Advanced Card Systems Ltd.
Card & Reader Technologies

APG8201
PINhandy 1

参考手册 V1.02

如有任何更改，恕不另行通知

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

目录

图目录

表目录

1.0. 简介

APG8201 PINhandy 1 是一款便携式手持智能卡设备。它支持连机和脱机两种模式，能够执行多种认证功能。配有内置键盘和图形液晶显示屏，支持多种语言和字母数字字符。APG8201 支持 PIN 码安全输入（SPE），通过设备内的认证进程防止 PIN 码受到安全攻击。APG8201 PINhandy 1 优质可靠，还可用作动态密码发生器和计算器。

2.0. 特性

- 纤巧、便携的手持设备
- 两种操作模式：
  - 连机
  - 脱机
- USB 供电（连机操作）：
  - USB 2.0 全速接口
  - 符合 CCID 标准
  - 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）
  - 支持协议和参数选择（PPS）
  - 支持 PC/SC 2.01 第 10 部分 PIN 码安全输入（SPE）
- 脱机操作：
  - 支持一次性密码（OTP），“质询——回应”和交易数据签名模式
  - 2 节 CR2032 电池供电
  - 智能电池管理，最长可使用 5 年（具体根据实际使用情况而定）
- 智能卡读写器：
  - 支持全尺寸的微处理器卡（T=0 和 T=1 协议）
  - 支持 ISO 7816 A 类卡
  - 卡身可以只插入一半
  - 短路保护
- 内置外围设备：
  - 图形液晶显示屏，支持标志和多种语言字符
  - 单音蜂鸣器
  - 耐用的触觉键盘，有 20 个硅胶按键
  - LCD 屏能够显示钥匙符号，用于识别 PIN 码安全输入模式
- 具有计算器和电子钱包的附加功能
- 支持 Android™ 3.1 及以上版本 [1]
- 符合下列标准：
  - ISO 7816
  - EMV™ Level 1 (接触)
  - MasterCard® 芯片验证计划（CAP）
  - MasterCard® 高级芯片身份验证（AA4C/PLA）
  - VISA® 动态密码验证（DPA）
  - PC/SC
  - CCID
  - CE
  - FCC
  - RoHS 2
  - FIPS 201 认证（美国）
  - Microsoft® WHQL

[1] *使用 ACS 定义的安卓库*

3.0. 支持的卡片类型

APG8201 PINhandy 1 主卡插槽支持符合 T=0 和 T=1 协议的 A 类（5 V）MCU 卡，也支持可在卡片内生成编程电压（VPP）的 EEPROM 微处理器卡。ATR 传递的编程参数如下：

- PI1 = 0 或 5
- I = 25 或 50

脱机模式下，APG8201 可自动执行 PPS，而在 USB 联机模式下，可手动执行 PPS。

若卡片产生的 ATR 指定了专用的操作模式（TA2 存在；TA2 中的 b5 位必须为 0），但 APG8201 读卡器不支持该指定模式，则 APG8201 会将卡片复位，使其置为协商模式。如果卡片不能被置为协商模式，APG8201 会拒绝读写该卡。

若卡片产生的 ATR 指定了协商模式（TA2 不存在时）和非默认的通信参数，APG8201 读卡器将执行 PPS 并尝试使用卡片 ATR 建议的通信参数。如果卡片不接受 PPS，读卡器会使用默认参数（F=372，D=1）。

对于上述参数的含义，请参考 ISO 7816-3。

4.0. 智能卡接口

APG8201 与智能卡之间的界面遵循 ISO 7816-2 和 ISO 7816-3，并进行了某些限制或提升来增强 APG8201 的实用功能：

- 智能卡电源 VCC（C1）
  - 插入的卡片电流消耗不得大于 54 mA。
- 编程电压 VPP（C6）
  - VPP 引脚断开
- 复位信号（C2）
  - 参见 EMV 2000 版 Book 1
- 时钟信号（C3）
  - 参见 EMV 2000 版 Book 1
- 地（C5）
  - 参见 EMV 2000 版 Book 1
- I/O 数据输入和输出（C7）
  - 参见 EMV 2000 版 Book 1

5.0. 卡片类型选择

激活插入的卡片之前，处于控制地位的电脑总是需要向 APG8201 发送适当的命令来选择卡片类型。对于基于 MCU 的卡片，若同时支持 T=0 和 T=1，则读卡器可通过协议与参数选择（PPS）选择 T=0 或 T=1 作为首选协议；若仅支持 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。

6.0. USB 接口

如《USB 规格书 2.0》所述，APG8201 通过 USB 连接计算机，支持 USB 全速模式（12 Mbps）。要使 APG8201 能够通过 USB 接口正常工作，必须先安装 ACS 专有设备驱动程序或 ACS PC/SC 设备驱动程序。

7.0. USB 通信协议（CCID）

APG8201 应当通过 USB 连接与主机（host）端建立接口。现有行业内规范 — CCID 标准，已经为 USB 芯片-智能卡接口设备定义了相关协议。CCID 涵盖了操作智能卡和 PIN 所需的全部协议。APG8201 上 USB 端点的配置和使用应当符合 CCID 标准。

APG8201 需要处理的几个基本 CCID 命令协议将在随后的章节中一一列举。

7.1. 计算机至读卡器

7.1.1. PC_to_RDR_IccPowerOn

此命令用于激活卡槽并返回卡片的 ATR。

偏移量	字段	大小	值	说明
0	*bMessageType*	1	62h	-
1	*dwLength*	4	00000000h	消息特定的数据长度。
2	*bSlot*	1	00-FFh	标识命令的插槽号
5	*bSeq*	1	00-FFh	命令的序号
6	*bPowerSelect*	1	01h	ICC 上的电压值： 01h - 5 V
7	*abRFU*	2	-	保留为将来使用

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

7.1.2. PC_to_RDR_IccPowerOff

此命令用于取消激活卡槽。

偏移量	字段	大小	值	说明
0	*bMessageType*	1	63h	-
1	*dwLength*	4	00000000h	消息特定的数据长度。
5	*bSlot*	1	00-FFh	标识命令的插槽号
6	*bSeq*	1	00-FFh	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。

7.1.3. PC_to_RDR_XfrBlock

此命令用于向 ICC 传输数据块。

偏移	字段	大小	值	说明
0	*bMessageType*	1	6Fh	-
1	*dwLength*	4	-	此消息中 abData 字段的大小
5	*bSlot*	1	00-FFh	标识命令的插槽号

偏移	字段	大小	值	说明
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。
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。

7.1.4. PC_to_RDR_GetParameters

此命令用于获取卡槽参数。

偏移	字段	大小	值	说明
0	*bMessageType*	1	6Ch	-
1	*dwLength*	4	00000000h	消息特定的数据长度。
5	*bSlot*	1	00-FFh	标识命令的插槽号
6	*bSeq*	1	00-FFh	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。

7.1.5. PC_to_RDR_ResetParameters

此命令用于重置卡槽参数。

偏移	字段	大小	值	说明
0	*bMessageType*	1	6Dh	-
1	*dwLength*	4	00000000h	消息特定的数据长度。
5	*bSlot*	1	00-FFh	标识命令的插槽号
6	*bSeq*	1	00-FFh	命令的序号。 ***注：****与插槽号无关，FFh 后直接过渡到 00h。*
7	*abRFU*	3	-	保留为将来使用

