Compatible PC/SC Series IC Card Reader

General Technical Manual

(Revision 2.38)

Quick-Ohm Küpper Co. GmbH

February 20, 2021

Please read this manual carefully before using. If any problem, please feel free to contact us, we will offer satisfied answer ASAP.

Contents

Document update records

Revision	Date	Update information
Revision 2.30	July. 6, 2018	Support MR880
Revision 2.32	Aug.1,2018	Add chapter 4.2.4 and 4.3.51 Fix chapter4.3.41 and 4.3.42
Revision 2.38	Jan.28,2021	Fix the extension command list Add SR176/SRIX label instruction introduction Fix Sam card instruction introduction Document formatting

1 Introduction

These series RFID Readers are the USB PC/SC interfaces. If the first time to connect with PC in Windows System, you need install the PC/SC CCID driver. The PC/SC interface adopts Win System itself building in driver and API, so the development work is easy.

Here we mentioned compatible USB PC/SC has a little bit of difference with the standard USB PC/SC interface. This is in order to compatible with more types of IC cards. Standard PC/SC just supports ISO14443A and IAO14443B.There is an Auto-detecting card sequence in the standard USB PC/SC Reader. Other type cards cannot be operated like the aboveAuto-detecting sequence, so we develop a way of operation that by sending detection commands.

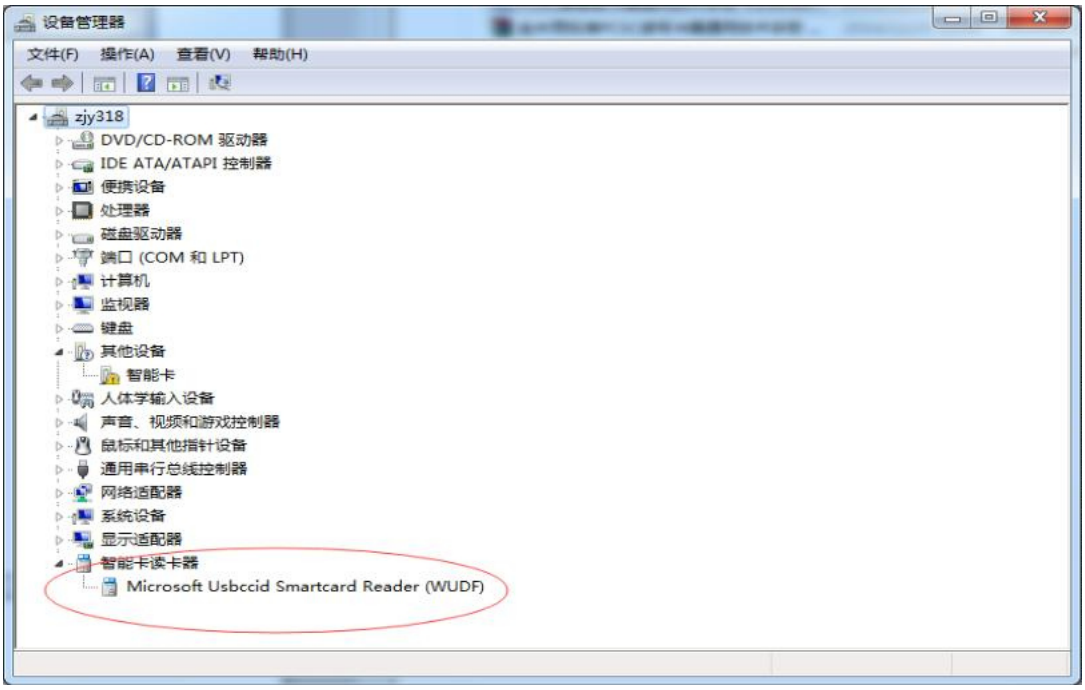
For the programmer'sdevelopment application convenient, we can offer VC、BC、VB、DELPHI programs(SDK). The programmercan enter into work quickly by using our offering SDK. If any further questions, please feel free to contact us or send mail to kontakt@quio-rfid.de, we are always at your service.

2 Device Installation

2.1 Driver Installation

Connect the PC/SC Readerwith your PC via USB interface, and then install the CCID Driver.

After that just click the “Device Manager” to find out the “PICC Interface”. The Microsoft USB CCID Driver is used like the following picture:



2.2 ATRGeneration

According to ISO 14443 Part 3 PICCs, if the reader detects a PICC, an ATR will be sent to the PCSC driver for identifying the PICC.In order to compatible with more types of contactless IC cards, the MR800 adopts getting the regularATR (without card information), so the return ATR format is like the following:

ByteNo.	Value(Hex)	Designation	Description
0	3B	Initial Header	
1	8N	T0	Higher nibble 8 means no TA1, TB1, and 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, and 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.
4 to 3+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
	00000000	RFU	RFU # 00 00 00 00
4+N	UU	TCK	XOR of all the bytes T0 to Tk

Note: For MR800 Reader, the ATR format:

ATR = {0x3B 8F 80 01 80 4F 0C A0 00 00 03 06 00 00 00 00 00 00 00 68}

3 APDU Description

There are two types of APDUformats for PC/SC CompatibleReaders: Standard APDU format (the Class in APDU is not 0xFF) and Non-Standard APDU format (the Class in APDU is 0xFF).In order to be compatible with the PC/SC standard, for Contactless Smart Cards and Contact SAM cards, we can directly send standard APDU commands to them except the Get_Data command that is to get the Card Reset information. Meantime, our these Series Readers support Contactless SmartCards and Contact SAM cards,sobefore using, you can choose which one you want to operate by switching the current operation. TheAPDU is {0xFF 00 FA 00 01 + CurSmartCard}.

Whatever for Contactless Smart Cards, Contact SAM cards or ContactlessStorage Cards, the first step is sending Get_Data APDU to get Card information.

3.1 Contactless Smart Cards

Contactless Smart Cardsadopt the standard APDU command format. For contactless ICCs, the IFD subsystem must construct an ATR from the fixed elements that identify the cards. If during this operation, you want to operate other Contact SAM card, you need switch the currentoperation to the designatedSAM Slot {APDU: 0xFF 00 FA 00 01 + CurSmartCard}to get the relevant card data.

3.2 SAM Cards

There are more than one SAM slots in the PC/SC CompatibleReaders. Before sending the standard APDU command, we need through Get_Data command to obtain SAM Card reset information. If during this operation, you want to operate other contactless ICCs, you need switch the currentoperation to the designated SmartCard.

Note: The current reading card type need be certificated by the SAM data during the operation.

3.3 Contactless StorageCards

Our PC/SC CompatibleReaders support storage cards like MIFARE one, MIFARE Ultra Light and so on. In order to be compatible with PC/SC standard, we defined the Non - Standard APDU. Before sending the Non - Standard APDU command, we need through Get_Data command to obtain Card SNR information.

4 Non - Standard APDU Commands

4.1 Common Error Code Definition

Get_Data APDUcommand is suitable for Contactless Smart Cards, SAM Cards and Contactless Storage Cards. But other Non - Standard APDU Commands are mainly suitable for Contactless Storage Cards. Standard APDU Commands are mainly suitable for Contactless Smart Cards and SAM Cards.

Common Error Codes:

Result	SW1 SW2	Meaning
Success	0x90 00	The operation is completed successfully.
Error	0x63 00	The operation is failed.
Error	0x6A 81	Function not supported
Error	0x6B 00	Wrong parameter P1 - P2

4.2 PC/SC Part 3

4.2.1 Get Data Command

This command will retrieve the SNR or RESET information of the present card.Before operation the card, this APDU command must be executed first. Because this APDU command containtheswitching Card Type.

APDU Format:

Command	CLA	INS	P1	P2	Le
Get Data	0xFF	0xCA	CardType	SubCardType	0x00

CardType and SubCardType Definition:

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/UltraLight/ MIFARE Plus Level1 (P1 = 00/02/03) Format:

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

MIFARE Plus Level0/2/3/1 for switch and ISO14443 - 4 Type ASmartCard (P1 = 03/01) Format:

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

ISO14443 - 4TypeBSmartCard/AT88F020 (P1=20/23) Format:

Response	Data Out		
Result	ATQB(12Bytes)	SW1	SW2

SR176/SRIX4K (SRI512) (P1=21/22) Format:

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

ISO15693 Tag (P1=40) Format:

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

ISO7816 SAM (P1=60) Format:

Response	Data Out		
Result	Reset Info(nByte)	SW1	SW2

Example:

TYPE A Card Request

Send: 0xFF CA 00 00 00

Receive: 0x04 72 AE A6 9E 04 00 08 90 00

ISO14443 Type A Smartcard Request

Send: 0xFF CA 01 00 00

Receive: 0x04 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 Request

Send: 0xFF CA 20 00 00

Receive: 0x50 C0 1281 89 54 46 22 08 00 80 A1 90 00

4.2.2 LoadAuthentication Keys

This command will just load (write) the keys in the IFD's designated memory. The key will be of two different types; the reader key and the card key. This command can be used for all kinds of contactless cards.

Reader Key: This key will be used to protect the transmission of secured data e.g. card key from the application to the reader. Example: The application may encrypt the data with one of the reader keys: It has to tell the reader that it has done so, and the number of the key used.

Card Key: This is the card specific key (e.g. for MIFARE it is MIFARE key). This key can be volatile or non-volatile.

The coding of the command provides the following mechanisms:

- Load keys in either container: Reader key container to be used for transmission protection and card key container for card authentication.
- Transmission of the loaded key in plain or encrypted using a key out of the reader key container: The key to be used is indicated in P1 by its number. P2 is indicating the address within the container, where the key shall be stored.

- The containers can be located in volatile or non-volatile memory.

Load Authentication KeysAPDUFormat:

Command	CLA	INS	P1	P2	Lc	Data
Load Keys	0xFF	0x82	KeyStructure	KeyIndex	1-16	Key (LSB)

Key Structure:

b7	b6	b5	b4	b3	b2	b1	b0	Description
X								0: Card Key; 1 Reader Key
	X							0: Plain Transmission, 1: Secured Transmission
		X						0: Keys are loaded into the IFD volatile memory 1: Keys are loaded into the IFD non-volatile memory.
			X	X	X	X	X	RFU

NOTE: The Non-volatile Key, which is stored in the Flash of the Reader, has storage time limitation. Users need pay more attention to it.

Key Storage Format:

Key Index	Card Key(Byte)	Reader Key(Byte)
0	16	16
1	16	-
.....	16	-
31	16	-

Note: {Card Key Index **0~31;** ReaderKey Index only**0}**

Key Index:1byte

Key Length: 1byte

When loading the Reader Key, the length of the Key must be 16bytes, or the Reader will return fail.

When loading the Card Key by way of plaintext, the Reader no any restriction for the Key length.

When loading the Card Key by way of ciphertext, the Key length must be 8bytes or 16bytes.

Key:N byte

Loadthe "Reader Key"or "Card Key" valueinto the Reader Writer.

Load Authentication KeysResponseFormat:

Response	Data Out	
Result	SW1	SW2

Example:

Load Keyinplaininto the Reader's Key storage location

Send: 0xFF 82 80 00 10 33 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF

Receive: 0x90 00

4.2.3 Authentication

The application provides the number of the key used for the MIFARE 1K/4K card authentication. The specific key must be already in the reader.Two typeauthentication keys: TYPE_A and TYPE_B.

Load Authentication Keys APDU Format:

Command	CLA	INS	P1	P2	P3	Data
Authentication	0xFF	0x88	High Address	Low Address	KeyType	KeyConfig+Key

P1/P2:

For MIFARE S50/70, this is the Block Address.

For AT88F020, The address is invalid. (P1=0, P2=0).

For MIFARE Plus Level 1/2/3, that is the AES Key Storage Block Address. (Note：Key Storage Block and Data Block are the corresponding relationship. Please refer to MIFARE Plus Datasheet.)

KeyType: {Only in MIFARE S50/S70, the byte is valid (Key_A - 0x60, Key_B - 0x61).}

KeyConfig:

b7	b6 - b0	Meaning
0	xxxxxxx	"xxxxxxx" means usingthe current input Key length.The Card is authenticated by the current Key.
1	xxxxxxx	"xxxxxxx" means thestored Key index in the Reader. The Card is authenticated by the stored Key in the Reader.

KEY:

If for KeyConfig, the Bit7 = 0, that means it's the key. The Key length is decided by the Card Type.

