TCOS 用户手册

TCOS User Manual

Version 1.5.0

Nov 6, 2016

Document No. : TCOS-UM-1.5.0

Confidential Proprietary - DO NOT COPY

版本历史记录：

版本	作者	日期	描述
V1.0	T.	May 28，2016	首版
V1.1	T.	Jun 2，2016	添加 NFC 应用
V1.2	T.	Jun 5，2016	添加 TypeA/B 转换，通信速率切换，UID 设置
V1.3	T.	Aug 2, 2016	添加 ISO14443-3B 层命令。14443B 的 3、4 层互相转换命令
V1.3.1	T.	Aug 7, 2016	明确了 CCK。修改了 14443B 的 3、4 层互相转换命令统一使用 CCK 来实现。
V1.3.2	T.	Sep 7,2016	修改 Set UID 命令支持 Type A 卡输出不同的 UID 长度 4，7，10 个字节。
V1.3.3	T.	Sep 9,2016	补充 Set UID 命令支持用户定义任意的 UID/PUPI 值
V1.4.0	T.	Otc 4,2016	增加 Set FSCI、Set SFGI、Set FWI 命令。修改 Set Type 命令，这个命令成功执行后卡内不再执行软复位。 添加 Test Protocol 命令，用于测试 APDU、WTX、FSDI。
V1.5.0	T.	Nov 8,2016	增加模拟 qPBOC 命令测试。

目　录

1 引用标准

[1] ISO 7816-1 first edition 1998-10-15
Identification cards - Integrated circuit(s) cards with contacts
Part 1: Physical characteristics

[2] ISO 7816-2 first edition 1999-03-01
Identification cards - Integrated circuit(s) cards with contacts
Part 2: Dimensions and location of the contacts

[3] ISO 7816-3 second edition 1997-12-15
Identification cards - Integrated circuit(s) cards with contacts
Part 3: Electronic signals and transmission protocols

[4] ISO 7816-4 first edition 1995-09-01
Amendment 1 1997-12-15
Identification cards - Integrated circuit(s) cards with contacts
Part 4: Interindustry commands for interchange

[5] Federal Information Processing Standards Publication
1980 December 2, FIPS PUB 81
DES Modes of Operation

[6] PBOC 第3、4、5、6、7、10部分

JR/T0025.4--2005

[7] EMV 2000
Integrated Circuit Card
Specification for Payment Systems
Version 4.0 December, 2000

[8] Visa ICC Specification version 1.4.0（VIS 1.4.0）

31 October 2001

[9] GB-T15150-1994 （ISO 8583-1987）

产生报文的银行卡一交换援文规范一金融交易内容

[10] GBT2659-2000 （ISO 3166-1-1997）
世界各国和地区名称代码

[11] Type 4 Tag Operation Technical Specification, Version 2.0, 2010-11-18
NFCForum-TS-Type-4-Tag_2.0

[12] NFC Data Exchange Format (NDEF), Version 1.0, 2006-07-24
NFCForum-TS-NDEF_1.0

[13] NFC Record Type Definition (RTD) Specification, Version 1.0, 2006-07-24
NFCForum-TS-RTD_1.0

[14] NFC Text Record Type Definition Technical Specification, Version 1.0, 2006-07-24
NFCForum-TS-RTD_Text_1.0

[15] NFC URI Record Type Definition Technical Specification, Version 1.0, 2006-07-24
NFCForum-TS-RTD_URI_1.0

2 定义和缩略语

缩略语	定义
CLA	命令类别
INS	命令代码
P1,P2	命令参数 1、命令参数 2
Lc	APDU 命令的数据长度
PCB	协议控制字节（protocol control byte ）
R-block	接收准备数据块
S-block	管理数据块
CRC	循环冗余码校验
NDEF	NFC Data Exchange Format（NFC 数据交换格式）
CC File	Capability Container File(能力容器文件)。CC 文件应该是一个只读 EF 文件，并且 FID 固定为 0xE103
NDEF file	存储 NDEF 数据的具体文件
Proprietary File	专用文件。专用文件是一个 EF 文件。里面保存的数据是专有数据（proprietary data）。用户私自定义的数据放这里
Proprietary File Control TLV	专用文件控制 TLV 记录.一条 TLV 记录占用 8 个字节。
NDEF message	NDEF 信息。一条 NDEF 信息包括一条或多条 NDEF 记录
NDEF record	NDEF 记录。一条 NDEF 记录包含一个有效载荷描述。这个描述是由一个类型，一个长度，一个可选的标识组成的。
FSCI	Frame Size for proximity Card Integer
FSC	Frame Size for proximity Card (卡片接收数据缓冲区大小)
SFGI	Start-up Frame Guard time Integer
SFGT	Start-up Frame Guard Time
FWI	Frame Waiting time Integer
FWT	Frame Waiting Time(帧等待时间)
FWT_{MIN}	Minimun Frame Waiting Time（最小帧等待时间）
FWT_{MAX}	Maximum Frame Waiting Time（最大帧等待时间）
FWT_{TEMP}	temporary Frame Waiting Time （临时帧等待时间）
WTX	Waiting Time eXtension（帧等待时间扩展）
WTXM	Waiting Time eXtension Multiplier（帧等待时间扩展乘数）
FSD	Frame Size for proximity coupling Device(读写器接收数据缓冲区大小)
FSDI	Frame Size for proximity coupling Device Integer

3 TCOS 综述

在 TCOS 中，仅仅存在一条称为卡片控制密钥的密钥(CCK)。这个密钥用于所有跟安全相关的命令中，使用 Change Key 命令可以修改这条密钥的值。密钥长度为 16 个字节，密钥的初始值为 16 个字节零，使用的算法为 DES 或 3DES 算法。

TCOS 特点：

- 符合 ISO14443-1/2/3/4 标准
- 非接触通信速率支持最高 424kbit/s，同时可配置支持 106/212/424kbit/s。
- 支持非接触通信测试 WTX。
- 支持 NFC Forum Type 4 Tag 应用（NFC-A 和 NFC-B）。
- TCOS 默认是 ISO14443 Type A 模式。可以通过命令设置成 ISO14443 Type B 模式。
- TCOS 默认通信速率是 106kbit/s。通信速率兼容性更好。
- TCOS 工作在 Type A 模式下，默认是每张卡的 UID 都是不同的，UID 长度是 4 个字节。TCOS 支持设置随机数为 UID 卡号或用户定义自己的 UID 卡号；同时支持设置 UID 长度为 4/7/10 个字节。
- TCOS 工作在 Type B 模式下，默认的每张卡 PUPI 是固定值，并且每张卡的 PUPI 都是唯一的。TCOS 支持命令设置 PUPI 值为随机数或者是用户定义的 PUPI 值。
- TCOS 支持用户通过命令设置 FSCI、SFGI、FWI 等底层通信参数，方便 ATM 机、POS 机、读写器等终端测试 ISO14443 协议的完整性和兼容性。

应用场景：

- POS 机、ATM 机、读写器、等各种终端的 L1、L2 测试
- qPBOC 模拟测试
- NFC 设备测试
- NFC 应用
- 授权应用（自动化设备授权、软件授权、安卓软件授权）

4 命令

4.1命令汇总表

CLA (hex)	INS (hex)	意义
00	84	取随机数（Get Challenge）
00	82	外部认证（External Authentication）
00	88	内部认证（Internal Authentication）
00	B0	读二进制数据（Read Binary）
00	D6	更新二进制数据（Update Binary）
90	B2	读系统信息（Read SysData）
90	04	初始化卡（Init Card）
90	24	修改密钥（Change Key）
90	DA	切换页（Change Page）
90	B0	读数据（Read Data）
90	D6	更新数据（Update Data）
90	B1	读寄存器（Read Register）
90	D7	更新寄存器（Update Register）
90	F0	测试协议(Test protocol)（APDU、WTX、FSDI）
90	F2	计算 CRC（Calculate CRC）
90	F4	设置类型（Set Type）（Type A/B）
90	F6	设置通信速率（Set Speed） (106/212/424kbit/s)
90	F8	设置 UID/PUPI（Set UID/PUPI）（固定或随机，长度，自定义）
90	FA	设置卡片接收缓冲区大小（Set FSCI）
90	FC	设置起始帧保护时间（Set SFGI）
90	FE	设置帧等待时间（Set FWI）
90	44	NFC 应用初始化（Init NFCApp）
90	14	停留 ISO14443-3 协议层(Stay ISO14443-3 Level)
90	14	退出 ISO14443-3 协议层(Exit ISO14443-3 Level)
00	36	读卡片全局唯一标识符（Read GUID）
00	06	更新卡片全局唯一标识符（Update GUID）

