PC/SC 兼容型读卡器
通用技术手册

(Revision 2.33)

北京金木雨电子有限公司

2018/8/16

在使用本产品前请详细阅读本说明书，如果有任何疑问，请联系我们，我们会给您详尽的解答

目录

1 简介

本系列读卡器采用 PC/SC USB 接口，在 Windows 下初次连接时需要安装 PC/SC 的驱动程序（CCID，在光盘上可以找到）。PC/SC 接口采用 Windows 操作系统自带驱动和 API 函数，优点是开发相对简单。

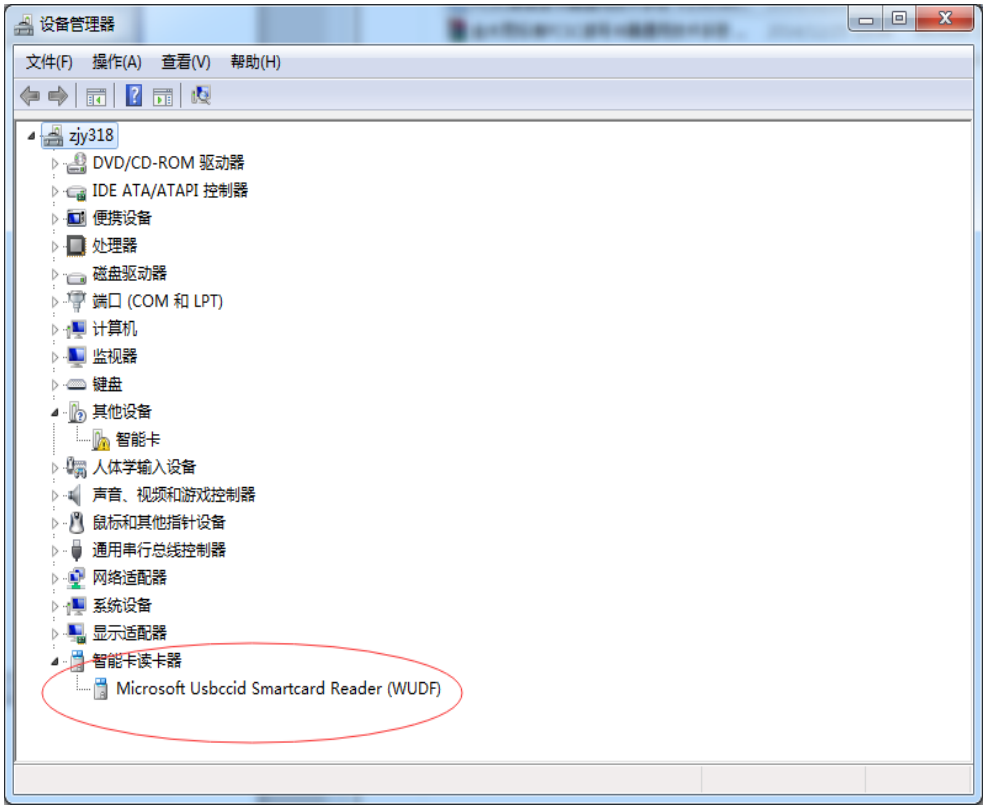
本系列读卡器采用兼容方式的 PC/SC 接口，与标准 PC/SC 有少许差异，这是因为为了兼容更多种类的卡片。标准的 PC/SC 一般只支持 ISO14443A 和 ISO14443B，在读卡器中有一个针对这些卡片的自动寻卡流程，而其他种类的卡片不方便参与到这个流程中，因此我们设计了发指令进行寻卡的操作方式，这是与标准 PC/SC 读卡器的区别所在。

为了便于开发者的应用，我们提供了 VC、BC、VB、DELPHI 例子程序，开发者可以通过例子程序快速地开展开发工作。如果在编写程序中依然有任何的问题，请随时联系我们的技术支持，或发送电子邮件到：jinmuyu@vip.sina.com 我们会给您满意的答复。

2 设备安装

2.1 驱动安装

将读卡器连接到电脑，安装驱动程序（产品光盘：\Chinese\桌面读写器\PCSC Interface\CCID Driver），安装后执行如下步骤可以检测读卡器是否连接好：**计算机->属性->设备管理器。**如下可以看到标注红色部分的 **Microsoft Usbccid Smartcard Reader(WUDF);**



2.2 上电复位信息（ATR）

按照 PC/SC Part3 协议规定，设备上电返回 SmartCard 复位信息 ATR，为了使读卡器能够阅读更多非接触卡，MR800 采用返回固定的复位信息（未包括卡片信息），ATR 信息格式如下：

Byte	Value（Hex）	Designation	Description
0	3B	Initial Header	
1	8N	T0	Higher nibble 8 means：no TA1，TB1，TC1 only TD1 is following. Lower nibble N is the number of historical bytes （HistByte 0 to HistByte N-1）
2	80	TD1	Higher nibble 8 means：no TA2，TB2，TC2 only TD2 is following. Lower nibble 0 means T = 0
3	01	TD2	Higher nibble 0 means no TA3，TB3，TC3，TD3 following. Lower nibble 1 means T = 1
4To3+N	80	T1	Category indicator byte，80 means A status indicator may be present in an optional COMPACT-TLV data object
	4F	Tk	Application identifier Presence Indicator
	0C		Length
	RID		Registered Application Provider Identifier （RID） # A0 00 00 03 06
	SS		Byte for standard
	C0 .. C1		Bytes for card name
	00 00 00 00	RFU	RFU # 00 00 00 00
4+N	UU	TCK	Exclusive-oring of all the bytes T0 to Tk

针对 MR800 读卡器，我们返回 ATR 信息如下：ATR = {3B 8F 80 01 80 4F 0C A0 00 00 03 06 00 00 00 00 00 00 00 68}

3 基本操作原则

PC/SC 兼容读卡器将 APDU 分为标准 APDU（APDU 中 Class 为非 0xFF）和非标准 APDU（APDU 中 Class 是 0xFF）。为了兼容 PC/SC 标准，对于非接触 SmartCard 和接触式 SAM 卡，除了 GetData 获取卡复位信息外，其余的标准 APDU 可以直接发送到 SmartCard 或 SAM 卡。因本系列读卡器支持非接触智能卡和接触智能卡（SAM），故在操作前也可以通过切换当前操作智能卡 APDU（APDU：FF 00 FA 00 01 CurSmartCard）切换当前操作智能卡（此处指的是接触和非接触智能卡之间的切换）。卡片操作流程见后面章节。对于存储卡，我们采用的是 Class =FF 非标准 APDU 指令操作，指令描述见后面章节。

不论是非接触 SmartCard，接触式 SAM 卡还是存储卡，所有对卡片的操作第一个步骤都是通过 GetData APDU 去获取卡片信息。

3.1 非接触智能卡（SmartCard）

非接触智能卡采用的是标准 APDU 指令，在发送标准 APDU 指令前我们需要通过 GetData 指令获取 SmartCard ATR 数据。若在操作过程中需要读取接触智能卡 SAM，需要通过指令切换到指定的 SAM Slot（APDU：FF 00 FA 00 01 CurSmartCard）去读取相关数据。

3.2 接触智能卡（SAM）

本系列读卡器都带有多个 SAM 插槽，在发送标准 APDU 指令前我们需要通过 GetData 指令获取 SAM 卡复位信息。若在操作过程中需要读取非接触智能卡，则需要通过切换指令切换到非接触 SmartCard。

如：读卡器所读非接触卡类型，在操作过程中需通过 SAM 数据认证。

3.3 非接触存储卡

本系列读卡器支持如 MIFARE One/Ultralight 等存储卡，为了兼容 PC/SC 标准，我们定义了非标准 APDU，在发送非标准 APDU 指令前我们需要通过 GetData 指令去寻卡，获取卡片序列号信息。

4 非标准 APDU 指令详解

4.1 返回状态信息

除了 GetData APDU 既可以对存储卡，也可以 SmartCard/SAM 进行操作外，其它非标准 APDU 主要是用来实现存储类卡片的操作；标准 APDU 主要是用来对 SmartCard/SAM 类卡片的操作。

返回信息状态如下（SW1/SW2）：

结果	SW1	SW2	错误注释
成功	90	00	操作成功
错误	63	00	操作失败
错误	6A	81	功能不支持
错误	6B	00	P1-P2参数错误

4.2 PC/SC Part3 部分

4.2.1 GetData

该 APDU 指令是获取卡片序列号或复位信息。在操作一张卡片前，须首先执行该 APDU，因其中包含了对读卡器读卡类型的切换。

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
GetData	FF	CA	CardType	SubCardType	00

CardType 和 SubCardType 定义如下：

ISO	CardType	SubCardType
ISO14443 Type A	00：ISO14443 A MIFARE card	00
	01：ISO14443 A Smartcard（ISO14443-4）	00
	02：MIFARE Ultra Light	00
	03：MIFARE Plus	00：MIFARE PLUS Level 0
		01：MIFARE PLUS Level 1
		02：MIFARE PLUS Level 2
		03：MIFARE PLUS Level 3
		04：MIFARE PLUS Level 1 for switch level
ISO14443 Type B	20：ISO14443 B Smartcard（ISO14443-4）	00
	21：SR176	00
	22：SRIX4K/SRI512	00
	23：AT88RF020	00
ISO15693	40：ISO15693 Tag（Only one Tag）	00（NXP/TI Tag）

ISO7816	60：ISO7816-Contact（T=0/T=1）	00：SAM1
		01：SAM2
		02：SAM3
		03：SAM4

MIFARE 1K/4K/UlraLight/MIFARE Plus Level 1 （P1 = 00/02/03）应答：

Response	**Data Out**		
Result	UID Len（1Byte） + UID （LSB- 4/7Byte） + ATQA（2byte） + SAK（1Byte）	SW1	SW2

MIFARE Plus Level 0/2/3/1 for switch 和 ISO14443 - 4 TypeA SmartCard （P1 = 03/01）应答：

Response	**Data Out**		
Result	UID Len（1Byte） + UID （LSB- 4/7Byte） + ATQA（2byte） + SAK（1Byte） +ATQA（nByte）	SW1	SW2

ISO14443 - 4 TypeB SmartCard/AT88F020（P1=20/23） 应答：

Response	**Data Out**		
Result	ATQB（12Byte）	SW1	SW2

SR176/SRIX4K（SRI512）（P1=21/22） 应答：

Response	**Data Out**		
Result	CHIPID（1Byte）+UID（8Byte）	SW1	SW2

ISO15693 Tag（P1=40）应答：

Response	**Data Out**		
Result	DSFID（1Byte）+UID（8Byte）	SW1	SW2

ISO7816 SAM（P1=60）应答：

Response	**Data Out**		
Result	Reset Info（nByte）	SW1	SW2

例如：

寻 TypeA 卡片：

Send： FF CA 00 00 00

Receive： 04 72 AE A6 9E 04 00 08 90 00

寻 ISO14443 TypeA Smartcard：

Send： FF CA 01 00 00

Receive： 04 50 3D CE EB 08 03 20 11 28 A1 53 43 41 5F 4F 5F 56 31 30 30 5F 54 64 90 00

ISO14443 TypeB SmartCard：

Send： FF CA 20 00 00

Receive： 50 C0 1281 89 54 46 22 08 00 80 A1 90 00

4.2.2 LoadKey

该 APDU 是用来保存卡片授权密钥和密钥传输时加密密钥。装载的密钥可以选择保存还是不保存，不保存的密钥暂时存放在 RAM 中，断电易失；保存的密钥保存于 Flash，断电后不丢失。MR800 最多保存卡片密钥个数是 32，且每个密钥最大长度是 16 字节，若授权密钥小于 16 字节，则取低字节密钥授权。最多保存读卡器密钥个数是 1 条。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
Load Key	FF	82	Key Structure	Key Index	1~16	Key Data （LSB）

Key Structure：

b7	b6	b5	b4	b3	b2	b1	b0	Description
X								0：卡片密钥 1：读卡器密钥
	X							0：明文传输 1：密文传输
		X						0：暂时存储 1：非易失性存储
			X	X	X	X	X	RFU

卡片密钥是用来对卡片授权的密钥，**读卡器密钥**是对卡片密钥载入时的加密密钥。加密方式是 3DES 加密，所以读卡器密钥必须是 16 字节。所有加密的卡片密钥必须是 8 字节的倍数，不够的在高字节补 00，如 MIFARE One 密钥是 FF FF FF FF FF FF 6 字节密钥，假如密钥下载选择密文传输，则先补 0 为 FF FF FF FF FF FF 00 00 （LSB..MSB） 然后再加密。若明文传输则不需要补 0。出厂默认所有密钥都为 0。

