



Advanced Card Systems Ltd.
Card & Reader Technologies

ACR128U Dual- Interface Reader



Application Programming Interface



Table of Contents

1.0.	Introduction	5
1.1.	Features.....	5
2.0.	Terms Used.....	6
3.0.	Architecture of ACR128.....	7
3.1.	Communication between the PCSC Driver and the ICC, PICC & SAM	7
3.2.	Communication between the PCSC Driver and the ACR128U Peripherals.....	8
3.3.	ACR128 Escape Command.....	8
4.0.	Hardware Description	9
4.1.	Reader Firmware Version.....	9
4.2.	LED Indicator	9
4.3.	Buzzer.....	11
4.4.	USB Interface	13
4.5.	ICC Interface (Contact Smart Card)	13
4.6.	SAM Interface (Contact Smart Card).....	13
4.7.	PICC Interface (Contactless Smart Card)	13
5.0.	PICC Interface Description	14
5.1.	ATR Generation.....	14
5.1.1.	ATR format for ISO 14443 Part 3 PICCs.	14
5.1.2.	ATR format for ISO 14443 Part 4 PICCs.	15
5.2.	ICC and PICC Interfaces Conflict Handling.....	16
5.2.1.	Reader Interface Usage	16
5.2.2.	Exclusive Mode Setting.....	16
5.3.	Automatic PICC Polling	18
5.4.	Manual PICC Polling.....	19
5.5.	Change the Default FWI, Polling Timeout And Transmit Frame Size Of The Activated PICC 19	
5.6.	Antenna Field ON/OFF	20
5.7.	Transceiver Setting.....	20
5.8.	PICC Setting	21
5.9.	PICC Polling For Specific PICC Types.....	22
5.10.	PICC T=CL Data Exchange Error Handling	22
5.11.	Auto PPS (Communication Speed Change).....	23
5.12.	Read and Update the RC531 Register.....	24
5.13.	Refresh the Interface Status	24
5.14.	Changing the ISO 7816 Extra Guard Time.....	25
6.0.	PICC Commands for General Purposes.....	26
6.1.	Get Data.....	26
7.0.	PICC Commands (T=CL Emulation) for MiFare 1K/4K MEMORY Cards	27
7.1.	Load Authentication Keys.....	27
7.1.1.	Authentication for MIFARE 1K/4K.....	28
7.1.2.	Authentication for MIFARE Ultralight-C	30
7.2.	Read Binary Blocks	31
7.3.	Update Binary Blocks	32
7.4.	Value Block Related Commands.....	33
7.4.1.	Value Block Operation	33
7.4.2.	Read Value Block.....	34
7.4.3.	Restore Value Block.....	35
8.0.	PICC Commands (T=CL Emulation) for SR176, SR1X512 and SR1X4K Memory Cards 36	



8.1.	Read Binary Blocks	36
8.2.	Update Binary Blocks	37
9.0.	PICC Commands for ISO 14443-4 Compliant Cards	38
Appendix A.	Simple PCSC Application Sample Code.....	40
Appendix B.	E-passport	42

Figures

Figure 1:	ACR128 Architecture.....	7
Figure 2:	ACR128 Peripherals and PC/SC Drivers	8
Figure 3:	LED of ACR128	9
Figure 4:	PICC and ICC Conflict Handling	16

Tables

Table 1:	LED Indicator	10
Table 2:	LED Control.....	10
Table 3:	Buzzer Event.....	11
Table 4:	Buzzer Duration value.....	11
Table 5:	Default LED and Buzzer Behaviors	12
Table 6:	USB Interface Wiring	13
Table 7:	ISO 14443 Part 3 ATR Format.....	14
Table 8:	ISO 14443 Part 4 ATR Format.....	15
Table 9:	Mode Configuration Setting	17
Table 10:	Current Mode Configuration Values.....	17
Table 11:	Register 0x23 –Automatic PICC Polling (Default value = 0x97 or 0x99 or 9F).....	18
Table 12:	Default Values for FWI, Polling Timeout, and Transmit Frame Size.....	19
Table 13:	Antenna Setting Values.....	20
Table 14:	RX Gain Setting Values.....	21
Table 15:	TX Mode Setting Values.....	21
Table 16:	PICC Setting Data Values	21
Table 17:	Card Type Values to configure device for Specific PICC detection	22
Table 18:	Error Handling Level Values.....	23
Table 19:	Connection Speed Values.....	23
Table 20:	Reader Interface Values.....	25
Table 21:	Get UID APDU Format (5 Bytes).....	26
Table 22:	Get UID Response Format (UID + 2 Bytes) if P1 = 0x00.....	26
Table 23:	Get ATS of an ISO 14443 A card (ATS + 2 Bytes) if P1 = 0x01	26
Table 24:	Response Codes	26
Table 25:	Load Authentication Keys APDU Format (11 Bytes).....	27
Table 26:	Load Authentication Keys Response Format (2 Bytes)	27



Table 27:	Load Authentication Keys Response Codes.....	27
Table 28:	Load Authentication Keys APDU Format (6 Bytes) #Obsolete	28
Table 29:	Load Authentication Keys APDU Format (10 Bytes).....	28
Table 30:	Authenticate Data Bytes (5 Byte)	28
Table 31:	Load Authentication Keys Response Format (2 Bytes)	28
Table 32:	Load Authentication Keys Response Codes.....	29
Table 33:	MIFARE 1K Memory Map	29
Table 34:	MIFARE 4K Memory Map	29
Table 35:	Get Challenge APDU Command.....	30
Table 36:	Get Challenge APDU Response	30
Table 37:	Send 3DES Response APDU Command.....	30
Table 38:	Send 3DES Response APDU Response	30
Table 39:	Read Binary APDU Frmat (5 Bytes).....	31
Table 40:	Read Binary Block Response Format (Multiply of 4/16 + 2 Bytes)	31
Table 41:	Read Binary Block Response Codes	31
Table 42:	Update Binary APDU Format (Multiple of 16 + 5 Bytes)	32
Table 43:	Update Binary Block Response Codes (2 Bytes).....	32
Table 44:	Value Block Operation APDU Format (10 Bytes).....	33
Table 45:	Value Block Operation Response Format (2 Bytes)	34
Table 46:	Value Block Operation Response Codes	34
Table 47:	Read Value Block APDU Format (5 Bytes).....	34
Table 48:	Read Value Block Response Format (4 + 2 Bytes)	34
Table 49:	Read Value Block Response Codes	35
Table 50:	Restore Value Block APDU Format (7 Bytes).....	35
Table 51:	Restore Value Block Response Format (2 Bytes).....	35
Table 52:	Restore Value Block Response Codes	35
Table 53:	Read Binary APDU Format (5 Bytes).....	36
Table 54:	Read Binary Block Response Format (Multiply of 4 + 2 Bytes) or (Multiply of 2 + 2 Bytes) 36	
Table 55:	Read Binary Block Response Codes	36
Table 56:	Update Binary APDU Format (Multiple of 4 + 5 Bytes) or (2 + 2 Bytes)	37
Table 57:	Update Binary Block Response Codes (2 Bytes).....	37
Table 58:	ISO 7816-4 APDU Format.....	38
Table 59:	ISO 7816-4 Response Format (Data + 2 Bytes)	38
Table 60:	Common ISO 7816-4 Response Codes.....	38



1.0. Introduction

The ACR128 is a powerful and efficient dual interface smart card reader which can be used to access ISO 7816 MCU cards and Mifare, ISO14443 Type A and B Contactless Cards. It makes use of the Microsoft CCID class driver and USB interface to connect to a PC and accept card commands from the computer application.

The ACR128 acts as the intermediary device between the PC and the Card where a command issued from the PC will be carried out by the reader, specifically, to communicate with the contactless tag, MCU card, SAM card, or the device peripherals (LED or buzzer). It has three interfaces namely the SAM, ICC and PICC interfaces and all these three interfaces follow the PC/SC specifications. The contact interface makes use of the APDU commands as defined in ISO7816 specifications. For contact card operations, refer to the related card documentation and the PC/SC specifications. This API document will discuss in detail how the PCSC APDU commands were implemented for the device peripherals and the Contactless Interface of ACR128.

1.1. Features

The ACR128 has the following features:

- A standard ICC landing type card acceptor is used to allow the user to perform more R/W operations with the contact card.
- A SAM socket is provided for highly secure applications.
- A built-in antenna is provided for PICC applications.
- User-Controllable Peripherals such as LED and Buzzer are implemented for total device control.
- The device is PCSC Compliant for three interfaces namely Contact, Contactless, and SAM Interface.
- The device makes use of the Microsoft CCID class driver framework for trouble-free installation.
- It makes use of USB V2.0 Interface (12 Mbps).
- It is firmware upgradeable through the RS232 interface with a special cable.
- It has intelligent support for Hybrid Cards and Combi-Cards and can detect the PICC even if it is inserted into the contact slot.
- It is ISO 7816 Parts 1-4 Compliant for Contact Smart Card Interface.
- It is ISO 14443 Parts 1-4 Compliant for Contactless Smart Card Interface.
- It uses the T=CL emulation for MiFare 1K/4K PICCs
- Multi-block transfer mode is provided for efficient PICC access.
- It supports high communication speed for PICCs that can reach a maximum speed of 848 kbps for DESFire.
- It implements an energy saving mode whereby the antenna field is turned off whenever no PICC is found, or the PICC is inactive to prevent the PICC from being exposed to the field all the time.