If KeyConfig, the Bit7 = 1, the Key does not exist.

Load Authentication Keys Response Format:

Response	Data Out	
Result	SW1	SW2

Example:

MIFARE1kCard Request, and to read the first Block Data

Send: 0xFF CA 00 00 00

Receive: 0x04 72 AE A6 9E 04 00 08 90 00

Send: 0xFF 88 00 01 60 06 FF FFFFFFFFFF

Receive: 0x90 00

Send: 0xFF B0 00 01 10

Receive: 0x00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF 90 00

MIFAREPlus Level3 Card Request, and to read the Data Block0

Send: 0xFF CA 03 03 00

Receive: 0x07 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: 0xFF 88 40 00 00 10 FF FFFFFFFFFFFFFFFFFFFFFFFFFFFF {The Key Address for Block1 is 0x4000 or 0x4001.}

Receive: 0x90 00

Send: 0xFF B0 00 01 10

Receive: 0x11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 90 00

AT88F020 CardRequest, and to read the Data Block9

Send: 0xFF CA 23 00 00

Receive: 0x50 00 04 E8 51 00 00 00 00 00 00 41 90 00

Send: 0xFF 88 00 00 00 08 00 00 00 00 00 00 00 00

Receive: 0x90 00

Send: 0xFF B0 00 09 08

Receive: 0x00 00 00 00 00 00 00 00 90 00

4.2.4 General Authenticate Command

The application provides the number of the key used for the MIFARE 1K/4K card authentication. The specific key must be already in the reader. Two type authentication keys: TYPE_A and TYPE_B.

APDU Format:(New PC/SC standards, recommended)

Command	CLA	INS	P1	P2	Lc	Data
Authentication	0xFF	0x86	0x00	0x00	0x05	Authentication Data

Authentication Data:

BYTE 1	BYTE2	BYTE3	BYTE4	BYTE5
versions(01h)	HighAddress	LowAddress	KeyType	Key number

HighAddress/ LowAddress:

For MIFARE S50/70, this is the Block Address.

For AT88F020, The address is invalid. (P1=0, P2=0).

For MIFARE Plus Level 1/2/3, that is the AES Key Storage Block Address. (Note：Key Storage Block and Data Block are the corresponding relationship. Please refer to MIFARE Plus Datasheet.)

KeyType:

Only in MIFARE S50/S70, the byte is valid (Key_A - 0x60, Key_B - 0x61).

Key number:

00h~1fh

Response Format:

Response	Data Out	
Result	SW1	SW2

Example:

MIFARE1kCard Request, and to read the first Block Data

Send: 0xFF 82 20 00 06 FF FFFFFFFFFF

Receive: 0x90 00

Send: 0xFF CA 00 00 00

Receive: 0x04 7E CE 4A A5 04 00 08 90 00

Send: 0xFF 86 00 00 05 01 00 01 60 00

Receiv: 0x90 00

Send: 0xFF B0 00 01 10

Receive: 0x00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF 90 00

MIFAREPlus Level3 Card Request, and to read the Data Block0

Send: 0xFF 82 20 01 10 FF FFFFFFFFFFFFFFFFFFFFFFFFFFFF

Receive: 0x90 00

Send: 0xFF CA 03 03 00

Receive: 0x07 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: 0xFF 86 00 00 05 01 40 00 00 01

{The Key Address for Block1 is 0x4000 or0x4001.}

Receive: 0x90 00

Send: 0xFF B0 00 01 10

Receive: 0x11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 90 00

AT88F020 Card Request, and to read the Data Block9

Send: 0xFF 82 20 02 08 00 00 00 00 00 00 00 00

Receive: 0x90 00

Send: 0xFF CA 23 00 00

Receive: 0x50 00 06 29 BB 00 00 00 00 00 00 41 90 00

Send: 0xFF 86 00 00 05 01 00 00 00 02

Receive: 0x90 00

Send: 0xFF B0 00 09 08

Receive: 0x00 01 02 03 04 05 06 07 90 00

4.2.5 ReadBinaryBlocks

This command is used for retrieving “data blocks” from the PICC. The data block/trailer block must be authenticated first.

Read Binary APDU Format

Command	CLA	INS	P1	P2	Le
Read Blocks	0xFF	0xB0	HighAddress	LowAddress	DataLen

P1/P2: the Block Address to be read

Data Len: the Data Length to be read {All of the Data to be read, LSB first}

- MIFARE 1K/4K 16 bytes
- MIFARE Plus 16 bytes {In Level3, Support Multi-Block Read}
- MIFARE Ultra light 4 bytes/Block, 4Blocks each time, total 16bytes
- SR176 2 bytes
- SR512 2 bytes
- SRIX4K 2 bytes
- AT88RF020 8 bytes
- ISO15693 Tag 4 bytes {Support Multi-Block Read}

This APDU command support Muti-Block read, but the Card itself also must support Muti-Block read. If for ISO15693 Tag, to read from the two continuous Blocks, the DataLen is 4*2=8. Note: This APDU for ISO15693 Tag read operation is just for the last Tag that be detected within antenna field.

If you want to operate the selected or designatedUID of the Tag, please reference the Chapter4.3.

Read Binary Block Response Format

Response	Data Out		
Result	Data	SW1	SW2

Example:

SR176 card request, to read the Block 10

Send: 0xFF CA 21 00 00

Receive: 0x20 42 2F 69 18 08 92 D0 02 90 00

Send: 0xFF B0 00 0A 02

Receive: 0x00 00 90 00

MIFARE Ultra light card request, to read the Block 10

Send: 0xFF CA 02 00 00

Receive: 0x07 04 24 A2 E1 BF 02 80 44 00 00 90 00

Send: 0xFF B0 00 0A 10

Receive: 0x11 22 33 44 00 00 00 00 00 00 00 00 00 00 00 00 90 00

Read ISO15693 Tag Block 10 and Block 11

Send: 0xFF CA 40 00 00

Receive: 0x00 3D 3D 08 17 00 01 04 E0 90 00

Send: 0xFF B0 00 0A 08

Receive: 0x00 00 00 00 00 00 00 00 90 00

4.2.6 UpdateBinaryBlocks

This command is used for writing “data blocks” into the PICC. The data block/trailer block must be authenticated.

Update Binary APDU Format (4 or 16 + 5 Bytes)

Command	CLA	INS	P1	P2	Lc	Data
Update Blocks	0xFF	0xD6	HighAddress	LowAddress	DataLen	Data

P1/P2: the Block Address to be written

Data Len: the Data Length to be written {All of the Data to be written, LSB first}

- MIFARE 1K/4K 16 bytes
- MIFARE Plus 16 bytes {In Level3, Support Multi-Block Write}
- MIFARE Ultra light 4 bytes
- SR176 2 bytes
- SR512 4 bytes
- SRIX4K 2 bytes
- AT88RF020 8 bytes
- ISO15693 Tag 4 bytes

This APDU command support Muti-Block write, but the Card itself also must support Muti-Block write. If for ISO15693 Tag, to write into the two continuous Blocks, the DataLen is 4*2=8. Note: This APDU for ISO15693 Tag read operation is just for the last Tag that be detected within antenna field.

If you want to operate the selected or designated UID of the Tag, please reference the Chapter4.3.

Update Binary Block Response Format

Response	Data Out	
Result	SW1	SW2

Example:

MIFARE 1k cardrequest, Write and Readthe first Data Block

Send: 0xFF CA 00 00 00

Receive: 0x04 72 AE A6 9E 04 00 08 90 00

Send: 0xFF 88 00 01 60 06 FF FFFFFFFFFF

Receive: 0x90 00

Send: 0xFF D6 00 01 10 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 00

Receive: 0x90 00

Send: 0xFF B0 00 01 10

Receive: 0x01 10 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 00

MIFARE Plus Level 1 cardrequest, Read and Write the fourthData Block

Send: 0xFF CA 03 01 00

Receive: 0x04 72 AE A6 9E 04 00 08 90 00

Send: 0xFF 88 00 04 60 06 FF FFFFFFFFFF

Receive: 0x90 00

Send: 0xFF D6 00 04 10 00 00 00 04 05 06 07 08 09 0A 0B 0C 0D 0E 01 00

Receive: 0x90 00

Send: 0xFF B0 00 04 10

Receive: 0xFF D6 00 04 10 00 00 00 04 05 06 07 08 09 0A 0B 0C 0D 0E 01 00

Read and WriteMIFARE Ultra light card, the tenthData Block

Send: 0xFF CA 02 00 00

Receive: 0x07 04 24 A2 E1 BF 02 80 44 00 00 90 00

Send: 0xFF D6 00 0A 04 00 01 02 03

Receive: 0x90 00

Send: 0xFF B0 00 0A 10

Receive: 0x00 01 02 03 00 00 00 00 00 00 00 00 00 00 00 00 90 00

Read and WriteMIFARE Plus Level 3 card, the firstData Block

Send: 0xFF CA 03 03 00

Receive: 0x07 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: 0xFF 88 40 00 00 10 FF FFFFFFFFFFFFFFFFFFFFFFFFFFFF

{the Key Address for Block 1 is 0x4000 or 0x4001}

Receive: 0x90 00

Send: 0xFF D6 00 01 10 00 00 00 04 05 06 07 08 09 0A 0B 0C 0D 0E 01 00

Receive: 0x90 00

Send: 0xFF B0 00 01 10

Receive: 0x00 00 00 04 05 06 07 08 09 0A 0B 0C 0D 0E 01 00 90 00

Write and Read SR176 card, the tenthData Block

Send: 0xFF CA 21 00 00

Receive: 0x20 42 2F 69 18 08 92 D0 02 90 00

Send: 0xFF D6 00 0A 02 00 01

Receive: 0x90 00

Send: 0xFF B0 00 0A 02

Receive: 0x00 01 90 00

AT88F020cardrequest, Read the Data Block 9

Send: 0xFF CA 23 00 00

Receive: 0x50 00 04 E8 51 00 00 00 00 00 00 41 90 00

Send: 0xFF 88 00 00 00 08 00 00 00 00 00 00 00 00

Receive: 0x90 00

Send: 0xFF D6 00 09 08 00 01 02 03 04 05 06 07

Receive: 0x90 00

Send: 0xFF B0 00 09 08

Receive: 0x00 01 02 03 04 05 06 07 90 00

Read ISO15693 Tagthe Block 10 and Block 11

Send: 0xFF CA 40 00 00

Receive: 0x00 3D 3D 08 17 00 01 04 E0 90 00

Send: 0xFF D6 00 0A 04 00 01 02 03

Receive: 0x90 00

Send: 0xFF B0 00 0A 04

Receive: 0x00 01 02 03 90 00

4.2.7 ValueBlockOperation

This command increments the value of a data object if the card supports this functionality.

Value Block Operation APDU Format

Command	CLA	INS	P1	P2	Lc	Data	
Value Block	0xFF	0xD7	HighAddress	LowAddress	0x05	VB_OP	VB_Value

P1/P2: the Block Address

VB_OP (1 Byte):

0x00 = Store the VB_Value into the block. The block will then be converted to a value block.

0x01 = Increment the value of the value block by the VB_Value.

0x02 = Decrement the value of the value block by the VB_Value.

VB_Value (4 Bytes):

The value used for value manipulation.{LSB first}

Value Block Operation Response Format

Response	Data Out	
Result	SW1	SW2

4.2.8 ReadValueBlock

This command is used for retrieving the value from the value block. This command is only valid for value block.

Read Value Block APDU Format (5 Bytes)

Command	CLA	INS	P1	P2	Lc
ReadValueBlock	0xFF	0xB1	High Address	Low Address	0x04

P1/P2: The value block to be accessed.