密钥存储结构：

Key Index	卡片密钥（Byte）	读卡器密钥（Byte）
0	16	16
1	16	-
……	16	-
31	16	-

（备注：卡片密钥索引 0~31，读卡器密钥索引只有 0）

应答：

Response	Data Out	
Result	SW1	SW2

例如：

明文传输 ReaderKey，不保存：

Send： FF 82 80 00 10 33 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF

Receive： 90 00

4.2.3 Authentication Command

该 APDU 主要用在对带有密钥保护的卡片进行授权。在 GetData 指令后，若卡片带有密钥保护功能，则需要通过这个 APDU 对卡片授权后才能对其进行读写操作。一般需要授权的卡片有：MIFARE S50/70、MIFARE Plus、AT88F020。授权可以采用已经存储的密钥或当前下载的密钥授权两种方式中的任意一种。

发送 APDU 格式（旧的 PC/SC 标准，不推荐使用）：

Command	Class	INS	P1	P2	P3	Data
Authenticate	FF	88	High Address	Low Address	Key Type	KeyCofig + KEY

High Address / Low Address：

对于 MIFARE S50/70 则卡片块地址。

对于 AT88F020，则该地址无效（P1=0，P2=0）。

对于 MIFARE Plus Level 1/2/3，则为 AES 密钥存储块的地址（注意 密钥存储块和数据块是一一对应关系，请参考 MIFARE Plus 数据手册）。

KeyType：密钥类型（仅仅在 MIFARE S50/S70，该字节有效：A Key－0x60，B Key－0x61）。

KeyConfig：

b7	b6-b0	Meaning
0	XXXXXXX	XXXXXXX表示用当前输入密钥KEY的长度，卡片采用当前密钥授权
1	XXXXXXX	XXXXXXX表示存储于读卡器密钥索引，卡片采用存储的密钥授权

注：此处与 PC/SC V2.01 版本协议有差别，需要最高位为 bit7=1，bit6~bit0 为密钥索引号。

KEY：若 KeyConfig Bit7 = 0，Key 表示密钥，密钥长度根据卡片类型的不同而不同；若 KeyConfig Bit7 = 1，Key 内容不存在。

应答：

Response	Data Out	
Result	SW1	SW2

例如：

寻 MIFARE S50 卡片，并且读第一块：

Send： FF CA 00 00 00

Receive： 04 72 AE A6 9E 04 00 08 90 00

Send： FF 88 00 01 60 06 FF FF FF FF FF FF

Receive： 90 00

Send： FF B0 00 01 10

Receive： 00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF 90 00

寻 MIFARE Plus Level 3 卡片，并且读数据块 0：

Send： FF CA 03 03 00

Receive： 07 04 8B AD 04 05 06 07 42 00 31 0C 75 77 84 02 4D 46 50 5F 45 4E 47 90 00

Send： FF 88 40 00 00 10 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF（数据块 1 对应的密钥地址是 0x4000 或 0x4001）

Receive： 90 00

Send： FF B0 00 01 10

Receive： 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 90 00

寻 AT88F020 卡片，并且读数据块 9：

Send： FF CA 23 00 00

Receive： 50 00 04 E8 51 00 00 00 00 00 00 41 90 00

Send： FF 88 00 00 00 08 00 00 00 00 00 00 00 00

Receive： 90 00

Send： FF B0 00 09 08

Receive： 00 00 00 00 00 00 00 00 90 00

4.2.4 General Authenticate Command

该指令功能同 4.2.3 章节。

AUTHENTICATION 命令使用存储在读写器内的密钥来验证，验证前需参考 4.2.2LoadKey 章节加载密钥到读卡器。其中会用到两种认证密钥：KEY A 和 KEY B。

发送 APDU 格式（10 字节）（新的 PC/SC 标准，建议使用）：

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 字节）

字节 1	字节 2	字节 3	字节 4	字节 5
版本 01h	High Address	Low Address	Key Type	Key number

High Address / Low Address： 2 字节 验证的存储块

对于 MIFARE S50/70 则卡片块地址。

对于 AT88F020，则该地址无效（P1=0，P2=0）。

对于 MIFARE Plus Level 1/2/3，则为 AES 密钥存储块的地址

（注意 密钥存储块和数据块是一一对应关系，请参考 MIFARE Plus 数据手册）。

KeyType： 1 字节

60h = 密钥用作 A 密钥进行认证

61h = 密钥用作 B 密钥进行认证

Key number： 1 字节

00h ~ 01Fh = 密钥位置

应答：

Response	**Data Out**	
Result	SW1	SW2

例如：

寻 MIFARE S50 卡片，并且读第一块：

Send： FF 82 20 00 06 FF FF FF FF FF FF（加载认证密钥，密钥编号 00）

Receive： 90 00

Send： FF CA 00 00 00

Receive： 04 7E CE 4A A5 04 00 08 90 00

Send： FF 86 00 00 05 01 00 01 60 00

Receive： 90 00

Send： FF B0 00 01 10

Receive： 00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF 90 00

寻 MIFARE Plus Level 3 卡片，并且读数据块 0：

Send： FF 82 20 01 10 FF FF FF FF FF FF FF FF FF FF FF FF FF FF（加载认证密钥，密钥编号 01）

Receive： 90 00

Send： FF CA 03 03 00

Receive： 07 04 8B AD 04 05 06 07 42 00 31 0C 75 77 84 02 4D 46 50 5F 45 4E 47 90 00

Send： FF 86 00 00 05 01 40 00 00 01 （数据块 1 对应的密钥地址是 0x4000 或 0x4001）

Receive： 90 00

Send： FF B0 00 01 10

Receive： 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 90 00

寻 AT88F020 卡片，并且读数据块 9：

Send： FF 82 20 02 08 00 00 00 00 00 00 00 00（加载认证密钥，密钥编号 02）

Receive： 90 00

Send： FF CA 23 00 00

Receive： 50 00 06 29 BB 00 00 00 00 00 00 41 90 00

Send： FF 86 00 00 05 01 00 00 00 02

Receive： 90 00

Send： FF B0 00 09 08

Receive： 00 01 02 03 04 05 06 07 90 00

4.2.5 ReadBinaryBlock

该 APDU 主要是根据 GetData APDU 指定的寻卡类型来读取卡片存储块的内容。若卡片带有密码保护，则读取卡片块内容前，先对卡片进行授权（APDU：4.2.4 章节 General Authenticate Command）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
Read Binary	FF	B0	High Address	Low Address	Data Len

P1/P2：所读块地址

DataLen：所读数据长度**（所有数据都是低字节在前）**

MIFARE 1K/4K	16字节
MIFARE Plus	16字节（Level 3支持多块读）
MIFARE Ultralight	每块4字节，但是一次能读出4块 = 16字节
SR176	2字节
SR512	2字节
SRIX4K	2字节
AT88RF020	8字节
ISO15693 Tag	4字节（支持多块读）

该 APDU 支持读多块指令（**注意：卡片也必须支持多块读**）。若读 ISO15693Tag 连续 2 块，那么 DataLen = 4x2 = 8。注意该 APDU 对 ISO15693 Tag 的读操作是对最后一次寻到的 Tag 操作，若对选择或指定 UID 的 tag 操作请参考 3.5 章节**非标准 APDU（自定义部分）**。

应答：

Response	Data Out		
Result	Data	SW1	SW2

例如：

读 SR176 卡片第 10 块：

Send： FF CA 21 00 00

Receive： 20 42 2F 69 18 08 92 D0 02 90 00

Send： FF B0 00 0A 02

Receive： 00 00 90 00

读 MIFARE Ultralight 第 10 块：

Send： FF CA 02 00 00

Receive： 07 04 24 A2 E1 BF 02 80 44 00 00 90 00

Send： FF B0 00 0A 10

Receive： 11 22 33 44 00 00 00 00 00 00 00 00 00 00 00 00 90 00

读 ISO15693 Tag 从第 10 块开始的 2 块（即第 10、11 块）：

Send： FF CA 40 00 00

Receive： 00 3D 3D 08 17 00 01 04 E0 90 00

Send： FF B0 00 0A 08

Receive： 00 00 00 00 00 00 00 00 90 00

4.2.6 UpdataBinaryBlock

写块操作会根据 GetData APDU 指定的寻卡类型来对其写操作。若卡片带有密码保护，则写卡片块内容前，先对卡片进行授权（APDU：4.2.4 章节 General Authenticate Command）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
UpdataBinary	FF	D6	High Address	Low Address	Data Len	Data

P1/P2：所写块地址

DataLen：所写数据长度**（所有数据都是低字节在前）**

MIFARE 1K/4K	16字节
MIFARE Plus	16字节（Level 3支持多块写）
MIFARE Ultralight	4字节
SR176	2字节
SR512	4字节
SRIX4K	2字节
AT88RF020	8字节
ISO15693 Tag	4字节

该 APDU 支持写多块指令（**注意：卡片也必须支持多块写**）。若写 ISO15693Tag 连续 2 块，则 DataLen = 4x2 = 8。 注意该 APDU 对 ISO15693 Tag 的读操作是对最后一次寻到的 Tag 操作，若对选择或指定 UID 的 tag 操作请参考 3.5 章节**非标准** APDU**（自定义部分）**。

应答：

Response	Data Out	
Result	SW1	SW2

例如：

寻 MIFARE S50 卡片，并且写读第一块：

Send: FF 82 20 00 06 FF FF FF FF FF FF（加载认证密钥，掉电保存，编号 00）
Receive: 90 00
Send: FF CA 00 00 00
Receive: 04 72 AE A6 9E 04 00 08 90 00
Send: FF 86 00 00 05 01 00 01 60 00
Receive: 90 00
Send: FF D6 00 01 10 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 00
Receive: 90 00
Send: FF B0 00 01 10
Receive: 01 10 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 00

寻 MIFARE Plus Level 1 卡片，并读写第 4 块：

Send: FF 82 20 00 06 FF FF FF FF FF FF（加载认证密钥，掉电保存，编号 00）
Receive: 90 00
Send: FF CA 03 01 00
Receive: 07 04 5C 53 3A AC 22 80 42 00 18 90 00
Send: FF 86 00 00 05 01 00 01 60 00
Receive: 90 00
Send: FF D6 00 04 10 00 00 00 04 05 06 07 08 09 0A 0B 0C 0D 0E 01 00
Receive: 90 00
Send: FF B0 00 04 10
Receive: FF D6 00 04 10 00 00 00 04 05 06 07 08 09 0A 0B 0C 0D 0E 01 00

读写 MIFARE Ultralight 第 10 块：

Send: FF CA 02 00 00
Receive: 07 04 24 A2 E1 BF 02 80 44 00 00 90 00
Send: FF D6 00 0A 04 00 01 02 03
Receive: 90 00
Send: FF B0 00 0A 10
Receive: 00 01 02 03 00 00 00 00 00 00 00 00 00 00 00 00 90 00

读写 MIFARE Plus Level 3 第 1 块：

Send: FF 82 20 01 10 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF（加载认证密钥，密钥编号 01）
Receive: 90 00
Send: FF CA 03 03 00
Receive: 07 04 8B AD 04 05 06 07 42 00 31 0C 75 77 84 02 4D 46 50 5F 45 4E 47 90 00
Send: FF 86 00 00 05 01 40 00 00 01 （数据块 1 对应的密钥地址是 0x4000 或 0x4001）

Receive： 90 00
Send： FF D6 00 01 10 00 00 00 04 05 06 07 08 09 0A 0B 0C 0D 0E 01 00
Receive： 90 00
Send： FF B0 00 01 10
Receive： 00 00 00 04 05 06 07 08 09 0A 0B 0C 0D 0E 01 00 90 00

写读 SR176 卡片第 10 块：

Send： FF CA 21 00 00
Receive： 20 42 2F 69 18 08 92 D0 02 90 00
Send： FF D6 00 0A 02 00 01
Receive： 90 00
Send： FF B0 00 0A 02
Receive： 00 01 90 00

寻 AT88F020 卡片，并且读数据块 9：

Send： FF 82 20 02 08 00 00 00 00 00 00 00 00（加载认证密钥，密钥编号 02）
Receive： 90 00
Send： FF CA 23 00 00
Receive： 50 00 06 29 BB 00 00 00 00 00 00 41 90 00
Send： FF 86 00 00 05 01 00 00 00 02
Receive： 90 00
Send： FF D6 00 09 08 00 01 02 03 04 05 06 07
Receive： 90 00
Send： FF B0 00 09 08
Receive： 00 01 02 03 04 05 06 07 90 00

读 ISO15693 Tag 从第 10 块开始的 2 块（即第 10、11 块）：

Send： FF CA 40 00 00
Receive： 00 3D 3D 08 17 00 01 04 E0 90 00
Send： FF D6 00 0A 04 00 01 02 03
Receive： 90 00
Send： FF B0 00 0A 04
Receive： 00 01 02 03 90 00