2.0. Terms Used

APDU: This term stands for Application Protocol Data Unit. An APDU is a communication unit, or a packet of data exchanged between two applications, in this case, a reader and a card.

ATR: The term ATR stands for Answer-to-Reset. This refers to the transmission sent by an ICC to the reader (IFD) in response to a RESET condition.

ATS: This term stands for Answer-to-Select. This refers to the transmission sent by a PICC Type A to the reader (PCD) in response to a SELECT condition.

ATQB: This term stands for Answer-to-Request. This refers to the transmission sent by a PICC Type B to the reader (PCD) in response to a REQUEST condition.

Card Insertion Event: This refers to the event when an ICC or a PICC is presented to the reader.

Card Removal Event: This refers to the event when an ICC or a PICC is removed from the reader.

CCID: This term stands for Chip/Smart Card Interface Devices. The CCID Standard is a specification for USB devices that interface with ICC or act as an interface with ICC/PICC.

Combi-Card: This is a smart card that supports both ICC and PICC interface but contains only one smart chip embedded in the card. Only one interface can operate at any given time.

Hybrid-Card: This is a smart card that consists of two or more embedded chip technologies inside, like the ICC and PICC smart chip. Both the ICC and PICC chips can operate at the same time.

ICC: This term stands for Integrated Circuit Card and refers to a plastic card containing an integrated circuit that is compliant with ISO 7816.

IFD: This term stands for Interface Device. This refers to a terminal, communication device, or machine wherein the integrated circuit card is electrically connected during the operation.

ISO 7816: This is the ISO standard for contact smart cards (ICC).

ISO 14443: This is the ISO standard for contactless smart cards (PICC).

PCD: This term stands for Proximity Coupling Device. This term refers to a Contactless Smart Card Reader.

PICC: This term stands for Proximity Integrated Circuit(s) Card. This refers to contactless cards which operate without mechanical contact to the IFD, i.e., uses magnetic coupling.

PC/SC: The term PC/SC stands for Personal Computer Smart Card which is a specification that facilitates the interoperability necessary to allow ICC/PICC technology to be effectively utilized in the PC environment.

SAM: This term stands for Security Access Module, a special MCU card used for security applications.

T=0: This refers to the character-oriented asynchronous half duplex transmission protocol for ICCs as described in ISO 7816.

T=1: This refers to the block-oriented asynchronous half duplex transmission protocol for ICCs as described in ISO 7816.

T=CL: This refers to the block-oriented asynchronous half duplex transmission protocol for PICCs as described in ISO 14443.

USB: This term stands for Universal Serial Bus which is a common device interface used in a PC environment.