Read Value Block Response Format

Response	Data Out		
Result	Value(4Bytes)	SW1	SW2

Example:

MIFARE Purse Block Initialization, Purse Increment, Purse Decrement, Purse Read

Send: 0xFF CA 00 00 00

Receive: 0x04 72 AE A6 9E 04 00 08 90 00

Send: 0xFF 88 00 01 60 06 FF FFFFFFFFFF

Receive: 0x90 00

Send: 0xFF D7 00 01 05 00 00 00 00 01

Receive: 0x90 00

Send: 0xFF B1 00 01 04

Receive: 0x00 00 00 01 90 00

Send: 0xFF D7 00 01 05 01 00 00 00 02

Receive: 0x90 00

Send: 0xFF B1 00 01 04

Receive: 0x00 00 00 03 90 00

Send: 0xFF D7 00 01 05 02 00 00 00 01

Receive: 0x90 00

Send: 0xFF B1 00 01 04

Receive: 0x00 00 00 02 90 00

MIFARE Plus Level1 Purse Block Initialization, Purse Increment, Purse Decrement, Purse Read

Send: 0xFF CA 03 01 00

Receive: 0x07 04 8C AF 04 05 06 07 42 00 18 90 00

Send: 0xFF 88 00 04 60 06 FF FFFFFFFFFF

Receive: 0x90 00

Send: 0xFF D7 00 04 05 00 00 00 00 01

Receive: 0x90 00

Send: 0xFF B1 00 04 04

Receive: 0x00 00 00 01 90 00

Send: 0xFF D7 00 04 05 01 00 00 00 02

Receive: 0x90 00

Send: 0xFF B1 00 04 04

Receive: 0x00 00 00 03 90 00

Send: 0xFF D7 00 04 05 02 00 00 00 01

Receive: 0x90 00

Send: 0xFF B1 00 04 04

Receive: 0x00 00 00 02 90 00

MIFARE Plus Level3 Purse Block Initialization, Purse Increment, Purse Decrement, Purse Read{Block =0x01}

Send: 0xFF CA 03 03 00

Receive: 0x07 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: 0xFF 88 40 00 00 10 FF FFFFFFFFFFFFFFFFFFFFFFFFFFFF

Receive: 0x90 00

Send: 0xFF D7 00 01 05 00 00 00 00 01

Receive: 0x90 00

Send: 0xFF B1 00 01 04

Receive: 0x00 00 00 01 90 00

Send: 0xFF D7 00 01 05 01 00 00 00 02

Receive: 0x90 00

Send: 0xFF B1 00 01 04

Receive: 0x00 00 00 03 90 00

Send: 0xFF D7 00 01 05 02 00 00 00 01

Receive: 0x90 00

Send: 0xFF B1 00 01 04

Receive: 0x00 00 00 02 90 00

4.2.9 RestoreValueBlock

This command is used to copy a value from a value block to another value block.

Restore Value Block APDU Format

Command	CLA	INS	P1	P2	Lc	Data
Restore Value Block	0xFF	0xD7	SourceHigh Address	Source Low Address	0x03	0x03+TargetAddress

P1/P2: Source Block Address

TargetAddress: Target Block Number (2 Bytes): The value block to be restored {MSB first}

Restore Value Block Response Format

Response	Data Out	
Result	SW1	SW2

Example:

MIFARE 1K/4K Purse Backup

Send: 0xFF CA 00 00 00

Receive: 0x04 16 49 D9 F1 04 00 08 90 00

Send: 0xFF 88 00 01 60 06 FF FFFFFFFFFF

Receive: 0x90 00

Send: 0xFF D7 00 01 05 00 00 00 00 01

Receive: 0x90 00

Send: 0xFF D7 00 01 03 03 00 02

Receive: 0x90 00

Send: 0xFF B1 00 02 04

Receive: 0x00 00 00 01 90 00

4.3 Non-Standard APDU(Custom)

Non-Standard APDU (Custom) is the extension of Non-Standard APDU function in PC/SC Part3.The Commands are mainly the extension for FF (INS = 00) Command. Our Non-Standard APDU (Custom) Commands extension functions include "SwitchCurrent Operation Smart Card ", "LCD Display", and "Beep/LED Control". For details, please refer to the following:

Extension Commands List:

Class	Ins	P1		P2	Le/Lc	Function
FF	00	ISO14443 Type A (0x00~0x1F)	MIFAREClass (0x00)	00		Set TypeA request mode
				01		HaltA card
			MIFAREPlus (0x03)	00		Switch Level0 to Level1/3
		ISO14443 TypeB (0x20~0x3F)	ISO14443SMARTB (0x20)	00		Set TypeB detecting card mode
				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)	Contact SAM (0x60)	10		Set SAMn PPSBaud
				11		Set SAMn RSTBaud
				12		Read SAMn RSTBaud
				14		AutoSet SAMn PPSBaud
		SYSTEM (0xE0~0xFF)	Smart card Switching (0xFA)	00		Smart card Switching (contactless and contact)
			RTC operation (0xFB) (for MR800/MR880)	00		Time Initialization
				01		Read time
				02		Set LCD show time

				03		Set LCD show date
			LCD&&LED operation (0xFC) (forMR800/MR880)	00		Set the display font type
				01		Read the display font type
				02		Showing specified number of characters
				03		Show pictures (download data)
				04		Erase LCD
				05		Set the boot image
				06		Set the standby interface
				07		LCD backlight control
				08		Flash image displayed in a specified format
			LCD operation (0xFC) (for MR880)	09		Any point display character string
				0A		Set font pixel
			Flash operation (load font type 0xFD)	00		Read Flash
				01		Write Flash
			RFU (0xFE)	-		System retains instruction
			System instruction (0xFF)	00		Get the serial number
				01		Get the version number (hardware and software)
				02		Set the LED status
				03		Set the buzzer status
				04		Set the antenna status
				05		Set Card Encryption Standard
				06		Restore the factory default
				07		Re-start the Reader

4.3.1 ISO14443 TYPEARequest

Set ISO14443 TYPE_A Card Requestmode. The default value is REQA (0x26) when power on.

APDU Format:

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

RequestMode:

REQA (0x26)

WUPA (0x52)

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.2 ISO14443A Card Halt

Set the current operating ISO14443A card into halt status.

APDU Format:

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

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.3 MIFARE PlusSwitch Level0 to Level1/3

After the Level 0 initialization finished, to switch Level0 to Level1 or Level3. Target Level depends on the card.The MIFARE Plus Card factory default is in Level0. If switched to other Level, some parameters need be written into the Card by sending WriteBinary APDU commands in advance. (Note: like Address Configuration Value 0x9000/0x9001/0x9002/0x9003)

APDU Format:

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

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.4 ISO14443 TYPE B Request

Set ISO14443 TYPE_BCard Requestmode. The default value is REQB (0x00) when power on.

APDU Format:

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

RequestMode:

REQB (0x00)

WUPB (0x01)

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.5 Halt Type_B

Set the current operating ISO14443B card into halt status.

APDU Format:

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

PUPI (8 bytes):for Type_B

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.6 AT88F020 Count

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
Count	FF	00	23	01	06	Signature

Signature (6 bytes)

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.7 AT88F020Deselect

APDU Format:

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

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.8 AT88F020Lock

APDU Format:

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

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.9 ST176 Block Lock

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
SR176 Lock	FF	00	30	00	01	LockData

LockData：

B_7	1: Write-Protect Blocks 14 and 15
	0: Allow write access
B_6	1: Write-Protect Blocks 12 and 13
	0: Allow write access
B_5	1: Write-Protect Blocks 10 and 11
	0: Allow write access
B_4	1: Write-Protect Blocks 8 and 9
	0: Allow write access
B_3	1: Write-Protect Blocks 6 and 7
	0: Allow write access
B_2	1: Write-Protect Blocks 4 and 5
	0: Allow write access
B_1	1: Write-Protect Blocks 2 and 3
	0: Allow write access
B_0	1: Write-Protect Blocks 0 and 1
	0: Allow write access

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Send: 0xFF CA 21 00 00

Receive: 0xEE 93 39 E9 0F 08 92 D0 02 90 00

Send: 0xFF 00 30 00 01 EF

Receive: 0x90 00

//read 0x0F block（1Byte Lock_REG + 4Bits Reserved + 4Bits CID）

Send: 0xFF B0 00 0F 02

Receive: 0xEF EE 90 00

Send: 0xFF D6 00 07 02 77 77

Receive: 0x63 00

Send: 0xFF B0 00 08 02

Receive: 0xAA AA 90 00

Send: 0xFF D6 00 08 02 88 88

Receive: 0x90 00

Send: 0xFF B0 00 08 02

Receive: 0x88 88 90 00

4.3.10SRIX Serial Cards Read UID

Read the 8-byte UID of the SRIX series card

APDU Format:

Command	Class	INS	P1	P2	Le
Read UID	FF	00	30	10	08

Answer:

Response	Data Out		
Result	UID（8Bytes）	SW1	SW2

4.3.11SRIX Serial Cards Authentication

SRIX series card certification, prevent replication.

APDU Format:

Command	Class	INS	P1	P2	Lc	DATA
Authentication	FF	00	30	11	06	Random

Answer:

Response	Data Out		
Result	Result(3Bytes)	SW1	SW2

4.3.12SRIX Serial Cards Return to Inventory

APDU Format:

Command	Class	INS	P1	P2	Lc
ReturntoInventory	FF	00	30	12	00

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.13SR Serial Cards Completion

The selected SRIX series card switches to the deactivated state.

APDU Format:

Command	Class	INS	P1	P2	Lc
Completion	FF	00	30	13	00

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Send: 0xFF CA 22 00 00

Receive: 0xE2 D0 02 1A 25 2B 03 01 48 90 00

Send: 0xFF 00 30 10 08

Receive: 0xD0 02 1A 25 2B 03 01 48 90 00

Send: 0xFF CA 22 00 00

Receive: 0x63 00

Send: 0xFF 00 30 12 00

Receive: 0x90 00

Send: 0xFF CA 22 00 00

Receive: 0xA3 D0 02 1A 25 2B 03 01 48 90 00

Send: 0xFF 00 30 13 00

Receive: 0x90 00

Send: 0xFF CA 22 00 00

Receive: 0x63 00

4.3.14SRIX Serial Cards 16 Slots Initiate Card

SRIX series card 16 slots initialization.

APDU Format:

Command	Class	INS	P1	P2	Le
Pcall16	FF	00	30	14	0x20

Answer:

Response	Data Out			
Result	Status	CardID	SW1	SW2

Status: 16Bytes, slot0~15result.

0x00:success; 0xE8:collision; 0xFF:NoCard.

Card ID: 16Bytes, Card IDs for 16 slots. The ID is only valid if the execution result of the current slot is successful.

4.3.15SR Serial Cards Select

Select the card by ID.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
Select	FF	00	30	15	01	CardID

CardID: 1Byte. The card ID that needs to be selected.

Answer:

Response	Data Out		
Result	CardID	SW1	SW2

CardID:selected id.

Example:

One SRIX512 and One SRIX4K were used for testing

Send: 0xFF 00 30 14 20

Receive: 0xFF FF 00 FF 00 FF FF FF FF FF FF FF FF FF FF FF 00 00 92 00 04 00 00 00 00 00 00 00 00 00 00 00 00 90 00

Send: 0xFF 00 30 15 01 92

Receive: 0x92 90 00

Read 0x7F Block(SRIX512 was selected, response failed)

Send: 0xFF B0 00 7F 04

Receive: 0x63 00

Send: 0xFF 00 30 15 01 04

Receive: 0x04 90 00

Send: 0xFF B0 00 7F 04

Receive: 0xFF FF FF FF 90 00

4.3.16ISO15693 Inventory

There are two ways to obtain the Tag UID. It contains "GetData command" and "Single&Multi TagDetect by sendingAPDU command". As toTag numbers depend on the antenna RF capability. Note that: This command has the same function with "GetData APDU command". Using this APDU command, the working mode will be automatically switched to ISO15693.