4.2.7 ValueBlockOperation

值块操作仅限于带有钱包功能的卡片，如：MIFARE S50/70，MIFAREPlus Level 1/3。值块操作包括：初始化钱包、充值、扣款。若卡片带有密码保护，则操作卡片块内容前，先对卡片进行授权（APDU：4.2.4 章节 General Authenticate Command）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
ValueBlock	FF	D7	HighAddress	LowAddress	05	VB_OP+VB_Value

P1/P2：块地址

VB_OP（1Byte）： 0x00-初始化钱包
0x01-充值
0x02-扣款

VB_Value（4Byte）： 钱包值，低字节在前。

应答：

Response	Data Out	
Result	SW1	SW2

4.2.8 ReadValueBlock

读钱包操作仅仅限于带有钱包功能的卡片，如：MIFARE S50/70，MIFARE Plus Level 1/3。若卡片带有密码保护，则读卡片块内容前，先对卡片进行授权（APDU：4.2.4 章节 General Authenticate Command）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
ReadValueBlock	FF	B1	HighAddress	LowAddress	04

P1/P2：所读块地址

应答：

Response	Data Out		
Result	Value（4Byte）	SW1	SW2

例如：

MIFARE S50 初始化钱包，充值，扣款，读钱包：

Send： FF 82 20 00 06 FF FF FF FF FF FF（加载认证密钥，掉电保存，编号 00）
Receive： 90 00
Send： FF CA 00 00 00
Receive： 04 72 AE A6 9E 04 00 08 90 00
Send： FF 86 00 00 05 01 00 01 60 00
Receive： 90 00
Send： FF D7 00 01 05 00 00 00 00 01
Receive： 90 00
Send： FF B1 00 01 04
Receive： 00 00 00 01 90 00
Send： FF D7 00 01 05 01 00 00 00 02
Receive： 90 00
Send： FF B1 00 01 04
Receive： 00 00 00 03 90 00
Send： FF D7 00 01 05 02 00 00 00 01
Receive： 90 00
Send： FF B1 00 01 04
Receive： 00 00 00 02 90 00

MIFARE Plus Level 1 初始化钱包，充值，扣款，读钱包：

Send: FF 82 20 00 06 FF FF FF FF FF FF（加载认证密钥，掉电保存，编号 00）
Receive: 90 00
Send: FF CA 03 01 00
Receive: 07 04 5C 53 3A AC 22 80 42 00 18 90 00
Send: FF 86 00 00 05 01 00 04 60 00
Receive: 90 00
Send: FF D7 00 04 05 00 00 00 00 01
Receive: 90 00
Send: FF B1 00 04 04
Receive: 00 00 00 01 90 00
Send: FF D7 00 04 05 01 00 00 00 02
Receive: 90 00
Send: FF B1 00 04 04
Receive: 00 00 00 03 90 00
Send: FF D7 00 04 05 02 00 00 00 01
Receive: 90 00
Send: FF B1 00 04 04
Receive: 00 00 00 02 90 00

MIFARE Plus Level 3 初始化钱包，充值，扣款，读钱包（Block =0x01）：

Send: FF 82 20 01 10 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF（加载认证密钥，密钥编号 01）
Receive: 90 00
Send: FF CA 03 03 00
Receive: 07 04 8B AD 04 05 06 07 42 00 31 0C 75 77 84 02 4D 46 50 5F 45 4E 47 90 00
Send: FF 86 00 00 05 01 40 00 00 01 （数据块 1 对应的密钥地址是 0x4000 或 0x4001）
Receive: 90 00
Send: FF D7 00 01 05 00 00 00 00 01
Receive: 90 00
Send: FF B1 00 01 04
Receive: 00 00 00 01 90 00
Send: FF D7 00 01 05 01 00 00 00 02
Receive: 90 00
Send: FF B1 00 01 04
Receive: 00 00 00 03 90 00
Send: FF D7 00 01 05 02 00 00 00 01
Receive: 90 00
Send: FF B1 00 01 04
Receive: 00 00 00 02 90 00

4.2.9 RestoreValueBlock

备份值块操作仅仅限于带有钱包功能的卡片，如：MIFARE S50/70，MIFARE Plus Level 1/3。备份值块操作时，目标值块和源值块需在同一个扇区。若卡片带有密码保护，则操作卡片块内容前，先对卡片进行授权（APDU：4.2.4 章节 General Authenticate Command）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
Restore Value Block	FF	D7	Source High Address	Source Low Address	03	03+TargetAddress

P1/P2：源块地址

TargetAddress：目标地址（2Byte，高地址在前）

应答：

Response	Data Out	
Result	SW1	SW2

例如：

MIFARE S50 备份值块

Send： FF 82 20 00 06 FF FF FF FF FF FF（加载认证密钥，掉电保存，编号 00）

Receive： 90 00

Send： FF CA 00 00 00

Receive： 04 72 AE A6 9E 04 00 08 90 00

Send： FF 86 00 00 05 01 00 01 60 00

Receive： 90 00

Send： FF D7 00 01 05 00 00 00 00 01

Receive： 90 00

Send： FF D7 00 01 03 03 00 02

Receive： 90 00

Send： FF B1 00 02 04

Receive： 00 00 00 01 90 00

4.3 非标准 APDU（自定义部分）

非标准 APDU（**自定义部分**）是对 PC/SC Part3 定义的非标准 APDU 功能的扩展。该部分指令是通过对 FF 类指令 INS = 00 进行扩展。该部分指令可以实现当前操作智能卡切换、LCD 显示、Beep/LED 控制等。具体内容见下列表：

扩展命令列表：

Class	Ins	P1		P2	Le/Lc	功能
FF	00	ISO14443 Type A （0x00~0x1F）	MIFARE Class （0x00）	00		设定TypeA寻卡模式
				01		HaltA卡片
			MIFARE Plus （0x03）	00		从Level 0切换到Level 1/3
		ISO14443 TypeB （0x20~0x3F）	ISO14443SMARTB （0x20）	00		TypeB寻卡模式
				01		HaltB
			AT88F020 （0x23）	00		AT88F020 COUNT
				01		AT88F020 Deselect
				02		AT88F020 Lock block
		ISO15693 （0x40~0x5F）	Tag （0x40）	00		MultiTag Inventory
				01		Stay Quiet
				02		Select Tag
				03		Reset to Ready
				04		Read Block
				05		Write Block
				06		Write AFI
				07		Lock AFI
				08		Write DSFID
				09		Lock DSFID
				0A		Get System info
				0B		Get M Blk Sec St
				0C		Lock Block
		ISO7816 （0x60~0x6F）	0x60	00		设置SAM1 PPSBaud
				01		设置SAM2 PPSBaud
				02		设置SAM3 PPSBaud
				03		设置SAM4 PPSBaud
				04		设置SAM1 RSTBaud
				05		设置SAM2 RSTBaud
				06		设置SAM3 RSTBaud
				07		设置SAM4 RSTBaud
		SYSTEM （0xE0~0xFF）	智能卡切换 （0xFA）	00		智能卡操作类别切换（非接触和接触）

			RTC 操作 （0xFB）	00		初始化时间
				01		读时间
				02		设定LCD显示时间
				03		设定LCD显示日期
			LCD&&LED数码管操作 （0xFC）	00		设置显示字体类型
				01		读取显示字体类型
				02		显示指定个数字符
				03		显示图片（直接下载数据）
				04		擦除LCD
				05		设定开机图片
				06		设定待机界面
				07		LCD背光控制
				08		按指定格式显示Flash图片
				09		在任意位置显示指定个数字符
				0A		更改默认点阵大小
			Flash操作 （字体下载0xFD）	00		读Flash
				01		写Flash
			RFU（0xFE）	-		系统保留指令
			系统指令 （0xFF）	00		获取序列号
				01		获取版本号（硬件&&软件）
				02		LED 灯控制
				03		蜂鸣器操作
				04		天线状态设置
				05		设置卡片加密标准
				06		恢复出厂默认值
				07		Reader重新启动
				FF		直接传输

4.3.1 Set ISO14443A 寻卡模式

设置 ISO14443 TypeA 寻卡模式。ISO14443 TypeA 寻卡模式上电默认值是 REQA（0x26）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
SetRequestModeA	FF	00	00	00	01	RequestMode

Request Mode：

0x26- REQA

0x52- WUPA

应答：

Response	Data Out	
Result	SW1	SW2

4.3.2 Halt TypeA 卡片

使符合 ISO14443 TypeA 卡片进入休眠模式。

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
Halt A	FF	00	00	01	00

应答：

Response	Data Out	
Result	SW1	SW2

4.3.3 MIFARE Plus 从 Level 0 切换到 Level 1/3

在 Level 0 初始化完毕后，可以通过该 APDU 从 Level 0 切换到 Level 1 或 Level 3。切换到的目标层级依据卡片类型而定。注意，在 MIFARE Plus 卡片出厂时，默认层级是 Level 0，在切换到其它 Level 前需要通过 WriteBinary APDU 写入一些块参数（如：切换前必须写入 0x9000/0x9001/0x9002/0x9003 地址设置值）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
SwitchLevel	FF	00	01	00	00

应答：

Response	Data Out	
Result	SW1	SW2

4.3.4 Set ISO14443 TypeB 寻卡模式

设置 ISO14443 TypeB 寻卡模式。ISO14443 TypeB 寻卡模式上电默认值是 REQB（0x00）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
SetRequestModeB	FF	00	20	00	01	RequestMode

RequestMode：

0x00- REQB

0x01- WUPB

应答：

Response	Data Out	
Result	SW1	SW2

4.3.5 Halt TypeB

使符合 ISO14443 TypeB 卡片进入休眠模式。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
HaltB	FF	00	20	01	04	PUPI

PUPI：TypeB 卡片伪标识符

应答：

Response	Data Out	
Result	SW1	SW2

4.3.6 AT88F020 Count

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
AT88F020Count	FF	00	23	00	06	Signature

Signature：6 字节

应答：

Response	Data Out	
Result	SW1	SW2

4.3.7 AT88F020 Deselect

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
AT88F020Deselect	FF	00	23	01	00

应答：

Response	Data Out	
Result	SW1	SW2

4.3.8 AT88F020Lock

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
AT88F020Lock	FF	00	23	02	04	LockData

应答：

Response	Data Out	
Result	SW1	SW2

4.3.9 ISO15693 Inventory

除了通过 GetData 获取 Tag 标签 UID 外，也可以通过该 APDU 实现寻单张或多张 Tag 标签，标签的数量要看天线承载能力。注意该指令和 GetData APDU 同样具有切换寻卡类型的功能，使用该 APDU，寻卡类型自动切换到 ISO15693。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
Inventory	FF	00	40	00	03	Type+Flag+AFI

Type：

0x00－寻一张标签（如Flag = x26）

0x01－寻多张标签

Flag：见 ISO15693 标准

AFI：指定所寻标签的应用标识符（AFI）

应答：

Response	Data Out		
Result	（（DSFID（1Byte）+UID（8Byte））*n	SW1	SW2

例如：

寻 ISO15693 单张 Tag：

Send： FF 00 40 00 03 00 26 00

Receive： 00 3D 3D 08 17 00 01 04 E0 90 00

Send： FF 00 40 01 09 22 3D 3D 08 17 00 01 04 E0（休眠）

Receive： 90 00

Send： FF 00 40 00 03 00 26 00

Receive： 63 00

Send： FF 00 40 03 09 22 3D 3D 08 17 00 01 04 E0

Receive： 00 3D 3D 08 17 00 01 04 E0 90 00

Send： FF 00 40 00 03 00 26 00

Receive： 00 3D 3D 08 17 00 01 04 E0 90 00

4.3.10 ISO15693 Stay Quiet

ISO15693 Tag 休眠。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
Stayquiet	FF	00	40	01	09	Flag+UID

Flag：见 ISO15693 标准（如：Flag =0x22）

UID：待休眠卡片 UID（8Byte）

应答：

Response	Data Out	
Result	SW1	SW2

4.3.11 ISO15693 Select Tag

ISO15693 Tag 选卡操作。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
SelectTag	FF	00	40	02	09	Flag+UID

Flag：见 ISO15693 标准（如：Flag = 0x22）

UID：卡片 UID（8Byte）

应答：

Response	Data Out	
Result	SW1	SW2

例如：

选择一张卡片，并进行读写操作：

Send： FF 00 40 00 03 00 26 00

Receive： 00 3D 3D 08 17 00 01 04 E0 90 00

Send： FF 00 40 02 09 22 3D 3D 08 17 00 01 04 E0

Receive： 9000

Send： FF 00 40 05 0E 12 00 00 00 00 00 00 00 00 0A 11 22 33 44

Receive： 9000

Send： FF 00 40 04 0B 12 00 00 00 00 00 00 00 00 0A 01

Receive： 11 22 33 44 90 00

4.3.12 ISO15693 Reset to Ready

ISO15693 Tag 从 Halt 到 Ready 状态。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
ResetToReady	FF	00	40	03	09	Flag+UID

