 05h 的值复制到值块 06h

APDU = {FF D7 00 05 02 03 06h}

例 4：使值块 05h 的值增加“5”

APDU = {FF D7 00 05 05 01 00 00 00 05h}
应答：90 00h [$9000]

4.1.2.4. 访问符合 PC/SC 标准的标签（ISO 14443-4)

所有符合 ISO 14443-4 标准的卡片（PICC）都可以理解符合 ISO 7816-4 规定的 APDU。ACR89U-A2 读写器与符合 ISO 14443-4 标准的卡片进行通信时，需要对 ISO 7816-4 规定的 APDU 和响应进行转换。ACR89U-A2 会在内部处理 ISO 14443 第 1-4 部分协议。

MIFARE 1K、4K、MINI 和 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。更多相关信息，请参阅“MIFARE Classic 存储标签的 PICC 命令”。

ISO 7816-4 规定的 APDU 报文的结构

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 规定的响应结构（数据 + 2 个字节）

响应	响应数据域		
结果	响应数据	SW1	SW2

通用的 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。

典型的操作顺序为：

- 出示标签并连接 PICC 界面
- 读取/更新标签的存储内容。

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

步骤 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”来获取 ATS。*

例： ISO 7816-4 APDU

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

APDU = {80 B2 80 00 08h}

类 = 80h；INS = B2h；P1 = 80h；P2 = 00h;

Lc = 无; Data In = 无； Le = 08h

应答：00 01 02 03 04 05 06 07h [$9000]

附录A.　非接触式应用的基本流程

步骤 0.启动应用程序，读写器会不断地进行 PICC 轮询和标签扫描。一旦发现并检测到标签，相应的 ATR 会被发送到 PC。

步骤 1.通过 T=1 协议连接“ACR89U PICC 界面”。

步骤 2.通过 APDU 交换访问 PICC。

..

步骤 N. 断开“ACR89U PICC 界面”的连接，关闭应用程序。

备注：

您可以关掉天线来节省电源。

- 关闭天线电源：FF 00 00 00 04 D4 32 01 00h
- 开启天线电源：FF 00 00 00 04 D4 32 01 01h

附录B. 访问 MIFARE DESFire 标签(ISO 14443-4)

MIFARE DESFire 支持 ISO7816-4 APDU 包模式和本地模式。一旦 MIFARE DESFire 标签被激活，发送至 MIFARE DESFire 标签的第一个 APDU 就会确定“命令的模式”。如果第一个 APDU 采用“本地模式”，则其余的 APDU 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 (DESFire)中读取 8 个字节的随机数

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

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

Lc = 01h; 命令数据 = 00h; Le = 00h (Le = 00h，表示最大长度)

应答：7B 18 92 9D 9A 25 05 21h [$91AF]

注：*状态码{91 AFh}由 MIFARE DESFire 标准定义，详情请参阅 DESFire 标准。*

例 2：DESFire 分页链接（ISO 7816 APDU 包模式）

在本例中，应用涉及到"分页链接"。

要获取 DESFire 卡的版本号：

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

应答：04 01 01 00 02 18 05 91 AFh [$91AF]

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

应答：04 01 01 00 06 18 05 91 AFh [$91AF]

步骤 3：发送 APDU {90 AF 00 00 00h}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 00h [$9100]

例 3：MIFARE DESFire 本地命令。

若本地 MIFARE DESFire 命令更易于操作，则我们可以向读写器发送不带 ISO 7816 包的本地 DESFire 命令。

从 ISO 14443-4 Type A PICC (MIFARE DESFire)中读取八个字节的随机数

APDU = {0A 00h}

应答：AF 25 9C 65 0C 87 65 1D D7h [$1DD7]

其中，第一个字节“AFh”是 MIFARE DESFire 卡片返回的状态码。

应用程序可以对[$1DD7]中的数据予以忽略。

例 4：MIFARE DESFire 分页链接 (本地模式)

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 {60h} 来获取第一个数据页。INS=60h

应答：AF 04 01 01 00 02 18 05h [$1805]

步骤 2：发送 APDU {AFh} 来获取第二个数据页。INS=AFh

应答：AF 04 01 01 00 06 18 05h [$1805]

步骤 3：发送 APDU {AFh} 来获取最后一个数据页。INS=AFh

应答：00 04 52 5A 19 B2 1B 80 8E 36 54 4D 40 26 04h [$2604]

注：*在 DESFire 本地模式下，如果响应的长度大于 1，则在响应中不会出现状态码[90 00h]。但是如果响应的长度小于 2，则会在响应中增加状态码[90 00h]以满足 PC/SC 的要求。最短的响应长度为 2。*

