Advanced Card Systems Ltd.
Card & Reader Technologies

ACR40T

USB SIM-sized Smart Card Reader

Reference Manual V1.01

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

Revision History

Date	Revision Description	Version Number
2022-11-14	• Initial version	1.00
2023-07-26	• Document formatting	1.01

Table of Contents

List of Figures

List of Tables

1.0. Introduction

The ACR40T USB Sim-sized smart card reader serves as an intermediary for communication between a computer and a smart card. Different smart cards have varying communication protocols and commands, making direct communication with a computer challenging. However, the ACR40T USB Sim-sized smart card reader offers a standardized interface for a broad range of smart cards, thus freeing software developers from the technical complexities of smart card operations. By handling the card's specifics, the ACR40T USB Sim-sized smart card reader allows programmers to focus on implementing the smart card system's functionality without worrying about the underlying technical details.

1.1. Reference Documents

The following related documents are available from www.usb.org

- Universal Serial Bus Specification 2.0 (also referred to as the USB specification), April 27, 2000
- Universal Serial Bus Common Class Specification 1.0, December 16, 1997
- Universal Serial Bus Device Class: Smart Card CCID Specification for Integrated Circuit(s) Cards Interface Devices, Revision 1.1, April 22, 2005

The following related documents can be ordered through www.ansi.org

- ISO/IEC 7816-1; Identification Cards – Integrated circuit(s) cards with contacts - Part 1: Physical Characteristics
- ISO/IEC 7816-2; Identification Cards – Integrated circuit(s) cards with contacts - Part 2: Dimensions and Locations of the contacts
- ISO/IEC 7816-3; Identification Cards – Integrated circuit(s) cards with contacts - Part 3: Electronic signals and transmission protocols

1.2. Symbols and Abbreviations

Abbreviation	Description
ATR	Answer-To-Reset
CCID	Chip/Smart Card Interface Device
ICC	Integrated Circuit Cards
IFSC	Information Field Sized for ICC for protocol T=1
IFSD	Information Field Sized for CCID for protocol T=1
NAD	Node Address
PPS	Protocol and Parameters Selection
RFU	Reserved for future use[1]
TPDU	Transport Protocol Data Unit
USB	Universal Serial Bus

Table 1: Symbols and Abbreviations

[1] *Must be set to zero unless stated differently.*

2.0. Smart Card Support

2.1. MCU Cards

ACR40T is a PC/SC-compliant smart card reader that supports ISO 7816 Class A, B, and C (5 V, 3 V, and 1.8 V) smart cards. It also works with MCU cards following either the T=0 and T=1 protocol.

The card ATR indicates the specific operation mode (TA2 present; bit 5 of TA2 must be 0) and when that particular mode is not supported by the ACR40T, it will reset the card to negotiable mode. If the card cannot be set to negotiable mode, the reader will then reject the card.

When the card ATR indicates the negotiable mode (TA2 not present) and communication parameters other than the default parameters, the ACR40T will execute the PPS and try to use the communication parameters that the card suggested in its ATR. If the card does not accept the PPS, the reader will use the default parameters (F=372, D=1).

For the meaning of the aforementioned parameters, please refer to ISO 7816-3.

2.2. Memory-based Smart Cards

ACR40T works with several memory-based smart cards such as:

- Cards following the I2C bus protocol (free memory cards) with maximum 128 bytes page with capability, including:
  - Atmel®: AT24C01/02/04/08/16/32/64/128/256/512/1024
- Cards with secure memory IC with password and authentication, including:
  - Atmel®: AT88SC153 and AT88SC1608
- Cards with intelligent 1 KB EEPROM with write-protect function, including:
  - Infineon®: SLE4418, SLE4428, SLE5518 and SLE5528
- Cards with intelligent 256 bytes EEPROM with write-protect function, including:
  - Infineon®: SLE4432, SLE4442, SLE5532 and SLE5542

