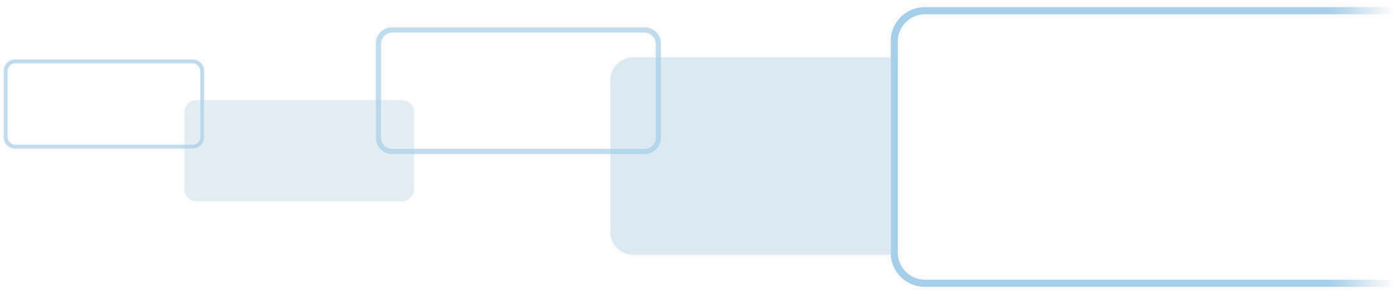
OMNIKEY® 5023
Software Developer Guide

PLT-03273, Rev. A.1

July 2018



hidglobal.com

Copyright

Trademarks

Revision history

Date	Description	Revision
July 2018	Removed support for MIFARE Ultralight/Ultralight C using PC/SC.	A.1
August 2017	Initial release.	A.O

Contacts

For additional offices around the world, see **www.hidglobal.com/contact/corporate-offices**

Americas and Corporate

611 Center Ridge Drive
Austin, TX 78753
USA
Phone: 866 607 7339
Fax: 949 732 2120

Asia Pacific

19/F 625 King's Road
North Point, Island East
Hong Kong
Phone: 852 3160 9833
Fax: 852 3160 4809

Europe, Middle East and Africa (EMEA)

Haverhill Business Park Phoenix Road
Haverhill, Suffolk CB9 7AE
England
Phone: 44 (0) 1440 711 822
Fax: 44 (0) 1440 714 840

Brazil

Condomínio Business Center
Av. Ermano Marchetti, 1435
Galpão A2 - CEP 05038-001
Lapa - São Paulo / SP
Brazil
Phone: +55 11 5514-7100

HID Global Technical Support: www.hidglobal.com/support

Contents

Section 1

1 Introduction

1.1 Product description

OMNIKEY® 5023 Readers open new market opportunities for system integrators seeking simple reader integration and development using standard interfaces, such as CCID (Circuit Card Interface Device). This reader works without needing to install or maintain drivers, eliminating complex software lifecycle management issues in the field and accelerating introduction into the market. Only an operating system driver is necessary, for example Microsoft CCID driver.

The OMNIKEY 5023 reader features include support for common high frequency card technologies, including ISO/IEC 14443 A/B, iCLASS®, MIFARE® and others.

It is also possible to add support for new card technologies in the future through device firmware upgrades.

1.2 Key features

- **CCID Support** – Removes the requirement to install drivers on standard operating systems to fully support capabilities of the reader board.
- **High frequency card technologies** – Supports common high frequency card technologies, including ISO/IEC 14443 A/B (including MIFARE/MIFARE DESFIRE etc), iCLASS 15693, Topaz, Sony FeliCa®, and others.
- **Rapid and Easy Integration** – No special driver installation is required.
- **iCLASS SE® Processor** - Provides support for processing of PACS data and secure key exchange and communication. Access of iCLASS Seos® data and HID PACS SIO across multiple technologies.

1.3 Command execution behavior

The OMNIKEY 5023 contains a Secure Element, which controls and manages the cryptographic access to the presented secure credentials. The reader's operational behavior is similar to the OMNIKEY 5427CK in CCID mode rather than the OMNIKEY 5x22 readers.