Flag：见 ISO15693 标准（如：Flag = x22）

UID：卡片 UID（8Byte）

应答：

Response	Data Out	
Result	SW1	SW2

4.3.13 ISO15693 WriteBlock

ISO15693 Tag 写块。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
WriteBlock	FF	00	40	05	0E	Flag + UID + BlockAddr + BlockData

Flag： 见 ISO15693 标准（如：Flag = 0x22 或 0x12（Selected tag））

UID： 卡片 UID（8Byte）

BlockAddr： 起始块地址（1Byte）

BlockData： 块数据（4 Byte）

应答：

Response	Data Out	
Result	SW1	SW2

4.3.14 ISO15693 Read Block

ISO15693 Tag 读块。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
ReadBlock	FF	00	40	04	0B	Flag + UID + BlockAddr + BlockNum

Flag： 见 ISO15693 标准（如：Flag = 0x22 或 0x12（Selected tag））

UID： 卡片 UID（8Byte）

BlockAddr： 起始块地址

BlockNum： 读取块数，不同的卡片支持读取块数不同（最小是 1）

应答：

Response	Data Out	
Result	SW1	SW2

4.3.15 ISO15693 Write AFI

写 ISO15693 Tag AFI。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
Write AFI	FF	00	40	06	0A	Flag+UID+AFI

Flag： 见 ISO15693 标准（如：Flag = 0x22 或 0x12（Selected tag））

UID： 卡片 UID（8Byte）

AFI： 新的 AFI

应答：

Response	Data Out	
Result	SW1	SW2

例如：

写 AFI

Send： FF 00 40 06 0A 22 3D 3D 08 17 00 01 04 E0 00

Receive： 90 00

4.3.16 ISO15693 Lock AFI

锁 ISO15693 Tag AFI。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
LockAFI	FF	00	40	07	09	Flag+UID

Flag：见 ISO15693 标准（如：Flag = 0x22 或 0x12（Selected tag））

UID：卡片 UID（8Byte）

应答：

Response	Data Out	
Result	SW1	SW2

4.3.17 ISO15693 Write DSFID

写 ISO15693 Tag DSFID。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
WriteDSFID	FF	00	40	08	0A	Flag+UID（8Byte）+DSFID

Flag：见 ISO15693 标准（如：Flag = 0x22 或 0x12（Selected tag））

UID：卡片 UID（8Byte）

DSFID：新的 DSFID

应答：

Response	Data Out	
Result	SW1	SW2

例如：

写 DSFID：

Send： FF 00 40 08 0A 22 3D 3D 08 17 00 01 04 E0 00

Receive： 90 00

4.3.18 ISO15693 Lock DSFID

锁 ISO15693 Tag DSFID。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data

LockDSFID	FF	00	40	09	09	Flag+UID

Flag： 见 ISO15693 标准（如：Flag = 0x22 或 0x12（Selected tag））

UID： 卡片 UID（8Byte）

应答：

Response	**Data Out**	
Result	SW1	SW2

4.3.19 ISO15693 Get System info

获取 ISO15693 Tag 系统信息

发送 APDU 格式：

Command	**Class**	**INS**	**P1**	**P2**	**Lc**	**Data**
GetSysInfo	FF	00	40	0A	09	Flag+UID

Flag： 见 ISO15693 标准（如：Flag = 0x22（需要带 UID），Flag = 0x02（可以不带 UID））

UID： 卡片 UID（8Byte）

应答：

Response	**Data Out**		
Result	System Info	SW1	SW2

SystemInfo： InfoFlag（1Byte）+UID（8Byte）+DSFID（1Byte）+AFI（1Byte）+Other（nByte）

例如：

Get system information：

Send： FF 00 40 0A 09 22 3D 3D 08 17 00 01 04 E0

Receive： 0F 3D 3D 08 17 00 01 04 E0 01 00 1B 03 01 90 00

4.3.20 ISO15693 Get Blocks Security

获取 ISO15693 Tag 块安全状态

发送 APDU 格式：

Command	**Class**	**INS**	**P1**	**P2**	**Lc**	**Data**
GetMultiBlkSecSt	FF	00	40	0B	0B	Flag + UID + StartAddr+Num

Flag： 见 ISO15693 标准（如：Flag = 0x22）

UID： 卡片 UID（8Byte）

StartAddr： 开始块（1Byte）

Num： 块数（最小 0）

应答：

Response	**Data Out**		
Result	BlockSecSta ×Num	SW1	SW2

例如：

获取 ISO15693 第 10、11、12 块安全状态：

Send： FF 00 40 0B 09 22 3D 3D 08 17 00 01 04 E0 0A 02

Receive： 00 00 00 90 00

4.3.21 ISO15693 Lock Block

锁 ISO15693 Tag DSFID。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
LockDSFID	FF	00	40	0C	0A	Flag+UID+BlockNO

Flag：见 ISO15693 标准（如：Flag = 0x22 或 0x12（Selected tag））

UID：卡片 UID（8Byte）

BlockNO：待锁块号

应答：

Response	Data Out	
Result	SW1	SW2

4.3.22 设置 SAM 波特率（SetPPS）

该功能主要是设置 SAM 卡通讯波特率。每个读卡器支持的 SAM 个数可能不同，详情请参考读卡器说明书（MR800 支持 2 个 SAM）。在发送 GetData APDU 复位 SAM 卡后，若修改 SAM 卡波特率（注：该 SAM 卡必须支持所设置波特率），可发送该 APDU 去设置（PPS）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
SetSamBaud	FF	00	60	SAMPPS	01	Baudrate

SAMPPS：

0- SAM1 SetPPS
1- SAM2 SetPPS
2- SAM3 SetPPS
3- SAM4 SetPPS

Baudrate：

0- 9600 （默认）
1- 19200
2- 38400
3- 55800
4- 57600
5- 115200

应答：

Response	Data Out	
Result	SW1	SW2

4.3.23 设置 SAM 复位波特率

该功能主要是设置 SAM 复位时采用的波特率。每个读卡器支持的 SAM 个数可能不同，详情请参考读卡器说明书（MR800 支持 2 个 SAM，MR880 支持 4 个 SAM）。一般默认情况下，

SAM 卡默认复位波特率是 9600，若想修改 SAM 复位波特率，在发送 GetData APDU 复位 SAM 卡前，可发送该 APDU 去设置 SAM 复位波特率 （注：该 SAM 卡必须支持所设置的复位波特率）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
SetRstSamBaud	FF	00	60	SAMRestBaudNO	01	Baudrate

SAMRestBaudNO：

4- SAM1 Reset Baudrate
5- SAM2 Reset Baudrate
6- SAM3 Reset Baudrate
7- SAM4 Reset Baudrate

Baudrate：

0- 9600 （默认）
1- 19200
2- 38400
3- 55800
4- 57600
5- 115200

应答：

Response	Data Out	
Result	SW1	SW2

4.3.24 切换当前操作智能卡

该功能主要实现非接触 SmartCard 和 接触的 SAM 之间切换。因为非接触 SmartCard 和 SAM 卡除了寻卡和复位使用非标准 APDU（GetData）外，其余都是发送标准的 APDU 指令。为了区分当前操作的是 SmartCard 还是 SAM 卡，通过此指令可以实现切换。在实际应用中，有时已经通过 GetData 寻到 smartcard 后，需要通过 SAM 卡进行认证，那么需要通过该 APDU 暂时将对智能卡的操作对象切换到 SAM，操作完毕后需再切换到 SmartCard。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
SwitchSmartCard	FF	00	FA	00	01	CurSmartCard

CurSmartCard：

0- 非接触智能卡
1- SAM1 卡
2- SAM2 卡
3- SAM3 卡
4- SAM4 卡

应答：

Response	Data Out	
Result	SW1	SW2

例如：

同时操作 SmartCard 和 SAM 卡，操作流程图：

上电初始化
上电复位 SAM1 卡：发送 APDU： GetData(SAM)，成功后 SAM1 为当前操做的智能卡，所以所有发送的标准 APDU 都是发送到 SAM1,如：发送取随机数 APDU：0084000008，直接发送到 SAM1.

↓

循环寻 SmartCard 卡
发送 APDU： GetData(SmartCard)，发送完毕后 SmartCard 为当前操做的智能卡(不管当前操作成功与否)，所以所有发送的标准 APDU 都是发送到 SmartCard,如：发送取自由数 APDU： 0084000008，直接发送到 SmartCard.

↓

寻 SmartCard 成功后，需通过 SAM1 验证 SmartCard 合法性并且读 smartcard
发送标准 APDU 到 SmartCard(此时为当前操作卡)获取验证数据，若获取数据成功则进入步骤 2。
切换当前智能卡到 SAM1，发送 APDU： SwitchSmartCard(SAM1)，若切换成功，发送标准 APDU 到 SAM1 验证从 smartcard 获取数据的合法性(此时 SAM1 为当前操作卡)
若要进一步对 SmartCard 操作，需发送 APDU： SwitchSmartCard(Smartcard)切换到 smartcard，切换成功后，当前操作卡又为 smartcard，可以进行 smartcard 的读写操作。

4.3.25 初始化 RTC 时间（仅 MR800/810/880 支持）

该功能实现对读卡器内部时钟初始化操作。若需要时间能掉电保持，需要配备电池。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
InitialRTC	FF	00	FB	00	08	Time

Time：年（High Byte）＋年（Low Byte） ＋月（Month）＋日（Date） ＋时（Hour） ＋分（Minute）＋秒（Second）＋星期（Week）

如：2010－4－12 12：01：00 星期一 时间数据是：07 DA 04 0C 0C 01 00 01

应答：

Response	Data Out	
Result	SW1	SW2

例如：

设置时间并读出时间：

Send： FF 00 FB 00 08 07 DA 04 0C 0C 01 00 01

Receive： 90 00

Send： FF 00 FB 01 08

Receive： 07 DA 04 0C 0C 03 15 01 90 00

4.3.26 读 RTC 时间（仅 MR800/810/880 支持）

该功能实现读取读卡器内部时钟。若需要时间能掉电保持，需要配备电池。

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
ReadRTC	FF	00	FB	01	08

应答：

Response	Data Out		
Result	Time	SW1	SW2

Time：年（High Byte）＋年（Low Byte） ＋月（Month）＋日（Date） ＋时（Hour） ＋分（Minute）＋秒（Second）＋星期（Week）

如：2010－4－12 12：01：00 星期一 时间数据是：07 DA 04 0C 0C 01 00 01

4.3.27 设定 RTC 时间显示-时间（仅 MR800/880 支持）

该功能主要是设置时间在 LCD 上的显示模式。若需要时间能掉电保持，需要配备电池。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
DisTime	FF	00	FB	02	03	EnableFag+Line+Column

EnableFag：日期显示使能（0-Disable，1-Enable）

Line：显示起始行（0~7 / 12）

Column：显示起始列（0~127 / 239）

如显示 12 点 10 分 10 秒，显示为：12：10：10

应答：

Response	Data Out	
Result	SW1	SW2

例如：

关闭时间显示

Send： FF 00 FB 02 03 00 00 00

Receive： 90 00

设定时间显示

Send： FF 00 FB 02 03 01 03 05

Receive： 90 00

4.3.28 设定 RTC 时间显示-日期（仅 MR800/880 支持）

该功能主要是设置日期在 LCD 上的显示模式。若需要时间能掉电保持，需要配备电池。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
DisDate	FF	00	FB	03	03	EnableFag+Line+Column

EnableFag：日期显示使能（0-Disable，1-Enable）

Line： 显示起始行（0~7 / 12）

Column： 显示起始列（0~127 / 239）

如显示 2010－04－16，显示为：10/04/16

应答：

Response	Data Out	
Result	SW1	SW2

例如：

关闭日期显示

Send： FF 00 FB 03 03 00 00 00

Receive： 90 00

设定日期显示

Send： FF 00 FB 03 03 01 03 05

Receive： 90 00

4.3.29 设定 LCD 显示字体类型（仅 MR800/880 支持）

MR800 支持简体中文、繁体中文和俄文三种字体。通过该指令可实现非英文显示字库切换。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
SetFontType	FF	00	FC	00	01	ChineseFontType

ChineseFontType：

0x01 简体中文（默认）

0x02 繁体中文

0x03 俄文

应答：

Response	Data Out	
Result	SW1	SW2

例如：

简体中文显示

Send： FF 00 FC 00 01 01

Receive： 90 00

4.3.30 读取 LCD 显示字体类型（仅 MR800/880 支持）

MR800 支持简体中文、繁体中文和俄文三种字体。通过该指令可获知当前支持的非英文字体类型。

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
ReadFontType	FF	00	FC	01	01