3.0. System Block Diagram

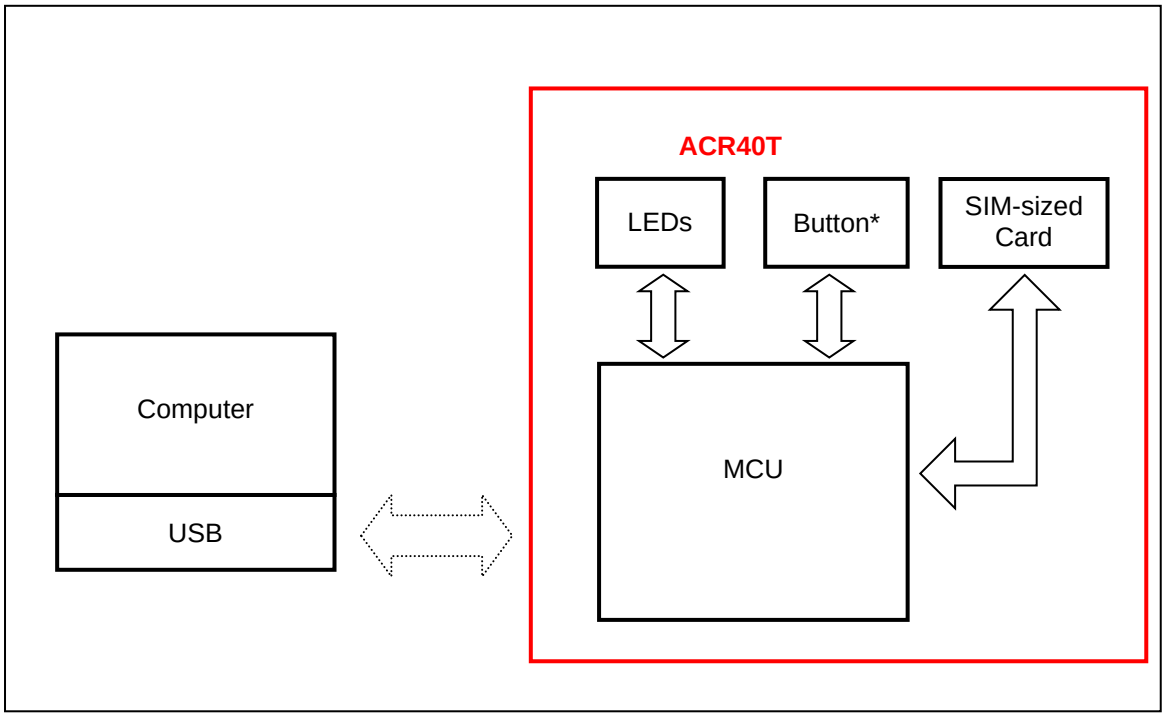

Figure 1: ACR40T Architecture

*Button is only available in model ACR40T-A6/7

4.0. USB Interface

4.1. Communication Parameters

ACR40T is connected to a computer through USB as specified in the USB Specification 2.0. ACR40T is working in full speed mode, i.e. 12 Mbps.

Pin	Signal	Function
1	V_{BUS}	+5 V power supply for the reader
2	D-	Differential signal transmits data between ACR40T and computer
3	D+	Differential signal transmits data between ACR40T and computer
4	GND	Reference voltage level for power supply

Table 2: USB Interface Wiring

4.2. Endpoints

ACR40T uses the following endpoints to communicate with the host computer:

Control Endpoint	For setup and control purpose
Bulk OUT	For command to be sent from host to ACR40T (data packet size is 64 bytes)
Bulk IN	For response to be sent from ACR40T to host (data packet size is 64 bytes)
Interrupt IN	For card status message to be sent from ACR40T to host (data packet size is 8 bytes)

5.0. User Interface

5.1. Status LED

ACR40T has three different LED behaviors to show various operation status, where:

- **Green LED -** Card and reader status under USB mode

Color	LED Activity	Status
Green	Slow flash (2 seconds/flash)	No card operation and the reader is waiting for PC instructions
	Fast flash	Data transferring between the reader and PC
	On	Card is connected and powered on

Table 3: Status LED

5.2. Configurable push button (ACR40T-A6/7 only)

ACR40T has a push button which can operate in various modes, where:

Mode	Description
0	Read button's status by sending escape command
1	Emulate card removal event
2	Disable button feature

Table 4: Operation mode of button

6.0. Smart Card Interface

The interface between the ACR40T and the inserted smart card follows the specification of ISO 7816-3 with certain restrictions or enhancements to increase the practical functionality of ACR40T.

6.1. Smart Card Power Supply VCC (C1)

The current consumption of the inserted card must not higher than 50 mA.

6.2. Programming Voltage VPP (C6)

According to ISO 7816-3, the smart card contact C6 (VPP) supplies the programming voltage to the smart card. Since all common smart cards in the market are EEPROM-based and do not require the provision of an external programming voltage, the contact C6 (VPP) has not been implemented as a normal control signal in the ACR40T.

6.3. Card Type Selection

The controlling computer must always select the card type through the proper command sent to the ACR40T prior to activating the inserted card. This includes both the memory cards and MCU-based cards.

For MCU-based cards, the reader allows to select the preferred protocol, T=0 or T=1. However, this selection is only accepted and carried out by the reader through the PPS when the card inserted in the reader supports both protocol types. Whenever an MCU-based card supports only one protocol type, T=0 or T=1, the reader automatically uses that protocol type, regardless of the protocol type selected by the application.

6.4. Interface for Microcontroller-based Cards

For microcontroller-based smart cards, only the contacts C1 (VCC), C2 (RST), C3 (CLK), C5 (GND) and C7 (I/O) are used. A frequency of 4.8 MHz is applied to the CLK signal (C3).

7.0. USB Communication Protocol

ACR40T shall interface with the host through the USB connection. A specification, namely CCID, has been released within the industry defining such a protocol for the USB chip-card interface devices. CCID covers all the protocols required for operating smart cards.

The configurations and usage of USB endpoints on ACR40T shall follow CCID Rev 1.0 Section 3.

An overview is summarized below:

1. *Control Commands* are sent on control pipe (default pipe). These include class-specific requests and USB standard requests. Commands that are sent on the default pipe report information back to the host on the default pipe.
2. *CCID Events* are sent on the interrupt pipe.
3. *CCID Commands* are sent on BULK-OUT endpoint. Each command sent to ACR40T has an associated ending response. Some commands can also have intermediate responses.
4. *CCID Responses* are sent on BULK-IN endpoint. All commands sent to ACR40T have to be sent synchronously (e.g., *bMaxCCIDBusySlots* is equal to 01h for ACR40T).

The ACR40T supported CCID features are indicated in its Class Descriptor:

Offset	Field	Size	Value	Description
0	*bLength*	1	-	Size of this descriptor, in bytes
1	*bDescriptorType*	1	-	CCID Functional Descriptor type
2	*bcdCCID*	2	-	CCID Specification Release Number in Binary-coded decimal
4	*bMaxSlotIndex*	1	-	One slot is available on ACR40T
5	*bVoltageSupport*	1	-	ACR40T can supply 1.8 V, 3 V, and 5 V to its slot
6	*dwProtocols*	4	-	ACR40T supports T=0 and T=1 protocol
10	*dwDefaultClock*	4	-	Default ICC clock frequency is 4.8 MHz
14	*dwMaximumClock*	4	-	Maximum supported ICC clock frequency is 4.8 MHz
18	*bNumClockSupported*	1	-	Does not support manual setting of clock frequency
19	*dwDataRate*	4	-	Default ICC I/O data rate is 12903 bps
23	*dwMaxDataRate*	4	-	Maximum supported ICC I/O data rate is 600 Kbps
27	*bNumDataRatesSupported*	1	-	Does not support manual setting of data rates
28	*dwMaxIFSD*	4	-	Maximum IFSD supported by ACR40T for protocol T=1 is 247
32	*dwSynchProtocols*	4	-	ACR40T does not support synchronous card
36	*dwMechanical*	4	-	ACR40T does not support special mechanical characteristics