此消息的响应是 *RDR_to_PC_Parameters* 消息。

7.1.6. PC_to_RDR_SetParameters

此命令用于设置卡槽参数。

偏移	字段	大小	值	说明
0	*bMessageType*	1	61h	-
1	*dwLength*	4	-	此消息中 abProtocolDataStructure 字段的大小
5	*bSlot*	1	00-FFh	标识命令的插槽号
6	*bSeq*	1	00-FFh	命令的序号
7	*bProtocolNum*	1	00h、01h	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
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，选择一个时钟频率转换因子 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	*bmTCCKST0*	1	00h、02h	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	00-FFh	两个字符间的额外保护时间。在通常的保护时间（12 etu）基础上增加 0-254 个 etu。 FFh 等同于 00h。
13	*bWaitingIntegerT0*	1	00-FFh	T=0 时 WI 用于定义 WWT

偏移	字段	大小	值	说明
14	*bClockStop*	1	00 03h	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

此消息的响应是 *RDR_to_PC_Parameters* 消息。

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，选择一个时钟频率转换因子 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	10h、11h、12h、13h、	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	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 B3-0 = CWI 值 0-Fh 有效
14	*bClockStop*	1	00-03h	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	00-FEh	商定的 IFSC 的大小
16	*bNadValue*	1	-	若 CCID 不支持默认值之外的任何值，值 = 00h

此消息的响应是 *RDR_to_PC_Parameters* 消息。

7.1.7. PC_to_RDR_Escape

此命令用于定义并访问扩展特性。

偏移	字段	大小	值	说明
0	*bMessageType*	1	6Bh	-
1	*dwLength*	4	00000000h	此消息的 abData 字段的大小
5	*bSlot*	1	00-FFh	标识命令的插槽号
6	*bSeq*	1	00-FFh	命令的序号
7	*abRFU*	3	-	保留为将来使用
10	*abData*	字节型数组	-	发送至 CCID 的数据块

7.1.7.1. 获取固件版本号（Get Firmware Version）

偏移	字段	大小	值	说明
10	*bcmdCode*	1	04h	-
11	*wcmdLength*	2	0000h	-
13	*abRFU*	2	-	保留为将来使用

例如：

bSendBuffer[0]=04h;

bSendBuffer[1]=00h;

bSendBuffer[2]=00h;

bSendBuffer[3]=00h;

bSendBuffer[4]=00h;

dwSendBufferLen=05h

SCARDStatus = SCardControl(hSAM, SCARD_CTL_CODE(3500), bSendBuffer, dwSendBufferLen, bRecvBuffer, dwRecvBufferLen, &dwRecvBufferLen)；

7.1.7.2. 显示 LCD 消息（Display LCD Message）

偏移	字段	大小	值	说明
10	*bcmdCode*	1	05h	-
11	*wcmdLength*	2	0020h	-
13	*abRFU*	2	-	保留为将来使用
15	*abData*	20	-	用计算机键盘输入数据，并显示在 LCD 上

例如：

bSendBuffer[0]=05h;

bSendBuffer[1]=00h;

bSendBuffer[2]=20h;

bSendBuffer[3]=00h;

bSendBuffer[4]=00h;

bSendBuffer[abData]=(31 32 33 20 20 20 20 20 … 20h)

dwSendBufferLen=25h

SCARDStatus = SCardControl(hSAM, SCARD_CTL_CODE(3500), bSendBuffer, dwSendBufferLen, bRecvBuffer, dwRecvBufferLen, &dwRecvBufferLen);

7.1.7.3. 读取密钥（Read Key）

偏移	字段	大小	值	说明
10	*bcmdCode*	1	06h	-
11	*wcmdLength*	2	0006h	-
13	*abRFU*	2	-	保留为将来使用
15	*timeout*	1	00h	-
16	*PinLength*	2	XXYYh	XXh：PIN 码的最大长度 YYh：PIN 码的最小长度
18	*KeyReturnCondition*	1	-	该值表示一种按位异或操作。 01：已达到最大长度 02：按 Enter 键 04：超时 08：按 Cancel 键
19	*StartPosition*	1	-	Bit7-4：0000 表示 LCD 上线;0001 表示 LCD 下线。 Bit3-0：表示显示位置。
20	*EchoLCDMode*	1	-	00： 显示 ASCII 码 01： 显示字符*

例如：

bSendBuffer[0]=06h;

bSendBuffer[1]=00h;

bSendBuffer[2]=06h;

bSendBuffer[3]=00h;

bSendBuffer[4]=00h;

bSendBuffer[abData]=(00 08 04 01 00 00h)

dwSendBufferLen=0Bh

SCARDStatus = SCardControl(hSAM, SCARD_CTL_CODE(3500), bSendBuffer, dwSendBufferLen, bRecvBuffer, dwRecvBufferLen, &dwRecvBufferLen);

7.1.7.4. 蜂鸣器响（Buzzer Beep）

偏移	字段	大小	值	说明
10	*bcmdCode*	1	08h	-
11	*wcmdLength*	2	0000h	-
13	*abRFU*	2	-	保留为将来使用

例如：

bSendBuffer[0]=08h;

bSendBuffer[1]=00h;

bSendBuffer[2]=00h;

bSendBuffer[3]=00h;

bSendBuffer[4]=00h;

dwSendBufferLen=05h

SCARDStatus = SCardControl(hSAM, SCARD_CTL_CODE(3500), bSendBuffer, dwSendBufferLen, bRecvBuffer, dwRecvBufferLen, &dwRecvBufferLen);

7.1.7.5. 设置读卡器选项命令（Set Reader Option Command）

偏移	字段	大小	值	说明
10	*bcmdCode*	1	13h	-
11	*wcmdLength*	2	0000h	-
14	*abRFU*	2	0000h	-
15	*ReaderOptions*	1	-	位定义。请参考**表 1.**

例如：

bSendBuffer[0]=13h;

bSendBuffer[1]=00h;

bSendBuffer[2]=00h;

bSendBuffer[3]=00h;

bSendBuffer[4]=00h;

bSendBuffer[5]=02h;

dwSendBufferLen=06h

SCARDStatus = SCardControl(hSAM, SCARD_CTL_CODE(3500), bSendBuffer, dwSendBufferLen, bRecvBuffer, dwRecvBufferLen, &dwRecvBufferLen);

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
未使用	未使用	未使用	未使用	未使用	616C 标签	EMV 模式	PPS 模式

表1 : ReaderOptions 位结构

7.1.8. PC_to_RDR_Secure

此命令用于输入验证或更改用的 PIN 码。

偏移	字段	大小	值	说明
0	*bMessageType*	1	69h	-
1	*dwLength*	4	-	此消息的 abData 字段的大小
5	*bSlot*	1	00-FFh	标识命令的插槽号
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。
8	*wLevelParameter*	2	0000h	RFU（TPDU 交换级别）
10	*abData*	字节型数组	-	该值取决于 wLevelParameters。若 wLevelParameters = 0000h 或 0001h，则 abData = abPINOperationDataStructure。