应答：

Response	Data Out		
Result	ChineseFontType	SW1	SW2

ChineseFontType： 0x01 简体中文（默认），0x02 繁体中文，0x03 俄文

例如：

Send： FF 00 FC 00 01 02

Receive： 90 00

Send： FF 00 FC 01 01

Receive： 02 90 00

4.3.31 LCD 点阵设定（仅 MR880 支持）

MR880 支持三种点阵显示，开机默认 32 点阵。通过改指令可以自由切换点阵大小

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
Display Picture	FF	00	FC	0A	01	DefineBitmap

DefineBitmap:

0x00 = 16 点阵大小

0x01 = 24 点阵大小

0x02 = 32 点阵大小(系统默认，重新上电会重置)

应答：

Response	Data Out	
Result	SW1	SW2

注：俄文仅支持 32 点阵，中文简体繁体支持 16,24,32 点阵。

例如：

Send: FF 00 FC 0A 01 00

Receive: 90 00

4.3.32 LCD 显示指定个数的中文、俄文或英文字体（仅 MR800/880 支持）

MR800 支持简体中文、繁体中文和俄文三种字体。该指令显示指定个数的字符（包括英文或中文）。注意一个中文字体占 2Byte，英文字体占 1Byte。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
Display Font	FF	00	FC	02	nByte	Configure + Row + Column + Display Data

Configure：

Bit0（NegativeDis）：正反显示 0-正显，1-反显

Bit2~1：

00-显示画面前不清屏幕

01-显示画面前只清除显示画面的行

10-显示画面前全部清屏

Bit3（BackLight）：0-背光不亮，1-背光亮

Bit4~7：RFU

Row:

0~7 (MR800 (1Row = 16 dot High))

0~7 (MR880 32 点阵(1Row = 32 dot High))

0~0x09 (MR880 24 点阵(1Row = 24 dot High))

0~0x0F (MR880 16 点阵(1Row = 16 dot High))

Column：0~127 / 239

DisplayData：显示内容（汉字相当于 2Byte），一行最多显示 16Byte。

应答：

Response	Data Out	
Result	SW1	SW2

例如：

在左上角显示简体“金木雨”三个字，正显、显示画面不清屏幕、背光不亮。

Send： FF 00 FC 02 09 00 00 00 BD F0 C4 BE D3 EA

Receive： 90 00

4.3.33 LCD 任意位置显示指定个数字符(仅 MR880 支持)

此指令跟“LCD 显示指定个数的中文、俄文或英文字体”指令功能基本相同，只不过它可以任意的点的位置（指定点的 X 坐标和 Y 坐标位置）进行显示。该指令显示指定个数的字符（包括英文或中文）,同时指定此字符串的点阵大小（共有 16 点阵，24 点阵和 32 点阵三种点阵可以选择）。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
Display Font	FF	00	FC	09	nByte	Configure + Row + Column + Display Data

Configure：

Bit0（NegativeDis）： 正反显示 0-正显，1-反显

Bit1: RFU

Bit2: 此位置 1 则显示之前清整个液晶屏，为 0 则不清屏

Bit3（BackLight）: 0-背光不亮， 1-背光亮

Bit4~5: （此点阵大小只对该命令有效）

01-用 16 点高的字符显示（汉字宽 16 点，ASC 码宽 8 点）

10-用 24 点高的字符显示（汉字宽 24 点，ASC 码宽 12 点）

11-用 32 点高的字符显示（汉字宽 32 点，ASC 码宽 16 点）

Bit6~7: RFU

Row 字节:

范围 0~127 ，即表示字符串显示的点行位置

Column 字节:

范围 0~239 ，即表示字符串显示的点列位置

DisplayData: 显示内容，显示不能超过 240 点。

应答：

Response	Data Out	
Result	SW1	SW2

例如：在第 03 点行第 04 点列的位置显示 24 点阵的“金木雨”三个字，正显、显示画面、不清屏幕、背光亮。

Send: FF 00 FC 09 09 28 03 04 BD F0 C4 BE D3 EA

Receive: 90 00

4.3.34 LCD 显示图片（直接发送图片数据）（仅 MR800/880 支持）

该功能实现显示规定大小的图片，大的图片可以分多次显示。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
Display Picture	FF	00	FC	03	nByte	Configure + Row + Column + PictureWidth + PictureHigh+ Display Data

Configure：

Bit0（Negative Dis）：正反显示 0-正显，1-反显

Bit2~1：

00-显示画面前不清屏幕

01-显示画面前只清除显示画面的行

10-显示画面前全部清屏

Bit3（Back Light）：0-背光不亮，1-背光亮

Bit4~7：RFU

Row（1row = 8 dot High）：0~7 / 15（开始列）

Column：0~127 / 239（开始行）

Picture Width：1~128 / 240，图片宽度

Picture High：1~8 / 16，图片高度

Display Data：显示图片内容（字节数= 宽度 x 高度）

应答：

Response	Data Out	
Result	SW1	SW2

4.3.35 LCD 擦除行（仅 MR800/880 支持）

MR800 为了方便清屏，用户可以分行清除字体或图片。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
EraseLCD	FF	00	FC	04	01/02	Row

Row（1row = 8 dot High）：Bit0~Bit7 分别代表 0~7 行（0-保持不变，1-擦除）

注： MR880 的 Lc=02, Row 为双字节变量，是 Bit0~Bit15 分别代表 0~15 行。

应答：

Response	Data Out	
Result	SW1	SW2

例如：

LCD 擦除所有行

Send： FF 00 FC 04 01 FF

Receive： 90 00

4.3.36 LCD 设定开机画面（仅 MR800/880 支持）

该功能实现默认开机画面设置。若没有设置，则开机默认显示金木雨开机画面。所有显示画面都保存于读卡器内 AT45DB321/AT45DB641(MR880)内。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
PowerOnPIC	FF	00	FC	05	08	Enable+SaveAddr+Width+High+StartLine+StartColumn+Time

Enable（1Byte）： 0-禁止显示开机画面，1-显示开机画面

SaveAddr（2Byte）： 开机画面保存于 ***Flash*** 中，地址低字节在前

Width（1Byte）： 图片宽度（1~128 / 240）

High（1Byte）： 图片高度（1~8 / 16）

StartLine（1Byte）： 显示开始行（0~7 / 15）

StartColumn（1Byte）： 显示开始列（0~127 / 239）

Time： 设定显示启动画面时间（单位：S）

应答：

Response	Data Out	
Result	SW1	SW2

备注：

- 若设置开机画面禁止，则后面参数无效。
- 开机画面保存在读卡器片外Flash中，字库占据开始的1303块（0~1302）（MR880为10360(0 ~ 10359)，用户不能进行擦写，供用户使用的块号是1303~8191（MR880为10360 ~ 16383），每块大小是512字节。
- 在使能开机画面前，需用FlashWrite APDU 写入画面数据到Flash SaveAddr地址中，否则显示画面为不确定，若画面大于512字节，则多余字节写入紧接的第2块。
- 画面大小=Width*High。

例如：

设定一个开机画面，本图片大小为 128*64.(要在 Flash 中先写入图片数据)

Send： FF 00 FD 01 84 05 17 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 01 07 3F 3F 3F
1F 07 01 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Receive： 90 00

Send： FF 00 FD 01 84 05 17 00 80

00 00 00 00 00 00 00 00 00 00 7C 7F 7F 7F 3F 3F
3F 3F 1F 1F 1F 0F 0F 07 07 03 7F FF FF FF FF FF
FF FF FF 7D 03 07 07 0F 0F 1F 1F 1F 3F 3F 3F 3F
7F 7F 7F 78 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 01 03 03 0D 39 71 31 0D 07 07
03 03 01 00 00 04 04 04 04 05 07 7F 27 05 04 04
0C 0C 00 00 30 37 37 37 35 34 3F 3F 37 35 34 37
37 30 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Receive： 90 00

Send： FF 00 FD 01 84 05 17 01 00

00 00 00 00 00 00 00 00 00 00 00 C0 F0 FC FF FF
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
BF 7F FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FC F0 80 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 83 A2 32 3A 2E 26 FE FE 26 3E 3A
62 22 02 00 04 0C 18 30 60 C0 00 FF 00 C0 60 30
18 18 08 00 00 FF FE 20 B8 90 FE FE 20 BA 03 FF
FC 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Receive： 90 00

Send： FF 00 FD 01 84 05 17 01 80

00 00 00 06 0F 0F 1F 1F 3F 3F 7F 7F 7F 7F 7F BF
FF EF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF FF FF FF F6 FF FF FF 7F
7F 7F 7F 7F 3F 3F 1F 1F 0F 0F 07 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Receive： 90 00

Send： FF 00 FD 01 84 05 18 00 00

00 00 00 00 00 00 80 80 C0 C0 E0 E0 E0 E3 EF DF
FF 7F FF FF FF FF FF FF FF FF FF FF FF FF FF FF
FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF EF
E3 E0 E0 E0 C0 C0 80 80 00 00 00 00 00 00 00 00
00 00 00 00 00 00 08 0E 06 01 05 05 05 1F 1D 05
05 05 01 00 00 02 0E 0C 09 0B 08 08 08 08 08 0B
0F 0C 00 00 00 00 0F 0F 09 0F 0F 00 0F 09 09 0F
0F 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Receive： 90 00

Send： FF 00 FD 01 84 05 18 00 80
00 00 00 00 00 00 00 00 00 00 03 1F FF FF FF FF
FF FF FF FF FF FF FF FE FE FC FF FF FF FF FF FF
FF EF FF FB FC FE FE FF FF FF FF FF FF FF FF FF
FF FF 1F 01 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 C0 C0 FF FF 87 36 5C 6C 27 7F 7D
05 C4 8C 00 00 04 06 06 F6 D6 96 96 96 96 96 96
BF B8 00 00 44 64 EF EF 5C F7 EF E0 EF B4 DC 6F
6F 6C 28 00 00 00 00 00 00 00 00 00 00 00 00 00

Receive： 90 00

Send： FF 00 FD 01 84 05 18 01 00
00 00 00 00 00 00 00 00 00 00 E0 E0 E0 E0 E0 C0
C0 C0 80 80 80 00 00 00 00 00 F0 FC FE FF FF FF
FF FE F8 E0 00 00 00 00 00 80 80 80 C0 C0 C0 E0
E0 E0 E0 E0 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 80 80 40 40 C0 80 80 00 00
80 C0 40 00 00 00 00 00 00 00 00 00 80 80 C0 C0
80 00 00 00 00 00 C0 C0 80 C0 80 00 C0 80 80 C0
80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Receive： 90 00

Send： FF 00 FD 01 84 05 18 01 80
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 C0 C0 C0
80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Receive： 90 00

Send： FF 00 FC 05 08 01 17 05 80 08 00 00 05

Receive： 90 00

4.3.37 LCD 设定待机画面（仅 MR800/880 支持）

该功能实现待机画面设置，若没有设置，则显示完毕用户界面后不会回到待机画面。。所有显示画面都保存于读卡器内 AT45DB321/AT45DB641(MR880)内。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
IdlePIC	FF	00	FC	06	08	Configure +SaveAddr+ Width+ High+StartLine+StartColumn+Time

Configure （1Byte）：

Bit0： 0-禁止显示待机画面，1-显示待机画面

Bit2~1：

00-显示画面前不清屏幕

01-显示画面前只清除显示画面的行

10-显示画面前全部清屏

Bit3（BackLight）：0-背光不亮，1-背光亮

Bit4~7：RFU

SaveAddr（2Byte）：待机画面保存于 ***Flash*** 中的地址，地址低字节在前。

Width（1Byte）：图片宽度（1~128 / 240）

High（1Byte）：图片高度（1~8 / 16）

StartLine（1Byte）：显示开始行（0~7 / 15）

StartColumn（1Byte）：显示开始列（0~127 / 239）

Time：设定多长时间未操作 LCD，进入待机画面（单位：S）

应答：

Response	Data Out	
Result	SW1	SW2

备注：

- 若设置待机画面禁止，则后面参数无效。
- 待机画面保存在读卡器片外Flash中，字库占据开始的1303块（0~1302）（MR880为10360块），用户不能进行擦写，供用户使用的块号是1303~8191（MR880为10360 ~ 16383），每块大小是512字节。
- 在使能待机画面前，需用FlashWrite APDU 写入画面数据到Flash SaveAddr地址中，否则显示画面为不确定，若画面大于512字节，则多余字节写入紧接的第2块。
- 画面大小=Width*High。
- 指令方法可以参考LCD设定开机画面的例程，但是要注意Flash的存储地址不能重复。