4.2命令 APDU 简表

名称	CLA	INS	P1	P2	Lc/Le	CData/RData
Get Challenge	00	84	00	00	Le=08	
ExtAuthen	00	82	00	00	Lc=08	密文
IntAuthen	00	88	00	00	Lc=08	随机数
Read Binary	00	B0	AddrH	AddrL	Le=00…FF	读出的二进制数
Update Binary	00	D6	AddrH	AddrL	Lc=01…FF	待更新二进制数
Read SysInfo	90	B2	9X	00	Le=00	
Init Card	90	04	00	00	Le=00	
Change Key	90	24	00	00	Lc=10	新卡片控制密钥值（CCK）
Change Page	90	DA	00	00	Lc=01	PageNum
Read Data	90	B0	AddrH	AddrL	Le=00…FF	读指定物理地址数据
Update Data	90	D6	AddrH	AddrL	Lc=01…FF	待更新指定物理地址数据
Read Register	90	B1	00	RegAddr	Le=01	Register value
Update Register	90	D7	00	RegAddr	Lc=01	Register value
Test Protocol	90	F0	01	XX	var	Var （测试 APDU）
	90	F0	02	XX	var	Var （测试 WTX）
	90	F0	03	XX	var	Var （测试 FSDI）
Calculate CRC	90	F2	AddrH	AddrL	Lc=02 or 04	NumOfPages + [Expect CRC]
Set Type	90	F4	CC	CC	Lc=01	0xBB or 0xAA
Set Speed	90	F6	CC	CC	Lc=01	TA1
Set UID	90	F8	CC	CC	Lc=01	0xFF or 0xAB (固定或随机)
	90	F8	DD	DD	Lc=01	0x04, 0x07, 0x0A （A 卡 UID 长度）
	90	F8	EE	EE	Lc=0xFF or 01	设置或清除用户自定义 UID/PUPI
Set FSCI	90	FA	CC	CC	Lc=01	0x00……0x08
Set SFGI	90	FC	CC	CC	Lc=01	0x00……0x0E
Set FWI	90	FE	CC	CC	Lc=01	0x00……0x0E
Init NFCApp	90	44	00	00	Le=00	00
Stay ISO14443-3	90	14	44	33	Lc=10	卡片控制密钥值（CCK）
*Exit ISO14443-3	90	14	44	34	Lc=10	CCK (TPDU 14443-3 层命令)
**Read GUID	00	36	00	00	Le=08	读出 8 个字节的 GUID 值
**Update GUID	00	06	00	00	Lc=08	新 GUID 值

备注：

1、加单星号（*）的命令是 TPDU 命令，是运行在 ISO1444-3 层的命令。不需要加 PCB、NAD、CID 等数据域的。

2、加双星号（**）的命令是双重命令，既是 ISO14443-3 层命令，又是 ISO14443-4 层命令。

4.3取随机数（Get Challenge）

任务	获取一个随机数	
条件	无	
输入	CLA	‘00’
	INS	‘84’
	P1	‘00’
	P2’	‘00’
	Le	’00’…’FF’
输出	数据 = 随机数	
备注	Le=00 时，返回 256 个字节随机数 Le=FF 时，返回 255 个字节随机数	

例子 1：取 8 个字节的随机数
CMD_APDU = 00 84 00 00 08
RSP_APDU = D8 7B FB 84 8A 82 CF 3E 90 00

例子 2：取 1 个字节的随机数
CMD_APDU = 00 84 00 00 01
RSP_APDU = E2 90 00

例子 3：取 4 个字节的随机数
CMD_APDU = 00 84 00 00 04
RSP_APDU = 1C C4 77 CF 90 00

例子 4：取 16 个字节的随机数
CMD_APDU = 00 84 00 00 10
RSP_APDU = 54 65 C8 AA DB A2 7A 56 1A 8B 79 5E CA CD EA 4B 90 00

例子 5：取 20 个字节的随机数
CMD_APDU = 00 84 00 00 14
RSP_APDU = CA 00 CB E4 53 52 88 06 81 5C 6C 87 9F 2A 57 EB 33 A2 95 7E 90 00

4.4外部认证（External Authentication）

任务	认证成功之后，才允许运行一些关键性的命令。	
条件	先取随机数命令。	
输入	CLA	‘00’
	INS	‘82’
	P1	‘00’
	P2	‘00’
	Lc	‘08’
	Data	加密数据
	Le	不存在
输出	无	
备注	如果认证失败，返回’63CX 外部认证命令最多允许尝试次数 10 次	

例子：

Step1: Get 8 bytes random number

CMD_APDU = 00 84 00 00 08

RSP_APDU = EA 03 B4 00 1F E1 01 9F 90 00

Step2: Execute ExtAuthen command to start DES decryption

CMD_APDU = 00 82 00 00 08 6E 4C 58 B4 FD FC E5 C4

RSP_APDU = 90 00

注释： $Key = "00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00"

$PlainText = “EA 03 B4 00 1F E1 01 9F”

$CipherText = TripleDesEcbEncrypt $PlainText, $Key

4.5 内部认证（Internal Authentication）

任务	用卡片中的密钥对终端发来的随机数加密。	
条件		
输入	CLA	‘00’
	INS	‘88’
	P1	‘00’
	P2	‘00’
	Lc	认证相关数据长度
	Data	认证相关数据
输出	加密后的数据	

例子：

CMD_APDU = 00 88 00 00 08 AA BB CC DD EE FF 11 22

RSP_APDU = A5 8F 2E C6 B8 19 50 57 90 00

备注：

1、新卡的默认密码是 16 个字节全零： 用户密钥 = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

2、卡内把命令中输入的 8 个字节数据用用户密钥加密，加密后的密文通过命令响应返回给终端。

4.6读二进制数据（Read Binary）

任务	读出 EEPROM 中的数据流	
条件	如果 Le 等于零，所有允许的数据将会被读出，直到最大的响应数据的大小	
输入	CLA	‘00’
	INS	‘B0’
	P1	P1 = 偏移量的高字节
	P2	P2 = 偏移量的低字节
	Le	期望读取的数据长度
输出	数据 = /*读出的数据流*/	

例子：

ResetCard

CMD_APDU = 00 B0 00 00 06

RSP_APDU = 00 00 00 00 00 00 90 00

CMD_APDU = 00 B0 00 00 20

RSP_APDU = 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 90 00

CMD_APDU = 00 D6 00 00 10 AA BB CC DD EE FF 00 11 22 33 44 55 66 77 88 99

RSP_APDU = 90 00

CMD_APDU = 00 B0 00 00 10

RSP_APDU = AA BB CC DD EE FF 00 11 - 22 33 44 55 66 77 88 99 90 00

4.7更新二进制数据（Update Binary）

任务	修改 EEPROM 中的数据流	
条件	无条件	
输入	CLA	‘00’
	INS	‘D6’
	P1	P1 = 偏移量的高字节
	P2	P2 = 偏移量的低字节*/
	Lc	写入的数据长度
	Data	数据 = /*输入待更新的数据流*/
输出	无	