3.0. Architecture of ACR128

3.1. Communication between the PCSC Driver and the ICC, PICC & SAM

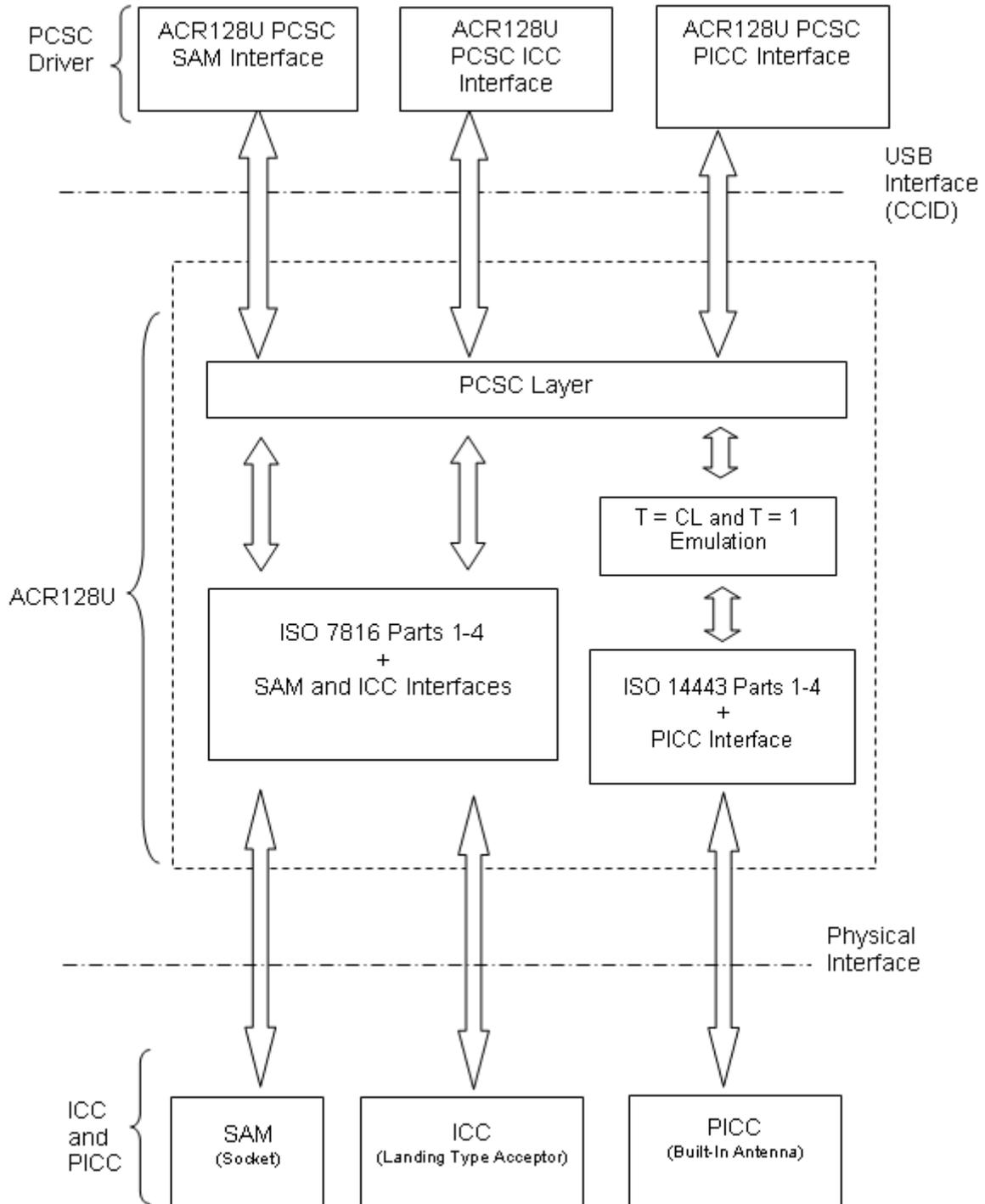


Figure 1: ACR128 Architecture

3.2. Communication between the PCSC Driver and the ACR128U Peripherals

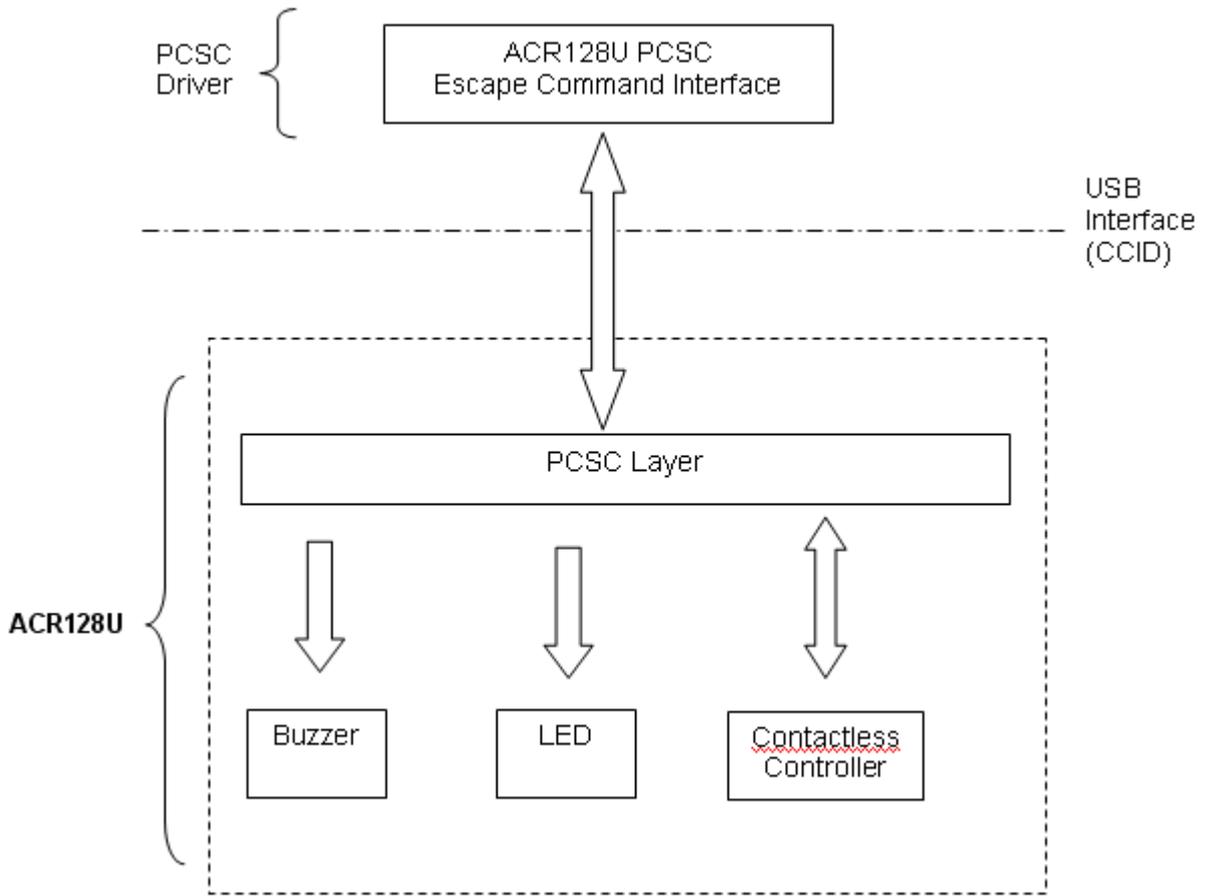


Figure 2: ACR128 Peripherals and PC/SC Drivers

3.3. ACR128 Escape Command

To send a direct command to the device, the driver uses the PC/SC SCardControl API. The dwControlCode parameter is defined as:

```
#define IOCTL_SMARTCARD_ACR128_ESCAPE_COMMAND SCARD_CTL_CODE(2079)
```

4.0. Hardware Description

4.1. Reader Firmware Version

To retrieve the reader firmware version of the device, issue the following command:

ACR128 Escape Command

Read Firmware Version	18	00
-----------------------	----	----

Response

Response Data	E1	00	00	00	01	Firmware Version [14h bytes]	RFU [0Ah bytes]
---------------	----	----	----	----	----	---------------------------------	--------------------

Example:

Firmware Version (HEX) = 41 43 52 31 32 38 55 5F 56 31 34 00 00 00 00 00 00 00 00 00

Firmware Version (ASCII) = "ACR128U_V14"

4.2. LED Indicator

The LEDs are used to show the state of the contact and contactless interfaces:

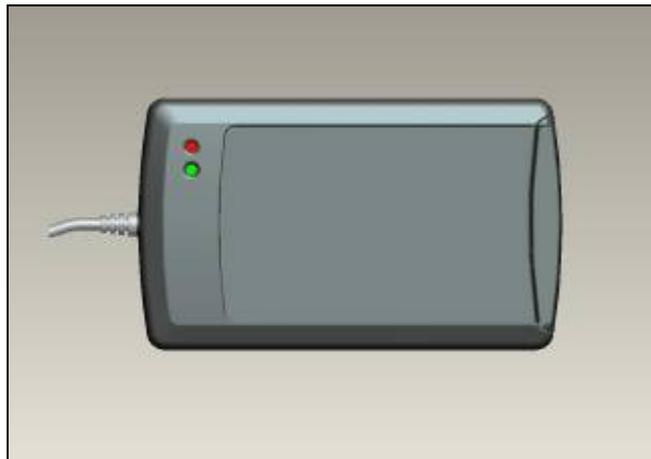


Figure 3: LED of ACR128



Reader States	Red LED PICC Indicator	Green LED ICC Indicator
1. No PICC is found	A single pulse per ~ 10 seconds	
2. PICC is present but not activated	Toggling ~ 0.3 Hz	
3. PICC is present and activated	ON	
4. PICC is operating	Blinking	
5. ICC is present and activated		ON
6. ICC is absent or not activated		OFF
7. ICC is operating		Blinking

Table 1: LED Indicator

CMD	Description	Description
Bit 0	RED LED	1 = ON; 0 = OFF
Bit 1	GREEN LED	1 = ON; 0 = OFF
Bit 2	RFU	RFU
Bit 3	RFU	RFU
Bit 4	RFU	RFU
Bit 5	RFU	RFU
Bit 6	RFU	RFU
Bit 7	RFU	RFU

Table 2: LED Control

To set the LED state of the device, issue the following command:

ACR128 Escape Command

Set LED State	29	01	CMD
---------------	----	----	-----

To read the current LED state of the device, issue the following command:

ACR128 Escape Command

Read LED State	29	00
----------------	----	----

Response

Response Data	E1	00	00	00	01	Status
---------------	----	----	----	----	----	--------

Use Tables 1 and 2 to format and interpret CMD and Status values.



4.3. Buzzer

A monotone buzzer is used to show the “Card Insertion” and “Card Removal” events.

Events	Buzzer
1. Card Insertion Event (ICC or PICC)	Beep
2. Card Removal Event (ICC or PICC)	Beep
3. Combi-Card (supports both ICC and PICC interfaces) is inserted in the contact card acceptor	2 Beeps
4. PICC is activated	1 beep per second (Default = Disabled)
5. PICC is activated (PPS Mode is activated). E.g. 424kps High Speed Mode	2 beeps per second (Default = Disabled)

Table 3: Buzzer Event

To set the Buzzer duration of the device, issue the following command:

ACR128 Escape Command

Set Buzzer Duration	28	01	Duration [Unit: 10 mS]
---------------------	----	----	------------------------

Value	Description
00	Turn Off
01 - FE	Buzzer duration x 10 mS
FF	Turn On

Table 4: Buzzer Duration value

**This command can be issued once the buzzer has died down so the response means that the buzzer state is OFF.*



CMD	MODE	Description
Bit 0	ICC Activation Status LED	To show the activation status of the ICC interface. 1 = Enable; 0 =Disable
Bit 1	PICC Polling Status LED	To show the PICC Polling Status. 1 = Enable; 0 =Disable
Bit 2	PICC Activation Status Buzzer	To make a beep per second to indicate that the PICC is activated. 1 = Enable; 0 =Disable
Bit 3	PICC PPS Status Buzzer #PICC Activation Status Buzzer must be enabled.	To make 2 beeps per second to indicate that the PICC PPS Mode is activated. 1 = Enable; 0 =Disable
Bit 4	Card Insertion and Removal Events Buzzer	To make a beep whenever a card insertion or removal event is detected. (For both ICC and PICC) 1 = Enable; 0 =Disabled
Bit 5	RC531 Reset Indication Buzzer	To make a beep when the RC531 is reset. 1 = Enable; 0 =Disabled
Bit 6	Exclusive Mode Status Buzzer. #Either ICC or PICC interface can be activated.	To make a beep when the exclusive mode is activated. 1 = Enable; 0 =Disable
Bit 7	Card Operation Blinking LED	To make the LED blink whenever the card (PICC or ICC) is being accessed.

Table 5: Default LED and Buzzer Behaviors

To set the LED and Buzzer behavior of the device, issue the following command:

ACR128 Escape Command

Set LED and Buzzer behavior	21	01	CMD
-----------------------------	----	----	-----

To read the current LED and Buzzer behavior of the device, issue the following command:

ACR128 Escape Command

Read LED and Buzzer behavior	21	00
------------------------------	----	----

Response

Response Data	E1	00	00	00	01	Status
---------------	----	----	----	----	----	--------

Use Table 5 to format and interpret CMD and Status values.

Note: The default CMD value is F3h. If you want a silent environment, just set the CMD value to 83h.



4.4. USB Interface

The ACR128U is connected to a computer through USB interface as specified in the USB Specification 2.0. The ACR128U is working in low speed mode, i.e. 12 Mbps.

Pin	Signal	Function
1	VBUS	+5V power supply for the reader (~200mA)
2	D-	Differential signal transmits data between ACR128U and PC.
3	D+	Differential signal transmits data between ACR128U and PC.
4	GND	Reference voltage level for power supply

Table 6: USB Interface Wiring

NOTE: In order for the ACR128U to function properly through USB interface, the ACS proprietary device driver has to be installed. Please refer to the Device Driver Installation Guide for more details. [VID = 0x072F; PID = 0x2100]

4.5. ICC Interface (Contact Smart Card)

A landing type Smart Card Acceptor is used for providing reliable operations. The minimum life cycle of the acceptor is about 300K times of card insertion and removal.

4.6. SAM Interface (Contact Smart Card)

One SAM socket is provided for high-security application requirement.

4.7. PICC Interface (Contactless Smart Card)

A built-in antenna is used for communication between the PCD and PICC.



5.0. PICC Interface Description

5.1. ATR Generation

If the reader detects a PICC, an ATR will be sent to the PCSC driver for identifying the PICC.