Offset	Field	Size	Value	Description
40	*dwFeatures*	4	-	ACR40T supports the following features: • Automatic ICC clock frequency change according to parameters • Automatic baud rate change according to frequency and FI,DI parameters • TPDU level change with ACR40T
44	*dwMaxCCIDMessageLength*	4	-	Maximum message length accepted by ACR40T is 271 bytes
48	*bClassGetResponse*	1	-	Insignificant for TPDU level exchanges
49	*bClassEnvelope*	1	-	Insignificant for TPDU level exchanges
50	*wLCDLayout*	2	-	No LCD
52	*bPINSupport*	1	-	With PIN Verification
53	*bMaxCCIDBusySlots*	1	-	Only 1 slot can be simultaneously busy

7.1. CCID Bulk-OUT Messages

7.1.1. PC_to_RDR_IccPowerOn

This command activates the card slot and returns ATR from the card.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	**62h**	-
1	*dwLength*	4	00000000h	Size of extra bytes of this message
5	*bSlot*	1	-	Identifies the slot number for this command
6	*bSeq*	1	-	Sequence number for command
7	*bPowerSelect*	1	-	Voltage that is applied to the ICC: 00h = Automatic Voltage Selection 01h = 5 V 02h = 3 V 03h = 1.8V
8	*abRFU*	2	-	Reserved for future use

The response to this command message is *RDR_to_PC_DataBlock* response message and the data returned is the Answer-to-Reset (ATR) data.

7.1.2. PC_to_RDR_IccPowerOff

This command deactivates the card slot.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	**63h**	-
1	*dwLength*	4	00000000h	Size of extra bytes of this message
5	*bSlot*	1	-	Identifies the slot number for this command

Offset	Field	Size	Value	Description
6	*bSeq*	1	-	Sequence number for command
7	*abRFU*	3	-	Reserved for future use

The response to this message is the *RDR_to_PC_SlotStatus* message.

7.1.3. PC_to_RDR_GetSlotStatus

This command gets the current status of the slot.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	**65h**	-
1	*dwLength*	4	00000000h	Size of extra bytes of this message
5	*bSlot*	1	-	Identifies the slot number for this command
6	*bSeq*	1	-	Sequence number for command
7	*abRFU*	3	-	Reserved for future use

The response to this message is the *RDR_to_PC_SlotStatus* message.

7.1.4. PC_to_RDR_XfrBlock

This command transfers data block to the ICC.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	**6Fh**	-
1	*dwLength*	4	-	Size of *abData* field of this message.
5	*bSlot*	1	-	Identifies the slot number for this command.
6	*bSeq*	1	-	Sequence number for command.
7	*bBWI*	1	-	Used to extend the CCIDs Block Waiting Timeout for this current transfer. The CCID will timeout the block after “this number multiplied by the Block Waiting Time” has expired.

Offset	Field	Size	Value	Description
8	*wLevelParameter*	2	-	Short APDU level, RFU = 0000h Extended APDU level: 0000h – the command APDU begins and ends with this command. 0001h – the command APDU begins with this command, and continues in the next PC_to_RDR_XfrBlock. 0002h – the abData field continues a command APDU and ends the APDU command. 0003h – the abData field continues a command APDU and another block is to follow. 0010h – empty abData field, continuation of response APDU is expected in the next RDR_to_PC_DataBlock.
10	*abData*	Byte array	-	Data block sent to the CCID.

The response to this message is the *RDR_to_PC_DataBlock* message.

7.1.5. PC_to_RDR_GetParameters

This command gets the slot parameters.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	**6Ch**	-
1	*DwLength*	4	00000000h	Size of extra bytes of this message
5	*BSlot*	1	-	Identifies the slot number for this command
6	*BSeq*	1	-	Sequence number for command
7	*AbRFU*	3	-	Reserved for future use

The response to this message is the *RDR_to_PC_Parameters* message.

7.1.6. PC_to_RDR_ResetParameters

This command resets slot parameters to its default value.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	**6Dh**	-
1	*DwLength*	4	00000000h	Size of extra bytes of this message
5	*BSlot*	1	-	Identifies the slot number for this command
6	*BSeq*	1	-	Sequence number for command
7	*AbRFU*	3	-	Reserved for future use

The response to this message is the *RDR_to_PC_Parameters* message.

7.1.7. PC_to_RDR_SetParameters

This command sets slot parameters.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	**61h**	-
1	*dwLength*	4	-	Size of extra bytes of this message
5	*bSlot*	1	-	Identifies the slot number for this command
6	*bSeq*	1	-	Sequence number for command
7	*bProtocolNum*	1	-	Specifies what protocol data structure follows: 00h = Structure for protocol T=0 01h = Structure for protocol T=1 The following values are reserved for future use: 80h = Structure for 2-wire protocol 81h = Structure for 3-wire protocol 82h = Structure for I2C protocol
8	*abRFU*	2	-	Reserved for future use
10	*abProtocolDataStructure*	Byte array	-	Protocol Data Structure

Protocol Data Structure for Protocol T=0 (*dwLength*=00000005h)