例子 1：在地址 0x0000 更新 16 个字节数据
ResetCard

CMD_APDU = 00 D6 00 00 10 AA BB CC DD EE FF 00 11 22 33 44 55 66 77 88 99
RSP_APDU = 90 00

CMD_APDU = 00 B0 00 00 10
RSP_APDU = AA BB CC DD EE FF 00 11 22 33 44 55 66 77 88 99 90 00

例子 2：在地址 0x5566 更新 16 个字节数据
ResetCard

CMD_APDU =00 D6 55 66 10 01 02 03 04 05 06 07 08 11 22 33 44 55 66 77 88
RSP_APDU = 90 00

CMD_APDU = 00 B0 55 66 10
RSP_APDU = 01 02 03 04 05 06 07 08 11 22 33 44 55 66 77 88 90 00

4.8读系统信息（Read SysInfo）

任务	读取卡片中系统信息：包括软件版本，物理卡号等	
条件	无条件限制	
输入	CLA	'90'
	INS	'B2'
	P1	'9X'
	P2	'00'
	Le	'00'
输出	数据 = /*读出的数据流*/	
备注		

例子 1：读 COS 版本
CMD_APDU = 90 B2 90 00 00
RSP_APDU = 54 43 4F 53 20 56 31 2E 34 2E 30 90 00

例子 2：读 Project build date
CMD_APDU = 90 B2 91 00 00
RSP_APDU = 4F 63 74 20 30 33 20 32 30 31 36 00 90 00

例子 3：读 Company ID
CMD_APDU = 90 B2 92 00 00
RSP_APDU = 54 49 41 4E 20 41 4E 20 58 49 4E 90 00

例子 4：读 Chip ID
CMD_APDU = 90 B2 93 00 00
RSP_APDU = 54 41 42 33 32 90 00

例子 5：读 Chip Physical ID
CMD_APDU = 90 B2 94 00 00
RSP_APDU = 4F 22 0A 15 48 5D FF 2D F5 0C 40 64 00 90 00

例子 6：读 Support Functions
CMD_APDU = 90 B2 95 00 00
RSP_APDU = 4E 46 43 20 71 50 42 4F 43 20 49 44 43 41 52 44 49 53 4F 31 34 34 34 33 41 26 42 20 49 4F 53 50 45
45 44 90 00

例子 7：读 Support Commands
CMD_APDU = 90 B2 96 00 00
RSP_APDU = 54 41 42 3D 46 34 7C 49 4F 3D 46 36 7C 55 49 44 3D 46 38 7C 46 53 43 49 3D 46 41 7C 53 46 47 49
3D 46 43 90 00

4.9初始化卡（Init Card）

任务	初始化一张新卡或重新初始化一张已初始化过的卡。 除了卡控制密钥以外，清除卡中所有的文件和密钥。 命令还实现更新 ATR 历史字节的功能。	
条件	1、 必须先发取随机数命令 2、 成功执行了外部认证命令	
输入	CLA	'90'
	INS	'04'
	P1	'00'
	P2	'00'
	Le	'00'
输出	无	
备注		

SW1	SW2	意义
'90'	'00'	successful
'65'	'81'	memory failure
'67'	'00'	invalid Lc value
'69'	'84'	referenced data invalidated(未取随机数)
'69'	'85'	conditions of use not satisfied
'69'	'88'	SM data objects incorrect
'6A'	'86'	incorrect parameters P1, P2

备注：成功执行了 init card 命令之后，卡中的 EEPROM 数据被全部清除。

例子：先在卡中写入内容，执行 Init Card 命令之后，验证内容被清除了
ResetCard

CMD_APDU = 00 D6 00 00 0B 12 34 56 78 90 A1 B2 C3 D4 E5 F6
RSP_APDU = 90 00

CMD_APDU = 00 B0 00 00 0B
RSP_APDU = 12 34 56 78 90 A1 B2 C3 D4 E5 F6 90 00

CMD_APDU = 00 84 00 00 08
RSP_APDU = B8 93 17 79 7F DB FC 9B 90 00

CMD_APDU = 00 82 00 00 08 49 5A AD D4 A6 EA 0D 75
RSP_APDU = 90 00

CMD_APDU = 90 04 00 00 00
RSP_APDU = 90 00

CMD_APDU = 00 B0 00 00 0B
RSP_APDU = 00 00 00 00 00 00 00 00 00 00 00 90 00

4.10 修改密钥（Change Key）

任务	修改为用户指定的密钥内容	
条件	需要外部认证成功执行了	
输入	CLA	'90'
	INS	'24'
	P1	'00'
	P2	'00'
	Lc	'10'
	Data	新密钥值
输出	无	
备注		

说明：
新卡的默认密码是 16 个字节全零： 用户密钥 = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
为了更加安全性，比如修改新的用户密钥 = 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F
用户密钥用于外部认证命令。只有成功执行了外部认证命令，才能执行 Init card 命令和 change key 的命令。

例子：
ResetCard

CMD_APDU = 00 84 00 00 08
RSP_APDU = EF A8 52 E9 43 E9 E9 B9 90 00

CMD_APDU =00 82 00 00 08 40 FD 13 5D 85 68 E9 CF
RSP_APDU = 90 00

CMD_APDU =00 24 00 00 10 40 41 42 43 44 45 46 47 48 49 4A 4B 4C 4D 4E 4F
RSP_APDU = 90 00

4.11 切换页（Change Page）

任务	选择不同的页（64K 空间为一页）	
条件		
输入	CLA	'90'
	INS	'DA'
	P1	'00'
	P2	'00'
	Lc	'01'
	Data	PageNum
输出	无	
备注		

例子：

ResetCard

CMD_APDU =90 DA 00 00 01 01

RSP_APDU = 90 00

CMD_APDU =90 DA 00 00 01 00

RSP_APDU = 90 00

4.12 读数据（Read Data）

任务	读出 EEPROM 中指定物理地址的数据流	
条件	如果 Le 等于零，所有允许的数据将会被读出，直到最大的响应数据的大小	
输入	CLA	‘90’
	INS	‘B0’
	P1	P1 = 物理地址高字节
	P2	P2 = 物理地址低字节
	Le	期望读取的数据长度
输出	数据 = /*读出的数据流*/	
备注	命令响应最大返回 256 个字节。	

例子：

ResetCard

CMD_APDU = 90 DA 00 00 01 01
RSP_APDU = 90 00

CMD_APDU =90 B0 10 00 10
RSP_APDU = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 90 00

CMD_APDU =90 B0 80 00 10
RSP_APDU = 67 00

4.13 更新数据（Update Data）

任务	修改 EEPROM 中指定物理地址的数据	
条件	无条件	
输入	CLA	‘90’
	INS	‘D6’
	P1	P1 = 物理地址高字节
	P2	P2 = 物理地址低字节*/
	Lc	写入的数据长度
	Data	数据 = /*输入待更新的数据流*/
输出	无	

例子：

ResetCard

CMD_APDU = 90 DA 00 00 01 01
RSP_APDU = 90 00

CMD_APDU =90 B0 10 00 10
RSP_APDU = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 90 00

CMD_APDU = 90 D6 10 00 08 A1 A2 A3 A4 A5 A6 A7 A8
RSP_APDU = 90 00

CMD_APDU =90 B0 10 00 10
RSP_APDU = A1 A2 A3 A4 A5 A6 A7 A8 00 00 00 00 00 00 00 00 90 00

4.14 读寄存器（Read Register）

任务	读出 MCU 中寄存器的数值	
条件	无	
输入	CLA	‘90’
	INS	‘B1’
	P1	P1 = 00
	P2	P2 = 寄存器地址（RegAddr）
	Le	‘01’
输出	数据 = 寄存器数值	
备注	一次只能读取一个寄存器数值	

例子：

ResetCard

CMD_APDU = 90 B1 00 87 01

RSP_APDU = 80 90 00

CMD_APDU = 90 B1 00 FA 01

RSP_APDU = 20 90 00

4.15 更新寄存器（Update Register）

任务	修改 MCU 中特殊功能寄存器的数值	
条件	无条件	
输入	CLA	‘90’
	INS	‘D7’
	P1	P1 = 00
	P2	P2 = 寄存器地址（RegAddr）
	Lc	‘01’
	Data	寄存器新值
输出	无	