5.1.1. ATR format for ISO 14443 Part 3 PICCs.

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

Table 7: ISO 14443 Part 3 ATR Format

Example:

ATR for MiFare 1K = [3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A]

ATR											
Initial Header	T0	TD1	TD2	T1	Tk	Length	RID	Standard	Card Name	RFU	TCK
3B	8F	80	01	80	4F	0C	A0 00 00 03 06	03	00 01	00 00 00 00	6A

Where:

- Length (YY)** = 0C
- RID** = A0 00 00 03 06 (PC/SC Workgroup)
- Standard (SS)** = 03 (ISO14443A, Part 3)
- Card Name (C0 ... C1)** = [00 01] (MIFare 1K)
[00 02] (Mifare 4K)
[00 03] (Mifare Ultralight)
FF [SAK] (Undefined)
[FF 0] (Mifare Mini)



5.1.2. ATR format for ISO 14443 Part 4 PICCs.

Byte	Value (Hex)	Designation	Description
0	3B	Initial Header	
1	8N	T0	Higher nibble 8 means there are no TA1, TB1 and TC1. Only TD1 follows. Lower nibble N is the number of historical bytes (HistByte 0 to HistByte N-1)
2	80	TD1	Higher nibble 8 means there are no TA2, TB2 and TC2. Only TD2 follows. Lower nibble 0 means T = 0
3	01	TD2	Higher nibble 0 means no TA3, TB3, TC3 and TD3 follow. Lower nibble 1 means T = 1
4 to 3 + N	XX XX XX XX	T1 Tk	Historical Bytes: ISO14443A: The historical bytes from ATS response. Refer to the ISO14443-4 specification. ISO14443B: The higher layer response from the ATTRIB response (ATQB). Refer to the ISO14443-3 specification.
4+N	UU	TCK	Exclusive-ORing of all the bytes T0 to Tk

Table 8: ISO 14443 Part 4 ATR Format

Example 1. Consider the ATR from DESFire as follows:

DESFire (ATR) = **3B 86 80 01 06 75 77 81 02 80 00**

ATR						
Initial Header	T0	TD1	TD2	ATS		TCK
				T1	Tk	
3B	86	80	01	06	75 77 81 02 80	00

This ATR has 6 bytes of ATS which is: [06 75 77 81 02 80]

NOTE: Use the APDU "FF CA 01 00 00" to distinguish the ISO14443A-4 and ISO14443B-4 PICCs and retrieve the full ATS if available. The ATS is returned for ISO14443A-3 or ISO14443B-3/4 PICCs.

Example 2. Consider the ATR from ST19XRC8E, which is as follows:

ST19XRC8E (ATR) = **3B 8C 80 01 50 12 23 45 56 12 53 54 4E 33 81 C3 55**

ATR						
Initial Header	T0	TD1	TD2	ATS		TCK
				T1	Tk	
3B	86	80	01	50	12 23 45 56 12 53 54 4E 33 81 C3	55



Since this card is compliant to ISO 14443 Type B, the response would be ATQB and it is 12 bytes long with no CRC-B.

Note: You can refer to the ISO7816, ISO14443 and PCSC standards for more details.

5.2. ICC and PICC Interfaces Conflict Handling

There are three different card interfaces available for ACR128 – one contact card interface (ICC), one contactless card interface (PICC) and one SAM card interface (SAM). Basically, all interfaces can operate at the same time.

For example, if an ICC is inserted into the contact card acceptor, the ACR128U ICC interface will be used to access the ICC. At the same time, the ACR128U PICC interface is available for PICC access.



Figure 4: PICC and ICC Conflict Handling

5.2.1. Reader Interface Usage

Case 1: If a normal PICC is inserted into the contact card acceptor, the ACR128U PICC interface will be used.

Case 2: If a Combi-Card, that supports both ICC and PICC interfaces, is inserted into the contact card acceptor, the ACR128U ICC interface will be used while the ACR128U PICC interface will be disabled. In such case, the Auto PCSC Polling Function for PICCs will be disabled.

Case 3: If a Hybrid card that consists of both ICC and PICC cards is inserted into the contact card acceptor, both the ACR128U ICC and PICC interfaces can be used to access the Hybrid card.

5.2.2. Exclusive Mode Setting

It may take some time for the reader to determine if a Combi-Card is inserted. To minimize the card detection time, we can enable the “Enforce ICC & PICC Exclusive Mode” setting.

To enforce ICC and PICC Exclusive Mode, issue the following command:



ACR128 Escape Command

Enforce ICC & PICC Exclusive Mode	2B	01	New Mode Configuration
-----------------------------------	----	----	------------------------

Mode	Description
00	Both ICC & PICC interfaces can be activated at the same time
01	Either the ICC or PICC interface can be activated at any given time but not both (default setting)

Table 9: Mode Configuration Setting

To read the current mode, issue the following command:

ACR128 Escape Command

Read Current Configuration Mode	2B	00
---------------------------------	----	----

Response

Response Data	E1	00	00	00	02	Mode Configuration	Current Mode
---------------	----	----	----	----	----	--------------------	--------------

Mode	Description
00	Exclusive Mode is not activated. PICC Interface is available
01	Exclusive Mode is activated now. PICC Interface is not available until the ICC interface is deactivated

Table 10: Current Mode Configuration Values

Note: Do not insert any card into the contact card acceptor while the PICC is activated, or the PICC may be deselected.



5.3. Automatic PICC Polling

Whenever the reader is connected to the PC, the PICC polling function will start the PICC scanning to determine if a PICC is placed on or removed within the range of the built-in antenna.

CMD	Description	Description
Bit 0	Auto PICC Polling	1 = Enable; 0 =Disable
Bit 1	Turn off Antenna Field if no PICC is found	1 = Enable; 0 =Disable
Bit 2	Turn off Antenna Field if the PICC is inactive.	1 = Enable; 0 =Disable
Bit 3	Activate the PICC when detected.	1 = Enable; 0 =Disable
Bit 5 .. 4	PICC Poll Interval for PICC	<Bit 5 – Bit 4> <0 – 0> = 250 msec <0 – 1> = 500 msec <1 – 0> = 1 sec <1 – 1> = 2.5 sec
Bit 6	Test Mode	1= Enable; 0= Disable (default)
Bit 7	Enforce ISO14443A Part 4	1= Enable; 0= Disable.

Table 11: Register 0x23 –Automatic PICC Polling (Default value = 0x97 or 0x99 or 9F)

The PICC polling function can be disabled by sending a command to the device through the PCSC Escape command sequence. To meet the energy saving requirement, special modes are provided for turning off the antenna field whenever the PICC is inactive, or no PICC is found. The reader will consume less current in this power saving mode.

To enable the Auto PICC Polling function, issue the following command:

ACR128 Escape Command

Enable Auto PICC Polling	23	01	9F
--------------------------	----	----	----

To disable the Auto PICC Polling function, issue the following command:

ACR128 Escape Command

Disable Auto PICC Polling	23	01	9E
---------------------------	----	----	----

To read the existing polling status, issue the following command:

ACR128 Escape Command

Read Existing Polling Status	23	00
------------------------------	----	----

Response

Response Data	E1	00	00	00	01	Status
---------------	----	----	----	----	----	--------

NOTE:

1. It is recommended to enable the option “Turn Off Antenna Field if the PICC is inactive”, so that the “Inactive PICC” will not be exposed to the field all the time, therefore preventing the PICC from “warming up”.
2. The longer the PICC Poll Interval is set, the more efficient energy saving is achieved. However, the response time of PICC Polling will become longer. The Idle Current Consumption in Power Saving Mode is about 60mA, while the Idle Current Consumption in Non-Power Saving mode is about 130mA. Idle Current Consumption corresponds to the setting wherein the PICC is not activated.
3. The reader will activate the ISO14443A-4 mode of the ISO14443A-4 compliant PICC automatically. Type B PICC will not be affected by this option.
4. The JCOP30 card comes with two modes: ISO14443A-3 (MIFARE 1K) and ISO14443A-4 modes. The application has to decide which mode should be selected once the PICC is activated.



5.4. Manual PICC Polling

If automatic PICC Polling is disabled, this command can be issued to determine if any PICC is within the detection range of the reader.

To manually detect PICC within range of the built-in antenna, issue the following command:

ACR128 Escape Command

Manual PICC Polling Function	22	01	0A
------------------------------	----	----	----

Response

Response Data	E1	00	00	00	01	Status
---------------	----	----	----	----	----	--------

Status

00	PICC is detected
FF	No PICC is detected

NOTE: This feature is useful for polling the PICC with a longer time interval, e.g., 30 sec.

5.5. Change the Default FWI, Polling Timeout And Transmit Frame Size Of The Activated PICC

For some special cases, the applications may have to change the FWI and Transmit Frame Size to meet the actual requirement. The parameter POLL_TIMEOUT is used for PICC Polling.

To change the FWI, Polling Timeout and Frame Size of the activated PICC, issue the following command:

ACR128 Escape Command

Change the FWI, Polling Timeout and FRAME SIZE	1F	03	New FWI	New Polling Timeout	New Frame Size
--	----	----	---------	---------------------	----------------

Parameter	Value
FWI	0B
Polling Timeout	08
Frame Size	64

Table 12: Default Values for FWI, Polling Timeout, and Transmit Frame Size

To read the existing FWI, Polling Timeout and Frame Size of the activated PICC, issue the following command:

ACR128 Escape Command

Change the FWI & FRAME SIZE	1F	00
-----------------------------	----	----

Response

Response Data	E1	00	00	00	03	FWI	Polling Timeout	Frame Size
---------------	----	----	----	----	----	-----	-----------------	------------

NOTE: Only the activated PICC will be affected by this command. Once the PICC is removed or a new PICC is



detected, the FWI and Frame size will be adjusted to conform to the new PICC requirement but the Polling Timeout will not be changed.