附录C.　访问 FeliCa 标签（ISO 18092）

典型的操作顺序为：

- 出示 FeliCa 标签，并与 PICC 接口建立连接
- 读取/更新标签的存储内容

步骤 1：与标签建立连接

ATR = 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 F0 11 00 00 00 00 8Ah

其中，

F0 11h = FeliCa 212K

步骤 2：读取内存块，不使用私有的 APDU。

<< 10 06 [8-byte NFC ID] 01 09 01 01 80 00h

>> 1D 07 [8-byte NFC ID] 00 00 01 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AAh [90 00h]

或

步骤 2：读取内存块，使用私有的 APDU。

<< FF 00 00 00 [13] D4 40 01 10 06 [8-byte NFC ID] 01 09 01 01 80 00h

其中，

[13h] 是私有数据“D4 40 01..80 00h”的长度。

D4 40 01h 是数据交换（DATA EXCHANGE）命令

>> D5 41 00 1D 07 [8-byte NFC ID] 00 00 01 00 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AAh [90 00h]

其中，D5 41 00h 是对 DATA EXCHANGE 的响应

注：*可以使用 APDU“FF CA 00 00 00h”来获取 NFC ID. 详情请参阅 Felica 标准的相关规定。*

附录D.　访问 NFC 论坛 1 类标签（ISO 18092）

典型的操作顺序为：

- 出示 Topaz 标签，并与 PICC 接口建立连接
- 读取/更新标签的存储内容

步骤 1：与标签建立连接

ATR = 3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 F0 04 00 00 00 00 9Fh

其中，

F0 04h = Topaz

步骤 2：读取内存地址 08 (Block 1:**Byte-0**），不使用私有 APDU

<< 01 08h

>> 18h [90 00h]

其中，响应数据 = 18h

或

步骤 2：读取内存地址 08h (Block 1:**Byte-0**），使用私有 APDU

<< FF 00 00 00 [05] D4 40 01 01 08h

其中，

[05h] 是私有 APDU 数据“D4 40 01 01 08h”的长度

D4 40 01h 是数据交换（Data Exchange）命令

01 08h 是要发送给标签的数据。

>> D5 41 00 18h [90 00h]

其中，响应数据 = 18h

提示：读取整个标签的存储内容

<< 00h

>> 11 48 18 26 ..00h [90 00h]

步骤 3：读取内存地址 08h（Block 1:Byte-0)更新为数据 FFh。

<< 53 08 FFh

>> FFh [90 00h]

其中，响应数据 = FFh

HR0	HR1
11_h	xx_h

EEPROM Memory Map

Type	Block No.	Byte-0 **(LSB)**	Byte-1	Byte-2	Byte-3	Byte-4	Byte-5	Byte-6	Byte-7 **(MSB)**	Lockable
UID	0	UID-0	UID-1	UID-2	UID-3	UID-4	UID-5	UID-6		Locked
Data	1	Data0	Data1	Data2	Data3	Data4	Data5	Data6	Data7	Yes
Data	2	Data8	Data9	Data10	Data11	Data12	Data13	Data14	Data15	Yes
Data	3	Data16	Data17	Data18	Data19	Data20	Data21	Data22	Data23	Yes
Data	4	Data24	Data25	Data26	Data27	Data28	Data29	Data30	Data31	Yes
Data	5	Data32	Data33	Data34	Data35	Data36	Data37	Data38	Data39	Yes
Data	6	Data40	Data41	Data42	Data43	Data44	Data45	Data46	Data47	Yes
Data	7	Data48	Data49	Data50	Data51	Data52	Data53	Data54	Data55	Yes
Data	8	Data56	Data57	Data58	Data59	Data60	Data61	Data62	Data63	Yes
Data	9	Data64	Data65	Data66	Data67	Data68	Data69	Data70	Data71	Yes
Data	A	Data72	Data73	Data74	Data75	Data76	Data77	Data78	Data79	Yes
Data	B	Data80	Data81	Data82	Data83	Data84	Data85	Data86	Data87	Yes
Data	C	Data88	Data89	Data90	Data91	Data92	Data93	Data94	Data95	Yes
Reserved	D									
Lock/Reserved	E	LOCK-0	LOCK-1	OTP-0	OTP-1	OTP-2	OTP-3	OTP-4	OTP-5	

Reserved for internal use
User Block Lock & Status
OTP bits

图5：Topaz 内存图

内存地址 = Block No * 8 + Byte No

例 1：内存地址 08h = 1 x 8 + 0 = Block 1:Byte-0 = Data0

例 2：内存地址 10h = 2 x 8 + 0 = Block 2:Byte-0 = Data8