1.4 Reference documents

Document number	Description
USB 2.0 Specification	Universal Serial Bus Revision 2.0 specification provides the technical details to understand USB requirements and design USB compatible products. Refer to http://www.usb.org/developers/docs/usb20_docs/.
CCID Specification	Specification for Integrated Circuit(s) Cards Interface Devices, revision 1.1.
PC/SC	PC/SC Workgroup Specifications version 2.01.9 April 2010.
PC/SC-3	PC/SC - Part 3 - Requirements for PC Connected Interface Devices V2.01.09, June 2007.
ISO 7816-4	Information technology - Identification cards - Integrated circuit(s) cards with contacts - Part 4: Inter-industry commands for interchange Rev. 2005.
ISO/IEC 14443	Identification cards - Contactless integrated circuit cards - Proximity cards.
5023 Reader Data Sheet	Provides a summary of the OMNIKEY 5023 Reader's features.
6700-902	iCLASS SE Processor User Guide.
NIST Special Publication 800-108	Recommendation for Key Derivation Using Pseudorandom Functions.

Note: HID Global is not allowed to support proprietary card layer protocols that may be implemented in the host device/application. For example, FeliCa application developers must contact Sony and MIFARE branded products must contact NXP to obtain these card layer protocols. HID Global is constantly expanding credential support in the reader, so, some card technologies support only the chip UID.

Contact HID Global Technical Support for further information: https://www.hidglobal.com/support

1.5 Abbreviations and definitions

Abbreviation	Description
APDU	Application Protocol Data Unit.
ATR	Answer To Reset.
ATS	Answer To Select.
CCID	Integrated Circuit(s) Cards Interface Device.
CE	Conformité Européenne.
CSN	Card Serial Number.
EMD	Electromagnetic Disturbance.
FCC	Federal Communications Commission.
ICAO	International Civil Aviation Organization.
ICC	Integrated Circuit Card.
IDm	Manufacture ID (FeliCa UID).
IFD	Interface Device.
ISO/IEC	International Organization for Standardization / International Electrotechnical Commission.
OID	Unique Object Identifier.
PC/SC	Personal Computer / Smart Card.
PCD	Proximity Coupling Device.
PICC	Proximity Integrated Circuit Card.
RFU	Reserved for Future Use.
UID	Universal ID.
USB	Universal Serial Bus.
VCD	Vicinity Coupling Device.
VICC	Vicinity Integrated Circuit Card.

This page is intentionally left blank.

Section 2

2 Host interfaces

The OMNIKEY® 5023 reader supports the following host interface:

- USB 2.0 Full Speed (12 Mbit/s) Device Port

2.1 USB

The device enumerates as a single device. The OMNIKEY 5023 USB protocol stack implements the following device class:

- CCID (Integrated Circuit Cards Interface Device, v1.1)

The USB CCID interface can be used to send Application Protocol Data Unit (APDU) to the reader. The OMNIKEY 5023 supports the standard PC/SC API (for example, `SCardConnect`, `SCardDisconnect`, `SCardTransmit`). Consequently, any application software using the PC/SC API commands should be able to communicate with the reader.

This page is intentionally left blank.

Section 3

3 Contactless card interface

The OMNIKEY® 5023 reader is compliant with CCID specifications. Data exchange with a host is done via Extended APDUs. Since the CCID specification does not define contactless protocols, T=1 protocol is emulated.

3.1 Polling mode

OMNIKEY 5023 supports a single polling mode.