5.6. Antenna Field ON/OFF

The antenna field used to detect the PICC within range can be turned on or off programmatically at any given time.

To turn on the antenna field of the device, issue the following command:

ACR128 Escape Command

Turn on Antenna	25	01	01
-----------------	----	----	----

To turn off the antenna field of the device, issue the following command:

ACR128 Escape Command

Turn off Antenna	25	01	00
------------------	----	----	----

To read the existing status of the built-in antenna, issue the following command:

ACR128 Escape Command

Read Antenna Status	25	00
---------------------	----	----

Response

Response Data	E1	00	00	00	01	Status
---------------	----	----	----	----	----	--------

Status

00	Antenna is turned off
01	Antenna if turned on

NOTE: Make sure that the Auto PICC Polling is disabled first before turning off the antenna field. To execute the manual PICC Polling, the antenna field must be enabled first.

5.7. Transceiver Setting

The Transceiver settings can be modified programmatically at any given time.

To modify the transceiver setting of the device, issue the following command:

ACR128 Escape Command

Modify Transceiver Setting	20	04	06	Antenna Setting	RX Gain	TX Mode
----------------------------	----	----	----	-----------------	---------	---------

Use Tables 13, 14 and 15 to format Antenna Setting, RX Gain, and TX Mode values.

CMD	Description
Bit7 – Bit4	Field Stop Time (Unit = 5 ms)
Bit3 – Bit 0	Setup Time (Unit = 10 ms)
33 or 12	Default Value

Table 13: Antenna Setting Values



CMD	Description
Bit7 – Bit3	RFU
Bit2	LP Filter Off
Bit1 – Bit 0	Receiver Gain
06	Default Value

Table 14: RX Gain Setting Values

CMD	Description
4B	Default Value

Table 15: TX Mode Setting Values

To read the existing transceiver setting of the device, issue the following command:

ACR128 Escape Command

Read Transceiver Setting	20	01
--------------------------	----	----

Response

Response Data	E1	00	00	00	04	06	Antenna Setting	RX Gain	TX Mode
---------------	----	----	----	----	----	----	-----------------	---------	---------

NOTE: The ANT_SETTING and RX_GAIN may have to be modified to access some non-standard PICCs.

5.8. PICC Setting

To modify the PICC setting of the device, issue the following command:

ACR128 Escape Command

Modify PICC Setting	2A	0C	Data [12 bytes]
---------------------	----	----	-----------------

Data	Description	Default Value
Byte 0	MOD_B1	08
Byte 1	COND_B1	3F
Byte 2	RX_B1	FF
Byte 3	MOD_B2	08
Byte 4	COND_B2	34
Byte 5	RX_B2	FF
Byte 6	MOD_A1	06
Byte 7	COND_A1	3F
Byte 8	RX_A1	9F
Byte 9	MOD_A2	06
Byte 10	COND_A2	05
Byte 11	RX_A2	9F

Table 16: PICC Setting Data Values



To read the existing PICC setting of the device, issue the following command:

ACR128 Escape Command

Read PICC Setting	2A	00
-------------------	----	----

Response

Response Data	E1	00	00	00	0C	Data [12 bytes]
---------------	----	----	----	----	----	-----------------

Use Tables 16 to format and interpret PICC Setting Data values.

NOTE: MOD_B1, COND_B1 ... RX_A2 may have to be modified to access some non-standard ISO14443 PICCs.

5.9. PICC Polling For Specific PICC Types

The PICC polling function can be configured to specifically detect ISO14443 Type A PICCs, ISO14443 Type B PICCs, or both types.

To configure the device to detect specific PICC within antenna range, issue the following command:

ACR128 Escape Command

Configure Device to Detect Specific PICC Type	20	02	Card Type	FF
---	----	----	-----------	----

Card Type	Description
01	ISO 14443 Type A PICCs Only
02	ISO 14443 Type B PICCs Only
03	Both ISO 14443 Type A and B PICCs

Table 17: Card Type Values to configure device for Specific PICC detection

Use Table 17 to determine the Card Type value.

To read the device signal output on the card detection process, issue the following command:

ACR128 Escape Command

Read PICC Detection Status	20	00
----------------------------	----	----

Response

Response Data	E1	00	00	00	01	Status
---------------	----	----	----	----	----	--------

Status

00	PICC is detected
FF	No PICC is detected

NOTE: It is recommended to specify the PICC types in the application so as to speed up the card detection process.

5.10. PICC T=CL Data Exchange Error Handling

To modify the Error Handling Level of T=CL protocol, issue the following command:

ACR128 Escape Command



Change Error Handling Level	2C	01	MODE
MODE	Description		
Bit5– Bit4	From PCD to PICC		
Bit1 – Bit 0	From PICC to PCD		
33	Default Value, Maximum Level		
11	Minimum Value		
00	No Error Handling		

Table 18: Error Handling Level Values

To read the existing Error Handling Level of the device, issue the following command:

ACR128 Escape Command

Read Error Handling Level	2C	00
---------------------------	----	----

Response

Response Data	E1	00	00	00	01	MODE
---------------	----	----	----	----	----	------

Use Table 18 to format and interpret the Error Handling Level Mode value.

5.11. Auto PPS (Communication Speed Change)

Whenever a PICC is recognized, the reader will try to change the communication speed between the PCD and PICC as defined by the Maximum Connection Speed. If the card does not support the proposed connection speed, the reader will try to connect to the card at a lower speed setting.

To set the maximum connection speed of the device, issue the following command:

ACR128 Escape Command

Set Maximum Connection Speed	24	01	Maximum Connection Speed
------------------------------	----	----	--------------------------

CMD	Description
00	106 kbps
01	212 kbps
02	424 kbps, Default value
03	848 kbps
FF	No Auto PPS

Table 19: Connection Speed Values

To read the existing Connection Speed Setting of the device, issue the following command:

ACR128 Escape Command

<i>Read Current Connection Speed</i>	<i>24</i>	<i>00</i>
--------------------------------------	-----------	-----------



Response

Response Data	E1	00	00	00	02	Max Conn Speed	Current Conn Speed
---------------	----	----	----	----	----	----------------	--------------------

Use Table 19 to format and interpret the Maximum and Current Connection Speed values, respectively.

NOTE: Normally, the application should know the maximum connection speed of the PICCs being used. The environment also affects the maximum achievable speed. The reader uses the proposed communication speed to communicate with the PICC. The PICC will become inaccessible if the PICC or environment does not meet the requirement of the proposed communication speed.

5.12. Read and Update the RC531 Register

To read the RC531 Register in the device, issue the following command:

ACR128 Escape Command

Read RC531 Register	19	01	Register No
---------------------	----	----	-------------

Response

Response Data	E1	00	00	00	01	Current Value
---------------	----	----	----	----	----	---------------

To update the RC531 Register in the device, issue the following command:

ACR128 Escape Command

Read RC531 Register	1A	02	Register No	New Value
---------------------	----	----	-------------	-----------

Response

Response Data	E1	00	00	00	01	Current Value
---------------	----	----	----	----	----	---------------

5.13. Refresh the Interface Status

To refresh the reader interface status in the device, issue the following command:

ACR128 Escape Command

Refresh Interface Status	2D	01	CMD
--------------------------	----	----	-----

Response

Response Data	E1	00	00	00	01	CMD
---------------	----	----	----	----	----	-----



CMD	Description
Bit0	ICC Interface
Bit1	PICC Interface
Bit2	Default Value, Maximum Level

Table 20: Reader Interface Values

Use Table 20 to format and interpret the reader interface values.

NOTE: This command is useful for refreshing the SAM status after a new SAM is inserted.

Example 1. Refresh the SAM status after a new SAM is inserted

- Step 1. Connect the "SAM Interface" in "Direct" connection mode.
- Step 2. Send the direct command "2D 01 04"
- Step 3. Disconnect the "SAM Interface"
- Step 4. Connect the "SAM Interface: again in either "Direct" or "Shared" connection mode.

Example 2. Refresh the ICC status (Reset the ICC)

- Step 1. Connect the "SAM Interface" in "Direct" or "Shared" connection mode.
- Step 2. Send the direct command "2D 01 01"

5.14. Changing the ISO 7816 Extra Guard Time

- Read the existing status = {2E 00}
- Refresh Interface = {2E 02 "Extra Guard Time for ICC" "Extra Guard Time for SAM"}
- Response = {E1 00 00 00 02 "Extra Guard Time for ICC" "Extra Guard Time for SAM"}

<Default Value>

Extra Guard Time for ICC = 1

Extra Guard Time for SAM = 1



6.0. PICC Commands for General Purposes

6.1. Get Data

The “Get Data command” will return the serial number or ATS of the connected PICC.

Command	Class	INS	P1	P2	Le
Get Data	FF	CA	00 01	00	00 (Full Length)

Table 21: Get UID APDU Format (5 Bytes)

Response	Data Out					
Result	UID (LSB)			UID (MSB)	SW1	SW2

Table 22: Get UID Response Format (UID + 2 Bytes) if P1 = 0x00

Response	Data Out		
Result	ATS	SW1	SW2

Table 23: Get ATS of an ISO 14443 A card (ATS + 2 Bytes) if P1 = 0x01

Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation failed.
Error	6A	81	Function is not supported.