4.3.38 LCD 背光控制（仅 MR800/880 支持）

该功能对 LCD 的背光进行控制。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
LCDBackLight	FF	00	FC	07	02	Mode+Time

Mode：

00-灭

01-常亮

02-规定时间亮（Time内容有效）

Time：仅仅在 Mode =2 才有效（单位：S）

应答：

Response	Data Out	
Result	SW1	SW2

例如：

LCD 背光灯亮 15 秒

Send： FF 00 FC 07 02 02 0F

Receive： 90 00

4.3.39 LCD 显示 Flash 中存储画面（仅 MR800/880 支持）

该功能实现保存画面显示。所有显示画面都保存于读卡器的串行 Flash 内。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
IdlePIC	FF	00	FC	08	09	Configure +DisAddr +Width+ High + StartLine+StartColumn

Configure（1Byte）：

Bit0： RFU

Bit2~1：

00-显示画面前不清屏幕

01-显示画面前只清除显示画面的行

10-显示画面前全部清屏

Bit3（BackLight）：0-背光不亮，1-背光亮

Bit4~7： RFU

DisAddr（2Byte）： 显示画面保存于 **Flash** 中，地址低字节在前

Width（1Byte）： 图片宽度（1~128 / 240）

High（1Byte）： 图片高度（1~8 / 16）

StartLine（1Byte）： 显示开始行（0~7 / 15）

StartColumn（1Byte）： 显示开始列（0~127 / 239）

应答：

Response	Data Out	
Result	SW1	SW2

备注：

- 显示画面保存在读卡器片外Flash中，字库占据开始的1303块（0~1302）（MR880为10360块），用户不能进行擦写，供用户使用的块号是1303~8191（MR880为10360 ~ 16383），每块大小是512字节。
- 在显示画面前，需用FlashWrite APDU 写入画面数据到Flash SaveAddr地址中，否则显示画面为金木雨默认画面，若画面大于512字节，则多余字节写入紧接的第2块。
- 画面大小=Width*High。

例如：

显示 Flash 中地址 1303 的存储画面

Send： FF 00 FC 08 09 0C 17 05 80 08 00 00

Receive： 90 00

4.3.40 读片外 Flash

片外 Flash 容量是 4Mbytes(MR880 是 8Mbytes)，字库占据开始的 1303 块（0~1302）（MR880 为 10360 块），用户不能进行擦写，供用户使用的块号是 1303~8191（MR880 为 10360 ~ 16383），

每块大小是 512 字节。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
ReadFlash	FF	00	FD	00	06	BlockAddr+ ByteAddr+ Len

BlockAddr：块地址（2Byte，高字节在前）

ByteAddr：块内字节起始地址（2Byte，高字节在前）

Len：所读字节长度（2Byte，高字节在前），len≤256

应答：

Response	Data Out		
Result	Flash Data	SW1	SW2

例如：

读 Flash 的 02 块中的 2Byte，起始地址 0002

Send： FF 00 FD 00 06 00 02 00 02 00 02

Receive： 18 08 90 00

4.3.41 写片外 Flash

片外 Flash 容量是 4Mbytes(MR880 是 8Mbytes)，字库占据开始的 1303 块（0~1302）（MR880 为 10360 块），用户不能进行擦写，供用户使用的块号是 1303~8191（MR880 为 10360 ~ 16383），每块大小是 512 字节。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
WriteFlash	FF	00	FD	01	04+n	BlockAddr+ ByteAddr+Data（n）

BlockAddr：块地址（2Byte，高字节在前）

ByteAddr：块内字节起始地址（2Byte，高字节在前）

Data：所写数据

应答：

Response	Data Out	
Result	SW1	SW2

例如：

给 0616 块写 1 字节数据，起始字节 00 02

Send： FF 00 FD 01 05 06 16 00 02 01

Receive： 90 00

4.3.42 获取产品序列号

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
GetSNR	FF	00	FF	00	0A

应答：

Response	Data Out

Result	Product SNR	SW1	SW2

例如：

Send： FF 00 FF 00 0A

Receive： 01 05 07 09 09 04 03 08 06 09 90 00

4.3.43 获取硬件版本和版本号

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
GetVer	FF	00	FF	01	04

应答：

Response	Data Out		
Result	Handware ver（2Byte）+Sosftware ver（2Byte）	SW1	SW2

例如：

Send： FF 00 FF 01 04

Receive： 01 00 02 02 90 00

4.3.44 LED 灯控制

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
LEDCtr	FF	00	FF	02	05	LED state + state Mask +T1 duration + T2 Duration + Number

LED Status：

BIT0 = 红灯最终状态（1-ON，0-OFF）
BIT1 = 绿灯最终状态（1-ON，0-OFF）
BIT2 = 蓝灯最终状态（1-ON，0-OFF）
BIT3 = 黄灯最终状态（1-ON，0-OFF）
BIT4 = 红灯闪动初始状态（1-ON，0-OFF）
BIT5 = 绿灯闪动初始状态（1-ON，0-OFF）
BIT6 = 蓝灯闪动初始状态（1-ON，0-OFF）
BIT7 = 黄灯闪动初始状态（1-ON，0-OFF）

LED Status Mask：

BIT0 = 红灯状态更新掩码（1-更新，0-不改变）
BIT1 = 绿灯状态更新掩码 （1-更新，0-不改变）
BIT2 = 蓝灯状态更新掩码 （1-更新，0-不改变）
BIT3 = 黄灯状态更新掩码 （1-更新，0-不改变）
BIT4~7 RFU

T1/T2：T1，T2 时间（单位：100ms），T＝T1+T2

Number：次数

应答：

Response	Data Out	
Result	SW1	SW2

例如：

四种颜色灯闪动两次，最终状态为所有灯全关

Send： FF 00 FF 02 05 F0 0F 0F 0F 02

Receive： 90 00

红色灯闪动两次，最终状态为红灯开

Send： FF 00 FF 02 05 F0 0F 0F 0F 02

Receive： 90 00

黄灯闪动，最终状态为红灯，执行两次

Send： FF 00 FF 02 05 81 09 0F 0F 02

Receive： 90 00

4.3.45 蜂鸣器控制

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
BuzzerCtr	FF	00	FF	03	05	Beep state + state Mask +T1 duration + T2 Duration + Number

BEEP Status：

BIT0 = BEEP最终状态（1-ON，0-OFF）

BIT4 = BEEP闪动初始状态（1-ON，0-OFF）

Status Mask：

BIT0 = Buzzer状态更新掩码（1-更新，0-不改变）

BIT4~7 RFU

T1/T2：T1，T2 时间（单位：100ms），T＝T1+T2

Number：次数

应答：

Response	Data Out	
Result	SW1	SW2

例如：

蜂鸣器闪动，更新状态掩码，闪动两次，重复两次。

Send： FF 00 FF 03 05 08 01 0F 0F 02

Receive： 90 00

4.3.46 天线状态设置

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
AtennaCtr	FF	00	FF	04	01	Antena status

Antena status：

00-关闭

01-打开

应答：

Response	Data Out	
Result	SW1	SW2

例如：

关闭天线

Send： FF 00 FF 04 01 00

Receive： 90 00

4.3.47 卡片加密方法设置

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	Data
EncrMode	FF	00	FF	05	01	Encrypt Standard

Encrypt Standard：

0x00-Philips

0x01-上海标准

应答：

Response	Data Out	
Result	SW1	SW2

例如：

设置上海标准加密方法

Send： FF 00FF 05 01 01

Receive： 90 00

4.3.48 恢复出厂默认值（系统重新启动）

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
FactoryDefault	FF	00	FF	06	00

应答：

Response	Data Out	
Result	SW1	SW2

例如：

Send： FF 00 FF 06 00

Receive： 90 00

4.3.49 系统重新启动

发送 APDU 格式：

Command	Class	INS	P1	P2	Le
Reboot	FF	00	FF	07	00

应答：

Response	Data Out	
Result	SW1	SW2

例如：

Send： FF 00 FF 07 00

Receive： 90 00

4.3.50 直接传输

将数据包经过 RF 直接发送到标签，可以发送读卡器不支持的命令。

发送 APDU 格式：

Command	Class	INS	P1	P2	Lc	CMD	TMO	命令数据域
DIRECT TRANSMIT	FF	00	FF	FF	待发送的字节数 2+n	命令	FWI	数据包 n