例子：

ResetCard

CMD_APDU = 90 D7 00 87 01 01

RSP_APDU = 90 00

CMD_APDU = 90 D7 00 FA 01 80

RSP_APDU = 90 00

4.16 测试协议（Test Protocol）

任务	测试 ISO14443 和 ISO7816 命令格式正确性	
条件	无条件	
输入	CLA	‘90’
	INS	F0’
	P1	P1 = ’01’ ---- 测试 APDU P1 = ’02’ ---- 测试 WTX P1 = ’03’ ---- 测试 FSDI
	P2	当 P1=01 时，P2=01 是命令格式 1，P2=02 是命令格式 2，P=03 是命令格式 3，P2-04 是命令格式 4。 当 P1=02 时，P2=01，表示卡片发送一个 WTX S-Block 给读写器 当 P1=03 时，用于设置卡中 FSDI 值，用于调试 Chain Block 情况
	Lc	Var
	Data	Var
输出	无	

测试 APDU 例子：

CMD_APDU = 90 F0 01 01 00 （command format 1）
RSP_APDU = 90 00

CMD_APDU = 90 F0 01 02 08 (command format 2)
RSP_APDU = 00 00 00 00 00 00 00 00 90 00

CMD_APDU = 90 F0 01 03 06 01 02 03 04 05 06 (command format 3)
RSP_APDU = 90 00

CMD_APDU = 90 F0 01 04 03 11 22 33 05 (command format 4)
RSP_APDU = 11 22 33 04 05 90 00

CMD_APDU = 90 F0 01 05 06 A1 B2 C3 D4 E5 F6 (响应数据等于命令输入数据)
RSP_APDU = A1 B2 C3 D4 E5 F6 90 00

CMD_APDU = 90 F0 01 06 08 AA BB CC DD EE FF 11 22 (响应数据等于整个命令 APDU 数据)
RSP_APDU = 90 F0 01 06 08 AA BB CC DD EE FF 11 22 90 00

测试 WTX 例子：

CMD_APDU = 90 F0 02 01 00 (卡片发送一个 WTX S-Block 到读写器)
RSP_APDU = 90 00

CMD_APDU = 90 F0 02 03 00 (卡片发送 3 个 WTX S-Block 到读写器)
RSP_APDU = 90 00

CMD_APDU = 90 F0 02 FF 00 (卡片发送 255 个 WTX S-Block 到读写器)
RSP_APDU = 90 00

CMD_APDU = 90 F0 02 01 06 （卡片发送一个 WTX S-Block 到读写器.并返回响应数据是 6 个字节）
RSP_APDU = 00 94 2C F2 A9 E9 90 00

//(卡片命令中循环 0x0013 * 256 次。期间卡片发送若干个 WTX S-Block 到读写器)
CMD_APDU = 90 F0 02 00 02 00 13
RSP_APDU = 90 00

//(卡片命令中循环 0x0030 * 256 次。期间卡片发送若干个 WTX S-Block 到读写器. 并返回响应数据是 5 个字节)
CMD_APDU = 90 F0 02 00 02 00 30 05
RSP_APDU = 00 C3 5C 32 DF 90 00

测试 FSDI 例子：

CMD_APDU = 90 F0 03 00 00 (Set FSDI = 0, FSD = 16 bytes)
RSP_APDU = 90 00

CMD_APDU = 90 F0 03 02 00 (Set FSDI = 2, FSD = 32 bytes)
RSP_APDU = 90 00

CMD_APDU = 90 F0 03 07 00 (Set FSDI = 7, FSD = 128 bytes)
RSP_APDU = 90 00

CMD_APDU = 90 F0 03 08 00 (Set FSDI = 8, FSD = 256 bytes)
RSP_APDU = 90 00

4.17 计算 CRC（Calculate CRC）

任务	计算 CRC 值	
条件		
输入	CLA	'90'
	INS	'F2'
	P1	StartAddrH
	P1	StartAddrL
	Lc	'02' or '04'
	Data	NumOfPages + (Expect CRC)
输出	无	
备注	页的大小是 512 bytes 如果 Expect CRC 没有出现，则命令响应返回 CRC 值	

4.18 设置类型（Set Type）

任务	设置卡片工作类型 ISO14443 Type A 或 Type B	
条件	无条件	
输入	CLA	‘90’
	INS	‘F4’
	P1	‘CC’
	P2	‘CC’
	Lc	‘01’
	Data	0xBB ---- Type B mode 0xAA --- Type A mode
输出	无	
备注	注意：命令执行成功后，并不是马上使用新的类型，需要等下一个卡片复位才使用新的类型。	

例子 1：设置卡片工作在 Type B 状态
ResetCard

CMD_APDU = 90 F4 CC CC 01 BB
RSP_APDU = 90 00

ResetCard

例子 2：设置卡片工作在 Type A 状态
ResetCard

CMD_APDU = 90 F4 CC CC 01 AA
RSP_APDU = 90 00

ResetCard

4.19 设置通信速率（Set Speed）

任务	设置卡片通信速率（106/212/424kbit/s）	
条件	无条件	
输入	CLA	‘90’
	INS	‘F6’
	P1	‘CC’
	P2	‘CC’
	Lc	‘01’
	Data	TA1
输出	无	
备注	注意：命令执行成功后，并不是马上使用新的 TA1 值，需要等下一个卡片复位才使用新的 TA1 值。 TA1 = 0xB3, 卡片支持 424、212、106kbit/s，强制双向速率一样 TA1 = 0x80, 卡片支持 106kbit/s，强制双向速率一样	

通信速率参数 TA1 的选择：

7	6	5	4	3	2	1	0
M	S8	S4	S2	0	R8	R4	R2

备注： bit7，M = 1，表示强制双向速率一致。=0，不强制。

bit6, S8 = 1.表示卡片发送速率支持 848kbit/s

bit5, S4 = 1.表示卡片发送速率支持 424kbit/s

bit4, S2 = 1.表示卡片发送速率支持 212kbit/s

bit3, 预留

bit2, R8 = 1.表示卡片接收速率支持 848kbit/s

bit1, R4 = 1.表示卡片接收速率支持 424kbit/s

bit0, R2 = 1.表示卡片接收速率支持 212kbit/s

TA1 取值按照强制双向速率一样，分类如下：

0xB3--- 卡片支持 424、212、106kbit/s，强制双向速率一样。

0xA2 --- 卡片支持 424、106kbit/s，强制双向速率一样。

0x91--- 卡片支持 212、106kbit/s，强制双向速率一样。

0x80--- 卡片支持 106kbit/s，强制双向速率一样。

TA1 取值按照双向速率可以不一样，情况一：
0x33--- 卡片支持发送数据速率 424、212、106kbit/s，接收数据速率 424、212、106kbit/s，双向速率可以不一样。
0x32--- 卡片支持发送数据速率 424、212、106kbit/s，接收数据速率 424、106kbit/s，双向速率可以不一样。
0x31--- 卡片支持发送数据速率 424、212、106kbit/s，接收数据速率 212、106kbit/s，双向速率可以不一样。
0x30--- 卡片支持发送数据速率 424、212、106kbit/s，接收数据速率 106kbit/s，双向速率可以不一样。

TA1 取值按照双向速率可以不一样，情况二：
0x23--- 卡片支持发送数据速率 424、106kbit/s，接收数据速率 424、212、106kbit/s，双向速率可以不一样。
0x22--- 卡片支持发送数据速率 424、106kbit/s，接收数据速率 424、106kbit/s，双向速率可以不一样。
0x21--- 卡片支持发送数据速率 424、106kbit/s，接收数据速率 212、106kbit/s，双向速率可以不一样。
0x20--- 卡片支持发送数据速率 424、106kbit/s，接收数据速率 106kbit/s，双向速率可以不一样。

TA1 取值按照双向速率可以不一样，情况三：
0x13--- 卡片支持发送数据速率 212、106kbit/s，接收数据速率 424、212、106kbit/s，双向速率可以不一样。
0x12--- 卡片支持发送数据速率 212、106kbit/s，接收数据速率 424、106kbit/s，双向速率可以不一样。
0x11--- 卡片支持发送数据速率 212、106kbit/s，接收数据速率 212、106kbit/s，双向速率可以不一样。
0x10--- 卡片支持发送数据速率 212、106kbit/s，接收数据速率 106kbit/s，双向速率可以不一样。