7.1.8.1. abPINOperationDataStructure

偏移	字段	大小	值	说明
10	*bPINOperation*	1	00-06h	用于表示 PIN 操作： 00h：PIN 码校验 01h：PIN 码更改
11	*abPINDataStructure*	字节型数组	-	PIN 码验证数据结构或 PIN 码更改数据结构

7.1.8.2. PIN 码验证数据结构（PIN Verification Data Structure）

偏移	字段	大小	值	说明
11	*bTimeOut*	1	-	超时时间（单位：秒）。若超时时间为 00h 秒，则 CCID 采用默认值。
12	*bmFormatString*	1	-	PIN 码格式选项的相关参数
13	*bmPINBlockString*	1	-	定义 APDU 命令中显示的 PIN 块的字节长度
14	*bmPINLengthFormat*	1	-	APDU 命令中允许插入的 PIN 码长度
15	*wPINMaxExtraDigit*	2	XXYYh	XX：最小 PIN 码长度（位） YY：最大 PIN 码长度（位）
17	*bEntryValidationCondition*	1	-	该值表示一种按位异或操作。 01h：已达到最大长度 02h：按校验键 04h：超时
18	*bNumberMessage*	1	00h 01h FFh	PIN 码验证管理中显示的消息数： 00h：无字符串 01h：bMsgIndex 所含索引对应的消息 FFh：默认的 CCID 消息
19	*wLangId*	2	-	用于显示消息的语言
21	*bMsgIndex*	1	-	读卡器 CCID 消息表中的消息索引（应为 00h）。该消息提示用户输入 PIN 码。
22	*bTeoPrologue*	3	-	用于 T=1 I-block prologue 域。只有 T=1 协议在用时有意义。
25	*abPINApdu*	字节型数组	-	待发送给 ICC 的 APDU

7.1.8.3. PIN 码更改数据结构（PIN Modification Data Structure）

偏移	字段	大小	值	说明
11	*bTimeOut*	1	-	超时时间（单位：秒）。若超时时间为 00h 秒，则 CCID 采用默认值。
12	*bmFormatString*	1	-	PIN 码格式选项的相关参数
13	*bmPINBlockString*	1	-	定义 APDU 命令显示的 PIN 块的字节长度
14	*bmPINLengthFormat*	1	-	APDU 命令允许插入的 PIN 码长度
15	*bInsertionOffsetOld*	1	-	当前 PIN 码插入位置的偏移字节数
16	*bInsertionOffsetNew*	1	-	新 PIN 码插入位置的偏移字节数

偏移	字段	大小	值	说明
17	*wPINMaxExtraDigit*	2	XXYYh	XXh：最小 PIN 码长度（位） Yyh：最大 PIN 码长度（位）
19	*bConfirmPIN*	1	00h、 01h、 02h、 03h	表示接受新 PIN 码前是否需要确认（如需确认，接受新 PIN 前用户需要输入新 PIN 码两次） 表示是否必须在同一 APDU 域输入并设置当前 PIN 码。 b0：（0/1） 若值为 0，则不需要确认 若值为 1，则需要确认 b1：（0/1） 若值为 0，则不需要输入当前 PIN 码（这种情况下，必然不用考虑 bInsertinoOffsetOld 的值） 若值为 1，则需要输入当前 PIN 码 b2 – b7：保留为将来使用
20	*bEntryValidationCondition*	1	-	该值表示一种按位异或操作。 01h：已达到最大长度 02h：按校验键 04h：超时
21	*bNumberMessage*	1	00h、 01h、 02h、 03h 或 FFh	PIN 码修改命令中显示的消息数： 00h：无消息 01h：bMsgIndex1 所含索引对应的消息 02h： bMsgIndex1 和 bMsgIndex2 所含索引对应的消息 03h：bMsgIndex1、bMsgIndex2 和 bMsgIndex3 所含索引对应的消息 FFh：默认的 CCID 消息
22	*wLangId*	2	-	用于显示消息的语言
24	*bMsgIndex1*	1	-	读卡器消息表中的消息索引（应为 00h 或 01h）。 若 bConfirmPIN = 00h 或其当前 PIN,会弹出该消息,提示用户输入新 PIN 码。
25	*bMsgIndex2*	1	-	读卡器消息表中的消息索引（应为 01h 或 02h）。 若 bConfirmPIN = 02h 或 03h,会弹出该消息,提示用户输入新 PIN 值。 若 bConfirmPIN = 01h,会弹出该消息,提示用户再次输入新 PIN 值。 （只有 bNumberMessage 不为空时才会出现）

偏移	字段	大小	值	说明
26	*bMsgIndex3*	1	-	读卡器消息表中的消息索引（应为 02h）。 需要确认时,会弹出该消息,提示用户再次输入新 PIN 值。 （只有 bNumberMessage = 3 时才会出现）
25 或 26 或 27	*bTeoPrologue*	3	-	用于 T=1 I-block prologue 域。只有在 T=1 协议在用时有意义。
28 或 29 或 30	*abPINApdu*	字节型数组	-	待发送给 ICC 的 APDU

此命令消息的响应是 RDR_to_PC_DataBlock 消息。

7.2. 读卡器至计算机

7.2.1. RDR_to_PC_DataBlock

此消息由 APG8201 发出，是对命令消息 PC_to_RDR_IccPowerOn、PC_to_RDR_XfrBlock 和 PC_to_RDR_Secure 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	-	此消息中 abData 字段的大小
5	*bSlot*	1	-	标识该命令的插槽号。与 Bulk-OUT 消息中的值相同。
6	*bSeq*	1	-	相应命令的序号。与 Bulk-OUT 消息中的值相同。
7	*bStatus*	1	-	插槽状态寄存器
8	*bError*	1	-	插槽错误寄存器
9	*bChainParameter*	1	00h	RFU（TPDU 交换级别）
10	*abData*	字节型数组	-	本字段包含由 CCID 返还的数据

7.2.2. RDR_to_PC_SlotStatus

此消息由 APG8201 发出，是对命令消息 PC_to_RDR_IccPowerOff 的响应。

偏移	字段	大小	值	说明
0	*bMessageType*	1	81h	-
1	*dwLength*	4	00000000h	消息特定的数据长度。
5	*bSlot*	1	-	标识该命令的插槽号。与 Bulk-OUT 消息中的值相同。
6	*bSeq*	1	-	相应命令的序号。与 Bulk-OUT 消息中的值相同。
7	*bStatus*	1	-	插槽状态寄存器
8	*bError*	1	-	插槽错误寄存器
9	*bClockStatus*	1	00h、01h、02h、03h	值： 00h = 时钟运行 01h = 时钟停于 L 状态 02h = 时钟停于 H 状态 03h = 时钟停止于未知状态 所有其他值保留为将来使用。

7.2.3. RDR_to_PC_Parameters

此消息由 APG8201 发出，是对命令消：

PC_to_RDR_GetParameters、*PC_to_RDR_ResetParameters* 和 *PC_to_RDR_SetParameters* 的响应。

偏移	字段	大小	值	说明
0	*bMessageType*	1	82h	-
1	*dwLength*	4	-	此消息中 abProtocolDataStructure 字段的大小
5	*bSlot*	1	-	标识命令的插槽号。与 Bulk-OUT 消息中的值相同。
6	*bSeq*	1	-	相应命令的序号。与 Bulk-OUT 消息中的值相同。
7	*bStatus*	1	-	插槽状态寄存器
8	*bError*	1	-	插槽错误寄存器
9	*bProtocolNum*	1	00h、01h	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