Lc：1个字节，待发送的字节数，最大值为255

CMD：命令为0：发送且接收，1：只发送

TMO：超时参数，FWI 值，对于 M1 卡的读写，FWI=4。当 CMD=1 时此字节无意义

命令数据域：经由 RF 发出的命令和数据

应答：

Response	响应数据域
DIRECT TRANSMIT	响应数据

例如：

MIFARE Ultralight C 卡片的数据块读写操作：

Send： FF CA 02 00 00（寻卡）

Receive： 07 04 15 BA 8A 7C 3B 80 44 00 00 90 00

Send： FF 00 FF FF 08 01 00 A2 09 01 02 03 04（写块09）

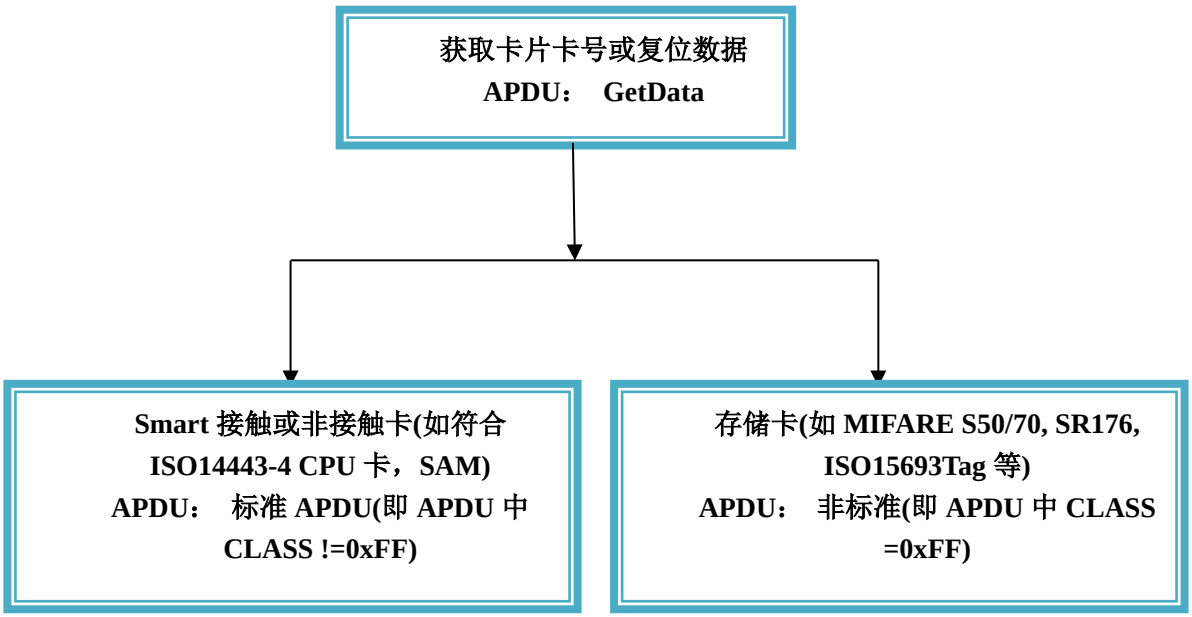
Receive： 90 00

Send： FF 00 FF FF 04 00 05 30 09（读块09起始的4个块）

Receive： 01 02 03 04 00 00 00 00 00 00 00 00 00 00 00 00 90 00

5 卡片操作流程

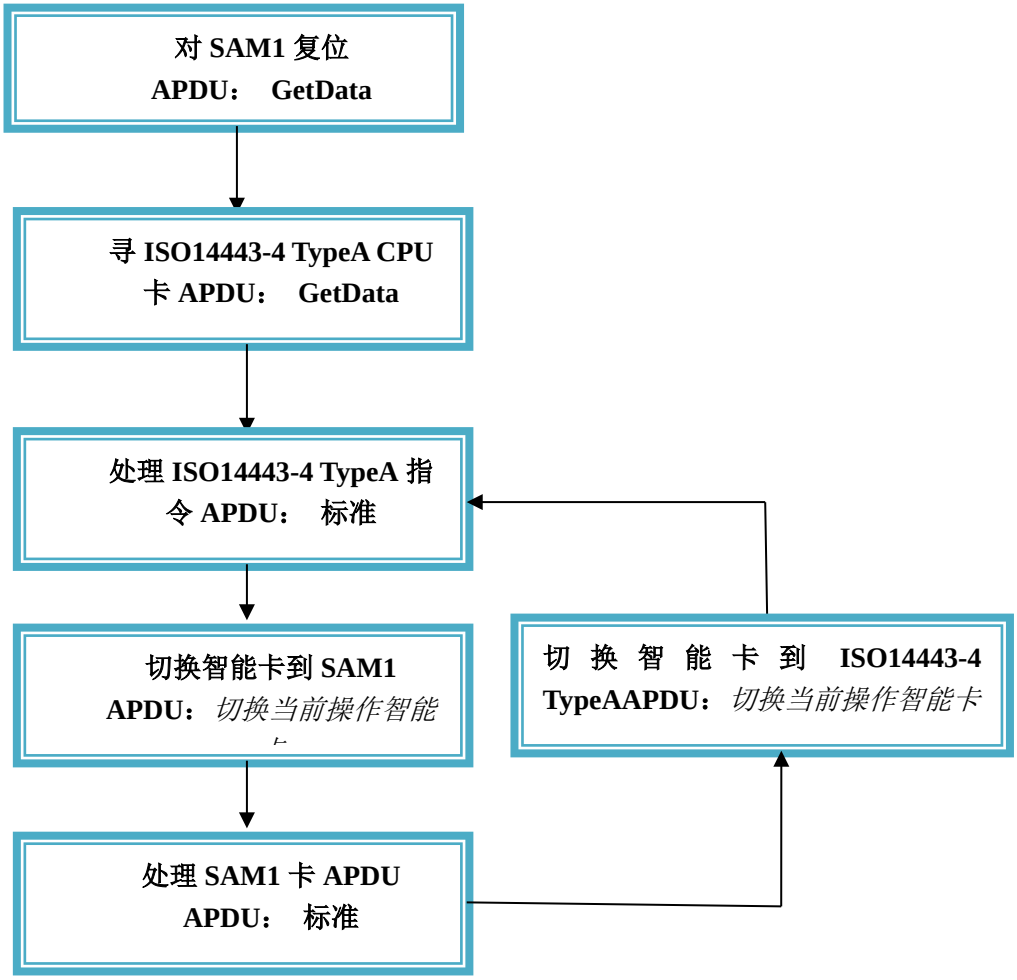
各种卡片操作基本流程如下：



在操作任何卡片前需要执行 GetData APDU 获取卡片基本信息（包括卡序列号，复位信息等），GetData 包含了读卡类型的切换，所以在对任何卡片执行操作前需执行该 APDU，获取卡片信息的同时，读卡器读卡类型也切换到这个类型上。

5.1 Smart 接触和非接触卡

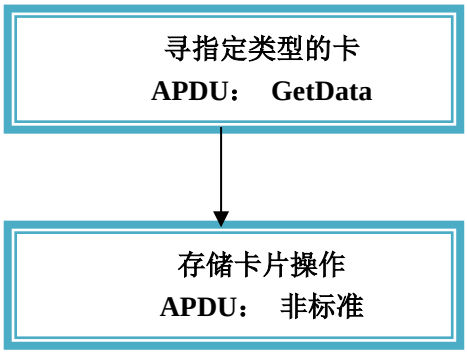
Smart 接触或非接触卡可以直接发送标准的 APDU 至卡片，假如需要同时操作非接触和接触的 Smart 卡（如：ISO14443-4 TypeA CPU 卡和 SAM1 卡） 卡片操作如下：



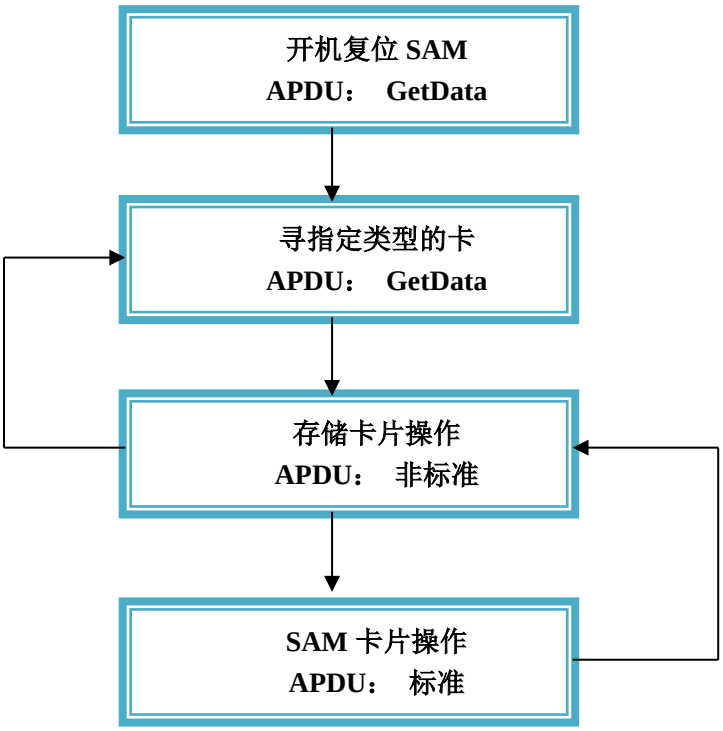
因为智能非接触和接触卡都采用的是标准 APDU，在对 SAM 卡复位后，若需要再对 SAM 进行操作，需要通过**切换智能卡类别**指令去切换当前操作智能卡，以保证数据是发送到指定类型的智能卡。若是智能卡和存储卡不需要切换，则执行完毕 GetData 后，当前操作类型就是 GetData 操作的卡片类型。

5.2 存储卡（非智能卡）

存储卡片的操作都是通过非标准的 APDU 来操作，主要操作如下：

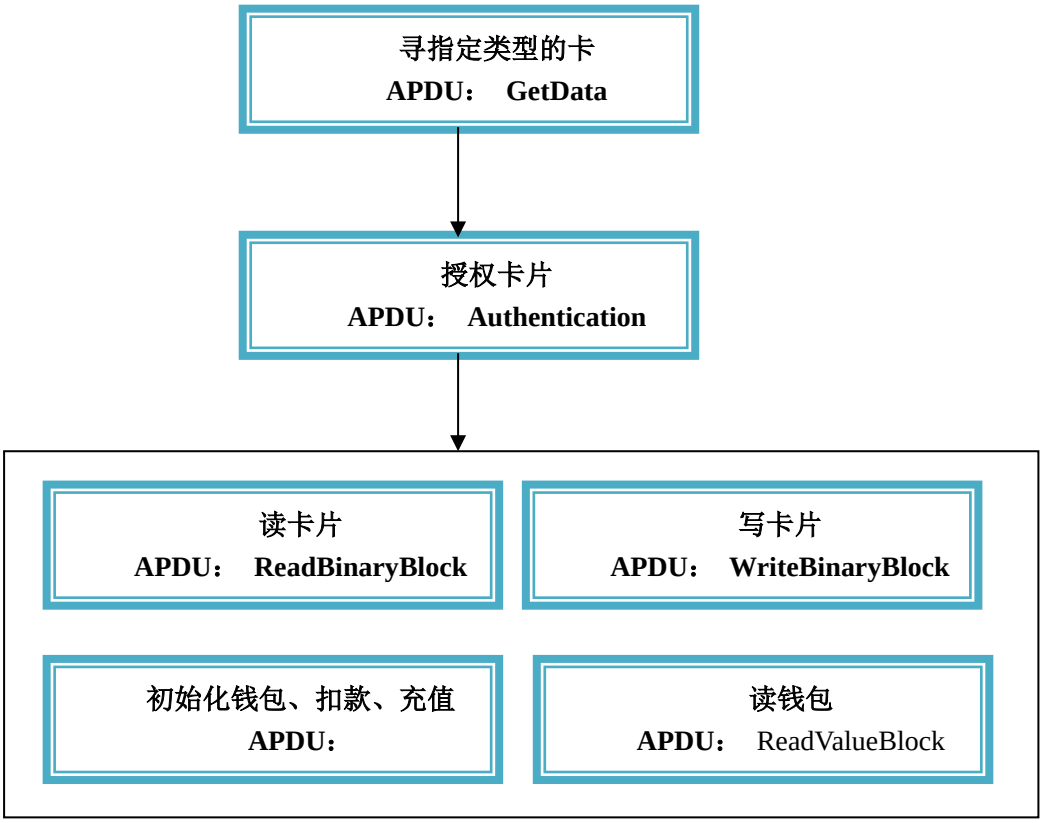


存储卡操作需要带 SAM 操作流程如下：



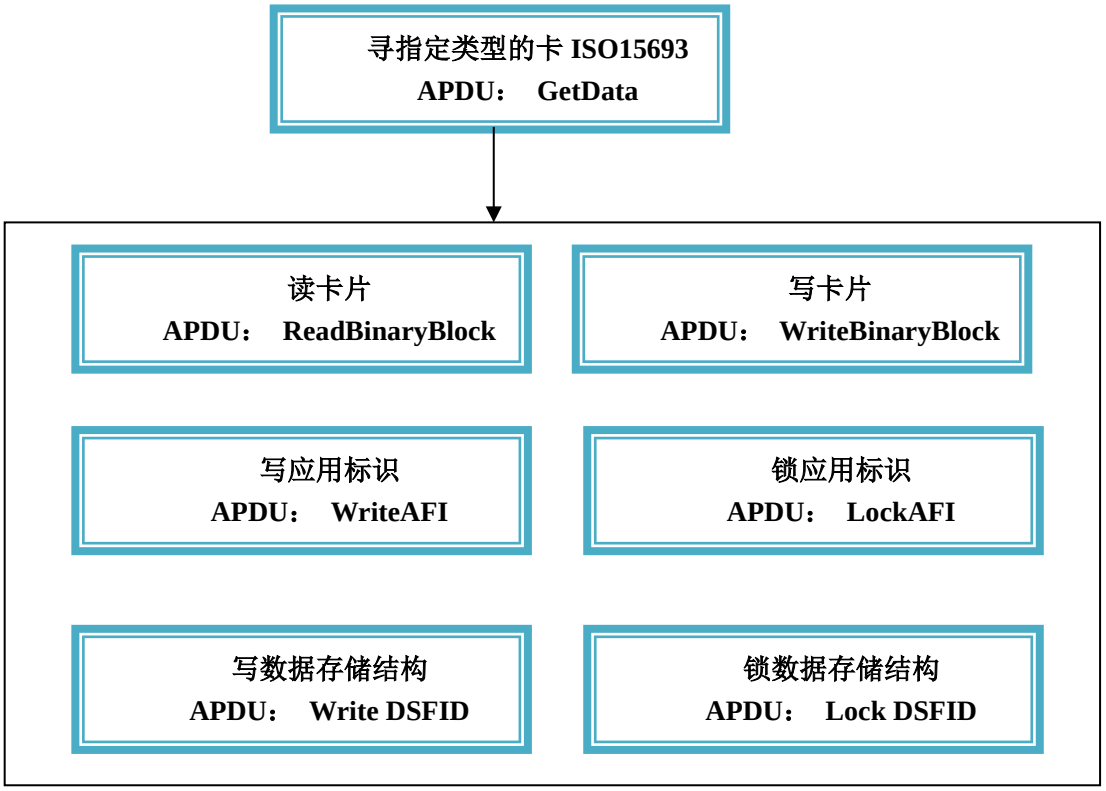
存储卡和单一 SAM 操作不需切换，若需要对多个 SAM 卡操作，则在操作这个 SAM 卡之前，需**切换智能卡类别**去切换指定 SAM 卡。

如常见的 MIFARE S50/70 卡片操作：



以上操作不带 SAM，若带 SAM 卡操作，见上面流程。

如 ISO15693Tag 操作：

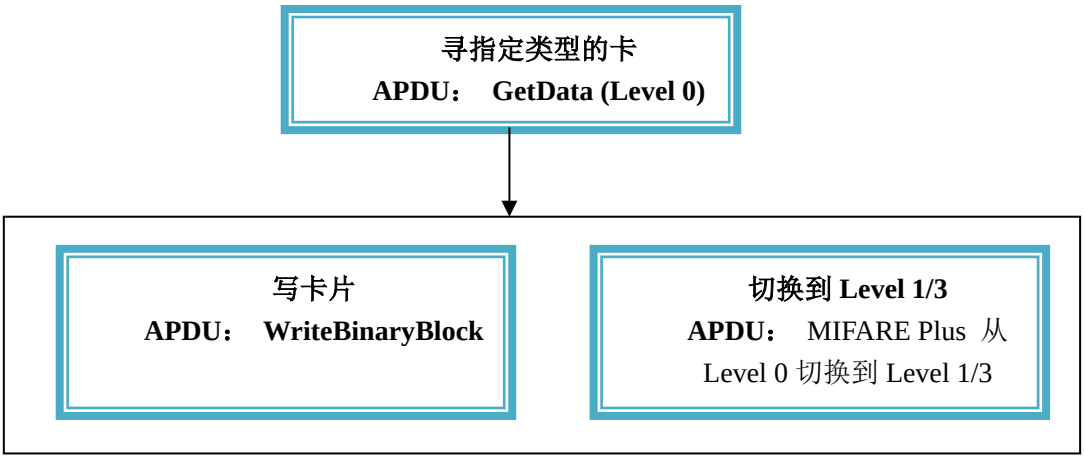


ISO15693 Tag 操作通过 ReadBinaryBlock 和 WriteBinaryBlock 仅仅针对最后寻到的一张 Tag，若需要对指定 UID 的一个 Tag 操作，可以参考**非标准 APDU（自定义部分）。**