Offset	Field	Size	Value	Description
10	*bmFindexDindex*	1	-	B7-4 – FI – Index into the table 7 in ISO/IEC 7816-3:1997 selecting a clock rate conversion factor. B3-0 – DI – Index into the table 8 in ISO/IEC 7816-3:1997 selecting a baud rate conversion factor.
11	*bmTCCKST0*	1	-	B0 – 0b, B7-2 – 000000b B1 – Convention used (b1=0 for direct, b1=1 for inverse) ***Note:*** *The CCID ignores this bit.*
12	*bGuardTimeT0*	1	-	Extra Guardtime between two characters. Add 0 to 254 etu to the normal guardtime of 12 etu. FFh is the same as 00h.
13	*bWaitingIntegerT0*	1	-	WI for T=0 used to define WWT
14	*bClockStop*	1	-	ICC Clock Stop Support 00h = Stopping the Clock is not allowed 01h = Stop with Clock signal Low 02h = Stop with Clock signal High 03h = Stop with Clock either High or Low

Protocol Data Structure for Protocol T=1 (dwLength=00000007h)

Offset	Field	Size	Value	Description
10	*bmFindexDindex*	1	-	B7-4 – FI – Index into the table 7 in ISO/IEC 7816-3:1997 selecting a clock rate conversion factor. B3-0 – DI – Index into the table 8 in ISO/IEC 7816-3:1997 selecting a baud rate conversion factor.
11	*BmTCCKST1*	1	-	B7-2 – 000100b B0 – Checksum type (b0=0 for LRC, b0=1 for CRC) B1 – Convention used (b1=0 for direct, b1=1 for inverse) ***Note:*** *The CCID ignores this bit.*
12	*BGuardTimeT1*	1	-	Extra Guardtime (0 to 254 etu between two characters). If value is FFh, then guardtime is reduced by 1 etu.
13	*BwaitingIntegerT1*	1	-	B7-4 = BWI values 0-9 valid B3-0 = CWI values 0-Fh valid

Offset	Field	Size	Value	Description
14	*bClockStop*	1	-	ICC Clock Stop Support 00h = Stopping the Clock is not allowed 01h = Stop with Clock signal Low 02h = Stop with Clock signal High 03h = Stop with Clock either High or Low
15	*bIFSC*	1	-	Size of negotiated IFSC
16	*bNadValue*	1	00h	Only support NAD = 00h

The response to this message is the *RDR_to_PC_Parameters* message.

7.2. CCID Bulk-IN Messages

7.2.1. RDR_to_PC_DataBlock

This message is sent by ACR40T in response to *PC_to_RDR_IccPowerOn*, and *PC_to_RDR_XfrBlock* messages.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	80h	Indicates that a data block is being sent from the CCID
1	*dwLength*	4	-	Size of extra bytes of this message
5	*bSlot*	1	-	Same value as in Bulk-OUT message
6	*bSeq*	1	-	Same value as in Bulk-OUT message
7	*bStatus*	1	-	Slot status register as defined in CCID Rev 1.1 Section 6.2.1
8	*bError*	1	-	Slot error register as defined in Error! Reference source not found. and in CCID Rev 1.1 Section 6.2.6
9	*bChainParameter*	1	-	Short APDU level, RFU = 00h Extended APDU level: 00h – the response APDU begins and ends in this command. 01h – the response APDU begins with this command, and is to continue. 02h – this abData field continues the response APDU and ends the response APDU. 03h – this abData field continues the response APDU and another block is to follow. 10h – empty abData field, continuation of command APDU is expected in the next PC_to_RDR_XfrBlock command.
10	*abData*	Byte array	-	This field contains the data returned by the CCID

7.2.2. RDR_to_PC_SlotStatus

This message is sent by ACR40T in response to *PC_to_RDR_IccPowerOff*, and *PC_to_RDR_GetSlotStatus* messages.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	81h	-
1	*dwLength*	4	00000000h	Size of extra bytes of this message
5	*bSlot*	1	-	Same value as in Bulk-OUT message
6	*bSeq*	1	-	Same value as in Bulk-OUT message
7	*bStatus*	1	-	Slot status register as defined in CCID Rev 1.0 Section 4.2.1
8	*bError*	1	-	Slot error register as defined in Error! Reference source not found. and in CCID Rev 1.0 Section 4.2.1
9	*bClockStatus*	1	-	Value: 00h = Clock running 01h = Clock stopped in state L 02h = Clock stopped in state H 03h = Clock stopped in an unknown state All other values are RFU

7.2.3. RDR_to_PC_Parameters

This message is sent by ACR40T in response to *PC_to_RDR_GetParameters*, *PC_to_RDR_ResetParameters* and *PC_to_RDR_SetParameters* messages.

Offset	Field	Size	Value	Description
0	*bMessageType*	1	82h	-
1	*dwLength*	4	-	Size of extra bytes of this message
5	*bSlot*	1	-	Same value as in Bulk-OUT message
6	*bSeq*	1	-	Same value as in Bulk-OUT message
7	*bStatus*	1	-	Slot status register as defined in CCID Rev 1.0 Section 4.2.1
8	*bError*	1	-	Slot error register as defined in Error! Reference source not found. and in CCID Rev 1.0 Section 4.2.1

Offset	Field	Size	Value	Description
9	*bProtocolNum*	1	-	Specifies what protocol data structure follows: 00h = Structure for protocol T=0 01h = Structure for protocol T=1 The following values are reserved for future use: 80h = Structure for 2-wire protocol 81h = Structure for 3-wire protocol 82h = Structure for I2C protocol
10	*abProtocolDataStructure*	Byte array	-	Protocol Data Structure as summarized in CCID Rev 1.0 Section 5.2.3

8.0. Host Programming API

8.1. Peripherals Control

The reader's peripherals control commands are implemented by using Escape Command (0x6B) in PC_to_RDR_Escape in USB mode.