TA1 取值按照双向速率可以不一样，情况四：
0x03--- 卡片支持发送数据速率 106kbit/s，接收数据速率 424、212、106kbit/s，双向速率可以不一样。
0x02--- 卡片支持发送数据速率 106kbit/s，接收数据速率 424、106kbit/s，双向速率可以不一样。
0x01--- 卡片支持发送数据速率 106kbit/s，接收数据速率 212、106kbit/s，双向速率可以不一样。
0x00--- 卡片支持发送数据速率 106kbit/s，接收数据速率 106kbit/s，双向速率可以不一样。

总结: 卡片总共可以支持 20 种不同的通信速率测试。

例子 1：卡片支持 424、212、106kbit/s，强制双向速率一样
ResetCard

CMD_APDU = 90 F6 CC CC 01 B3
RSP_APDU = 90 00

ResetCard

4.20 设置 UID/PUPI（Set UID/PUPI）

任务	设置卡片使用固定数或者是随机数作为UID/PUPI 以及设置 UID 长度和用户自定义 UID/PUPI 值	
条件	无条件	
输入	CLA	‘90’
	INS	‘F8’
	P1P2	‘CC CC’ – 设置 UID/PUPI 取固定值或动态值 ‘DD DD’ – 设置 UID 长度 ‘EE EE’ – 设置或清除用户自定义 UID/PUPI 值
	Lc	‘01’ or ‘0B’
	Data	0xFF or 0xAB (P1P2 须取值为 0xCCCC) 0x04 or 0x07 or 0x0A (P1P2 须取值为 0xDDDD) 0xFF | 10 个字节用户自定义 UID/PUPI 值 (P1P2 须取值为 0xEEEE) 0x00 (P1P2 须取值为 0xEEEE)
输出	无	
备注	注意：命令执行成功后，并不是马上使用新的 UID/PUPI 值，需要等下一个卡片复位才使用新的 UID/PUPI 值。 0xFF ---- 使用固定数值作为 UID/PUPI (P1P2 须取值为 0xCCCC) 0xAB --- 使用随机数作为 UID/PUPI (P1P2 须取值为 0xCCCC) 0x04 --- 设置 Type A 卡的 UID 长度为 4 个字节 0x07 --- 设置 Type A 卡的 UID 长度为 7 个字节 0x0A --- 设置 Type A 卡的 UID 长度为 10 个字节 0x00 --- 清除用户自定义 UID/PUPI 值 Type A 卡对应的是 UID。Type B 卡对应的是 PUPI。 注意：为了最好的兼容性，10 个字节用户自定义 UID/PUPI 值的第 1，4 和 7 个字节不能取值为 0x88，否则有些读写器寻卡失败。	

补充: 当执行自定义 UID 值时，UID 的第一个字节 UID0 取值规定：不能取’X8’和’XF’。

正确设置用户自定义 UID/PUPI 范例如下：

CMD_APDU = 90 F8 EE EE 0B FF 11 22 33 44 55 66 77 88 99 AA

CMD_APDU = 90 F8 EE EE 0B FF 00 00 00 00 00 00 00 00 00 00

错误设置用户自定义 UID/PUPI 范例如下：

CMD_APDU = 90 F8 EE EE 0B FF 88 22 33 44 55 66 77 88 99 AA （UID[0]不能为 0x88）

CMD_APDU = 90 F8 EE EE 0B FF 11 22 33 88 55 66 77 88 99 AA （UID[3]不能为 0x88）

CMD_APDU = 90 F8 EE EE 0B FF 11 22 33 44 55 66 88 88 99 AA （UID[7]不能为 0x88）

CMD_APDU = 90 F8 EE EE 0B FF 00 00 00 00 00 00 88 00 00 00 （UID[7]不能为 0x88）

例子 1：设置卡片使用固定数值的 UID/PUPI

ResetCard

CMD_APDU = 90 F8 CC CC 01 FF
RSP_APDU = 90 00

ResetCard

例子 2：设置卡片使用随机数作为 UID/PUPI

ResetCard

CMD_APDU = 90 F8 CC CC 01 AB
RSP_APDU = 90 00

ResetCard

例子 3：设置 Type A 卡的 UID 长度为 7 个字节

ResetCard

CMD_APDU = 90 F8 DD DD 01 07
RSP_APDU = 90 00

ResetCard

例子 4：设置 Type A 卡的 UID 长度为 10 个字节

ResetCard

CMD_APDU = 90 F8 DD DD 01 0A
RSP_APDU = 90 00

ResetCard

例子 5：设置用户自定义 UID/PUPI 值

ResetCard

CMD_APDU = 90 F8 EE EE 0B FF 11 22 33 44 55 66 77 88 99 AA
RSP_APDU = 90 00

ResetCard

例子 6：清除用户自定义 UID/PUPI 值

ResetCard

CMD_APDU = 90 F8 EE EE 01 00
RSP_APDU = 90 00

ResetCard

4.21 设置卡片接收缓冲区大小（Set FSCI）

任务	设置卡片接收缓冲区大小	
条件	无条件	
输入	CLA	‘90’
	INS	‘FA’
	P1	‘CC’
	P2	‘CC’
	Lc	‘01’
	Data	FSCI
输出	无	
备注	FSCI 的取值范围：0 <= FSCI <= 8 注意：命令执行成功后，并不是马上使用新的 FSCI 值，需要等下一个卡片复位才使用新的 FSCI 值。 在 TCOS 中，FSCI 默认值是 0x08，即 FSC 是 256 个字节。	

FSCI	0	1	2	3	4	5	6	7	8
FSC(bytes)	16	24	32	40	48	64	96	128	256

例子 1：设置卡片接收缓冲区是 256 个字节

ResetCard

CMD_APDU = 90 FA CC CC 01 08

RSP_APDU = 90 00

ResetCard

4.22 设置起始帧保护时间（Set SFGI）

任务	设置起始帧保护时间	
条件	无条件	
输入	CLA	‘90’
	INS	‘FC’
	P1	‘CC’
	P2	‘CC’
	Lc	‘01’
	Data	SFGI
输出	无	
备注	SFGI 的取值范围：0 <= SFGI <= 14 注意：命令执行成功后，并不是马上使用新的 SFGI 值，需要等下一个卡片复位才使用新的 SFGI 值。 SFGT = (256 x 16/fc) x 2^SFGI 在 TCOS 中，SFGI 默认值是 0x01，即 SFGT 是 604us。	

注意:当设置SFGI取值过大时,由于卡延时过大,读写器端有种感觉好像卡片没有响应一样。

For SFGI = 0, SFGT is minimal (~ 302 us);For SFGI = 14, SFGT is maximal (~ 4949 ms)

在 Type A 卡中，SFGI 是定义在 ATS 的 TB1 字节中，TB1 的最低四位是代表 SFGI。
在 Type B 卡中，SFGI 是定义在 ATQB 中，在 ATQB 的可选字节中。

SFGI 与 SFGT 关系表															
SFGI	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
SFGT	302us	604us	1.2ms	2.4ms	4.8ms	9.6ms	19ms	38ms	77ms	154ms	309ms	618ms	1.2ms	2.3s	4.9s

例子 1：设置卡片的起始帧保护时间为 604us
ResetCard

CMD_APDU = 90 FC CC CC 01 01
RSP_APDU = 90 00

ResetCard

4.23 设置帧等待时间（Set FWI）

任务	设置帧等待时间	
条件	无条件	
输入	CLA	‘90’
	INS	‘FE’
	P1	‘CC’
	P2	‘CC’
	Lc	‘01’
	Data	FWI
输出	无	
备注	FWI 的取值范围：0 <= FWI <= 14 注意：命令执行成功后，并不是马上使用新的 FWI 值，需要等下一个卡片复位才使用新的 FWI 值。 FWT = (256 x 16/fc) x 2^FWI 在 TCOS 中，FWI 默认值是 0x08，即 FWT 是 77ms。	