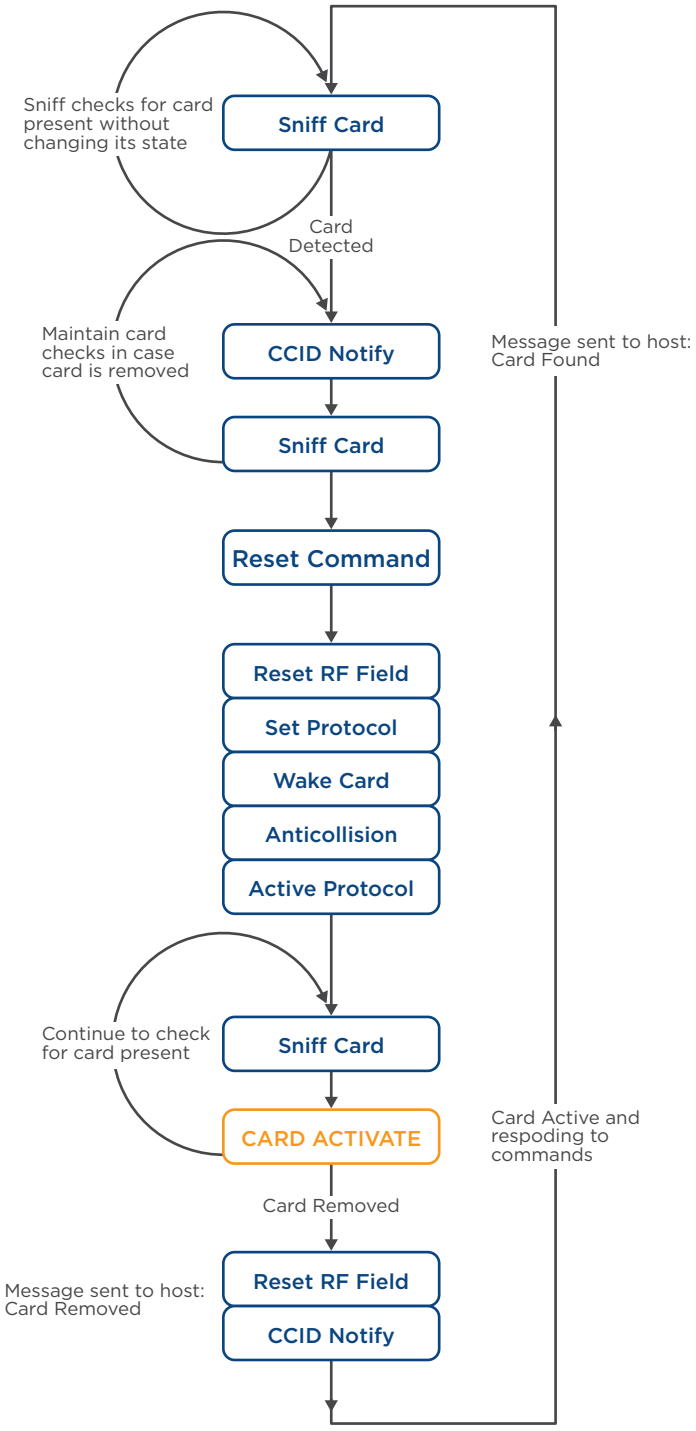
This polling mode operates as follows:

1. The reader polls for cards automatically using a set sequence of card protocols. It is possible to enable or disable each protocol individually and change the sequence.

   The factory default sequence is:

   - ISO/IEC 14443 Type A and Topaz
   - ISO/IEC 14443 Type B
   - iCLASS ISO/IEC 15693
   - FeliCa
   - ISO/IEC 15693
2. When a card or cards are found the host application is notified through CCID.
3. When the host powers up the card the relevant anti-collision procedure is executed to achieve the selection of a single card. The reader to card airspeed is set to the highest value supported by both reader and card. Where applicable the card is put into the T=CL protocol state. The card details (that is ATR) are sent to the host.
4. APDU layer communication is now possible through the CCID interface.
5. The reader continues to poll for card removal, whereupon it sends an appropriate CCID message to the host application.
6. On card removal, the cycle is repeated.

Polling operation



Section 4

4 Contactless protocol support

4.1 ISO/IEC 14443 Type A

The OMNIKEY® 5023 reader supports all ISO/IEC 14443 Type A compliant cards. Anti-collision is as described in ISO/IEC 14443-3:2001(E) section 6.4. Protocol mode when supported is T=CL as described in ISO/IEC 14443-4. ISO/IEC 14443A cards supported by the reader include, but are not limited to the following:

- MIFARE Classic
- MIFARE Plus
- MIFARE DESFire EV1
- MIFARE Ultralight, Ultralight C
- Topaz

The OMNIKEY 5023 reader allows accessing any T=CL card directly through PC/SC. MIFARE Classic and MIFARE Plus in MIFARE Classic emulation mode are supported by the PC/SC commands.

By default, the card normally is switched to the highest possible speed supported by both the reader and the card. This is as described in the card ATS, but can be limited by the configuration options as described below. The maximum speed supported by the reader is 848Kbit/s. Protocol mode will then be enabled when supported by the card.

Configurable ISO14443A parameters:

Item	Description
iso14443aEnable	Enables or disables support for ISO/IEC 14443 Type A.
iso14443aRxTxBaudRate	Sets the maximum baud rate in the PCD to PICC/PICC to PCD direction.
mifareKeyCache	Enable or disable MIFARE key caching.
mifarePreferred	Prefers MIFARE mode of a card.

4.2 ISO/IEC 14443 Type B

The OMNIKEY 5023 reader supports all ISO/IEC 14443 Type B compliant cards. Protocol activation when supported is T=CL according to ISO/IEC 14443-3:2001(E) Section 7.