8.1.1. Get Firmware Version Command

The Get Firmware Version command is used to get the reader's firmware version.

Get Firmware Version Format (5 Bytes)

Command	Class	INS	P1	P2	Lc
Get Firmware Version	E0h	00h	00h	19h	00h

Get Firmware Version Response Format (5 Bytes + Firmware Message Length)

Response	Class	INS	P1	P2	Le	Data Out
Result	E1h	00h	00h	00h	Number of Bytes to Received	Firmware Version

Example: Response = E1h 00h 00h 00h 0Bh 41h 43h 52h 34h 30h 54h 2Dh 50h 31h 30h 30h

Firmware Version (HEX) = 41h 43h 52h 34h 30h 54h 2Dh 50h 31h 30h 30h

Firmware Version (ASCII) = "ACR40T-P100 "

8.1.2. Get Card Voltage Selection Sequence

The Get Card Voltage Selection Sequence command is used to get the card voltage power up sequnce.

Get Card Voltage Selection Sequence Format (5 Bytes)

Command	Class	INS	P1	P2	Lc
Get Card Voltage Selection Sequence	E0h	00h	00h	0Bh	00h

Get Card Voltage Selection Response Format (5 Bytes + Get Card Voltage Selection)

Response	Class	INS	P1	P2	Le	Data Out
Result	E1h	00h	00h	00h	Number of Bytes to Received	Card Voltage Power Up Sequnce

Where

Card Voltage Power Up Sequnce Card Voltage Power Up Sequnce (1 Byte)
00h = Class C => Class B => Class A
01h = Class A only
02h = Class B only
03h = Class C onlye
04h = Class A => Class B => Class C

8.1.3. Set Card Voltage Selection

The Set Card Voltage Selection Sequence command is used to set the card voltage power up sequence.

Set Card Voltage Selection Format (5 Bytes)

Command	Class	INS	P1	P2	Lc	Data In
Set Card Voltage Selection	E0h	00h	00h	0Bh	01h	Card Voltage Power Up Sequnce

Set Card Voltage Selection Response Format (5 Bytes + Get Card Voltage Selection)

Response	Class	INS	P1	P2	Le	Data Out
Result	E1h	00h	00h	00h	Number of Bytes to Received	Card Voltage Power Up Sequnce

Where

Card Voltage Power Up Sequnce Card Voltage Power Up Sequnce (1 Byte)
00h = Class C => Class B => Class A
01h = Class A only
02h = Class B only
03h = Class C only
04h = Class A => Class B => Class C

8.1.4. Write Customer Data

This command writes user customized data (maximum size of customer data: 64 bytes)

Note: *Please contact us at info@acs.com.hk or contact an Advanced Card Systems Ltd. Sales Representative for the details regarding this command.*

8.1.5. Read Customer Data

This command is used for reading Customer Data:

Note: *Please contact us at info@acs.com.hk or contact an Advanced Card Systems Ltd. Sales Representative for the details regarding this command.*

8.1.6. Change Customer PIN

This command Change Customer PIN and Read Only Pin

Note: *Please contact us at info@acs.com.hk or contact an Advanced Card Systems Ltd. Sales Representative for the details regarding this command.*

8.1.7. Select operation mode for push button

This command configurate the operation mode for the push button.

Config push button

Command	Class	INS	P1	P2	Lc	Data In
Config push button	E0h	00h	00h	E3h	01h	Mode

Config push button Response Format (5 Bytes + Current MODE + 2 Bytes Return Code)

Response	Class	INS	P1	P2	Le	Data Out	
Result	E1h	00h	00h	00h	Number of Bytes to Received	Current Mode	Return Code

Where

Current Mode Current Mode (1 Byte)

00h = Read button's status by sending escape command

01h = Press to remove the card (card removal event)

02h = Disable Button's function

FFh = Read current button operation mode

Return Code

Results	SW1 SW2	Meaning
OK	90 00h	The button mode change successfully
ERROR	67 00h	Button mode change unsuccessfully

8.1.8. Read the Status of push button

This command read the status of the push button.(For button mode 0 only)

Read the push button's status (5 Bytes)

Command	Class	INS	P1	P2	Lc
Read button status	E0h	00h	00h	E3h	02h

Read button status Response Format (5 Bytes + Button status + 2 Bytes Return Code)

Response	Class	INS	P1	P2	Le	Data Out		
Result	E1h	00h	00h	00h	Number of Bytes to Received	Button status	Current Mode	Return Code

Where:

Button status Button status(1 Byte)
00h = Not pressed
01h = Pressed

Current Mode Current Mode (1 Byte)
00h = Read button's status by sending escape command
01h = Press to remove the card (card removal event)
02h = Disable Button's function
FFh = Read current button operation mode

Return Code

Results	SW1 SW2	Meaning
OK	90 00h	Command completed successfully
ERROR	67 00h	Command Error

8.2. Memory Card Command Set

8.2.1. Memory Card – 1, 2, 4, 8, and 16 kilobit I2C Card

8.2.1.1. SELECT_CARD_TYPE

This command powers down and up the selected card inserted in the card reader and performs a card reset.