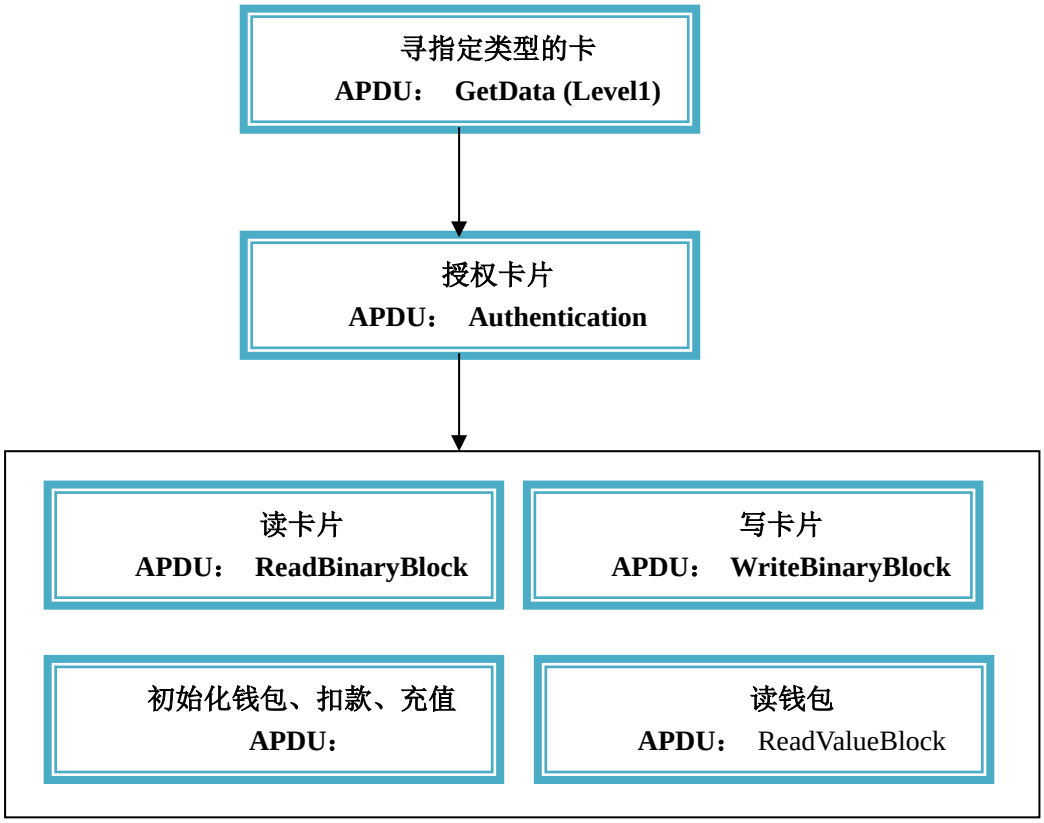
如 MIFARE Plus 卡片操作如下：

MIFARE Plus 卡片结构见附录，在 GetData 中针对 MIFARE Plus 有不同的 GetData 指令，是因为 MIFARE Plus 分为 4 个安全级别（Level 0~Level 3），不同的安全级别对寻卡操作不同，有的只需要寻卡片序号，有的需要寻卡后需要对卡片进行复位操作。其中 MIFARE Plus Level 1 兼容原来的 MIFARE One，所有操作同 MIFARE One。

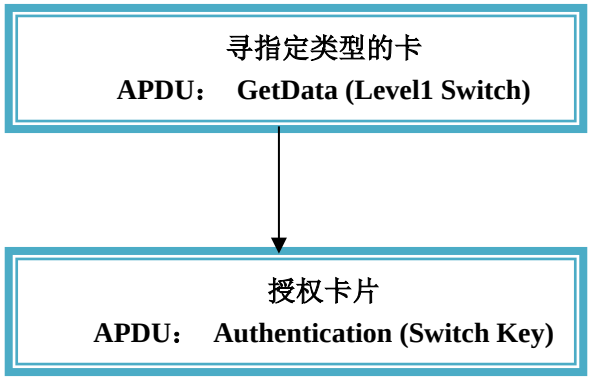
Level 0 操作如下：



Level 1 操作：

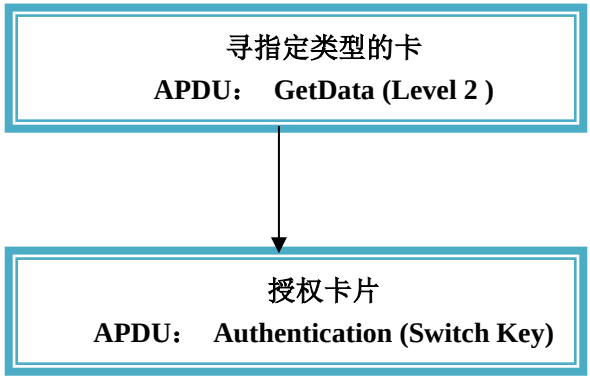


Level 1 Switch 操作：



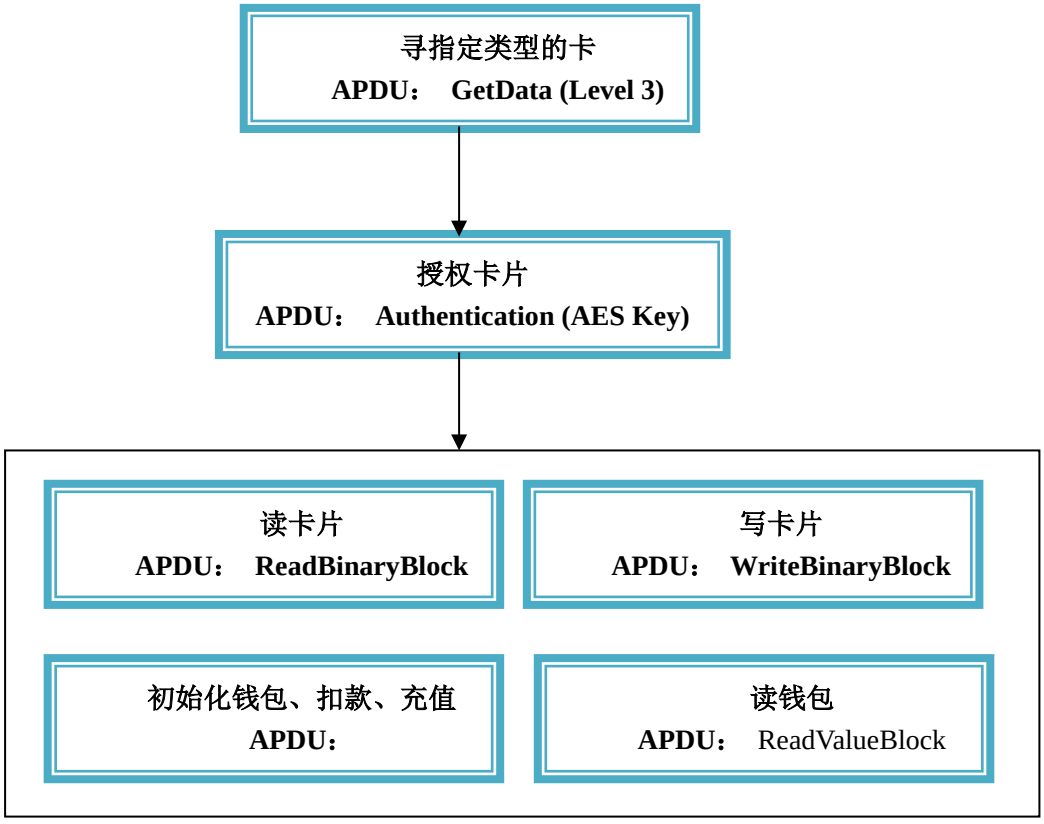
注意从 Level 1 切换到其它 Level，GetData 寻卡类型有区别，假如想从 Level 1 切换到 Level 2，那么 Switch Key 就用 Switch Key2。

Level 2 操作：



假如想从 Level 2 切换到 Level 3，那么 Switch Key 就用 Switch Key3。

Level 3 操作：



其他类别卡片操作基本类似，基本都是用到 GetData、ReadBinaryBlock、WriteBinaryBlock 指令操作，若需要对寻卡参数进行设定，请参考**非标准 APDU（自定义部分）。**

对于 LCD 操作、时钟操作、当前智能卡操作切换、SAM 复位 baudrate、LED、蜂鸣器等操作请参考**非标准 APDU（自定义部分）。**

附录 A

MIFARE Plus Level 3 的数据及密钥存储结构和 MIFARE One 有所区别，结构如下：

块 相 对地址		块地址	对应密钥块地址
Sector0			
Block0	数据块	0x0000	A 密 钥 ： 0x4000 B 密 钥 ： 0x4001
Block1	数据块	0x0001	
Block2	数据块	0x0002	
Block3	数据块	0x0003	
Sector1			
Block0	数据块	0x0004	A 密 钥 ： 0x4002 B 密 钥 ： 0x4003
Block1	数据块	0x0005	
Block2	数据块	0x0006	
Block3	数据块	0x0007	
….			
Sector31			
Block0	数据块	0x007C	A 密 钥 ： 0x403E B 密 钥 ： 0x403F
Block1	数据块	0x007D	
Block2	数据块	0x007E	
Block3	数据块	0x007F	
配置块			
	MFP Configuration Block	0xB000	
	Installation Identifier	0xB001	
	ATS Information	0xB002	
	Field Configuration Block	0xB003	
Key 块			
	AES Sector Keys	0x4000~0x403F	
	AES Sector Keys	0x4040~0x404F	
	Originality Key	0x8000	
	Card Master Key	0x9000	
	Card Configuration Key	0x9001	
	Level 2 switch Key	0x9002	
	Level 3 switch Key	0x9003	
	SL1 Card Authentication Key	0x9004	
	Select VC Key	0xA000	
	Proximity Check Key	0xA001	
	VC Polling ENC Key	0xA080	
	VC Polling MAC Key	0xA081	

注意：

1、 蓝色和黄色部分是关联部分。即数据区和密钥区对应部分（仅仅是在 Level 2/3 才对应，因只有级别 2/3 才使用到 AES 密钥认证）。

2、 在安全级别 Level 1，是和 MIFARE classic 兼容的，每个扇区最后一块为密钥和配置块。

3、 AES 密钥分为 A/B 密钥是人为划分，是为了同 MIFARE classic 概念相同。在 PLUS 内部一个扇区是对应地址连续的 AES 密钥块。

4、 主要掌握如下 key：

AES Sector Keys：

在 Level 2/3 中对数据的授权采用 AES Key 授权。该密钥可以在 Level 0 写入，或者通过 AES Sector Keys 对卡片授权而修改 AES Key。

CardMasterKey：

通过对该 Key 的授权，可以改变 **Card Configuration Key 和 Level 2/3 switch Key**

Card Configuration Key：

通过对该 key 的授权，可以改变 MFP Configuration Block 配置块内容。

Level 2 switch Key：

通过对该 key 的授权，可以从 Level 1 切换 Level 2。

Level 3 switch Key：

通过对该 key 的授权，可以从 Level 2 切换 Level 3，或从 Level 1 切换到 Level 3 。

5、 在 Level 0，除了出厂写入的用户不能修改的密钥外，都可以以明文方式写入，一般在 Level 0 做初始化操作。注意，必须在该安全级别写入 0x9000～0x9003 块。

6、 Level 3 级别支持明文、AES 加密、加密且带 MAC 方式读写方式。本读卡器采用的是最保密的方式读写 MIFARE Plus 块：加密且带 MAC 方式。

1 文件修改记录

日期	版本号	修改内容
2018.03.27	V1.12	增加 4.2.4　General Authenticate Command 章节 增加 4.3.45　直接传输章节
2018.05.11	V1.13	增加对 MR880 的指令介绍 修改 4.3.36，4.3.37 的说明
2018.06.13	V1.14	修改对读写 Flash 的用户可操作地址范围。 添加对支持字库的说明。
2018.07.05	V2.00	修正文字拼写问题
2018.08.16	V2.3.3	修正个别命令参数的错误描述