在 Type A 卡中，FWI 是定义在 ATS 的 TB1 字节中，TB1 的最高四位是代表 FWI。
在 Type B 卡中，FWI 是定义在 ATQB 中，在 ATQB[11]字节中，这个字节的最高四位代表 FWI。

FWI 与 FWT 关系表															
FWI	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
FWT	302us	604us	1.2ms	2.4ms	4.8ms	9.6ms	19ms	38ms	77ms	154ms	309ms	618ms	1.2ms	2.3s	4.9s

读写器是根据帧等待时间（FWT）判断卡片是否有响应或者是响应超时。读写器在发送了一个命令之后，在等待的 FWT 时间之内，如果没有收到卡片的响应，则认为卡片操作异常，读写器发送超时报告给用户。

如果卡片内由于需要执行 RSA 等需要大量时间处理的情况下，处理时间一般超过了帧等待时间（FWT），则卡片应该发出 WTX（等待时间扩展）管理块（S-Block）给读写器，这样读写器才不认为卡片是没有响应的。

问：FWT 在什么情况下有效？
答：对 Type A 卡而言，是响应了 RATS 命令之后起效。对 Type B 卡而言，是响应了 ATTRIB 命令之后起效。

注意：如果，FWI设置值过小，比如，小于4，容易出现卡片内部处理时间超过FWT值，而读写器提示用户是没有接到卡片响应的情况。（其实，卡片执行命令是成功的）。利用这个特点，来产生一个技巧：可以通过设置小的FWI值，评估COS中，命令的处理时间，或者是评估芯片的写EEPROM的时间性能参数。

例子 1：设置卡片的帧等待时间为 77ms
ResetCard

CMD_APDU = 90 FE CC CC 01 08
RSP_APDU = 90 00

ResetCard

4.24 NFC 应用初始化（Init NFCApp）

任务	卡片执行了这个命令才可以当 NFC Type 4 Tag 卡片使用	
条件		
输入	CLA	'90'
	INS	'44'
	P1	'00'
	P2	'00'
	Le	'00'
输出	无	
备注		

例子：

ResetCard

CMD_APDU = 90 44 00 00 00

RSP_APDU = 90 00

4.25 停留 ISO14443-3 协议层(Stay ISO14443-3 Level)

任务	卡片执行了这个命令,并且复位后,卡片就处于等待接收 ISO14443-3 层命令的状态. 【 ISO14443-4 ➔ ISO14443-3 】	
条件	命令是需要卡片处于 ISO14443-4 协议层时执行	
输入	CLA	'90'
	INS	'14'
	P1	'44'
	P2	'33'
	Lc	'10'
	Data	卡片控制密钥(CCK)
输出	无	
备注	1、 暂时只实现了 ISO14443 Type B 协议的。Type A 协议的将来再添加实现方法。 2、 卡片先完成了 REQB 和 ATTRIB 命令，之后处于等待 14443-3 层私有命令或特殊命令的状态。	

注意：普通用户或不了解 ISO1444-3 部分协议的用户，请不要随意执行这个命令,否则用户可能不懂得或者读写器不支持用户把卡转换成通常的 ISO14443-4 层协议状态。

例子：

ResetCard

CMD_APDU = 90 F4 CC CC 01 BB

RSP_APDU = XX XX

ResetCard

CMD_APDU = 90 14 44 33 10 XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX

RSP_APDU = 90 00

4.26 *退出 ISO14443-3 协议层(Exit ISO14443-3 Level)

任务	卡片执行了这个命令,并且复位后,卡片不再停留在循环接收 ISO14443-3 层命令的状态. 【 ISO14443-3 ➔ ISO14443-4 】	
条件	命令是需要卡片处于 ISO14443-3 协议层时执行	
输入	CLA	'90'
	INS	'14'
	P1	'44'
	P2	'34'
	Lc	'10'
	Data	卡片控制密钥（CCK）
输出	无	
备注	1、 暂时只实现了 ISO14443 Type B 协议的。Type A 协议的将来再添加实现方法。 2、 卡片先执行了 REQB 和 ATTRIB 命令，然后执行这个命令，卡片就可以返回通常的 ISO14443-4 层协议状态了。 3、 这个命令属于 TPDU。所以没有 PCB、NAD、CID 这些。	

例子：
ResetCard

REQB 命令

ATTRIB 命令

CMD_TPDU = 90 14 44 34 10 XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
RSP_TPDU = 90 00

4.27 **读卡片全局唯一标识符（Read GUID）

任务	读出卡片唯一标识符. 【 ISO14443-3 或 ISO14443-4 】	
条件		
输入	CLA	'00'
	INS	'36'
	P1	'00'
	P2	'00'
	Le	'08'
输出	8 个字节的 GUID	
备注	1、命令既属于 TPDU 又属于 APDU。即卡片处于两种不同的协议层都可以执行这个命令。	

ISO14443-4 协议层例子：

ResetCard

CMD_APDU = 00 36 00 00 08

RSP_APDU = XX XX XX XX XX XX XX XX 90 00

ISO14443-3 协议层例子：

ResetCard

REQB 命令

ATTRIB 命令

CMD_TPDU = 00 36 00 00 08

RSP_TPDU = XX XX XX XX XX XX XX XX 90 00

4.28 **更新卡片全局唯一标识符（Update GUID）

任务	更新卡片唯一标识符. 【 ISO14443-3 或 ISO14443-4 】	
条件		
输入	CLA	'00'
	INS	'06'
	P1	'00'
	P2	'00'
	Lc	'08'
	Data	新 GUID
输出	无	
备注	1、命令既属于 TPDU 又属于 APDU。即卡片处于两种不同的协议层都可以执行这个命令。	

ISO14443-4 协议层例子：

ResetCard

CMD_APDU = 00 06 00 00 08 XX XX XX XX XX XX XX XX

RSP_APDU = 90 00

ISO14443-3 协议层例子：

ResetCard

REQB 命令

ATTRIB 命令

CMD_TPDU = 00 06 00 00 08 XX XX XX XX XX XX XX XX

RSP_TPDU = 90 00

5 NFC 应用

初始化 NFC 应用命令：
CMD_ADPU = 90 44 00 00 00
RSP_ADPU = 90 00

6.1 读 NFC Forum Type 4 Tag 标签信息

NFC Forum Type 4 Tag Version 2.0 读标签操作步骤如下：

Resetcard

Setp1：Select NFC AID
CMD_ADPU = 00 A4 04 00 07 D2 76 00 00 85 01 01
RSP_ADPU = 90 00

Setp2：Select CC File
CMD_ADPU = 00 A4 00 0C 02 E1 03
RSP_ADPU = 90 00

Step3：Read CC File
CMD_ADPU = 00 B0 00 00 0F
RSP_ADPU = …… 90 00

Step4：Select NDEF File
CMD_ADPU = 00 A4 00 0C 02 E1 04
RSP_ADPU = 90 00

Step5：Read NLEN field in NDEF File
CMD_ADPU = 00 B0 00 00 02
RSP_ADPU = …… 90 00

Step6：Read NDEF message in NDEF File
CMD_ADPU = 00 B0 00 00 00
RSP_ADPU = …… 90 00

6.2 写 NFC Forum Type 4 Tag 标签信息

NFC Forum Type 4 Tag Version 2.0 标签写入电话号码：13800138000 操作步骤如下：

Resetcard

Setp1：Select NFC AID
CMD_ADPU = 00 A4 04 00 07 D2 76 00 00 85 01 01
RSP_ADPU = 90 00

Setp2：Select CC File
CMD_ADPU = 00 A4 00 0C 02 E1 03
RSP_ADPU = 90 00

Step3：Read CC File
CMD_ADPU = 00 B0 00 00 0F
RSP_ADPU = 00 0F 20 00 80 00 80 04 06 E1 04 00 40 00 00 90 00

Step4：Select NDEF File
CMD_ADPU = 00 A4 00 0C 02 E1 04
RSP_ADPU = 90 00

Step5：Read NLEN field in NDEF File
CMD_ADPU = 00 B0 00 00 02
RSP_ADPU = 00 03 90 00

Step6：Read NDEF message in NDEF File
CMD_ADPU = 00 B0 00 02 13
RSP_ADPU = 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 90 00