Note: *This command can only be used after the logical smart card reader communication has been established using the SCardConnect( ) API. For details of SCardConnect( ) API, please refer to PC/SC specification.*

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Card Type**
FFh	A4h	00h	00h	01h	01h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.1.2. SELECT_PAGE_SIZE

This command chooses the page size to read the smart card. The default value is 8-byte page write. It will reset to default value whenever the card is removed or the reader is powered off.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Page Size**
FFh	01h	00h	00h	01h	

Where:

Page size = 03h for 8-byte page write
= 04h for 16-byte page write
= 05h for 32-byte page write
= 06h for 64-byte page write
= 07h for 128-byte page write

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.1.3. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	INS	Byte Address		MEM_L
		MSB	LSB	
FFh	B0h			

Where:

Byte Address Memory address location of the memory card

MEM_L Length of data to be read from the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

BYTE 1	…	…	BYTE N	SW1	SW2

Where:

BYTE x Data read from memory card

SW1 SW2 = 90 00h if no error

8.2.1.4. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	Byte 1	....	....	Byte n
		MSB	LSB					
FFh	D0h							

Where:

Byte Address Memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Data to be written to the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.2. Memory Card – 32, 64, 128, 256, 512, and 1024 kilobit I2C Card

8.2.2.1. SELECT_CARD_TYPE

This command powers down and up the selected card that is inserted in the card reader and performs a card reset.

Note: *This command can only be used after the logical smart card reader communication has been established using the SCardConnect( ) API. For details of SCardConnect( ) API, please refer to PC/SC specifications.*

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Card Type**
FFh	A4h	00h	00h	01h	02h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.2.2. SELECT_PAGE_SIZE

This command chooses the page size to read the smart card. The default value is 8-byte page write. It will reset to default value whenever the card is removed or the reader is powered off.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Page size**
FFh	01h	00h	00h	01h	

Where:

Data TPDU to be sent to the card

Page size = 03h for 8-byte page write
= 04h for 16-byte page write
= 05h for 32-byte page write
= 06h for 64-byte page write
= 07h for 128-byte page write

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.2.3. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	INS	Byte Address		MEM_L
		MSB	LSB	
FFh				

Where:

INS = B0h for 32 kilobit, 64 kilobit, 128 kilobit, 256 kilobit and 512 kilobit iic card
= 1011 000*b for 1024 kilobit iic card,
where * is the MSB of the 17 bit addressing

Byte Address Memory address location of the memory card

MEM_L Length of data to be read from the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

BYTE 1	…	…	BYTE N	SW1	SW2

Where:

BYTE x Data read from memory card

SW1 SW2 = 90 00h if no error

8.2.2.4. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	Byte 1	....	....	Byte n
		MSB	LSB					
FFh								

Where:

INS = D0h for 32 kilobit, 64 kilobit, 128 kilobit, 256 kilobit, 512 kilobit iic card
= 1101 000*b for 1024 kilobit iic card,
where * is the MSB of the 17 bit addressing

Byte Address Memory address location of the memory card

MEM_L	Length of data to be written to the memory card
Byte x	Data to be written to the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.3. Memory Card – ATMEL AT88SC153

8.2.3.1. SELECT_CARD_TYPE

This command powers up and down the selected card that is inserted in the card reader and performs a card reset. It will also select the page size to be 8-byte page write.

Note: *This command can only be used after the logical smart card reader communication has been established using the SCardConnect( ) API. For details of SCardConnect( ) API, please refer to PC/SC specifications.*

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Card Type**
FFh	A4h	00h	00h	01h	03h

Response Data Format *(abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.3.2. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**
FFh		00h		

Where:

INS = B0h for reading zone 00b
= B1h for reading zone 01b
= B2h for reading zone 10b
= B3h for reading zone 11b
= B4h for reading fuse

Byte Address Memory address location of the memory card

MEM_L Length of data to be read from the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

BYTE 1	…	…	BYTE N	SW1	SW2

Where:

BYTE x Data read from memory card

SW1 SW2 = 90 00h if no error

8.2.3.3. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**	**Byte 1**	**....**	**....**	**Byte n**
FFh		00h						

Where:

INS = D0h for writing zone 00b
= D1h for writing zone 01b
= D2h for writing zone 10b
= D3h for writing zone 11b
= D4h for writing fuse

Byte Address Memory address location of the memory card

MEM_L Length of data to be written to the memory card

MEM_D Data to be written to the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.3.4. VERIFY_PASSWORD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU							
CLA	**INS**	**P1**	**P2**	**Lc**	**Pw(0)**	**Pw(1)**	**Pw(2)**
FFh	20h	00h		03h			

Where:

Pw(0),Pw(1),Pw(2) Passwords to be sent to memory card

P2 = 0000 00rp_b
where the two bits “rp” indicate the password to compare
r = 0: Write password,
r = 1: Read password,
p : Password set number,
rp = 01 for the secure code.

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2 ErrorCnt
90h	

Where:

SW1 = 90h

SW2 (ErrorCnt) = Error Counter. FFh indicates the verification is correct. 00h indicates the password is locked (or exceeded the maximum number of retries). Other values indicate the current verification has failed.

8.2.3.5. INITIALIZE_AUTHENTICATION

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	P1	P2	Lc	Q(0)	Q(1)	…	Q(7)
FFh	84h	00h	00h	08h				

Where:

Q(0),Q(1)…Q(7) Host random number, 8 bytes

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.3.6. VERIFY_AUTHENTICATION

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	P1	P2	Lc	Ch(0)	Ch(1)	…	Ch(7)
FFh	82h	00h	00h	08h				

Where:

Ch(0),Ch(1)…Ch(7) Host challenge, 8 bytes

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.4. Memory Card – ATMEL AT88C1608

8.2.4.1. SELECT_CARD_TYPE

This command powers down and up the selected card that is inserted in the card reader and performs a card reset. It will also select the page size to be 16-byte page write.