APDU Format:

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

Type:

Single Tag Detect (0x00)

Multi Tag Detect (0x01)

Flag: Refer to ISO15693 Standard(eg: Flag=0x26)

AFI: the designated Tag AFI

Answer:

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

Example:

ISO15693 Single TagDetect

Send: 0xFF 00 40 00 03 00 26 00

Receive: 0x00 3D 3D 08 17 00 01 04 E0 90 00

Send: 0xFF 00 40 01 09 22 3D 3D 08 17 00 01 04 E0 (Stay Quiet)

Receive: 0x90 00

Send: 0xFF 00 40 00 03 00 26 00

Receive: 0x63 00

Send: 0xFF 00 40 03 09 22 3D 3D 08 17 00 01 04 E0

Receive: 0x00 3D 3D 08 17 00 01 04 E0 90 00

Send: 0xFF 00 40 00 03 00 26 00

Receive: 0x00 3D 3D 08 17 00 01 04 E0 90 00

4.3.17ISO15693 Stay Quiet

APDU Format:

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

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22)

UID: Tag UID(8byte, required)

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.18ISO15693 Select Tag

APDU Format:

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

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22)

UID: Tag UID(8byte, required)

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

ISO15693 Read/Write Blocks

Send: 0xFF 00 40 00 03 00 26 00

Receive: 0x00 3D 3D 08 17 00 01 04 E0 90 00

Send: 0xFF 00 40 02 09 22 3D 3D 08 17 00 01 04 E0

Receive: 0x90 00

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

Receive: 0x90 00

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

Receive: 0x11 22 33 44 90 00

4.3.19ISO15693 Reset to Ready

ISO15693 Tag is from Halt to Ready.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
SelectTag	FF	00	40	03	Len	Flag+UID

Len:01/09,Depending on the Flag definition

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID: Tag UID(8byte, optional)

Len and Flag example:

Len	Flag	UID	备注
01h	Bit6~Bit5=01b	NULL	New firmware support

09h	Bit6~Bit5=01b	8Bytes Random	
09h	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.20ISO15693 Read Block

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
ReadBlock	FF	00	40	04	Len	Data

Len: 03/0B,Depending on the Flag definition

Data: Flag(1Byte) + UID(8Bytes) + BlockAddr(1Byte) + BlockNum(1Byte)

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID : Tag UID(optional)

BlockAddr: the start Block Address

BlockNum: Block number(>=1)

Len and Flag example:

Len	Flag	UID	备注
03h	Bit6~Bit5=01b	NULL	New firmware support
0Bh	Bit6~Bit5=01b	8Bytes Random	
0Bh	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out		
Result	DATA(BlockNum*4)	SW1	SW2

4.3.21ISO15693 WriteBlock

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
WriteBlock	FF	00	40	05	Len	Data

Len: 06/0E,Depending on the Flag definition

Data: Flag(1Byte) + UID(8Bytes) + BlockAddr(1Byte) + BlockData(4Byte)

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID : Tag UID(optional)

BlockAddr: the start Block Address

BlockData: Block data

Len and Flag example:

Len	Flag	UID	备注
06h	Bit6~Bit5=01b	NULL	New firmware support
0Eh	Bit6~Bit5=01b	8Bytes Random	
0Eh	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.22ISO15693 Write AFI

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
WriteAFI	FF	00	40	06	Len	Flag+UID+AFI

Len: 02/0A,Depending on the Flag definition

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID: Tag UID(8Byte, optional)

AFI: New AFI

Len and Flag example:

Len	Flag	UID	备注
02h	Bit6~Bit5=01b	NULL	New firmware support
0Ah	Bit6~Bit5=01b	8Bytes Random	
0Ah	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

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

Receive: 0x90 00

4.3.23ISO15693 Lock AFI

APDU Format:

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

Len: 01/09,Depending on the Flag definition

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID: Tag UID(8Byte, optional)

Len and Flag example:

Len	Flag	UID	备注
01h	Bit6~Bit5=01b	NULL	New firmware support
09h	Bit6~Bit5=01b	8Bytes Random	
09h	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.24ISO15693 Write DSFID

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
WriteDSFID	FF	00	40	08	Len	Flag+UID+DSFID

Len: 02/0A,Depending on the Flag definition

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID: Tag UID(8Byte, optional)

DSFID: New DSFID

Len and Flag example:

Len	Flag	UID	备注
02h	Bit6~Bit5=01b	NULL	New firmware support
0Ah	Bit6~Bit5=01b	8Bytes Random	
0Ah	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.25ISO15693 Lock DSFID

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
LockDSFID	FF	00	40	09	Len	Flag + UID

Len: 01/09,Depending on the Flag definition

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID: Tag UID(8Byte, optional)

Len and Flag example:

Len	Flag	UID	备注
01h	Bit6~Bit5=01b	NULL	New firmware support
09h	Bit6~Bit5=01b	8Bytes Random	
09h	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.26ISO15693 Get System Info

APDU Format:

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

Len: 01/09,Depending on the Flag definition

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID: Tag UID(8Byte, optional)

Len and Flag example:

Len	Flag	UID	备注
01h	Bit6~Bit5=01b	NULL	New firmware support
09h	Bit6~Bit5=01b	8Bytes Random	
09h	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out		
Result	SystemInfo	SW1	SW2

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

4.3.27ISO15693 Get Blocks Security

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
GetBlkSecSt	FF	00	40	0B	Len	Data

Len: 03/0B,Depending on the Flag definition

Data: Flag(1Byte) + UID(8Bytes) + StartAddr (1Byte) + Num(1Byte)

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID : Tag UID(optional)

StartAddr: the start Block Address

Num: Block number(n+1. if n = 0, read start block state)

Len and Flag example:

Len	Flag	UID	备注
03h	Bit6~Bit5=01b	NULL	New firmware support
0Bh	Bit6~Bit5=01b	8Bytes Random	
0Bh	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out		
Result	BlockSecSta * Num	SW1	SW2

4.3.28ISO15693 Lock Block

APDU Format:

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

Len: 02/0A,Depending on the Flag definition

Flag: Refer to ISO15693 Standard(eg:Flag = 0x22 or 0x12)

UID: Tag UID(8Byte, optional)

BkNO: The Block Number to be locked

Len and Flag example:

Len	Flag	UID	备注
02h	Bit6~Bit5=01b	NULL	New firmware support
0Ah	Bit6~Bit5=01b	8Bytes Random	
0Ah	Bit6~Bit5=10b	8Bytes UID	

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

ISO15693 one Tag

Inventory

Send: 0xFF 00 40 00 03 00 26 00

Receive: 0x02 C9 A7 95 0C 00 01 04 E0 90 00

Quiet

Send: 0xFF 00 40 01 09 22 C9 A7 95 0C 00 01 04 E0

Receive: 0x90 00

Inventory

Send: 0xFF 00 40 00 03 00 26 00

Receive: 0x63 00

Reset to Ready

Send: 0xFF 00 40 03 09 22 C9 A7 95 0C 00 01 04 E0

Receive: 0x90 00

Inventory

Send: 0xFF 00 40 00 03 00 26 00

Receive: 0x02 C9 A7 95 0C 00 01 04 E0 90 00

Select

Send: 0xFF 00 40 02 09 22 C9 A7 95 0C 00 01 04 E0

Receive: 0x90 00

Read Block 0Ah

Send: 0xFF 00 40 04 03 12 0A 01

Receive: 0xFF FF FF FF 90 00

WriteBlock 0Ah

Send: 0xFF 00 40 05 06 12 0A 01 02 03 04

Receive: 0x90 00

Read Block 0Ah

Send: 0xFF 00 40 04 03 12 0A 01

Receive: 0x01 02 03 04 90 00

Write AFI

Send: 0xFF 00 40 06 02 12 0A

Receive: 0x90 00

Write DSFID

Send: 0xFF 00 40 08 02 12 0A

Receive: 0x90 00

Inventory

Send: 0xFF 00 40 00 03 00 26 00

Receive: 0x0A C9 A7 95 0C 00 01 04 E0 90 00 //DSFID

GetSysteminfo

Send: 0xFF 00 40 0A 01 12

Receive: 0x0F C9 A7 95 0C 00 01 04 E0 0A 0A 90 00

GetBlockSecurity

Send: 0xFF 00 40 0B 03 12 0A 02

Receive: 0x00 00 00 90 00

ISO15693 multi tags

Inventory

Send: 0xFF 00 40 00 03 01 26 00

Receive: 0x33 DF 11 08 17 00 01 04 E0 0A C9 A7 95 0C 00 01 04 E0 90 00

Select tag1

Send: 0xFF 00 40 02 09 22 C9 A7 95 0C 00 01 04 E0

Receive: 0x90 00

Read Block 0Ah

Send: 0xFF 00 40 04 03 12 0A 01

Receive: 0xFF FF FF FF 90 00

WriteBlock 0Ah

Send: 0xFF 00 40 05 06 12 0A 01 02 03 04

Receive: 0x90 00

Read Block 0Ah

Send: 0xFF 00 40 04 03 12 0A 01

Receive: 0x01 02 03 04 90 00

Select tag2

Send: 0xFF 00 40 02 09 22 DF 11 08 17 00 01 04 E0

Receive: 0x90 00

Read Block 0Ah

Send: 0xFF 00 40 04 03 12 0A 01

Receive: 0xFF FF FF FF 90 00

WriteBlock 0Ah

Send: 0xFF 00 40 05 06 12 0A 01 02 03 04

Receive: 0x90 00

Read Block 0Ah

Send: 0xFF 00 40 04 03 12 0A 01

Receive: 0x01 02 03 04 90 00

Select tag1

Send: 0xFF 00 40 02 09 22 C9 A7 95 0C 00 01 04 E0

Receive: 0x90 00

WriteBlock 0Ah

Send: 0xFF 00 40 05 06 12 0A FF FF FF FF

Receive: 0x90 00

Read Block 0Ah

Send: 0xFF 00 40 04 03 12 0A 01

Receive: 0xFF FF FF FF 90 00

Select tag2

Send: 0xFF 00 40 02 09 22 DF 11 08 17 00 01 04 E0

Receive: 0x90 00

WriteBlock 0Ah

Send: 0xFF 00 40 05 06 12 0A FF FF FF FF

Receive: 0x90 00

Read Block 0Ah

Send: 0xFF 00 40 04 03 12 0A 01

Receive: 0xFF FF FF FF 90 00

4.3.29Set SAM Baud Rate (Set PPS)

This function is aim to set SAM Baud Rate. The SAM card slots which can be supported by each reader are different. (MR800 supports 2 SAM cards). After sending GetData APDU to Reset SAM card, if you want to modify the SAM Baud Rate, you can use this APDU command to set the new Baud Rate Value. (Note: this SAM card must support the Baud Rate to be set).

APDU Format:

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

APDU Format(Recommended):

Command	Class	INS	P1	P2	Lc	Data
SetSamBaud	FF	00	60	10	02	SAMn+Baudrate

SAMn:

0x00 - SAM1 SetPPS

0x01 - SAM2 SetPPS

0x02 - SAM3 SetPPS

0x03 - SAM4 SetPPS

Baudrate:

0x00 - 9600(Default)

0x01 - 19200

0x02 - 38400

0x03 - 55800

0x04 - 57600

0x05 - 115200

0x06 - 230400

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.30Automatically Set SAM Baud Rate (Set PPS)

This function is aim to according to SAM Reset Information (ATR) parameters in the automatic set of SAM baudrate.Default is not enable. This parameter is set to save when power is off.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
AutoSetBaud	FF	00	60	14	01	Status

Status:

0x00: disenable

0x01: enable

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.31Set SAM Baud Rate after Reset (through PPSS)

This function is aim to set SAM Baud Rate after Reset. The SAM card slots which can be supported by each Reader are different. (MR800 has 2 SAM slots). Usually, the default SAM Baud Rate after Reset is 9600bps. Before sending GetData APDU to Reset SAM card, if you want to modify the SAM Baud Rate after Reset, you can use this APDU command to set the new Baud Rate Value. (Note: this SAM card must support the Baud Rate to be set). This parameter is set to save when power is off.

APDU Format 1:

Command	Class	INS	P1	P2	Lc	Data
SetRstBaud	FF	00	60	SAMn	01	Baudrate

SAMn:

0x04 – SAM1 Reset Baudrate

0x05 – SAM2 Reset Baudrate

0x06 – SAM3 Reset Baudrate

0x07 – SAM4 Reset Baudrate

Baudrate:

0x00 - 9600(Default)

0x01 - 19200

0x02 - 38400

0x03 - 55800

0x04 - 57600

0x05 - 115200

0x06 – 230400

APDU Format 2(Recommended):

Command	Class	INS	P1	P2	Lc	Data
SetSamBaud	FF	00	60	11	02	SAMn+Baudrate

SAMn:

0x00 - SAM1 SetPPS

0x01 - SAM2 SetPPS

0x02 - SAM3 SetPPS

0x03 - SAM4 SetPPS

Baudrate:

0x00 - 9600(Default)

0x01 - 19200

0x02 - 38400

0x03 - 55800

0x04 - 57600

0x05 - 115200

0x06 - 230400

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.32Read SAM Baud Rate after Reset

APDU Format:

Command	Class	INS	P1	P2	Lc
ReadRstBaud	FF	00	60	12	SAMn

SAMn: The SAM card slots which can be supported.(eg:MR800 has 2 SAM slots)

Answer:

Response	Data Out		
Result	RstBaud	SW1	SW2

RstBaud(nBytes): SAM1RstBaud + ... + SAMnRstBaud

SAMnRstBaud:

0x00 - 9600

0x01 - 19200

0x02 - 38400

0x03 - 55800

0x04 - 57600

0x05 - 115200

0x06 - 230400

4.3.33Current Operating Smart Card Switch

This function is aim to switch between SmartCard and SAM card. Except "CardRequest" and "Reset" commands by using non-standard APDU (GetData) commands, the rest commands all use Standard APDU commands. In order to distinguish the current operation Card is a SmartCard or SAM card, you can use this command to know it. Inreal application, sometimes after detected the SmartCard via GetData command, and also need be authenticated by SAM card, so now you need temporarily switch the current operation SmartCard to SAM Card via APDU command. After finished the authentication, then back to the SmartCard operation.