T=0 的协议数据结构（*bProtocolNum=0，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，选择一个时钟频率转换因子 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	*bmTCCKST0*	1	00h、02h	若协议为 T=0，B0 – 0b，B7-2 – 000000b B1 – 使用的约定（b1=0：正向约定；b1=1：反向约定）
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（12 etu）基础上增加 0-254 个 etu。 FFh 等同于 00h。
13	*bWaitingIntegerT0*	1	00-FFh	T=0 时 WI 用于定义 WWT

偏移	字段	大小	值	说明
14	*bClockStop*	1	00-03h	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（*bProtocolNum=1，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，选择一个时钟频率转换因子 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	10h、 11h、 12h、 13h、	若协议为 T=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）
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。
13	*BwaitingIntegerT1*	1	00-9Fh	B7-4 = BWI B3-0 = CWI
14	*bClockStop*	1	00-03h	支持 ICC 时钟停止： 00 = 不允许停止时钟 01 = 时钟信号为低时停止 02 = 时钟信号为高时停止 03 = 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	00-FFh	CCID 用的节点地址

7.2.4. RDR_to_PC_Escape

此消息由 APG8201 发出，是对命令消息 *PC_to_RDR_Escape* 的响应。

偏移	字段	大小	值	说明
0	bMessageType	1	83h	-
1	dwLength	4	-	此消息中 abData 字段的大小
5	bSlot	1	-	标识命令的插槽号。与 Bulk-OUT 消息中的值相同。
6	bSeq	1	-	相应命令的序号。与 Bulk-OUT 消息中的值相同。

偏移	字段	大小	值	说明
7	bStatus	1	-	插槽状态寄存器
8	bError	1	-	插槽错误寄存器
9	abRFU	1	00h	保留为将来使用
10	abData	字节型数组	-	由 CCID 发送的数据

7.2.4.1. 获取读卡器的特定标签（Get Reader Specific Tag）

偏移	字段	大小	值	说明
10	*bRespType*	1	80h	-
11	*wcmdLength*	1	04h	-
12	*abData*	4	00h	-

7.2.4.2. 获取固件版本号（Get Firmware Version）

偏移	字段	大小	值	说明
10	*bRespType*	1	84h	-
11	*wcmdLength*	2	0004h	-
13	*abRFU*	2	0000h	保留为将来使用
15	*abData*	4	-	固件版本值

7.2.4.3. 显示 LCD 消息（Display LCD Message）

偏移	字段	大小	值	说明
10	*bRespType*	1	85h	-
11	*wcmdLength*	2	00h	-
12	*abRFU*	2	00h	保留为将来使用

7.2.4.4. 读取密钥（Read Key）

偏移	字段	大小	值	说明
10	*bRespType*	1	86h	-
11	*wResLength*	2	-	-
13	*abRFU*	2	0000h	-
15	*abData*	字节型数组	-	由读卡器发送的数据。

7.2.4.5. 蜂鸣器响 （Buzzer Beep）

偏移	字段	大小	值	说明
10	*bRespType*	1	88h	-
11	*abRFU*	4	-	-

7.2.4.6. LCD 生产测试功能（LCD Production Test Function）

偏移	字段	大小	值	说明
10	*bRespType*	1	8Fh	-
11	*wResLength*	2	0000h	-
13	*abRFU*	2	0000h	-

7.2.4.7. 设置读卡器选项命令（Set Reader Option Command）

偏移	字段	大小	值	说明
10	*bRespType*	1	93h	-
11	*wResLength*	2	0000h	-
13	*abStatus*	2	-	0000h：成功 0001h：BAD_PARAMETER

7.2.4.8. 生产测试命令 （Production Test Command）

偏移	字段	大小	值	说明
10	*bRespType*	1	8Ch	-
11	*wResLength*	2	0000h	-
13	*abStatus*	2	-	0000h：成功 0001h：BAD_PARAMETER

7.2.4.9. 认证（Authentication）

偏移	字段	大小	值	说明
10	*bRespType*	1	8Dh	-
11	*wResLength*	2	0008h	-
13	*abStatus*	2	-	0000h：成功 0001h：BAD_PARAMETER
15	*abData*	8	-	8 字节的认证数据

7.2.5. RDR_to_PC_NotifySlotChange

APG8201 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个 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）。 如果在指定位置不存在插槽，则该字段的这 2 个位返回 00b。 例如：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

7.2.6. RDR_to_PC_HardwareError

设置 *bHardwareErrorCode* 的任意位都会发送此消息。

偏移	字段	大小	值	说明
0	*bMessageType*	1	51h	-
1	*bSlot*	1	00-FFh	ICC 插槽号
2	*bSeq*	1	00-FFh	发生硬件错误时，bulk out 命令的序号
3	*bHardwareErrorCode*	1	XYh	该值表示根据以下值执行的按位异或操作。 01h = 过载电流 更多错误条件和值待定义。

如需了解详细的操作流程，请参见 CCID 规范。

8.0. 连机操作模式

连机模式下，用户可选择 SPE 功能进行卡片和读卡器之间的认证。脱机模式下，用户可选择脱机模式 PLA 功能和一些附加功能，例如计算器。

连机模式下，为了执行卡片和读卡器之间的一些基本操作，用户可选择 PCSC API SCardConnect、SCardTransmit、SCardControl 等发送命令给读卡器。SCardControl API 可用于 PIN 码安全验证，PIN 码安全更改和不同直接命令（Escape Comamnd）间的通讯。

8.1. SCardConnect API

```
LONG WINAPI SCardConnect(
  _In_   SCARDCONTEXT hContext,
  _In_   LPCTSTR szReader,
  _In_   DWORD dwShareMode,
  _In_   DWORD dwPreferredProtocols,
  _Out_  LPSCARDHANDLE phCard,
  _Out_  LPDWORD pdwActiveProtocol
);
```

8.2. SCardTransmit API

```
LONG WINAPI SCardTransmit(
  _In_          SCARDHANDLE hCard,
  _In_          LPCSCARD_IO_REQUEST pioSendPci,
  _In_          LPCBYTE pbSendBuffer,
  _In_          DWORD cbSendLength,
  _Inout_opt_   LPSCARD_IO_REQUEST pioRecvPci,
  _Out_         LPBYTE pbRecvBuffer,
  _Inout_       LPDWORD pcbRecvLength
);
```

8.3. SCardControl API

```
LONG WINAPI SCardControl(
  _In_   SCARDHANDLE hCard,
  _In_   DWORD dwControlCode,
  _In_   LPCVOID lpInBuffer,
  _In_   DWORD nInBufferSize,
  _Out_  LPVOID lpOutBuffer,
  _In_   DWORD nOutBufferSize,
  _Out_  LPDWORD lpBytesReturned
);
```

8.4. PIN 码安全验证

此应用发送登录命令（PIN 码安全输入）

例如. “00 20 00 01 08 24 12 34 FF FF FF FF FFh”