Note: *This command can only be used after the logical smart card reader communication has been established using the SCardConnect( ) API. For details of SCardConnect( ) API, please refer to PC/SC specifications.*

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Card Type**
FFh	A4h	00h	00h	01h	04h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.4.2. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**Zone Address**	**Byte Address**	**MEM_L**
FFh				

Where:

INS = B0h for reading user zone

= B1h for reading configuration zone or reading fuse

Zone Address = 0000 $0A_{10}A_9A_8$b where A_{10} is the MSB of zone address

= don't care for reading fuse

Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

= 1000 0000b for reading fuse

MEM_L Length of data to be read from the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

BYTE 1	…	…	BYTE N	SW1	SW2

Where:

BYTE x Data read from memory card

SW1 SW2 = 90 00h if no error

8.2.4.3. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	**INS**	**Zone Address**	**Byte Address**	**MEM_L**	**Byte 1**	**…**	**…**	**Byte n**
FFh								

Where:

INS = D0h for writing user zone
= D1h for writing configuration zone or writing fuse

Zone Address = 0000 $0A_{10}A_9A_8$b where A_{10} is the MSB of zone address
= Don't care for writing fuse

Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card
= 1000 0000b for writing fuse

MEM_L Length of data to be written to the memory card

Byte x Data to be written to the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.4.4. VERIFY_PASSWORD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	**INS**	**P1**	**P2**	**Lc**	**Data**			
FFh	20h	00h	00h	04h	RP	Pw(0)	Pw(1)	Pw(2)

Where:

Pw(0),Pw(1),Pw(2) Password to be sent to memory card

RP = 0000 $rp_2p_1p_0$b
where the four bits "$rp_2p_1p_0$" indicate the password to compare:
r = 0: Write password,
r = 1: Read password,
$p_2p_1p_0$: Password set number.
($rp_2p_1p_0$ = 0111 for the secure code)

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2 ErrorCnt
90h	

Where:

SW1 = 90h

SW2 (ErrorCnt) = Error Counter. FFh indicates the verification is correct. 00h indicates the password is locked (or exceeded the maximum number of retries). Other values indicate the current verification has failed.

8.2.4.5. INITIALIZE_AUTHENTICATION

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	P1	P2	Lc	Q(0)	Q(1)	…	Q(7)
FFh	84h	00h	00h	08h				

Where:

Byte Address Memory address location of the memory card

Q(0),Q(1)…Q(7) Host random number, 8 bytes

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.4.6. VERIFY_AUTHENTICATION

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	P1	P2	Lc	Q1(0)	Q1(1)	…	Q1(7)
FFh	82h	00h	00h	08h				

Where:

Byte Address Memory address location of the memory card

Q1(0),Q1(1)…Q1(7) Host challenge, 8 bytes

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.5. Memory Card – SLE4418/SLE4428/SLE5518/SLE5528

8.2.5.1. SELECT_CARD_TYPE

This command powers up and down the selected card that is inserted in the card reader and performs a card reset.

Note: *This command can only be used after the logical smart card reader communication has been established using the SCardConnect( ) API. For details of SCardConnect( ) API, please refer to PC/SC specifications.*

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Card Type**
FFh	A4h	00h	00h	01h	05h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.5.2. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**Byte Address**		**MEM_L**
		MSB	**LSB**	
FFh	B0h			

Where:

MSB Byte Address = 0000 00A_9A_8b is the memory address location of the memory card

LSB Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be read from the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

BYTE 1	…	…	BYTE N	SW1	SW2

Where:

BYTE x Data read from memory card

SW1, SW2 = 90 00h if no error

8.2.5.3. READ_PRESENTATION_ERROR_COUNTER_MEMORY_CARD (SLE4428 and SLE5528)

This command is used to read the presentation error counter for the secret code.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**P1**	**P2**	**MEM_L**
FFh	B1h	00h	00h	03h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

ERRCNT	DUMMY 1	DUMMY 2	SW1	SW2

Where:

ERRCNT Error Counter. FFh indicates that the last verification is correct. 00h indicates that the password is locked (exceeded the maximum number of retries). Other values indicate that the last verification has failed.

DUMMY Two bytes dummy data read from the card

SW1 SW2 = 90 00h if no error

8.2.5.4. READ_PROTECTION_BIT

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**Byte Address**		**MEM_L**
		MSB	**LSB**	
FFh	B2h			

Where:

MSB Byte Address = 0000 00A_9A_8b is the memory address location of the memory card

LSB Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of protection bits to be read from the card, in multiples of 8 bits. Maximum value is 32.

MEM_L = 1 + INT((number of bits - 1)/8)

For example, to read 8 protection bits starting from memory 0010h, the following pseudo-APDU should be issued:

FF B2 00 10 01h

Response Data Format (abData field in the RDR_to_PC_DataBlock)

PROT 1	…	…	PROT L	SW1	SW2

Where:

PROT y Bytes containing the protection bits

SW1, SW2 = 90 00h if no error

The arrangement of the protection bits in the PROT bytes is as follows:

PROT 1								PROT 2								…							
P8	P7	P6	P5	P4	P3	P2	P1	P16	P15	P14	P13	P12	P11	P10	P9	..	..	..	..	..	..	P18	P17

Where:

Px is the protection bit of BYTE x in the response data

'0' byte is write protected

'1' byte can be written

8.2.5.5. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	Byte 1	....	....	Byte N
		MSB	LSB					
FFh	D0h							

Where:

MSB Byte Address = 0000 00A_9A_8b is the memory address location of the memory card

LSB Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Data to be written to the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.5.6. WRITE_PROTECTION_MEMORY_CARD

Each byte specified in the command is used in the card to compare the byte stored in a specified address location. If the data match, the corresponding protection bit is irreversibly programmed to '0'.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	INS	Byte Address		MEM_L	Byte 1	....	....	Byte N
		MSB	LSB					
FFh	D1h							

Where:

MSB Byte Address = 0000 00A_9A_8b is the memory address location of the memory card

LSB Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Byte values to be compared with the data in the card starting at *Byte Address*. BYTE 1 is compared with the data at Byte Address; BYTE N is compared with the data at (Byte Address+N-1).