Step7：update NDEF message in NDEF File
CMD_ADPU = 00 D6 00 00 12
RSP_ADPU = 00 10 D1 01 0C 55 05 31 33 38 30 30 31 33 38 30 30 30 90 00

Step8: check content correction
CMD_ADPU = 00 B0 00 00 20
RSP_ADPU = 00 10 D1 01 0C 55 05 31 33 38 30 30 31 33 38 30
30 30 00 00 00 00 00 00 00 00 00 00 00 00 00 00 90 00

6 GUID 应用

GUID —-- 全局唯一标识符

6.1 读 GUID

以下是 MCR0110 读写器操作 Type B 卡读 GUID 的实例：

```
ResetCard

   ShowMessage "Get PUPI"
   ExecApduCmd "FF CA 00 00 00"
   CheckSw1Sw2 "90 00"

// 使 SDIO011 读写器处于 RAW 模式
// RAW 命令格式：FF CC 00 00 Lc data
ShowMessage "使 SDIO011 读写器处于 RAW 模式"
ExecApduCmd "FF CC 00 00 02 97 01"
CheckSw1Sw2 "90 00"


// 在 RAW 模式下，发送 REQB 命令：
// 针对 Type B 卡发的 RAW 命令格式 = FF CC 00 00 Lc AE 03 01 00 01 Lc data
ExecApduCmd "FF CC 00 00 09 AE 09 01 00 01 03 05 00 01"
CheckSw1Sw2 "90 00"
//RSP_APDU = 00 60 00 00 00 00 50 00 00 00 00 D1 03 86 05 00 80 80 90 00
//命令响应的前 6 个字节是没有用的，直接舍弃掉就行。


// 在 RAW 模式下，发送 ATTRIB 命令：
ExecApduCmd "FF CC 00 00 0F AE 09 01 00 01 09 1D 00 00 00 00 00 08 01 02"
CheckSw1Sw2 "90 00"
//RSP_APDU = 00 08 00 00 00 00 02 90 00


// 在 RAW 模式下，发送读身份证物理号码命令：
ExecApduCmd "FF CC 00 00 0B AE 09 01 00 01 05 00 36 00 00 08"
CheckSw1Sw2 "90 00"
//RSP_APDU = 00 50 00 00 00 00 23 16 D8 55 85 38 91 86 90 00 90 00
```

6.2 更新 GUID

以下是 MCR0110 读写器操作 Type B 卡更新 GUID 的实例：

```
Resetcard
   ShowMessage "Get PUPI"
   ExecApduCmd "FF CA 00 00 00"
   CheckSw1Sw2 "90 00"


// 使 SDIO011 读写器处于 RAW 模式
// RAW 命令格式：FF CC 00 00 Lc data
ShowMessage "使 SDIO011 读写器处于 RAW 模式"
ExecApduCmd "FF CC 00 00 02 97 01"
CheckSw1Sw2 "90 00"


// 在 RAW 模式下，发送 REQB 命令：
// 针对 Type B 卡发的 RAW 命令格式 = FF CC 00 00 Lc AE 03 01 00 01 Lc data
ExecApduCmd "FF CC 00 00 09 AE 09 01 00 01 03 05 00 01"
CheckSw1Sw2 "90 00"
//RSP_APDU = 00 60 00 00 00 00 50 00 00 00 00 D1 03 86 05 00 80 80 90 00
//命令响应的前 6 个字节是没有用的，直接舍弃掉就行。


// 在 RAW 模式下，发送 ATTRIB 命令：
ExecApduCmd "FF CC 00 00 0F AE 09 01 00 01 09 1D 00 00 00 00 00 08 01 02"
CheckSw1Sw2 "90 00"
//RSP_APDU = 00 08 00 00 00 00 02 90 00


// 在 RAW 模式下，发送更新模拟二代证物理卡号的命令：
ExecApduCmd "FF CC 00 00 13 AE 09 01 00 01 0D 00 06 00 00 08 01 02 03 04 05 06 07 08"
CheckSw1Sw2 "90 00"

// 在 RAW 模式下，发送读身份证物理号码命令：
ExecApduCmd "FF CC 00 00 0B AE 09 01 00 01 05 00 36 00 00 08"
CheckSw1Sw2 "90 00"
//RSP_APDU = 00 50 00 00 00 00 23 16 D8 55 85 38 91 86 90 00 90 00
```

7 qPBOC 应用模拟测试

卡片支持 qPBOC 的命令：Select PPSE、Select PBOC、GPO、Read Record、Get Data 等。命令是模拟的，返回的数据是固定的，不随着应用流程的执行而变化。用于测试终端的模拟流程。

命令测试流程如下：

//选择交易环境 PPSE

CMD_ADPU = 00 A4 04 00 0E 32 50 41 59 2E 53 59 53 2E 44 44 46 30 31

RSP_ADPU = 6F 2F 84 0E 32 50 41 59 2E 53 59 53 2E 44 44 46 30 31 A5 1D BF 0C 1A 61 18 4F 07 A0 00 00 03 33 01 01 50 0A 50 42 4F 43 20 44 45 42 49 54 87 01 01 90 00

//选择 PBOC 应用

CMD_ADPU = 00 A4 04 00 07 A0 00 00 03 33 01 01

RSP_ADPU = 6F 5F 84 07 A0 00 00 03 33 01 01 A5 54 50 0B 50 42 4F 43 20 43 72 65 64 69 74 87 01 01 9F 38 18 9F 66 04 9F 02 06 9F 03 06 9F 1A 02 95 05 5F 2A 02 9A 03 9C 01 9F 37 04 5F 2D 08 7A 68 65 6E 66 72 64 65 9F 11 01 01 9F 12 0F 43 41 52 44 20 49 4D 41 47 45 20 30 30 33 32 BF 0C 05 9F 4D 02 0B 0A 90 00

//初始化应用

CMD_ADPU = 80 A8 00 00 23 83 21 FE 00 00 80 00 00 00 00 00 01 00 00 00 00 00 00 01 56 00 00 00 00 00 01 56 16 11 06 00 2C C4 2C C4

RSP_ADPU = 80 0E 7C 00 08 01 02 00 10 01 03 00 18 01 03 01 90 00

//Read Record : SFI = 1,RecordNum = 1

CMD_ADPU = 00 B2 01 0C 00

RSP_ADPU = 70 2C 57 11 62 28 00 01 00 00 11 17 D3 01 22 01 01 23 45 67 89 9F 1F 16 30 31 30 32 30 33 30 34 30 35 30 36 30 37 30 38 30 39 30 41 30 42 90 00

//Read Record : SFI = 1,RecordNum = 2

CMD_ADPU = 00 B2 02 0C 00

RSP_ADPU = 70 12 5F 20 0F 46 55 4C 4C 20 46 55 4E 43 54 49 4F 4E 41 4C 90 00

//Read Record : SFI = 2,RecordNum = 1

CMD_ADPU = 00 B2 01 14 00

RSP_ADPU = 70 81 B0 90 81 80 22 91 03 A5 E3 12 0F 2D 28 62 09 11 76 AA 2B D4 E2 4D 69 E7 EE F7 B9 19 5C 91 EA 00 88 AE CF F4 7E DF A0 BE EF 7C 39 1D F3 B0 5F 71 7D CC 06 FF C8 EE FF 90 BA 14 21 2B 8A 52 AD 48 B3 32 77 B2 E2 30 D4 0B 3E 76 DC 59 77 89 26 F1 D8 73 9E 10 6C D7 41 DE 06 A7 42 3D FB A2 5E 02 F1 2E 54 3D 13 D1 B4 71 80 65 26 02 49 81 B7 D2 6B 4B F6 E5 55 86 04 CC C2 89 F5 9E 8A 80 2F 45 FB 3D 9E 67 9F 32 01 03 92 24 8B 64 3D 1E AF 2E A7 84 AC 20 53 03 C9 0E 74 5E A2 EF A5 CB F0 2C C4 7D 47 83 3B B7 B2 7E CC 69 62 38 5A 4B 8F 01 80 90 00