Table 24: Response Codes

Examples:

- To get the serial number of the connected PICC

```
UINT8 GET_UID[5]={0xFF, 0xCA, 0x00, 0x00, 0x00};
```
- To get the ATS of the connected ISO 14443 A PICC

```
UINT8 GET_ATS[5]={0xFF, 0xCA, 0x01, 0x00, 0x00};
```

7.0. PICC Commands (T=CL Emulation) for MiFare 1K/4K MEMORY Cards

7.1. Load Authentication Keys

The “Load Authentication Keys command” will load the authentication keys into the reader. The authentication keys are used to authenticate the particular sector of the Mifare 1K/4K Memory Card. Two kinds of locations for authentication keys are provided, volatile and non-volatile.

Command	Class	INS	P1	P2	Lc	Data In
Load Authentication Keys	FF	82	Key Structure	Key Number	06	Key (6 bytes)

Table 25: Load Authentication Keys APDU Format (11 Bytes)

Key Structure (1 Byte):

- 0x00 = Key is loaded into the reader’s volatile memory.
- 0x20 = Key is loaded into the reader’s non-volatile memory.
- Other = Reserved.

Key Number (1 Byte):

- 0x00 ~ 0x1F = Non-volatile memory for storing keys. The keys are permanently stored in the reader and will not be erased even when the reader is disconnected from the PC. It can store up to 32 keys.
- 0x20 (Session Key) = Volatile memory for storing a temporary key. The key will be erased once the reader is disconnected from the PC. Only 1 volatile key is provided. The volatile key can be used as a session key for different sessions. Default Value = {FF FF FF FF FF FF}

Key (6 Bytes):

The key value loaded into the reader. E.g. {FF FF FF FF FF FF}

Response	Data Out	
Result	SW1	SW2

Table 26: Load Authentication Keys Response Format (2 Bytes)

Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation failed.

Table 27: Load Authentication Keys Response Codes



7.1.1. Authentication for MIFARE 1K/4K

The “Authentication command” uses the keys stored in the reader to do authentication with the MIFARE 1K/4K card (PICC). Two types of authentication keys are used: TYPE_A and TYPE_B.

Command	Class	INS	P1	P2	P3	Data In
Authentication	FF	88	00	Block Number	Key Type	Key Number

Table 28: Load Authentication Keys APDU Format (6 Bytes)
#Obsolete

Command	Class	INS	P1	P2	Lc	Data In
Authentication	FF	86	00	00	05	Authenticate Data Bytes

Table 29: Load Authentication Keys APDU Format (10 Bytes)

Byte1	Byte 2	Byte 3	Byte 4	Byte 5
Version 0x01	0x00	Block Number	Key Type	Key Number

Table 30: Authenticate Data Bytes (5 Byte)

Block Number: 1 Byte. This is the memory block to be authenticated.

Key Type: 1 Byte

0x60 = Key is used as a TYPE A key for authentication.

0x61 = Key is used as a TYPE B key for authentication.

Key Number: 1 Byte

0x00 ~ 0x1F = Non-volatile memory for storing keys. The keys are permanently stored in the reader and will not be erased even when the reader is disconnected from the PC. It can store up to 32 keys.

0x20 = Volatile memory for storing keys. The keys will be erased when the reader is disconnected from the PC. Only 1 volatile key is provided. The volatile key can be used as a session key for different sessions.

NOTE: For MIFARE 1K Card, it has a total of 16 sectors and each sector consists of 4 consecutive blocks. E.g. Sector 0x00 consists of Blocks {0x00, 0x01, 0x02 and 0x03}; Sector 0x01 consists of Blocks {0x04, 0x05, 0x06 and 0x07}; the last sector 0x0F consists of Blocks {0x3C, 0x3D, 0x3E and 0x3F}.

Once the authentication is done successfully, there is no need to do the authentication again provided that the blocks to be accessed belong to the same sector. Please refer to the MIFARE 1K/4K specification for more details.

Response	Data Out	
Result	SW1	SW2

Table 31: Load Authentication Keys Response Format (2 Bytes)



Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation failed.

Table 32: Load Authentication Keys Response Codes

Sectors (Total 16 sectors. Each sector consists of 4 consecutive blocks)	Data Blocks (3 blocks, 16 bytes per block)	Trailer Block (1 block, 16 bytes)	} 1K Bytes
Sector 0	0x00 ~ 0x02	0x03	
Sector 1	0x04 ~ 0x06	0x07	
..			
..			
Sector 14	0x38 ~ 0x0A	0x3B	
Sector 15	0x3C ~ 0x3E	0x3F	

Table 33: MIFARE 1K Memory Map

Sectors (Total of 32 sectors. Each sector consists of 4 consecutive blocks)	Data Blocks (3 blocks, 16 bytes per block)	Trailer Block (1 block, 16 bytes)	} 2K Bytes
Sector 0	0x00 ~ 0x02	0x03	
Sector 1	0x04 ~ 0x06	0x07	
...			
...			
Sector 30	0x78 ~ 0x7A	0x7B	
Sector 31	0x7C ~ 0x7E	0x7F	

Sectors (Total of 8 sectors. Each sector consists of 16 consecutive blocks)	Data Blocks (15 blocks, 16 bytes per block)	Trailer Block (1 block, 16 bytes)	} 2K Bytes
Sector 32	0x80 ~ 0x8E	0x8F	
Sector 33	0x90 ~ 0x9E	0x9F	
...			
...			
Sector 38	0xE0 ~ 0xEE	0xEF	
Sector 39	0xF0 ~ 0xFE	0xFF	

Table 34: MIFARE 4K Memory Map

Examples:

1. To authenticate Block 0x04 with the following characteristics: TYPE A, non-volatile, key number 0x05, from PC/SC V2.01(Obsolete).

APDU = {FF 88 00 04 60 05};

2. Similar to the previous example, if we authenticate Block 0x04 with the following characteristics:



TYPE A, non-volatile, key number 0x05, from PC/SC V2.07

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

NOTE: MIFARE Ultralight does not need authentication since it provides free access to the user data area.

7.1.2. Authentication for MIFARE Ultralight-C

Two dedicated “Authentication commands” are used to perform 3DES Authentication with the Ultralight-C tag.

Command	Class	INS	P1	P2	P3	Data In
Get Challenge	FF	00	00	00	02	1A 00

Table 35: Get Challenge APDU Command

Response	Data Out		
Get Challenge	AF + 8 bytes Challenge Data	SW1	SW2

Table 36: Get Challenge APDU Response

Command	Class	INS	P1	P2	P3	Data In
Get Challenge	FF	00	00	00	11	AF + 16 bytes 3DES Response Data

Table 37: Send 3DES Response APDU Command

Response	Data Out		
Get Challenge	00 + 8 bytes 3DES Response Data	SW1	SW2

Table 38: Send 3DES Response APDU Response

7.1.2.1. Typical 3DES Authentication sequence

1. Issue “Get Challenge APDU Command”

<< FF 00 00 00 02 1A 00

>> AF + 8 bytes Challenge Data + 90 00

2. Perform 3DES cacluation based on { 8 bytes Challenge Data + 3DES key }

3. Issue “Send 3DES Response APDU Command”

<< FF 00 00 00 11 AF + 16 bytes 3DES response data + 90 00

>> 00 + 8 bytes 3DES response data. If the authentication is success

Or

>> 63 08. If the authentication is failed.

In case the Authentication is failed, the tag will become “De-Selected”. It requires to “Poll the tag” again.



7.2. Read Binary Blocks

The Read Binary Blocks command is used for retrieving multiple data blocks from the PICC. The data block/trailer block must be authenticated first before executing the Read Binary Blocks command.

Command	Class	INS	P1	P2	Le
Read Binary Blocks	FF	B0	00	Block Number	Number of Bytes to Read

Table 39: Read Binary APDU Frmat (5 Bytes)

where:

Block Number: 1 Byte. This is the starting block.

Number of Bytes to Read: 1 Byte. The length of the bytes to be read can be a multiple of 16 bytes for MIFARE 1K/4K or a multiple of 4 bytes for MIFARE Ultralight

Maximum of 16 bytes for MIFARE Ultralight.

Maximum of 48 bytes for MIFARE 1K. (Multiple Blocks Mode; 3 consecutive blocks)

Maximum of 240 bytes for MIFARE 4K. (Multiple Blocks Mode; 15 consecutive blocks)

Example 1:

0x10 (16 bytes). The starting block only. (Single Block Mode)

Example 2:

0x40 (64 bytes). From the starting block to starting block+3. (Multiple Blocks Mode)

NOTE: For security considerations, the Multiple Block Mode is used for accessing Data Blocks only. The Trailer Block is not supposed to be accessed in Multiple Blocks Mode. Please use Single Block Mode to access the Trailer Block.

Response	Data Out		
Result	Data (Multiply of 4/16 Bytes)	SW1	SW2

Table 40: Read Binary Block Response Format (Multiply of 4/16 + 2 Bytes)

Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation failed.