用户输入 PIN 码后，读卡器发送 SECURE PIN VERIFY 命令给卡片。

特定的状态码：

SW1 SW2	含义
90 00h	没有错误。
63 Cxh	错误的 PIN 码

SW1 SW2	含义
64 01h	取消输入 PIN 码

对于现有的 ACS 智能卡，PIN 码输入次数设为 3。如果第三次输入仍错误，智能卡将被锁定。

注：*对于 APG8201 一般读卡器，无论输入的 PIN 码是否正确，都不会显示任何文本。*

8.5. PIN 码安全更改

可用以下两种方法调用 SECURE PIN MODIFY 命令：

1. 对于显式 Secure Pin Modify 命令，主机应分别发送两个命令，即 SECURE PIN ENTRY 命令和 SECURE PIN MODIFY，例如：

   PIN 码安全输入（主机->读卡器）：69 1C 00 00 00 00 F3 00 00 00 00 00 89 47 04 0C 04 07 01 09 04 00 00 00 00 00 20 00 02 08 2C FF FF FF FF FF FF FFh

   LCD 显示屏："Enter auth. PIN:"

   用户输入 12 位长的 PUK 码：例如 3 3 3 3 3 3 1 1 1 1 1 1

   LCD 显示屏："Card inserted"

   PIN 码安全输入（读卡器->主机）：80 02 00 00 00 00 F3 00 00 00 90 00h

   PIN 码安全更改（主机->读卡器）：69 1F 00 00 00 00 F4 00 00 00 01 00 89 47 04 00 00 0C 04 00 03 01 09 04 01 00 00 00 00 24 01 01 08 24 FF FF FF FF FF FF FFh

   LCD 显示屏："NEW PIN: (key)"

   用户输入 4 位长的新 PIN 码：例如 1 2 3 4，然后按"OK"

   LCD 显示屏："CONFIRM PIN: (key)"

   用户再次输入 4 位长的新 PIN 码：例如 1 2 3 4，然后按"OK"

   LCD 显示屏："Card inserted"

   PIN 码安全更改（读卡器->主机）：80 02 00 00 00 00 F4 00 00 00 90 00h

2. 对于隐式 SECURE PIN MODIFY，主机应发送命令以进行验证和更改，例如：

   PIN 码安全更改（主机->读卡器）：69 29 00 00 00 00 CF 00 00 00 01 00 89 47 04 00 08 0C 04 03 03 03 09 04 00 01 02 00 00 00 00 24 00 01 10 24 FF FF FF FF FF FF FF 24 FF FF FF FF FF FF FFh

   LCD 显示屏："Enter auth. PIN:"

   用户输入 4 位长的正确 PIN 码：例如 1 2 3 4，然后按"OK"

   LCD 显示屏："NEW PIN: (key)"

   用户输入 4 位长的新 PIN 码：例如 4 3 2 1，然后按"OK"

   LCD 显示屏："CONFIRM PIN: (key)"

   用户再次输入 4 位长的新 PIN 码：例如 4 3 2 1，然后按"OK"

   LCD 显示屏："Card inserted"

   PIN 码安全更改（读卡器->主机）：80 02 00 00 00 00 CF 00 00 00 90 00h

如果隐式命令中的旧 PIN 码错误,卡片响应应为"63 Cxh",其中 x 是剩余的尝试输入次数，如果达到该次数后输入的 PIN 码仍然是错误的，会显示错误码“69 83h”。

8.6. 直接命令

读卡器提供不同的直接命令，例如获取读卡器的特定标签（Get Reader Specific Tag）、获取固件版本号（Get Firmware Version），显示 LCD 消息（Display LCD Message）、 读取键盘的键值（Read Key from keypad），蜂鸣器响（Buzzer Beep）以及一些其他仅用于生产的命令。

8.7. 获取固件版本号

直接命令码“04 00 00 00 00h”用于获取固件版本号。

获取固件版本号（主机->读卡器）04 00 00 00 00h

获取固件版本号（读卡器->主机）84 00 02 00 00 30 31 39 5Ah

8.8. 显示 LCD 消息

直接命令码“05 00 20 00 00h…”用于在 LCD 屏幕上显示文本。

获取固件版本号（主机->读卡器）：05 00 20 00 00 31 32 33 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20h