//Read Record : SFI = 2,RecordNum = 2

CMD_ADPU = 00 B2 02 14 00

RSP_ADPU = 70 81 83 93 81 80 81 7B 58 E9 92 D0 32 B7 F0 C0 B5 E0 AA 14 6F 53 FD D2 0D E1 B3 BF D9 BF D2 8D 0D 7B 5D 4B 69 A6 2E 14 42 84 7E C0 FC ED 37 C4 1A 65 3A C8 AE FF 68 07 04 60 7E 7D 6E DB B6 83 FD F8 AE 3C BA 63 FD 2F B9 38 45 D9 DA 06 F5 B6 CC 09 E8 07 A0 B6 9D 5C F6 FA FF DE C6 5A 3E 00 C5 60 94 7E 48 22 FD 74 D0 A4 99 44 93 C9 D5 E9 2F 83 63 4C 1E E7 7B C8 05 F8 38 A9 A7 9E 11 47 87 B6 5F 6B 74 B9 90 00

//Read Record : SFI = 2,RecordNum = 3

CMD_ADPU = 00 B2 03 14 00

RSP_ADPU = 70 81 97 9F 46 81 80 96 B3 DA 7D EA DD 6F 2D 3B 40 0F 6C B9 07 15 5A 7B 68 6E E6 DE 3D F2 FD 2A CD 61 AA 25 C3 3E 3C A0 34 3F AC 6A 0C 9A D0 6A 32 A4 8A ED 7D 95 07 BF 01 54 F9 08 7A AF BD 0C F0 F7 2A 70 08 E0 9F 89 43 9C 0C 79 85 25 11 07 5C 86 AF 85 78 7A DB 05 1C 0F 41 CD 54 34 74 A7 FF 05 9C 78 1D C4 06 62 87 CD 97 14 35 CF 0E 8A BD 26 CD 4A B2 B6 B5 3A EE BF FC 1B 73 05 9A B6 50 8B 59 09 FA 0A D2 9F 47 03 01 00 01 9F 48 0A 89 DD 91 7D 3A 28 8B 7B DD 55 90 00

//Read Record : SFI = 3,RecordNum = 1

CMD_ADPU = 00 B2 01 1C 00

RSP_ADPU = 70 1B 5A 08 62 28 00 01 00 00 11 17 5F 24 03 30 12 31 5F 25 03 95 07 01 9F 08 02 00 30 90 00

//Read Record : SFI = 3,RecordNum = 2

CMD_ADPU = 00 B2 02 1C 00

RSP_ADPU = 70 4E 8E 12 00 00 00 00 00 00 00 00 41 03 42 03 5E 03 43 03 1F 00 9F 0D 05 F0 20 04 00 00 9F 0E 05 00 50 88 00 00 9F 0F 05 F0 20 04 98 00 9F 63 10 11 22 33 44 55 66 77 88 00 00 00 00 00 00 00 00 9F 4A 02 82 02 5F 28 02 01 56 9F 07 02 FF C0 90 00

//Read Record : SFI = 3,RecordNum = 3

CMD_ADPU = 00 B2 03 1C 00

RSP_ADPU = 70 44 8C 18 9F 02 06 9F 03 06 9F 1A 02 95 05 5F 2A 02 9A 03 9F 21 03 9C 01 9F 37 04 8D 1A 8A 02 9F 02 06 9F 03 06 9F 1A 02 95 05 5F 2A 02 9A 03 9F 21 03 9C 01 9F 37 04 9F 42 02 01 56 5F 30 02 02 01 5F 34 01 01 90 00

//获取 PIN 尝试次数

CMD_ADPU = 80 CA 9F 17 00

RSP_ADPU = 9F 17 01 03 90 00

//持卡人认证--脱机 PIN 码验证

CMD_ADPU = 00 20 00 80 08 24 12 34 FF FF FF FF FF

RSP_ADPU = 90 00

//连续脱机交易上限(JR/T 0025 专有数据) [9F58]

CMD_ADPU = 80 CA 9F 58 00

RSP_ADPU = 9F 58 01 03 90 00

//连续脱机交易上限(JR/T 0025 专有数据) [9F59]

CMD_ADPU = 80 CA 9F 59 00

RSP_ADPU = 9F 59 01 07 90 00

//此次ATC寄存器值 [9F36]

CMD_ADPU = 80 CA 9F 36 00

RSP_ADPU = 9F 36 02 00 02 90 00

//上次联机ATC寄存器值 [9F13]

CMD_ADPU = 80 CA 9F 13 00

RSP_ADPU = 9F 13 02 00 00 90 00

//电子现金余额 [9F79]

CMD_ADPU = 80 CA 9F 79 00

RSP_ADPU = 9F 79 06 00 00 00 01 00 00 90 00

8 附录

7.1 非接 Type A 卡和 Type B 卡重要参数表

参数	Type A			Type B		
	所在域	所在字节	所在位	所在域	所在字节	所在位
FSCI	ATS	T0	低四位	ATQB	ATQB[10]	高四位
DR/DS	ATS	TA1	DR：b2…b0 DS：b6…b4	ATQB	ATQB[9]	DR：b2…b0 DS：b6…b4
SFGI	ATS	TB1	低四位	ATQB	ATQB[12]可选	高四位
FWI	ATS	TB1	高四位	ATQB	ATQB[11]	高四位
NAD	ATS	TC1	b0	ATQB	ATQB[11]	b1
CID	ATS	TC1	b1	ATQB	ATQB[11]	b0

7.2 命令响应码（状态字节）表

SW1 (hex)	SW2 (hex)	警告和执行状态	
90	00	成功（命令正确完成）	
61	XX	SW2 表示还有可用的返回字节的个数	
62	00	Warning value in EEPROM unchanged	no information given(无进一步信息)
	81		part of return data may be corrupted(回送的数据可能有错)
	82		end of file or record reached before reading Le bytes
	83		selected file is invalidated(选择文件无效)
	84		file control information not in required format
63	00	Warning value in EEPROM changed	no information given（认证失败）
	81		the last write has filled up the file
	82		memory modification successful after retry
	CX		the last significant nibble of SW2 is a counter valued from 0 to 15. verification or authentication is unsuccessful, the counter indicates number of retries.
65	00	Execution error	no information given
	81		memory failure(写 EEPROM 操作失败)

表 7－1 警告和执行状态代码

SW1 (hex)	SW2 (hex)	检查错误状态	
67	00	wrong length Lc or Le	
68	00	functions in CLA not supported	no information given
	81		logical channel not supported
	82		secure messaging not supported
69	00	command not allowed	no information given
	81		command incompatible with file structure
	82		security status not satisfied（不满足安全状态）
	83		verification or authentication methods blocked(密钥锁定（算法锁定）)
	84		referenced data invalidated(未取随机数)
	85		conditions of use not satisfied(使用条件不满足)
	86		no current EF
	87		expected SM data objects missing(MAC 丢失)
	88		SM data objects incorrect(MAC 不正确)
6A	00	wrong parameter(s) P1,	no information given
	80		incorrect parameters in the data field(数据域参数不正确)（文件已经

		P2	存在）
	81		function not supported
	82		file not found(未找到文件)
	83		record not found(未找到记录)
	84		not enough memory space in the file(文件空间不足)
	85		Lc inconsistent with TLV structure
	86		incorrect parameters P1, P2(P1 或 P2 不正确)
	87		Lc inconsistent with P1, P2
	88		key not found
6B	00	wrong parameter(s) - the offset is outside the EF(参数错(偏移地址超出了 EF))	
6C	XX	wrong length Le, SW2 indicates the proper length(Le 不正确，实际长度应为 xx)	
6D	00	wrong INS(INS 不正确)	
6E	00	class not supported(无效的 CLA)	
6F	00	no precise diagnosis	no information given
	80		RAM failure - proprietary
	81		ROM checksum error - proprietary
	82		File already existed - proprietary
	83		File structure not supported - proprietary
	84		Not enough space - proprietary
93	03		应用已被永久锁定，卡片锁定

表 7－2 检查错误状态代码