Table 41: Read Binary Block Response Codes

Example 1: Read 16 bytes from the binary block 0x04 (MIFARE 1K or 4K)

APDU = {FF B0 00 04 10}

Example 2: Read 240 bytes starting from the binary block 0x80 (MIFARE 4K). Block 0x80 to Block 0x8E (15 blocks)

APDU = {FF B0 00 80 F0}



7.3. Update Binary Blocks

The Update Binary Blocks command is used for writing multiple data blocks into the PICC. The data block/trailer block must be authenticated first before executing the Update Binary Blocks command.

Command	Class	INS	P1	P2	Lc	Data In
Update Binary Blocks	FF	D6	00	Block Number	Number of Bytes to Update	Block Data (Multiple of 16 Bytes)

Table 42: Update Binary APDU Format (Multiple of 16 + 5 Bytes)

where:

Block Number: 1 Byte. This is the starting block to be updated.

Number of Bytes to Update: 1 Byte. The number of bytes to be updated can be multiple of 16 bytes for MIFARE 1K/4K or multiple of 4 bytes for MIFARE Ultralight.

Maximum of 48 bytes for MIFARE 1K. (Multiple Blocks Mode; 3 consecutive blocks)

Maximum of 240 bytes for MIFARE 4K. (Multiple Blocks Mode; 15 consecutive blocks)

Block Data (Multiple of 16 + 2 Bytes, or 6 bytes): The data to be written into the binary blocks.

Example 1: 0x10 (16 bytes). The starting block only. (Single Block Mode)

Example 2: 0x30 (48 bytes). From the starting block to starting block+2. (Multiple Blocks Mode)

NOTE: For security considerations, the Multiple Block Mode is used for accessing Data Blocks only. The Trailer Block is not supposed to be accessed in Multiple Blocks Mode. Please use Single Block Mode to access the Trailer Block.

Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation failed.

Table 43: Update Binary Block Response Codes (2 Bytes)

Examples:

- Update the binary block 0x04 of MIFARE 1K/4K with Data {00 01 .. 0F}
APDU = {FF D6 00 04 10 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F}
- Update the binary block 0x04 of MIFARE Ultralight with Data {00 01 02 03}
APDU = {FF D6 00 04 04 00 01 02 03}



7.4. Value Block Related Commands

The data block can be used as value block for implementing value-based applications.

7.4.1. Value Block Operation

The Value Block Operation command is used for manipulating value-based transactions, e.g., increment a value of the value block, etc.

Command	Class	INS	P1	P2	Lc	Data In	
Value Block Operation	FF	D7	00	Block Number	05	VB_OP	VB_Value (4 Bytes) {MSB .. LSB}

Table 44: Value Block Operation APDU Format (10 Bytes)

where:

Block Number: 1 Byte. The value block to be manipulated.

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. This command is only valid for value block.

0x02 = Decrement the value of the value block by the VB_Value. This command is only valid for value block.

VB_Value: 4 Bytes. The value of this data, which is a signed long integer (4 bytes), is used for value manipulation.

Example 1: Decimal - 4 = {0xFF, 0xFF, 0xFF, 0xFC}

VB_Value			
MSB			LSB
FF	FF	FF	FC

Example 2: Decimal 1 = {0x00, 0x00, 0x00, 0x01}

VB_Value			
MSB			LSB
00	00	00	01



Response	Data Out	
Result	SW1	SW2

Table 45: Value Block Operation Response Format (2 Bytes)

Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation failed.

Table 46: Value Block Operation Response Codes

7.4.2. Read Value Block

The Read Value Block command is used for retrieving the value from the value block. This command is only valid for value block.

Command	Class	INS	P1	P2	Le
Read Value Block	FF	B1	00	Block Number	00

Table 47: Read Value Block APDU Format (5 Bytes)

where, **Block Number** : 1 Byte. The value block to be accessed.

Response	Data Out		
Result	Value {MSB .. LSB}	SW1	SW2

Table 48: Read Value Block Response Format (4 + 2 Bytes)

where, **Value** : This is 4 Bytes long. This is the value returned from the card. The value is a signed long integer (4 bytes).

Example 1: Decimal - 4 = {0xFF, 0xFF, 0xFF, 0xFC}

Value			
MSB			LSB
FF	FF	FF	FC

Example 2: Decimal 1 = {0x00, 0x00, 0x00, 0x01}

Value			
MSB			LSB
00	00	00	01

Results	SW1	SW2	Meaning
---------	-----	-----	---------



Success	90	00	The operation is completed successfully.
Error	63	00	The operation failed.

Table 49: Read Value Block Response Codes

7.4.3. Restore Value Block

The Restore Value Block command is used to copy a value from a value block to another value block.

Command	Class	INS	P1	P2	Lc	Data In	
Value Block Operation	FF	D7	00	Source Block Number	02	03	Target Block Number

Table 50: Restore Value Block APDU Format (7 Bytes)

where:

Source Block Number: 1 Byte. The value of the source value block will be copied to the target value block.

Target Block Number: 1 Byte. This is the value block to be restored. The source and target value blocks must be in the same sector.

Response	Data Out	
Result	SW1	SW2

Table 51: Restore Value Block Response Format (2 Bytes)

Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation failed.

Table 52: Restore Value Block Response Codes

Examples:

- Store a value "1" into block 0x05
APDU = {FF D7 00 05 05 00 00 00 01}
Answer: 90 00 [\$9000]
- Read the value block 0x05
APDU = {FF B1 00 05 00}
Answer: 00 00 00 01 90 00 [\$9000]
- Copy the value from value block 0x05 to value block 0x06
APDU = {FF D7 00 05 02 03 06}
Answer: 90 00 [\$9000]
- Increment the value block 0x05 by "5"
APDU = {FF D7 00 05 05 01 00 00 00 05}
Answer: 90 00 [\$9000]



8.0. PICC Commands (T=CL Emulation) for SR176, SRIX512 and SRIX4K Memory Cards

8.1. Read Binary Blocks

The “Read Binary Blocks command” is used for retrieving a multiple of “data blocks” from the PICC.

Command	Class	INS	P1	P2	Le
Read Binary Blocks	FF	B0	00	Block Number	Number of Bytes to Read

Table 53: Read Binary APDU Format (5 Bytes)

Block Number (1 Byte):

The starting block.

Number of Bytes to Read (1 Byte):

Multiply of 4 bytes for SRIX512 or SRIX4K or multiply of 2 bytes for SR176

Maximum 32 bytes for SR176

Maximum 64 bytes for SRIX512

Maximum 252 bytes for SRIX4K

Response	Data Out		
Result	Data (Multiply of 4 Bytes) Or 2 Bytes (SR176 only)	SW1	SW2

Table 54: Read Binary Block Response Format (Multiply of 4 + 2 Bytes) or (Multiply of 2 + 2 Bytes)

Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation is failed.

Table 55: Read Binary Block Response Codes

Examples:

// Read 4 bytes from the binary block 0x07 (SR176 or SRIX512 or SRIX4K)
APDU = {FF B0 00 07 04}

// Read 252 bytes starting from the binary block 0x07 (SRIX4K)
APDU = {FF B0 00 07 FC}



8.2. Update Binary Blocks

The “Update Binary Blocks command” is used for writing a multiple of “data blocks” into the PICC.

Command	Class	INS	P1	P2	Lc	Data In
Update Binary Blocks	FF	D6	00	Block Number	Number of Bytes to Update	Block Data (Multiple of 4 Bytes) Or 2 Bytes (SR176 only)

Table 56: Update Binary APDU Format (Multiple of 4 + 5 Bytes)
or (2 + 2 Bytes)

Block Number (1 Byte):

The starting block to be updated.

Number of Bytes to Update (1 Byte):

Multiply of 4 bytes for SRIX512/SRIX4K or 2 bytes for SR176.
Maximum 2 bytes for SR176
Maximum 64 bytes for SRIX512
Maximum 252 bytes for SRIX4K

Block Data (Multiply of 4, or 2 bytes):

The data to be written into the binary block/blocks.

Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation is failed.

Table 57: Update Binary Block Response Codes (2 Bytes)

Examples:

```
// Update the binary block 0x07 of SRIX512 or SRIX4K with Data {00 01 02 03}
APDU = {FF D6 00 07 04 00 01 02 03}
// Update the binary block 0x07 of SRIX512 or SRIX4K with Data {00 01 02 03 04 05 06 07}
APDU = {FF D6 00 07 08 00 01 02 03 04 05 06 07}
// Update the binary block 0x07 of SR176 with Data {00 01}
APDU = {FF D6 00 07 02 00 01}
```

Hints: It must disable the “PICC Polling Option Bits 1 & 2” before decreasing the 32-Bit Counter. Once the “OTP Update” is completed, the “PICC Polling Option Bits 1 & 2” can be enabled again.



9.0. PICC Commands for ISO 14443-4 Compliant Cards

Basically, all ISO 14443-4 compliant cards (PICCs) can interpret the ISO 7816-4 APDUs. The ACR128U Reader has to communicate with the ISO 14443-4 compliant cards by using ISO 7816-4 APDUs and responses. ACR128U will handle the ISO 14443 Parts 1-4 protocols internally.