APDU Format:

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

CurSmartCard:

0x00 - SmartCard

0x01 - SAM1Card

0x02 - SAM2Card

0x03 - SAM3Card

0x04 - SAM4Card

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

To operate the SmartCard and SAM card at the same time, the detailsas following:

Initialization when Power On:

Reset SAM1 card when power on. Send APDU {GetData(SAM)} command; if success, SAM1 is the current operating Card. In this situation, all of the sending Standard APDU commands will be sent to it.

Example:

Send and Get random from SAM1:

APDU= 0x00 84 00 00 08

SmartCard Cyclic Detection:

Send APDU {GetData(SmartCard)} command; Whatever the returned result is right or not. SmartCard is the current operating Card. In this situation, all of the sending Standard APDU commands will be sent to it.

Example:

Send and Get random from SmartCard:

APDU= 0x00 84 00 00 08

After detected the SmartCard successfully, the SmartCard need be authenticated by SAM card, and then to read the SmartCard. Send APDU {GetData(SmartCard)} command to choose the SmartCard to be the current operating Card, then to get the authentication data. If the authentication data is correct, then go to the next operation step.

Example:

Switch current operating SmartCard to SAM1 Card:

Send APDU {SwitchSmartCard(SAM1)} command; if success, then to send Standard ADPU command to SAM1 to authenticate the data from SmartCard is validity or not.(Note: SAM1 is the current operating Card).

If you want to operate the SmartCard again, You need change the current SAM1 to the SmartCard via sending APDU {SwitchSmartCard(SmartCard)} command. If success, now the current operating Card is SmartCard

4.3.34RTC Initialization

This function is aim to do initialization for theInternal Clock of the Reader. If the time need be kept when power off, the readerneed be equipped with battery. Readers are required to support RTC.

APDU Format:

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

Time:

Year (High Byte) + Year(Low Byte) + Month + Date + Hour + Minute + Second + Week

Example:

2010 - 4 - 12 12:01:00 Monday: Time = 0x07 DA 04 0C 0C 01 00 01

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Set and Read Time

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

Receive: 0x90 00

Send: 0xFF 00 FB 01 08

Receive: 0x07 DA 04 0C 0C 03 15 01 90 00

4.3.35RTC Time Read

This function is aim to read the display time format on LCD. If the time need be kept when power off, the readerneed be equipped with battery. Readers are required to support RTC.

APDU Format:

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

Answer:

Response	Data Out		
Result	Time	SW1	SW2

Time:

Year (High Byte) + Year(Low Byte) + Month + Date + Hour + Minute + Second + Week

Example:

2010 - 4 - 12 12:01:00 Monday : Time = 0x07 DA 04 0C 0C 01 00 01

4.3.36RTC TimeDisplay

This function is aim to set the display time format on LCD. If the time need be kept when power off, the readerneed be equipped with battery.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
DisTime	FF	00	FB	02	03	Data

Data: EnableFag(1Byte) + Line(1Byte) + Column(1Byte)

EnableFag: Date display enable (0-Disable, 1-Enable)

Line: The start display line (0~7 or 0~12) (LCD resolution:128*64 or 240*128)

Column: The start display column (0~127 or 0~239)(Same as above)

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Time Display OFF

Send: 0xFF 00 FB 02 03 00 00 00

Receive: 0x90 00

Time Display ON

Send: 0xFF 00 FB 02 03 01 03 05

Receive: 0x90 00

4.3.37RTC DateDisplay

This function is aim to set the display date format on LCD. If the dateneed be kept when power off, the readerneed be equipped with battery.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
DisDate	FF	00	FB	03	03	Data

Data: EnableFag(1Byte) + Line(1Byte) + Column(1Byte)

EnableFag: Date display enable (0-Disable, 1-Enable)

Line: The start display line (0~7 or 0~12) (LCD resolution:128*64 or 240*128)

Column: The start display column (0~127 or 0~239)(Same as above)

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Date Display OFF

Send: 0xFF 00 FB 03 03 00 00 00

Receive: 0x90 00

Date Display ON

Send: 0xFF 00 FB 03 03 01 03 05

Receive: 0x90 00

4.3.38Set The Date Display Format

MR88x special instruction.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
DateFormat	FF	00	FB	04	01	USdateformat

Usdateformat:

0x00 : YYYY-MM-DD(Default)

0x01 : MM-DD-YYYY

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.39Set Display Character Set

MR800/MR880 support Simplify Chinese,Traditional Chinese and Russian character. For none ASCII characters in instruction will use this set to display the font.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
Display Font	FF	00	FC	00	01	FontType

FontType:

0x01: Simplify Chinese

0x02: Traditional Chinese

0x03: Russian

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Send: 0xFF 00 FC 00 01 01

Receive: 0x90 00

4.3.40Read Display Character Set Setting

Read display character set setting.

APDU Format:

Command	Class	INS	P1	P2	Le
Display Font	FF	00	FC	01	01

Answer:

Response	Data Out		
Result	CharacterSet	SW1	SW2

Example:

Send: 0xFF 00 FC 00 01 02

Receive: 0x90 00

Send: 0xFF 00 FC 01 01

Receive: 0x02 90 00

4.3.41Set Display Font Pixel

MR880 support 16, 24 and 32 pixel display fonts. This instruction could switch the display font pixel.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
SetFontPixel	FF	00	FC	0A	01	FontPiexl

FontPiexl:

0x00 = 16pixel

0x01 = 24pixel

0x02 = 32pixel (system default, not save after repower)

Answer:

Response	Data Out	
Result	SW1	SW2

Remark: Russian support 32 pixel only.

Example:

Send: FF 00 FC 0A 01 00

Receive: 90 00

4.3.42LCD Display Character String

This function is aim to set the specified number of characters (including English and/or Chinese, Russian) on LCD.

One Chinese font - 2Byte

One Russian font - 1Byte

One English font - 1Byte

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
Display Font	FF	00	FC	02	nByte	Data

Data: Configure(1Byte) + **Row**(1Byte) + **Column**(1Byte) + **DisplayData**(nBytes)

Configure:

Bit	Value	Instructions
B0	0	Positive Display
	1	Negative Display
B2~B1	00	Before showing new information on the screen, no any delete the old ones
	01	Before showing new information on the screen, only to clear the row of the showed screen
	10	Before showing new information on the screen, delete the all old ones
B3	0	BackLight off
	1	BackLight on
B7~b4	RFU	RFU

Row:

Value	Instructions
0~7	LCD resolution 128*64(1Row = 16 dot High)
0~7	LCD resolution 240*128 32 pixel font (1Row = 32 dot High)
0~0x09	LCD resolution 240*128 24 pixel font (1Row = 24 dot High)
0~0x0F	LCD resolution 240*128 16 pixel font (1Row = 16 dot High)

Column: (0~7 or 0~12) (LCD resolution:128*64 or 240*128)

Display Data: One Chinese font - 2Byte, One ASCII or Russian font – 1Byte.

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Showing "QUIO" on the top left corner of LCD displayer with "Positive Display", "Before

showing new information on the screen, no any delete the old ones" and"BackLight off" .

Send: 0xFF 00 FC 02 09 00 00 00 BD F0 C4 BE D3 EA

Receive: 0x90 00

4.3.43LCD Display Character String at Any Point

MR88x special instruction.

This function is aim to set the specified number of characterson LCD at ANY POINT.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
Display Font	FF	00	FC	09	nByte	Data

Data: Configure(1Byte) + **Row**(1Byte) + **Column**(1Byte) + **DisplayData**(nBytes)

Configure:

Bit	Value	Instructions
B0	0	Positive Display
	1	Negative Display
B1	RFU	RFU
B2	0	keep screen
	1	clear screen
B3	0	BackLight off
	1	BackLight on
B5~B4	01	display character with 16 pixel font
	10	display character with 24 pixel font
	11	display character with 32 pixel font
B7~b6	RFU	RFU

Row: 0 ~ 127 dot

Column: 0 ~ 239 dot

Display Data: One Chinese font - 2Byte, One ASCII or Russian font – 1Byte.

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Showing "QUIO" on the top left corner of LCD displayer with "Positive Display", "Before showing new information on the screen, no any delete the old ones" and"BackLight off" .

Send: 0xFF 00 FC 02 09 00 00 00 BD F0 C4 BE D3 EA

Receive: 0x90 00

4.3.44Display Picture on LCD(Directly send picture data)

This function is aim to display the specified size picture. If the showing picture is large, it needs several times to be displayed.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
DisPicture	FF	00	FC	03	nByte	Data

Data: Configure(1Byte) + **Row**(1Byte) + **Column**(1Byte) + **PictureWidth**(1Byte) + **PictureHigh**(1Byte) + **DisplayData**(nBytes)

Configure:

Bit	Value	Instructions
B0	0	Positive Display
	1	Negative Display
B2~B1	00	Before showing new information on the screen, no any delete the old ones
	01	Before showing new information on the screen, only to clear the row of the showed screen
	10	Before showing new information on the screen, delete the all old ones
B3	0	BackLight off
	1	BackLight on
B7~b4	RFU	RFU

Row（1Row = 8 dot High）: 0~7 or 0~15(LCD resolution:128*64 or 240*128)

Column:0 ~ 127 or 0~239 (Same as above)

PictureWidth: 1~128 or 0~240, Width of the picture(Same as above)

PictureHigh: 1~8 or 0~16, Picture height(Same as above)

DisplayData: Picture data to be displayed (Bytes = Width*Height)

Answer:

Response	Data Out	
Result	SW1	SW2

4.3.45Delete Row on LCD

MR80x special instruction.

For the convenience of the screen to be cleared, the user can remove fonts or picture according to each row.

APDU Format:

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

Row（1Row = 8 dot High）: "Bit0 ~ Bit7" means 0 to 7 rows. (0-keep, 1- delete)

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Delete the whole rows on the LCD

Send: 0xFF 00 FC 04 01 FF

Receive: 0x90 00

4.3.46Delete Row on LCD

MR88x special instruction.

For the convenience of the screen to be cleared, the user can remove fonts or picture according to each row.

APDU Format:

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

Row（1Row = 8 dot High）: "Bit0 ~ Bit15" means 0 to 15 rows. (0-keep, 1- delete)

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Delete the whole rows on the LCD

Send: 0xFF 00 FC 04 02 FFFF

Receive: 0x90 00

4.3.47Set Boot Screen on LCD

This function is aim to set the boot screen on LCD when power on. If no setting, it will show the default screen. All of the screen pictures will be stored in Flash AT45DB321/AT45DB641.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
PowerOnPIC	FF	00	FC	05	08	Data

Date: Enable(1Byte) + **SaveAddr**(2Byte) + **Width**(1Byte) + **High**(1Byte) + **StartLine**(1Byte) + **StartColumn**(1Byte) + **Time**(1Byte)

Enable: 0-Enable Boot Screen, 1-Disable Boot Screen

SaveAddr: Save the Boot Screen in the Flash; Address LSB first

Width: Width of the image (1~128 or 1~240)(LCD resolution:128*64 or 240*128)

High: Image height (1~8 or 1~16) (Same as above)

StartLine: Displaystart line (0~7 or 1~15) (Same as above)

StartColumn: Display start column (0~127 or 1~239) (Same as above)

Time: To set the time of the Boot Screen (Unit: S)

Answer:

Response	Data Out	
Result	SW1	SW2

Note:

If disable Boot Screen, the following parameters are meaningless.