获取固件版本号（读卡器->主机）85 00 00 00 00h

8.9. 读取密钥

用户使用直接命令码“06 00 06 00 00 00 08 04 01 00 00h”读取键盘的键值。

获取固件版本号（主机->读卡器）：06 00 06 00 00 00 08 04 01 00 00h

获取固件版本号（读卡器->主机）：86 00 09 00 00 31 31 32 33 34 35 36 37 38h

8.10. 蜂鸣器响

直接命令码“08 00 00 00 00h”用于发出哔声。

蜂鸣器响（主机->读卡器）08 00 00 00 00h

蜂鸣器响（读卡器->主机）88 00 00 00 00h

9.0. 设备控制

本节将对系统智能卡设备的 IOCTL 进行介绍。

9.1. 操作流程（PC/SC 2.0 第 10 部分）

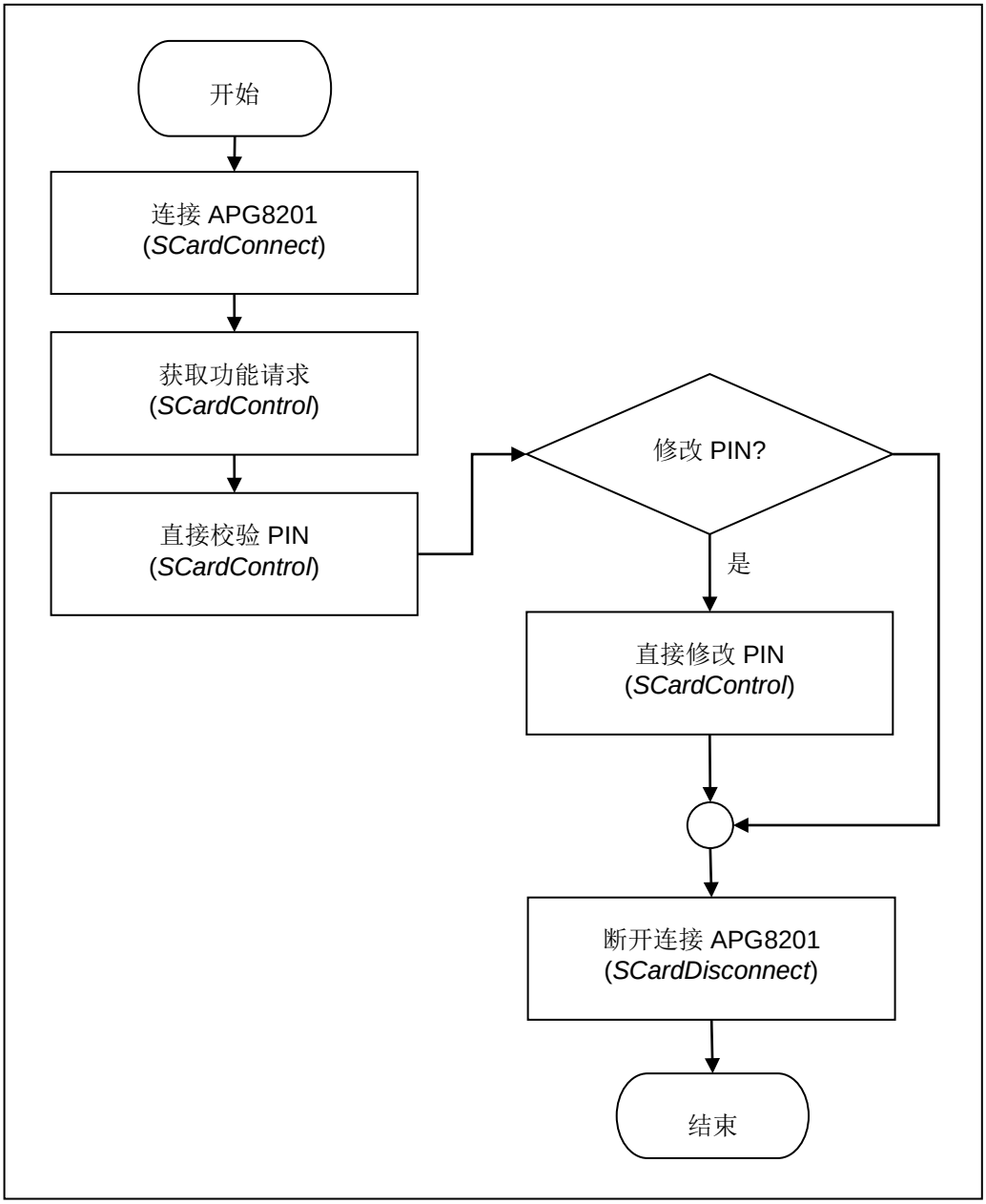


图1 :操作流程图

要使用 PIN 码验证和修改功能，*SCardControl* API 在调用时必须带有“获取功能请求”控制码，此 API 会返回读写器支持的功能列表。

APG8201 仅支持直接校验 PIN、直接修改 PIN 以及 IFD PIN 属性三种功能。要使用这些功能，您可以从此列表中获得相应的控制码。如需了解更多信息，请参考 PC/SC 2.0 规范第 10 部分。

9.2. 具体的 ScardControl 函数

```
LONG SCardControl(
SCARDHANDLE hCard,
DWORD dwControlCode,
LPCVOID lpInBuffer,
DWORD nInBufferSize,
LPVOID lpOutBuffer,
DWORD nOutBufferSize,
LPDWORD lpBytesReturned
);
#define IOCTL_SMARTCARD_GET_FIRMWARE_VERSION SCARD_CTL_CODE(2078)
#define IOCTL_SMARTCARD_DISPLAY_LCD_MESSAGE SCARD_CTL_CODE(2079)
#define IOCTL_SMARTCARD_READ_KEY SCARD_CTL_CODE(2080
// PC/SC 2.0 Part 10
#define CM_IOCTL_GET_FEATURE_REQUEST SCARD_CTL_CODE(3400)
```

注：*数据以小端格式存储，最低有效字节（LSB）在前面。另外，必须在源代码中对 SCardControl 命令予以声明。*

9.3. 智能卡设备 IOCTL

9.3.1. CM_IOCTL_GET_FEATURE_REQUEST

CM_IOCTL_GET_FEATURE_REQUEST 会返回读写器支持的功能列表。

hCard	由函数 *SCardConnect* 返回的引用值
dwControlCode	CM_IOCTL_GET_FEATURE_REQUEST
lpInBuffer	NULL
nInBufferSize	必须是 *lpInBuffer* 的 **sizeof**(ULONG)
lpOutBuffer	根据 PC/SC 2.0 规范第 10 部分的规定，定义了以下功能：

```
#define FEATURE_VERIFY_PIN_START 0x01
#define FEATURE_VERIFY_PIN_FINISH 0x02
#define FEATURE_MODIFY_PIN_START 0x03
#define FEATURE_MODIFY_PIN_FINISH 0x04
#define FEATURE_GET_KEY_PRESSED 0x05
#define FEATURE_VERIFY_PIN_DIRECT 0x06
#define FEATURE_MODIFY_PIN_DIRECT 0x07
#define FEATURE_MCT_READERDIRECT 0x08
#define FEATURE_MCT_UNIVERSAL 0x09
#define FEATURE_IFD_PIN_PROP 0x0A
#define FEATURE_ABORT 0x0B
```

APG8201 支持以下功能：

```
#define FEATURE_VERIFY_PIN_DIRECT 0x06
#define FEATURE_MODIFY_PIN_DIRECT 0x07
#define FEATURE_IFD_PIN_PROP 0x0A
```

如果使用的APG8201支持PC/SC 2.0第10部分规定，您将获得以下数据：

06 04 XX XX XX XX 07 04 XX XX XX XX 0A 04 XX XX XX XXh

其中，XX XX XX XXh 为功能的控制代码。

nOutBufferSize *lpOutBuffer* 的 **sizeof**(ULONG)

lpBytesReturned 指向一个 *DWORD* 变量的指针，该变量用于接收存储进缓冲区的数据的大小（字节数），而该缓冲区由 *lpOutBuffer* 指定。

9.3.2. FEATURE_VERIFY_PIN_DIRECT

hCard 由函数 *SCardConnect* 返回的引用值

dwControlCode CM_IOCTL_GET_FEATURE_REQUEST

lpInBuffer

偏移	数据域	大小	值	说明
0	*bTimeOut*	1	-	秒数。若值等于 00h，则使用默认值
1	*bTimeOut2*	1	00h	不支持。第一次按键后的秒数
2	*bmFormatString*	1	-	PIN 码格式选项的几个参数，更多信息请参阅 **bmFormatString 说明**。
3	*bmPINBlockString*	1	-	定义 APDU 命令中 PIN 数据块的字节长度。更多信息请参阅 **bmPINBlockString 说明**。
4	*bmPINLengthFormat*	1	-	允许在 APDU 命令中加入 PIN 码长度。更多信息请参阅 **bmPINLength 格式**。
5	*wPINMaxExtraDigit*	2	XXYYh	XXh：PIN 码最大长度（位数） YYh：PIN 码最小长度（位数）
7	*bEntryValidationCondition*	1	-	该值是一个位 OR 运算 01h = 达到最大长度 02h = 按下确认键 04h = 出现超时
8	*bNumberMessage*	1	FFh	PIN 验证显示的消息数量
9	*wLangId*	2	0409h	消息的语言
11	*bMsgIndex*	1	00h	消息索引（应为 00h）
12	*bTeoPrologue*	3	000000h	要使用的 T=1 信息块（I-block）起始域（填写 00h）
15	*ulDataLength*	4	-	待发送至 ICC 的数据的长度。
19	*abData*	-	-	待发送至 ICC 的数据

nInBufferSize 19 + *uLDataLength*

lpOutBuffer

偏移	数据域	大小	值	说明
0	*abStatus*	2	-	6400h：SPE 操作超时 6401h：通过“取消”按钮取消了 SPE 操作。 6402h：两次输入的“新 PIN”不一致，PIN 修改操作失败 6403h：用户输入的 PIN 太短或太长，不符合最短/最长 PIN 码长度限制 ***注：APG8201 在 PIN 输入过程中检查 PIN 的长度，将不再返回此状态。*** 6B80h：传递的结构参数无效 SW1SW2：来自卡片的结果

nOutBufferSize 2

lpBytesReturned 指向一个 *DWORD* 变量的指针，该变量用于接收存储进缓冲区的数据的长度（字节数），而该缓冲区由 *lpOutBuffer* 指定。