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.5.7. PRESENT_CODE_MEMORY_CARD (SLE4428 and SLE5528)

This command is used to submit the secret code to the memory card to enable the write operation with the SLE4428 and SLE5528 card, the following actions are executed:

1. Search a '1' bit in the presentation error counter and write the bit to '0'.
2. Present the specified code to the card.
3. Try to erase the presentation error counter.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU						
CLA	INS	P1	P2	MEM_L	CODE	
					Byte 1	Byte 2
FFh	20h	00h	00h	02h		

Where:

CODE Two bytes secret code (PIN)

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2 ErrorCnt
90h	

Where:

SW1 = 90h

SW2 (ErrorCnt) = Error Counter. FFh indicates successful verification. 00h indicates that the password is locked (or exceeded the maximum number of retries). Other values indicate that current verification has failed.

8.2.6. Memory Card – SLE4432/SLE4442/SLE5532/SLE5542

8.2.6.1. SELECT_CARD_TYPE

This command powers down and up the selected card that is inserted in the card reader and performs a card reset.

Note: *This command can only be used after the logical smart card reader communication has been established using the SCardConnect() API. For details of SCardConnect() API, please refer to PC/SC specifications.*

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU					
CLA	**INS**	**P1**	**P2**	**Lc**	**Card Type**
FFh	A4h	00h	00h	01h	06h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.6.2. READ_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**
FFh	B0h	00h		

Where:

Byte Address = $A_7A_6A_5A_4$ $A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be read from the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

BYTE 1	…	…	BYTE N	SW1	SW2

Where:

BYTE x Data read from memory card

SW1, SW2 = 90 00h if no error

8.2.6.3. READ_PRESENTATION_ERROR_COUNTER_MEMORY_CARD (SLE 4442 and SLE 5542)

This command is used to read the presentation error counter for the secret code.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**P1**	**P2**	**MEM_L**
FFh	B1h	00h	00h	04h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

ERRCNT	DUMMY 1	DUMMY 2	DUMMY 3	SW1	SW2

Where:

ERRCNT Error counter. 07h indicates that the last verification is correct. 00h indicates that the password is locked (exceeded the maximum number of retries). Other values indicate that the last verification has failed.

DUMMY Three bytes dummy data read from the card

SW1 SW2 = 90 00h if no error

8.2.6.4. READ_PROTECTION_BITS

This command is used to read the protection bits for the first 32 bytes.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU				
CLA	**INS**	**P1**	**P2**	**MEM_L**
FFh	B2h	00h	00h	04h

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

PROT 1	PROT 2	PROT 3	PROT 4	SW1	SW2

Where:

PROT y Bytes containing the protection bits from protection memory

SW1, SW2 = 90 00h if no error

The arrangement of the protection bits in the PROT bytes is as follows:

PROT 1								PROT 2								…							
P8	P7	P6	P5	P4	P3	P2	P1	P16	P15	P14	P13	P12	P11	P10	P9	..	..	..	..	..	..	P18	P17

Where:

Px is the protection bit of BYTE x in the response data

'0' byte is write protected

'1' byte can be written

8.2.6.5. WRITE_MEMORY_CARD

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**	**Byte 1**	**....**	**....**	**Byte N**
FFh	D0h	00h						

Where:

Byte Address = $A_7A_6A_5A_4\ A_3A_2A_1A_0$b is the memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Data to be written to the memory card

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.6.6. WRITE_PROTECTION_MEMORY_CARD

Each byte specified in the command is internally in the card compared with the byte stored at the specified address and if the data match, the corresponding protection bit is irreversibly programmed to '0'.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU								
CLA	**INS**	**P1**	**Byte Address**	**MEM_L**	**Byte 1**	**....**	**....**	**Byte N**
FFh	D1h	00h						

Where:

Byte Address = $000A_4\ A_3A_2A_1A_0$b (00h to 1Fh) is the protection memory address location of the memory card

MEM_L Length of data to be written to the memory card

Byte x Byte values to be compared with the data in the card starting at Byte Address. BYTE 1 is compared with the data at Byte Address; BYTE N is compared with the data at (Byte Address+N-1).

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error

8.2.6.7. PRESENT_CODE_MEMORY_CARD (SLE 4442 and SLE 5542)

To submit the secret code to the memory card to enable the write operation with the SLE 4442 and SLE 5542 card, the following actions are executed:

1. Search a '1' bit in the presentation error counter and write the bit to '0'.
2. Present the specified code to the card.
3. Try to erase the presentation error counter.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU							
CLA	INS	P1	P2	MEM_L	CODE		
					Byte 1	Byte 2	Byte 3
FFh	20h	00h	00h	03h			

Where:

CODE Three bytes secret code (PIN)

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2 ErrorCnt
90h	

Where:

SW1 = 90h

SW2 (ErrorCnt) = Error Counter. 07h indicates that the verification is correct. 00h indicates the password is locked (exceeded the maximum number of retries). Other values indicate that the current verification has failed.

8.2.6.8. CHANGE_CODE_MEMORY_CARD (SLE 4442 and SLE 5542)

This command is used to write the specified data as new secret code in the card.

The current secret code must have been presented to the card with the *PRESENT_CODE* command prior to the execution of this command.

Command Format (*abData* field in the *PC_to_RDR_XfrBlock*)

Pseudo-APDU							
CLA	INS	P1	P2	MEM_L	CODE		
					Byte 1	Byte 2	Byte 3
FFh	D2h	00h	01h	03h			

Response Data Format (*abData* field in the *RDR_to_PC_DataBlock*)

SW1	SW2

Where:

SW1 SW2 = 90 00h if no error