The Boot Screen stored in external FLASH of the reader. Thefonts stored totallyin the 1303(0~1302) Blocks with MR800, 10360(0 ~ 10359) blocks with MR880, the user can not erase or set the above Blocks. Forusers'use the block number is 1303 ~ 8191 of MR800, forusers' use the block number is 10360~ 16383 of MR880, each block size is 512 bytes.

Before the Boot Screen Enable, the Screen picture data need be written into the Flash "SaveAddr" via "FlashWrite APDU" command.If the picture is larger than 512 bytes, the extra bytes will be written into the following block.

The image dimension = Width*High

Example:

Set a Boot Screen picture on MR800, the picture is 128*64. (The picture data need be written into the FLASH)

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 7C 7F 7F7F 3F 3F

3F 3F 1F 1F1F 0F 0F 07 07 03 7F FF FF FF FF FF

FF FFFF 7D 03 07 07 0F 0F 1F 1F1F 3F 3F 3F 3F

7F 7F7F 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 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 7F7F7F7F 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 E0E0 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

Receive: 90 00

Send:

FF 00 FD 01 84 05 18 00 80

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 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

Receive: 90 00

Send:

FF 00 FD 01 84 05 18 01 00

00 00 00 00 00 00 00 00 00 E0 E0E0E0E0 C0

C0 C0 80 80 80 00 00 00 00 F0 FC FE FF FF FF

FF FE F8 E0 00 00 00 00 80 80 80 C0 C0C0 E0

E0 E0 E0 E0 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 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

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 C0 C0C0

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

Receive: 90 00

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

Receive: 90 00

4.3.48Set Standby Screen on LCD

This function is aim to set the standby screen. If no setting, after finished the User's interface display, it won't return to the standby screen. All pictureswill be stored in FLASH AT45DB321/AT45DB641.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
IdlePIC	FF	00	FC	06	08	Data

Date: Configure (1Byte) + **SaveAddr**(2Byte) + **Width**(1Byte) + **High**(1Byte) + **StartLine**(1Byte) + **StartColumn**(1Byte) + **Time**(1Byte)

Configure:

Bit	Value	Instructions
B0	0	Enable Standby Screen
	1	Disable Standby Screen
B2~B1	00	Before showing new information on the screen, no any delete the old ones
	01	Before showing new information on the screen, only to clear the row of the showed screen
	10	Before showing new information on the screen, delete the all old ones
B3	0	BackLight off
	1	BackLight on
B7~b4	RFU	RFU

SaveAddr: Save the Boot Screen in the Flash; Address LSB first

Width: Width of the image (1~128 or 1~240)(LCD resolution:128*64 or 240*128)

High: Image height (1~8 or 1~16) (Same as above)

StartLine: Displaystart line (0~7 or 1~15) (Same as above)

StartColumn: Display start column (0~127 or 1~239) (Same as above)

Time: Set operation interval time, if no further operation, then the LCD screen enter into the standby screen (Unit: S).

Answer:

Response	Data Out	
Result	SW1	SW2

Note:

If the Disable StandbyScreen, the following parameters are meaningless.

The Boot Screen stored in external FLASH of the reader. Thefonts stored totallyin the 1303(0~1302) Blocks with MR800, 10360(0 ~ 10359) blocks with MR880, the user can not erase or set the above Blocks. Forusers' use the block number is 1303 ~ 8191 of MR800, forusers' use the block number is 10360~ 16383 of MR880, each block size is 512 bytes.

Before the StandbyScreen Enable, the Screen picture data need be written into the Flash SaveAddr via "FlashWrite APDU" command. If the picture is larger than 512 bytes, the extra bytes will be written into the following block.

The image dimension = Width*High

As to how to operate, please refer to the SDK for "Set Boot Screen on LCD", but notice the storage address in FLASH.

4.3.49LCD Backlight Control

APDU Format:

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

Mode:

00 - OFF

01 - ON

02 - Specified time on (Time data is valid)

Time: Only in "Mode=2"are valid (Unit:S)

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

LCD BackLight is on and last 15s

Send: 0xFF 00 FC 07 02 02 0F

Receive: 0x90 00

4.3.50LCD Display Picture ofStored in FLASH

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
DisplayPIC	FF	00	FC	08	09	Data

Data: Configure (1Byte) + **DisAddr**(2Byte) + **Width**(1Byte) + **High**(1Byte) + **StartLine**(1Byte) + **StartColumn**(1Byte)

Configure:

Bit	Value	Instructions
B0	RFU	RFU
B2~B1	00	Before showing new information on the screen, no any delete the old ones
	01	Before showing new information on the screen, only to

		clear the row of the showed screen
	10	Before showing new information on the screen, delete the all old ones
B3	0	BackLight off
	1	BackLight on
B7~B4	RFU	RFU

DisAddr:the savedAddress in the Flash; LSB first.

Width: Width of the image (1~128 or 1~240)(LCD resolution:128*64 or 240*128)

High: Image height (1~8 or 1~16) (Same as above)

StartLine: Displaystart line (0~7 or 1~15) (Same as above)

StartColumn: Display start column (0~127 or 1~239) (Same as above)

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Display the Screen Picture from Address1303 in FLASH

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

Receive: 0x90 00

4.3.51Read Data from FLASH

The Flash on MR801/MR811 is AT45DB321 (MR881 isAT45DB641).From address0 to address1302, these are used to store the fonts (MR881 is address0 to address10359), so please don't read or write them. For users' use the block number is 1303 ~ 8191 of MR801/MR811, for users' use the block number is 10360~ 16383 of MR881, each block size is 512 bytes.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
ReadFlash	FF	00	FD	00	06	Data

Data: BlockAddr (2Byte) + **ByteAddr**(2Byte) + **Len**(2Byte)

BlockAddr: Block Address (MSB first)

ByteAddr: The start byte address in Block (MSB first)

Len : The length of Byte to be read (MSB first),Len≤256

Answer:

Response	Data Out		
Result	Flash Data	SW1	SW2

Example:

Read 2bytes from Block2 in Flash, the start address is 0002

Send: 0xFF 00 FD 00 06 00 02 00 02 00 02

Receive: 0x18 08 90 00

4.3.52Write Data into FLASH

The Flash on MR801/MR811 is AT45DB321 (MR881 isAT45DB641).From address0 to address1302, these are used to store the fonts (MR881 is address0 to address10359), so please don't read or write them. For users' use the block number is 1303 ~ 8191 of MR801/MR811, for users' use the block number is 10360~ 16383 of MR881,each block size is 512 bytes.

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
WriteFlash	FF	00	FD	01	04+n	Data

Data: BlockAddr (2Byte) + **ByteAddr**(2Byte) + **nData** (nBytes)

BlockAddr: Block Address (MSB first)

ByteAddr: The start byte address in Block (MSB first)

nData: Data to be written

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Write 1bytes into Block0616 in Flash, the start address is 0002

Send: 0xFF 00 FD 01 05 06 16 00 02 01

Receive: 0x90 00

4.3.53Get Device SNR

APDU Format:

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

Answer:

Response	Data Out		
Result	Product SNR	SW1	SW2

Example:

Send: 0xFF 00 FF 00 0A

Receive: 0x01 05 07 09 09 04 03 08 06 09 90 00

4.3.54Get Hardware and Firmware Version

APDU Format:

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

Answer:

Response	Data Out		
Result	HandwareVer（2Byte）+ Software Ver（2Byte）	SW1	SW2

Example:

Send: 0xFF 00 FF 01 04

Receive: 0x01 00 02 02 90 00

4.3.55Set LED

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
LEDCtr	FF	00	FF	02	05	Data

Data:LEDstate(1Byte)+**StateMask**(1Byte)+**T1**(1Byte)+**T2** (1Byte)+**Number**(1Byte)

LEDState:

BIT0 = Red light final state (1 - ON, 0 - OFF)

BIT1 = Green light final state (1 - ON, 0 - OFF)

BIT2 = Blue light final state (1 - ON, 0 - OFF)

BIT3 = Yellow light final state (1 - ON, 0 - OFF)

BIT4 = Red light flashing in the initial state (1 - ON, 0 - OFF)

BIT5 = Green light flashing in the initial state (1 - ON, 0 - OFF)

BIT6 = Blue light flashing in the initial state (1 - ON, 0 - OFF)

BIT7 = Yellow light flashing in the initial state (1 - ON, 0 - OFF)

StateMask:

BIT0 = Red state update mask (1 - Update, 0- Maintenance)

BIT1 = Green state update mask (1 - Update, 0- Maintenance)

BIT2 = Blue state update mask (1 - Update, 0- Maintenance)

BIT3 = Yellow state update mask (1 - Update, 0- Maintenance)

BIT4~7: RFU

T1/T2: T1, T2 time (Unit: 100ms), T = T1+T2

Number: Times

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Four kinds of lights twinkle two times. And then all of them are OFF

Send: 0xFF 00 FF 02 05 F0 0F 0F0F 02

Receive: 0x90 00

Red twinkles two times. And then light ON

Send: 0xFF 00 FF 02 05 11 01 0F 0F 02

Receive: 0x90 00

Yellow twinkles, and then Red is ON. This state will be executed two times.

Send: 0xFF 00 FF 02 05 81 09 0F 0F 02

Receive: 0x90 00

4.3.56Set Buzzer

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
BuzzerCtr	FF	00	FF	03	05	Data

Data: BeepState(1Byte)+**StateMask**(1Byte)+**T1**(1Byte)+**T2** (1Byte)+**Number**(1Byte)

BeepStatus:

BIT0 = BEEP final state (1 - ON, 0 - OFF)

BIT4 = BEEP initial state (1 - ON, 0 - OFF)

StatusMask:

BIT0 = Buzzer status update mask (1 - Update, 0 - Maintenance)

BIT4~7 RFU

T1/T2: T1, T2 time (Unit: 100ms), T = T1+T2

Number: Times

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Buzzer beeps two times with status update mask. This state will be executed two times.

Send: 0xFF 00 FF 03 05 08 01 0F 0F 02

Receive: 0x90 00

4.3.57Set Antenna State

APDU Format:

Command	Class	INS	P1	P2	Lc	Data
AntennaCtr	FF	00	FF	04	01	Antenna state

Antenna state:

0x00 - Close

0x01 - Open

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Close Antenna

Send: 0xFF 00 FF 04 01 00

Receive: 0x90 00

4.3.58Set Card Encryption Mode

APDU Format:

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

EncryptMode:

0x00-Philips

0x01-Shanghai Standard

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Set Shanghai Encryption Mode

Send: 0xFF 00 FF 05 01 01

Receive: 0x90 00

4.3.59Reader Reset to Factory Default (Repower on)

APDU Format:

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

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Send: 0xFF 00 FF 06 00

Receive: 0x90 00

4.3.60System Reboot

APDU Format:

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

Answer:

Response	Data Out	
Result	SW1	SW2

Example:

Send: 0xFF 00 FF 07 00

Receive: 0x90 00

4.3.61Direct RF Transaction

Send data stream over RF interface to card and receive the data.

APDU format:

Command	CLA	INS	P1	P2	Lc	CMD	TMO	Data
Transmit	FF	00	FF	FF	LEN	CMD	FWI	RF Data

LEN: the length of Data

CMD: 0: Send commands and receive data.

1: Send only.