9.3.3. FEATURE_MODIFY_PIN_DIRECT

hCard 由函数 *SCardConnect* 返回的引用值

dwControlCode CM_IOCTL_GET_FEATURE_REQUEST

lpInBuffer

偏移	数据域	大小	值	说明
0	*bTimeOut*	1	-	秒数。若值等于 00h，则使用默认值
1	*bTimeOut2*	1	00h	不支持。第一次按键后的秒数
2	*bmFormatString*	1	-	PIN 码格式选项的几个参数，更多信息请参阅 **bmFormatString 说明**。
3	*bmPINBlockString*	1	-	定义 APDU 命令中 PIN 数据块的长度（字节数）。更多信息请参阅 **bmPINBlockString 说明**。
4	*bmPINLengthFormat*	1	-	允许在 APDU 命令中加入 PIN 码长度。更多信息请参阅 **bmPINLength 格式**。
5	*bInsertionOffsetOld*	1	-	当前 PIN 码的插入位置偏移（字节）

偏移	数据域	大小	值	说明
6	*bInsertionOffsetNew*	1	-	新 PIN 码的插入位置偏移（字节）
7	*wPINMaxExtraDigit*	2	XXYYh	XXh：PIN 码最大长度（位数） YYh：PIN 码最小长度（位数）
9	*bConfirmPIN*	1	00h, 01h, 02h, 03h	表示是否要在新的 PIN 码生效前进行确认（意思是用户要输入两次新 PIN 后，该 PIN 才会生效） 表示是否要在相同的 APDU 域输入并设置当前 PIN 码 b0：(0/1) 如果为 0 = 无需进行确认 如果为 1 = 需要进行确认 b1：(0/1) 如果为 0 = 无需输入当前 PIN（在此情况下，*bInsertinoOffsetOld* 的值不能被考虑在内） 如果为 1 = 需要输入当前 PIN b2 – b7：RFU
10	*bEntryValidationCondition*	1	-	该值是一个位 OR 运算 01h = 达到最大长度 02h = 按下确认键 04h = 出现超时
11	*bNumberMessage*	1	FFh	PIN 验证显示的消息数量
12	*wLangId*	2	0409h	消息的语言
14	*bMsgIndex1*	1	00h	第一条提示信息的索引
15	*bMsgIndex2*	1	01h	第二条提示信息的索引
16	*bMsgIndex3*	1	02h	第三条提示信息的索引
17	*bTeoPrologue*	3	000000h	要使用的 T=1 信息块（I-block）起始域 （填写 00h）。
20	*ulDataLength*	4	-	待发送至 ICC 的数据的长度。
24	*abData*		-	待发送至 ICC 的数据

nInBufferSize 24 + ulDataLength

lpOutBuffer

偏移	数据域	大小	值	说明
0	*abStatus*	2	-	6400h：SPE 操作超时 6401h：通过“取消”按钮取消了 SPE 操作。 6402h：两次输入的“新 PIN”不一致，PIN 修改操作失败 6403h：用户输入的 PIN 太短或太长，不符合最短/最长 PIN 码长度限制 ***注：****APG8201 在 PIN 输入过程中检查 PIN 的长度，将不再返回此状态。* 6B80h：传递的结构参数无效 SW1SW2：来自卡片的结果

nOutBufferSize 2

lpBytesReturned 指向一个 *DWORD* 变量的指针，该变量用于接收存储进缓冲区的数据的大小（字节数），而该缓冲区由 *lpOutBuffer* 指定。

9.3.4. FEATURE_IFD_PIN_PROP

hCard 由函数 SCardConnect 返回的引用值。

dwControlCode 由 CM_IOCTL_GET_FEATURE_REQUEST 返回。

lpInBuffer NULL

LpOutBuffer

偏移	数据域	大小	值	说明
0	*wLcdLayout*	2	0210h	显示特性：2 行，每行 16 个字符
2	*bEntryValidationCondition*	1	07h	支持超时时间到、达到 PIN 码最大长度以及按下确认按钮这三种条件
3	*bTimeOut2*	1	00h	0 = IFD 不能识别 *bTimeOut* 和 *bTimeOut2* 1 = IFD 可以识别 *bTimeOut* 和 *bTimeOut2*

nOutBufferSize 4

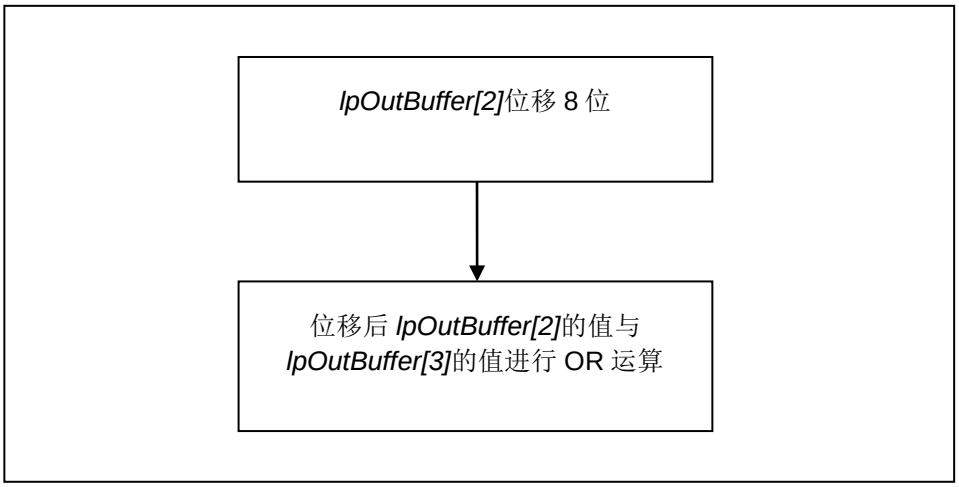
lpBytesReturned 指向一个 *DWORD* 变量的指针，该变量用于接收存储进缓冲区的数据的大小（字节数），而该缓冲区由 *lpOutBuffer* 指定。

9.3.5. IOCTL_SMARTCARD_GET_FIRMWARE_VERSION

IOCTL_SMARTCARD_GET_FIRMWARE_VERSION 用于启用 *Get Firmware Version* 命令。

9.3.5.1. 固件版本号

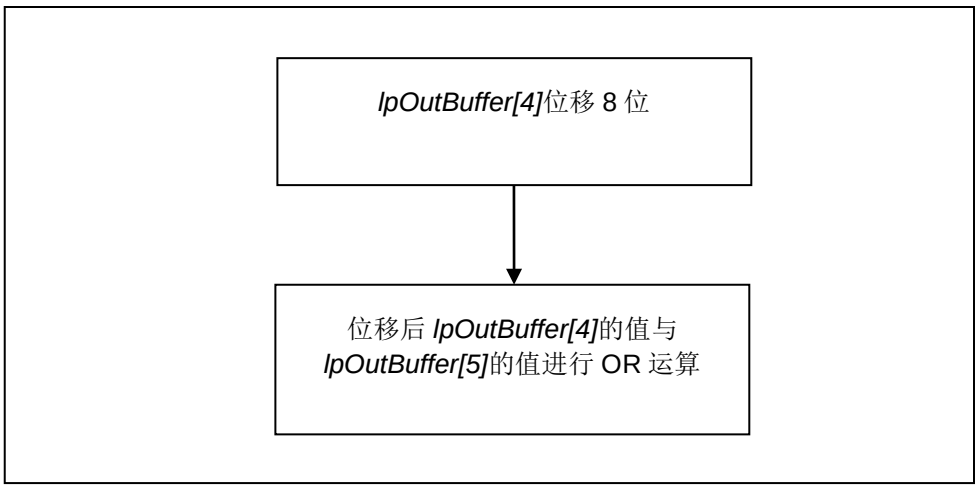
为了获得设备的固件版本号，要将接收到的缓冲数据的第三个数据元位移 8 位，之后再将位移后的结果与缓冲数据的第四个数据元进行 OR 运算。