Command	Class	INS	P1	P2	Lc	Data In	Le
ISO 7816 Part 4 Command					Length of the Data In		Expected length of the Response Data

Table 58: ISO 7816-4 APDU Format

Response	Data Out		
Result	Response Data	SW1	SW2

Table 59: ISO 7816-4 Response Format (Data + 2 Bytes)

Results	SW1	SW2	Meaning
Success	90	00	The operation is completed successfully.
Error	63	00	The operation failed.

Table 60: Common ISO 7816-4 Response Codes

Example 1: ISO7816-4 APDU: To read 8 bytes from an ISO 14443-4 Type B PICC (ST19XR08E)

APDU = {80 B2 80 00 08}

Class = 0x80

INS = 0xB2

P1 = 0x80

P2 = 0x00

Lc = None

Data In = None

Le = 0x08

Answer: 00 01 02 03 04 05 06 07 [\$9000]

Example 2: DESFIRE ISO7816-4 APDU Wrapping. To read 8 bytes random number from an ISO 14443-4 Type A PICC (DESFIRE)

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

Class = 0x90

INS = 0x0A (DESFIRE Instruction)

P1 = 0x00

P2 = 0x00

Lc = 0x01

Data In = 0x00

Le = 0x00 (Le = 0x00 for maximum length)

Answer: 7B 18 92 9D 9A 25 05 21 [\$91AF]



The status code [91 AF] is defined in the DESFIRE specification. Please refer to the DESFIRE specification for more details.

Example 3: DESFIRE Frame Level Chaining (ISO 7816 wrapping mode). In this example, the application has to do the “Frame Level Chaining” to get the version of the DESFIRE card.

Step 1: Send an APDU {90 60 00 00 00} to get the first frame. INS=0x60

Answer: 04 01 01 00 02 18 05 91 AF [\$91AF]

Step 2: Send an APDU {90 AF 00 00 00} to get the second frame. INS=0xAF

Answer: 04 01 01 00 06 18 05 91 AF [\$91AF]

Step 3: Send an APDU {90 AF 00 00 00} to get the last frame. INS=0xAF

Answer: 04 52 5A 19 B2 1B 80 8E 36 54 4D 40 26 04 91 00 [\$9100]

Example 4: DESFIRE Native Command. We can send Native DESFire Commands to the reader without ISO7816 wrapping if we find that the Native DESFire Commands are easier to handle. To read 8 bytes random number from an ISO 14443-4 Type A PICC (DESFIRE)

APDU = {0A 00}

Answer: AF 25 9C 65 0C 87 65 1D D7[\$1DD7]

In which, the first byte “AF” is the status code returned by the DESFire Card.

The Data inside the blanket [\$1DD7] can simply be ignored by the application.

Example 5: DESFIRE Frame Level Chaining (Native Mode). In this example, the application has to do the “Frame Level Chaining” to get the version of the DESFIRE card.

Step 1: Send an APDU {60} to get the first frame. INS=0x60

Answer: AF 04 01 01 00 02 18 05[\$1805]

Step 2: Send an APDU {AF} to get the second frame. INS=0xAF

Answer: AF 04 01 01 00 06 18 05[\$1805]

Step 3: Send an APDU {AF} to get the last frame. INS=0xAF

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

NOTE: Once the DESFire Tag is activated, the first APDU sent to the DESFire Tag will determine the “Command Mode”. If the first APDU is in “Native Mode”, the rest of the APDUs must be in “Native Mode” format.



Appendix A. Simple PCSC Application Sample Code.

```
#include <windows.h>
#include <winbase.h>
#include <wincard.h>
#include <winerror.h>

void main(void)
{

    SCARDCONTEXT      hContext;
    SCARDHANDLE       hCard, hSAM;
    LONG              SCARDStatus;
    DWORD            ActiveProtocol;

    BYTE             bSendBuffer[257], rRecvBuffer[257];
    DWORD           dwSendBufferLen = 0, dwRecvBufferLen =257;
    BYTE            ReaderPICC[]="ACR128U PICC Interface";
    BYTE            ReaderSAM[]="ACR128U SAM Interface";

    SCARD_IO_REQUEST ioSendPci;

    LPSCARD_READERSTATE m_ReaderState;

    m_ReaderState.szReader          = NULL;
    m_ReaderState.pvUserData        = NULL;
    m_ReaderState.dwCurrentState    = SCARD_STATE_UNAWARE;
    m_ReaderState.dwEventState     = SCARD_STATE_CHANGED;

    // Step 1. Establish a PCSC context //
    SCARDStatus = SCardEstablishContext(SCARD_SCOPE_USER, NULL, NULL, &hContext);

    if (SCARDStatus != SCARD_S_SUCCESS)
    {
        // Error Handling
    }

    // Step 2 Connect to the SAM Interface //
    // We will use the SAM Interface for sending PCSC Direct Commands
    m_ReaderState.szReader = ReaderSAM;
    m_dwRetCode = SCardGetStatusChange(hContext, 0, &m_ReaderState, 1);

    if (m_dwRetCode == SCARD_S_SUCCESS)
    {
        if (m_ReaderState.dwEventState & SCARD_STATE_PRESENT)
        {
            // If a SAM is inserted, connect it with the parameter //
            ///SCARD_SHARE_SHARED ///
            SCARDStatus = SCardConnect( hContext, ReaderSAM,
                                       SCARD_SHARE_SHARED, SCARD_PROTOCOL_T0 |
                                       SCARD_PROTOCOL_T1, &hSAM, &ActiveProtocol);
        }
        else
        {
            // If no SAM is inserted, connect it with the parameter //
            ///SCARD_SHARE_DIRECT ///
            SCARDStatus = SCardConnect( hContext, ReaderSAM,
                                       SCARD_SHARE_DIRECT, SCARD_PROTOCOL_T0 |
                                       SCARD_PROTOCOL_T1, &hSAM, &ActiveProtocol);
        }
    }
}
```



```
// Step 3. Polling for a PICC
m_ReaderState.szReader = ReaderPICC;
m_dwRetCode = SCardGetStatusChange(hContext, 0, &m_ReaderState, 1);

if (m_dwRetCode == SCARD_S_SUCCESS)
{
    if (m_ReaderState.dwEventState & SCARD_STATE_PRESENT)
        m_bCardInserted = true;
    else
        m_bCardInserted = false;
}

// If a PICC is detected //

// Step 4 Connect to the PICC Interface //
SCARDStatus = SCardConnect( hContext, ReaderPICC, SCARD_SHARE_SHARED,
                            SCARD_PROTOCOL_T1,
                            &hCard, &ActiveProtocol);

ioSendPci.dwProtocol = ActiveProtocol;
ioSendPci.cbPciLength = sizeof(SCARD_IO_REQUEST);

// Step 5. Begin a transaction //
SCARDStatus = SCardBeginTransaction(hCard);

// Step 6. Send something to the connected PICC //
// Ask for the serial number of the PICC //
bSendBuffer[0]=0xFF;
bSendBuffer[1]=0xCA;
bSendBuffer[2]=0x00;
bSendBuffer[3]=0x00;
bSendBuffer[4]=0x00;
dwSendBufferLen=0x05

SCARDStatus = SCardTransmit( hCard, &ioSendPci, bSendBuffer, dwSendBufferLen,
                             NULL, bRecvBuffer, &dwRecvBufferLen);

// Make a beep sound by sending a PCSC Direct Command to the SAM Interface //
bSendBuffer[0]=0x28;
bSendBuffer[1]=0x01;
bSendBuffer[2]=0x80;
dwSendBufferLen=0x03

SCARDStatus = SCardControl( hSAM, SCARD_CTL_CODE(2079), bSendBuffer,
                            dwSendBufferLen, bRecvBuffer, dwRecvBufferLen,
                            &dwRecvBufferLen);

// Do something here //

// Step 7. End the transaction //
SCARDStatus = SCardEndTransaction(hCard, SCARD_LEAVE_CARD);

// Step 8. Terminate the connection //
SCARDStatus = SCardDisconnect(hCard, SCARD_LEAVE_CARD);
SCARDStatus = SCardDisconnect(hSAM, SCARD_LEAVE_CARD);

// Step 9. Release the PCSC context //
SCARDStatus = SCardReleaseContext(hContext);

...
return;

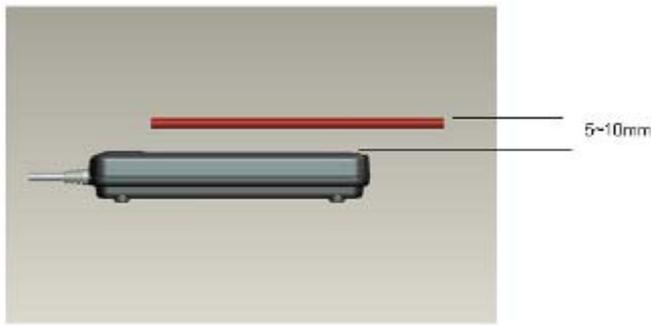
}
```

Appendix B. E-passport

1. Recommended ICAO E-Passport Placement



2. In case the E-Passport is not accessible, try to place the E-Passport by 5~10mm above the reader.



3. In case the E-Passport is still not accessible, please change operating speed to 106kbps. Set the Connection Speed to default 106k bps = {24 01 FF}.

NOTE: Please refer to Sec. 5.11 for more details on Auto PPS Direct Command.