FWI: Timeout parameter. Operate the M1 card, FWI = 4. When CMD=1, this byte is meaningless

RF Data: the data will send over RF interface

Answer:

Response	Data Out		
Result	Data	SW1	SW2

Response State

Result	SW1 SW2	Meaning
Success	90 00	The operation is completed successfully.
Error	63 00	The operation is failed.
Error	6A 81	No such function

Example:

MIFARE Ultralight card data block read and write operations

Send: 0xFF CA 00 00 00

Receive: 0x07 04 15 BA 8A 7C 3B 80 44 00 00 90 00

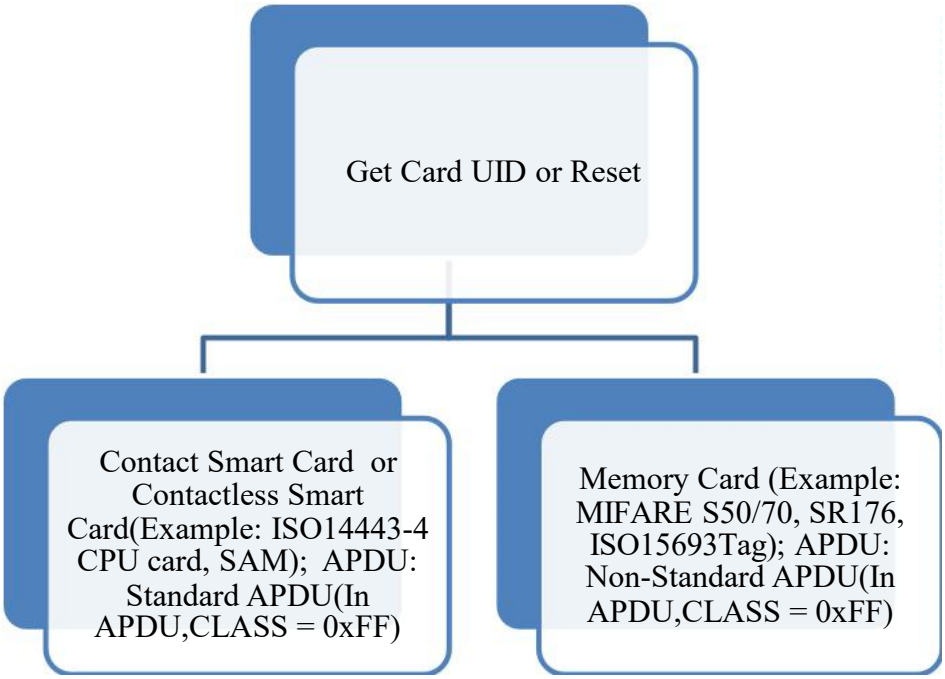
Send: 0xFF 00 FF FF 08 01 00 A2 04 01 02 03 04(write data block 4)

Receive: 0x90 00

Send: 0xFF 00 FF FF 04 00 05 30 04

Receive: 0x01 02 03 04 00 00 00 00 00 00 00 00 00 00 00 00 90 00

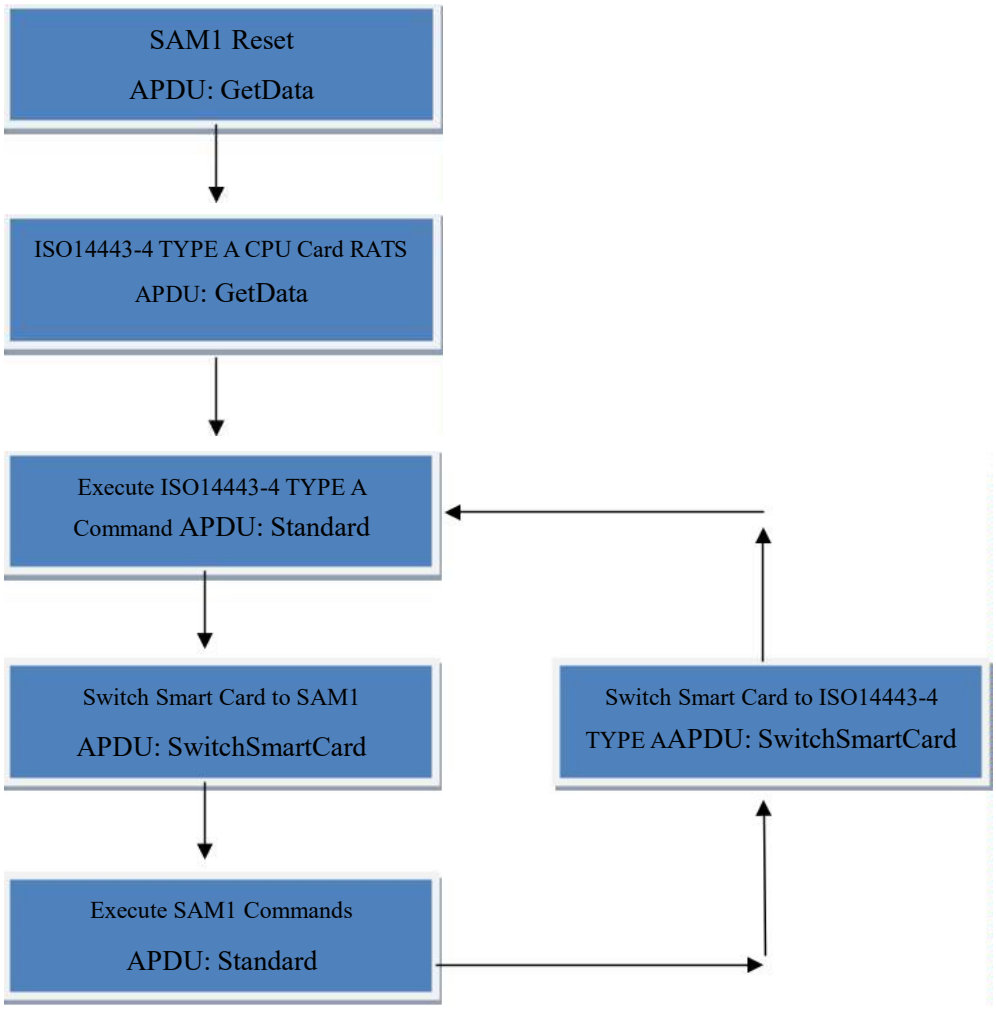
5 Card Operation Procedures



Before to operate any card, first you need send GetData APDU command to obtain the card basic information (Card serial number, reset information, etc.). GetData command contains the type of card to be switched, so once this APDU command executed, the current reading card type is also chosen.

5.1 Smart Contact Card and Contactless Card

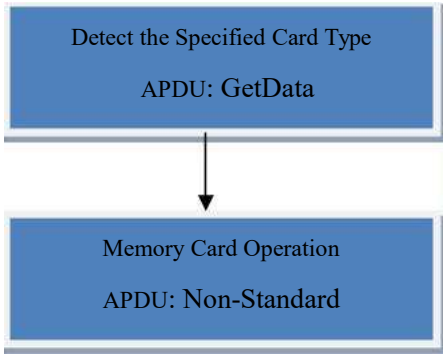
For Smart Contact Card and Contactless Card operation, you can directly send standard APDU commands. If you need to operate the above both Cards (ISO14443-4 Type A and SAM1 card) at the same time, the procedures are as the following:



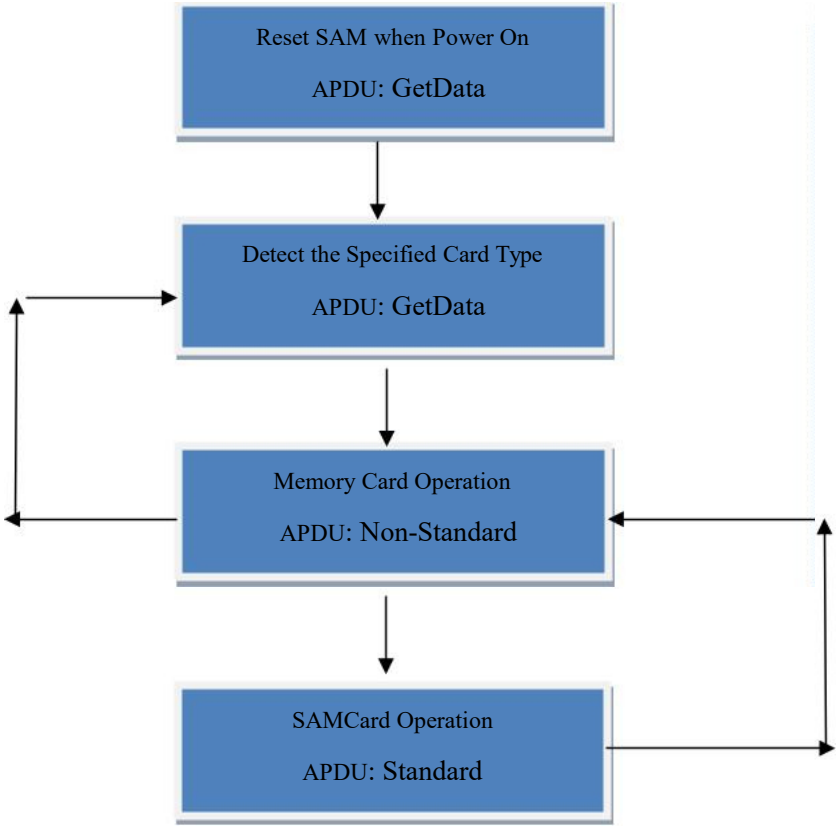
For Contact and Contactless Smart Cards, the operationsare both using the standard APDU. After reset the SAM card, if you still want to operate it, and you need switch the current operated smartcard to SAM card by using the SwitchSmartCardcommand. To make sure the data was sent to the specified Card Type. If the Smart Card and the Memory Card do not need be switched, then after the GetData operation, now the Card type is the GetData operation Card type.

5.2 Memory Cards

Memory Cards' operations are both by sending Non-Standard APDU commands.

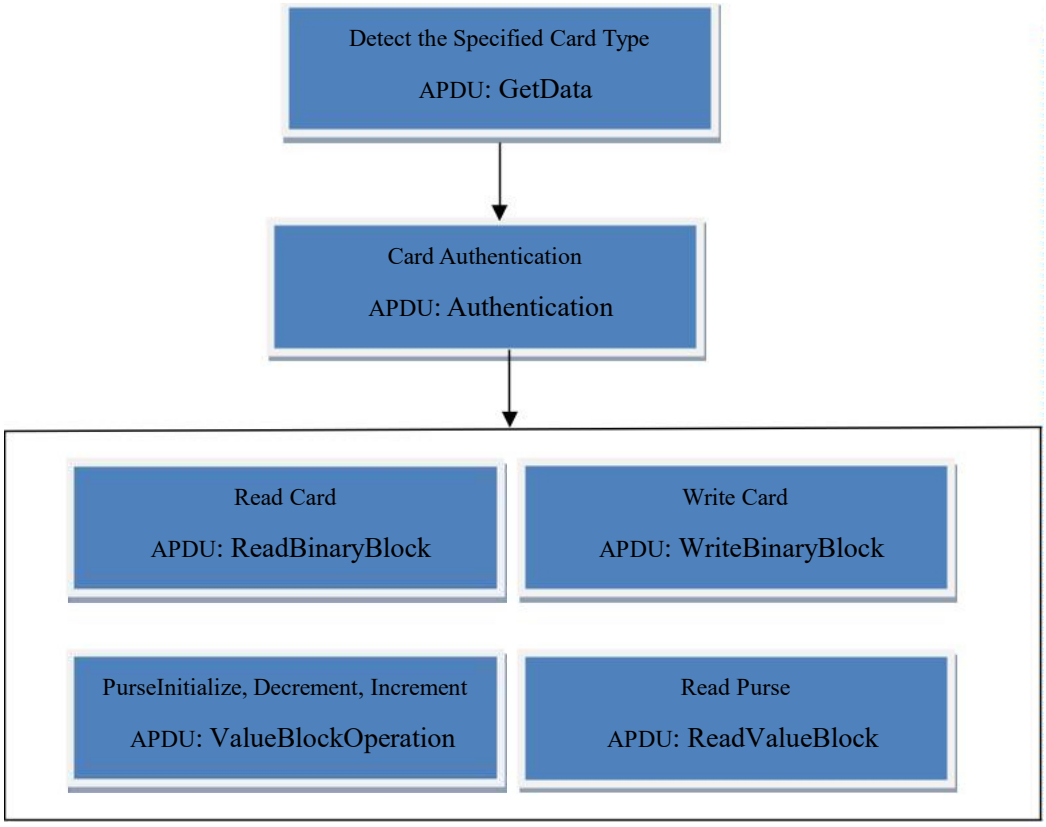


If the memory cards operation with SAM operation, the details as follows:



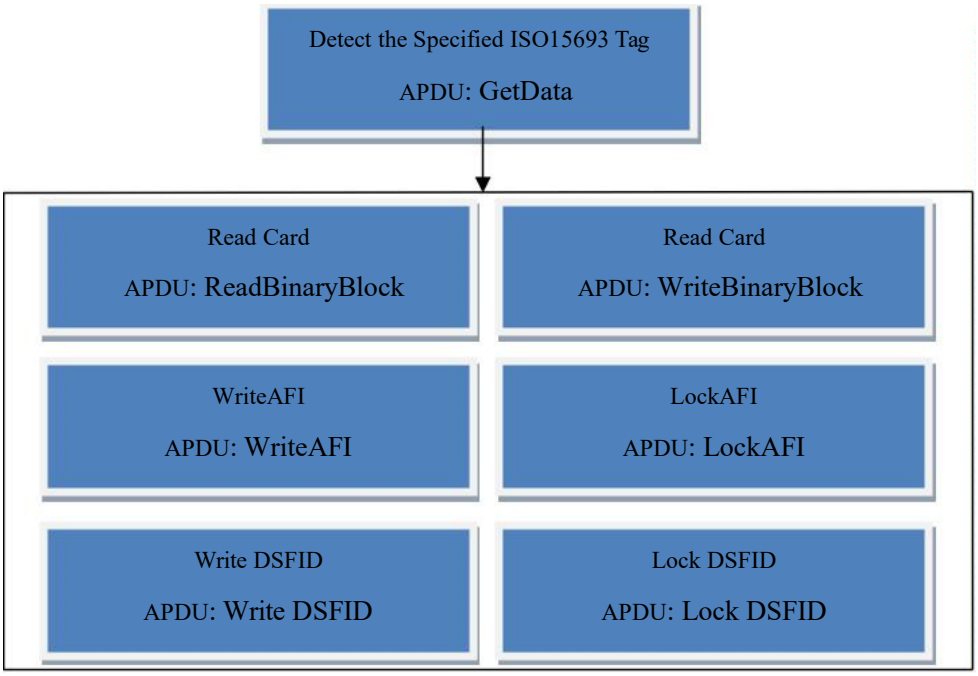
For the memory cards and single SAM operation are no need to be switched. If you need operate more than one SAM card, then before to operate this SAM card, it needs to be switched to the specified SAM card via SwitchSmartCard APDU command.