例如：*Firmware_Version = (Common.RecvBuff[2] << 8) | Common.RecvBuff[3]);*

9.3.5.2. LCD

为了获得设备 LCD 的信息，要将接收到的缓冲数据的第五个数据元位移 8 位，之后再将位移后的结果与缓冲数据的第六个数据元进行 OR 运算。



输入数据：

hCard	由函数 *SCardConnect* 返回的引用值
dwControlCode	IOCTL_SMARTCARD_GET_FIRMWARE_VERSION

输出数据：

lpOutBuffer	命令输出的值

nOutBufferSize *lpOutBuffer* 的 **sizeof**(ULONG)

lpBytesReturned 指向一个 *DWORD* 变量的指针，该变量用于接收存储进缓冲区的数据的大小（字节数），而该缓冲区由 *lpOutBuffer* 指定。

偏移	数据域	大小	值	说明
0	*abStatus*	2	0000h	成功
2	*wACR83Firmware*	2		-
4	*LCD*	2		-

9.3.6. IOCTL_SMARTCARD_DISPLAY_LCD_MESSAGE

IOCTL_SMARTCARD_DISPLAY_LCD_MESSAGE 用于启用 *Display LCD Message* 命令。

hCard 由函数 *SCardConnect* 返回的引用值

dwControlCode IOCTL_SMARTCARD_DISPLAY_LCD_MESSAGE

lpInBuffer 设置 Display LCD Message 选项的值

nInBufferSize *lpInBuffer* 的 **sizeof**(ULONG)

偏移	数据域	大小	值	说明
0	*abLCDmessage*	0-32	-	LCD 消息（最多 32 个字符）

输出数据：

lpOutBuffer 命令输出的值

nOutBufferSize *lpOutBuffer* 的 **sizeof**(ULONG)

lpBytesReturned 指向一个 *DWORD* 变量的指针，该变量用于接收存储进缓冲区的数据的大小（字节数），而该缓冲区由 *lpOutBuffer* 指定。

偏移	数据域	大小	值	说明
0	*abStatus*	2	0000h 0001h	成功 BAD_PARAMETER

9.3.7. IOCTL_SMARTCARD_READ_KEY

IOCTL_SMARTCARD_READ_KEY 用于启用 *Read Key* 命令。

输入数据：

hCard 由函数 *SCardConnect* 返回的引用值

dwControlCode IOCTL_SMARTCARD_READ_KEY

lpInBuffer 设置 Display LCD Message 选项的值

nInBufferSize *lpInBuffer* 的 **sizeof**(ULONG)

偏移	数据域	大小	值	说明
0	*bTimeOut*	1	-	秒数。若值等于 00h，则使用默认值。
1	*wPINMaxExtraDigit*	2	XXYYh	XXh：PIN 码最大长度（位数） YYh：PIN 码最小长度（位数）
3	*bKeyReturnCondition*	1	-	该值是一个位 OR 运算 01h：达到最大长度 02h：按下了按键[E] 04h：发生超时 08h：按下了按键[C]
4	*bEchoLCDStartPosition*	1	-	开始位置（0 – 31）
5	*bEchoLCDMode*	1	-	00h：通过按键输入的数据在 LCD 上显示为 ASCII 格式 01h：通过按键输入的数据在 LCD 上显示为星号“*”。

输出数据：

lpOutBuffer 命令输出的值

nOutBufferSize *lpOutBuffer* 的 **sizeof**(ULONG)

lpBytesReturned 指向一个 *DWORD* 变量的指针，该变量用于接收存储进缓冲区的数据的大小（字节数），而该缓冲区由 *lpOutBuffer* 指定。

偏移	数据域	大小	值	说明
0	*abStatus*	2	0000h 0001h	成功 BAD_PARAMETER
2	*bKeyReturnCondition*	1	31h 32h 33h 34h	达到最大长度 按下了按键[E] 发生超时 按下了按键[C]
3	*abNumericInputKeys*	0-32	-	-

附录A. 设置 bKeyReturnCondition

bKeyReturnCondition	OR 操作数
若达到 PIN 码最大长度	01h
若 APG8201 设备的 KEY_E 被按下	02
若 APG8201 会话超时时间到	04h
若 APG8201 设备的 KEY_C 被按下	08h
若 APG8201 设备的 KEY_BACK 被按下	10h
若 APG8201 设备的 KEY_FN 被按下	20h

注：*将值设为对特定的 OR 操作数进行 OR 运算。*

附录B. 响应错误代码

下表汇总了 APG8201（CCID）可能返回的错误代码：

错误代码	状态
0001h	BAD_PARAMETER
0083h	SLOTERROR_LCDCOMMANDERROR
0084h	SLOTERROR_WRONGCONFIRMPIN
0085h	SLOTERROR_UNKNOWN_LCD
0086h	SLOTERROR_MAXPINSIZE_EQUAL_ZERO
00EFh	SLOTERROR_PIN_CANCELLED
00F0h	SLOTERROR_PIN_TIMEOUT

附录C. bmFormatString 说明

位号	说明
Bit 7	系统单位的类型指示符 若为 0h：系统的单位是比特（Bit） 若为 1h：系统的单位是字节（Byte） 该位可量化下一个参数（单位移动）
Bit 6 – 3	定义格式化后 PIN 在 APDU 命令中的位置（相对于 Lc 后的第一个数据）。该位置基于系统单位的类型指示符来确定（最大值为 1111，15 个系统单位）
Bit 2	PIN 对齐的位掩码 若为 0h：数据左对齐 若为 1h：数据右对齐
Bit 1-0	PIN 格式类型 00h：二进制 01h：BCD 10h：ASCII

附录D. bmPINBlockString 说明

位号	说明
Bit 7 - 4	加入 APDU 命令的 PIN 长度的大小，单位为比特。（如果该值为 0，则不在 APDU 命令中加入有效的 PIN 长度）。
Bit 3 - 0	PIN 长度信息：经过对齐和格式化后 PIN 数据块的大小，单位为字节

附录E. bmPINLength 格式

位号	说明
Bit 7-5	RFU
Bit 4	系统单位的类型指示符 若为 0h：系统的单位是比特（Bit） 若为 1h：系统的单位是字节（Byte）
Bit 3 - 0	根据前述参数，表示 PIN 长度在 APDU 命令中的位置（最大值为 1111，15 个系统单位）。