MIFARE S50/70 Operation



The above operations are no SAM card operation. If with SAM card operation, please refer to the above relevant operations.

ISO15693Tag Operation

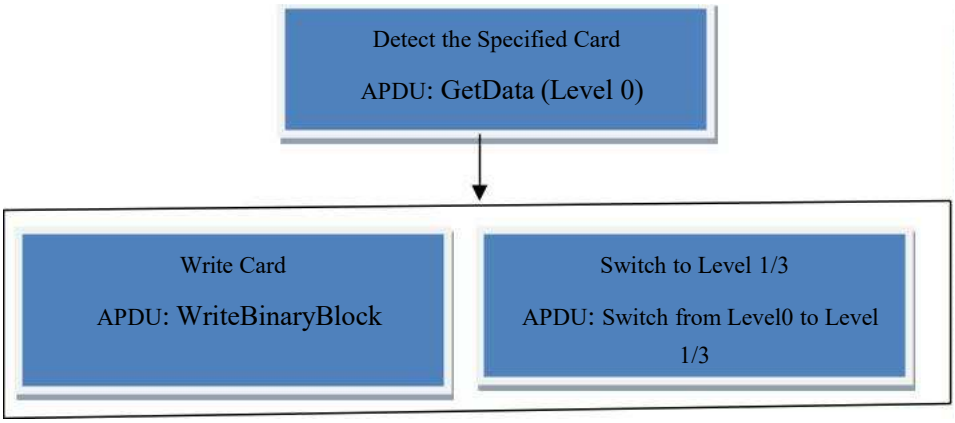


Theoperationsfor ISO15693 Tag are via "ReadBinaryBlock" and "WriteBinaryBlock" APDU commands. It's just suitablefor the last detected one Tag. If you need to operate one tag which is specified the UID, please refer to the Non-Standard APDU (Custom Parts).

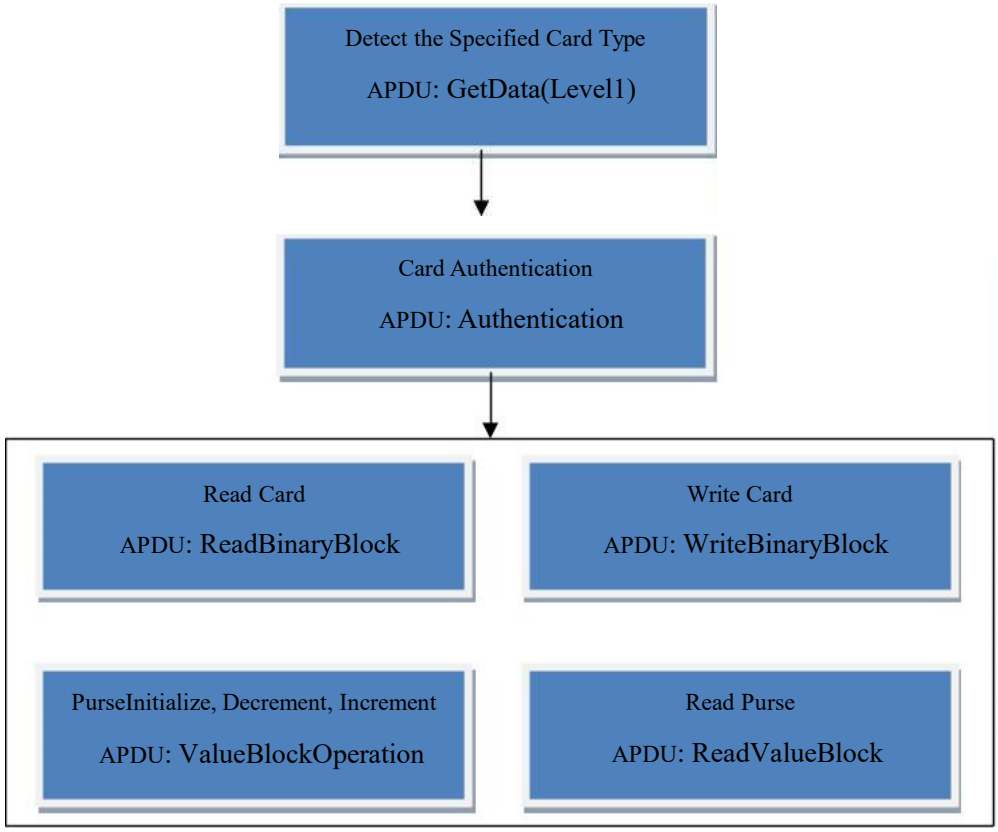
MIFARE PlusOperation

For Mifare Plus card, please refer to the Appendix and there are different instructions in GetData. MifarePlus is divided into four security levels. The different security level has the different operationsequence. Some of operations are just to detect the card serial number. After detected the card, some of operations are just "reset operation". Mifare plus Level1 is compatible with Mifare one, all operations are the same to the Mifareone.

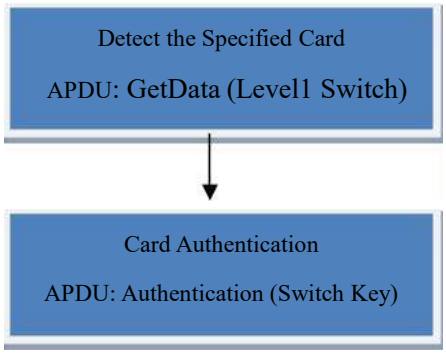
Level 0 Operation:



Level 1 Operation:

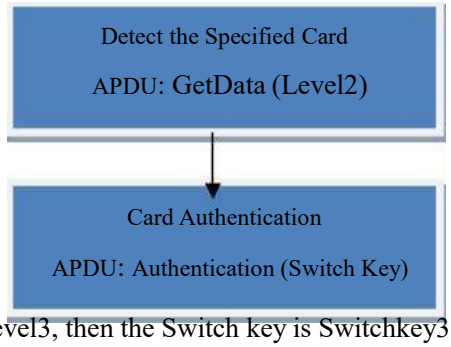


Level 1 Switch Operation:



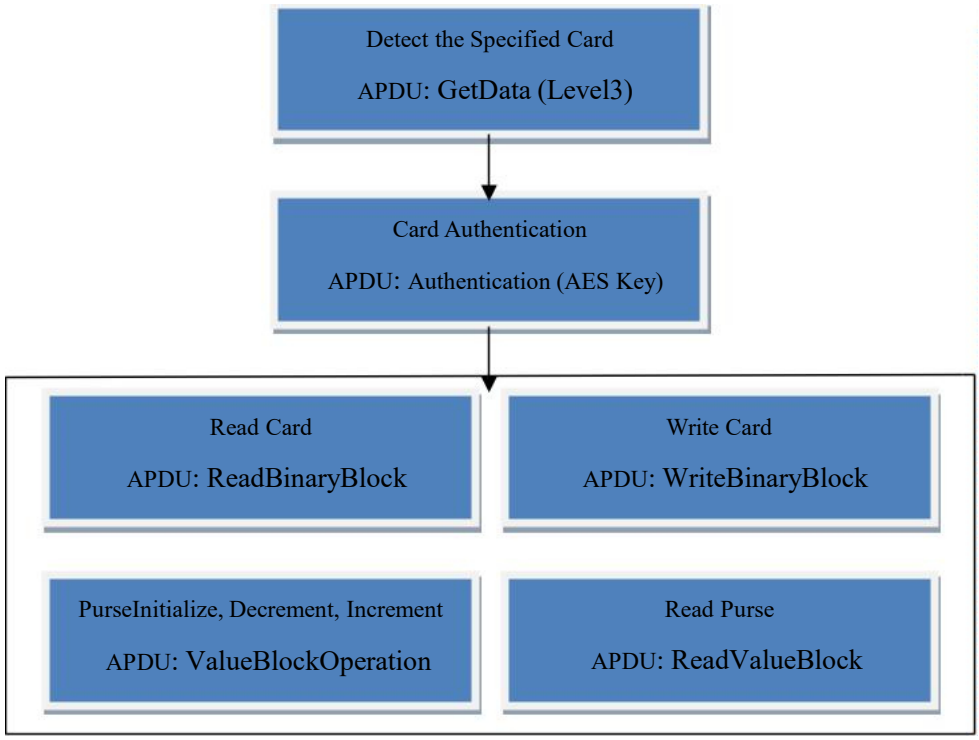
Note that to switch from the Level1 to the other Level, the detected Card Type via GetData is different. If you want to switch Level1 to Level2, then the Switch key is Switchkey2.

Level 2 Operation:



If you want to switch Level2 to Level3, then the Switch key is Switchkey3.

Level 3 Operation:



The operations for other Type Cards are similar. The basic APDU commands contain GetData, ReadBinaryBlock, WriteBinaryBlock and etc. If you need to set Card Parameter, please refer to Non-Standard APDU (Custom Parts).

"LCD&ClockOperations", "SmartcardSwitch ", "SAM to Reset Baudrate", "LED and Buzzer operation", etc. Please refer to Non-Standard APDU (Custom Parts).

Appendix A

For the Data and Key storage structures, there are some difference between MifarePlus Level3 and M one. The details as follows:

Block Relative Address		Block Address	Key Address
Sector0			
Block0	Data block	0x0000	A Key: 0x4000 B Key: 0x4001
Block1	Data block	0x0001	
Block2	Data block	0x0002	
Block3	Level1:KeyA + KeyB Level3: Data+Config	0x0003	
Sector1			
Block0	Data block	0x0004	A Key: 0x4002 B Key: 0x4003
Block1	Data block	0x0005	
Block2	Data block	0x0006	
Block3	Level1:KeyA+Key B Level3: Data+Config	0x0007	
….			
Sector31			
Block0	Data block	0x007C	A Key: 0x403E B Key: 0x403F
Block1	Data block	0x007D	
Block2	Data block	0x007E	
Block3	Level1:KeyA + KeyB Level3: Data+Config	0x007F	
Sector32			
Block0	Data block	0x80	A Key: 0x4040 B Key: 0x4041
Block1	Data block	0x81	
…	Data block	…	
Block15	Level1:KeyA + KeyB Level3: Data+Config	0x8F	
…			
Sector39			
Block0	Data block	0xF0	A Key: 0x404E B Key: 0x404F
Block1	Data block	0xF1	
…	Data block	…	
Block15	Level1:KeyA + KeyB	0xFF	

	Level3: Data+Config		
Configuration block			
	MFP Configuration Block	0xB000	
	Installation Identifier	0xB001	
	ATS Information	0xB002	
	Field Configuration Block	0xB003	
Key block			
	AES Sector Keys	0x4000~0x403F	
	AES Sector Keys	0x4040~0x404F	
	Originality Key	0x8000	
	Card Master Key	0x9000	
	Card Configuration Key	0x9001	
	Level2 switch Key	0x9002	
	Level3 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	