The card will normally be switched to the highest possible speed supported by both the reader and the card. This is as described in the card ATS, but can be limited by the configuration options as described below. The maximum speed supported by the reader is 848 kbps. Protocol mode will then be enabled when supported by the card.

Configurable ISO14443B parameters:

Item	Description
iso14443bEnable	Enables or disables support for ISO/IEC 14443 Type B.
iso14443bRxTxBaudRate	Sets the maximum baud rate in the PCD to PICC/PICC to PCD direction.

4.3 iCLASS 15693

Access to the following iCLASS® card data is through the proprietary set of pseudo APDUs:

- iCLASS Legacy
- iCLASS Legacy Elite
- iCLASS SE®
- iCLASS Seos®

Configurable iCLASS parameters:

Item	Description
iCLASS15693Enable	Enables or disables support for iCLASS 15693 card polling.
iCLASS15693DelayTime	Sets minimum chip response to reader command delay.
iCLASS15693Timeout	Sets time to wait for response to a command.
iCLASSActallTimeout	Sets time to wait for response to ACT/ACTALL.

4.4 FeliCa

FeliCa support is limited to card selection with only one card present (no anti-collision) and IDm retrieval.

Configurable FeliCa parameters:

Item	Description
felicaEnable	Enables FeliCa card polling.
felicaRxTxBaudRate	Sets the maximum baud rate in both directions.

Section 5

5 Contactless card communication

Before communicating with a contactless card, it will be necessary to select the card and in some cases, authenticate with a known key. For a USB-connected host with an operating system the card selection is done automatically. To enhance the user experience OMNIKEY® 5023 supports so-called key caching that reduces the number of authentication calls required to access certain areas of a card that use the same key. Key caching is disabled by default.

Communication with MIFARE Classic, MIFARE Plus (SL1) and iCLASS® credentials is normally done using the PC/SC APDUs described in the next section. However, MIFARE DESFire cards are only supported using T=CL pass through commands and the user must handle all of these details of the encryption, authentication, reading writing etc., in their application code. The following sections include the PC/SC commands required to communicate with a card. Examples of communication with some specific card types are included in the next chapter.

5.1 PC/SC commands

5.1.1 Command set

The PC/SC command set for contactless cards is defined in Section 3.2 of the document "*Interoperability Specification for ICCs and Personal Computer Systems - Part 3. Requirements for PC-Connected Interface Devices*", and is available from the PC/SC Workgroup website http://www.pcscworkgroup.com. The commands use the standard APDU syntax and standard SCardTransmit API, but use the reserved value of the CLA byte of "FF".

PC/SC commands

Instruction	Description	Comments
0xB0	Read Binary.	Read data from a credential.
0xC2	Increment/Decrement.	Supported for MIFARE cards.
0xCA	Get Data.	Fully supported.
0x82	Load Keys.	Partially supported.
0x86	General Authenticate.	Supported for HID iCLASS cards and MIFARE.
0x20	Verify.	Not supported.
0xD6	Update Binary.	Write data to credential.

Common SW1SW2 return codes

SW1SW2	Definition
0x9000	Operation successful.
0x6700	Wrong length (Lc or Le).
0x6A81	Function not supported.
0x6B00	Wrong parameter (P1 or P2).
0x6F00	Operation failed.

5.1.2 0xCA – Get Data

This command is used to retrieve specific information relating to the card itself, such as card serial number, rather than data on the card itself. The items which can be retrieved are listed in the following table.

Get Data command APDU

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xCA	0x00 0x01	0x00	–	–	XX

General: Works with any type of card, unless P1 = 0x01 (see below)

Get Data command response

P1	Card type	Data Out	SW1SW2	
0x00	ISO/IEC 14443 Type A	4, 7 or 10-byte UID	0x9000	Operation successful.
	ISO/IEC 14443 Type B	4-byte PUPI		
	FeliCa	8-byte IDm		
	iCLASS 14443 Type B / 15693	8-byte CSN		
0x01	ISO/IEC 14443 Type A	n Historical bytes		
	Other	–	0x6A81	Function not supported.

Note:

- For the ISO/IEC 14443 Type A Innovision Jewel card, the data field is 7 bytes of 0x00.
- The number of historical bytes returned is limited to 15.

5.1.3 0x82 – Load Keys

This command allows the application to load keys onto the reader. That includes MIFARE keys, iCLASS keys and secures session keys. All keys except MIFARE keys must be loaded during secure session. MIFARE keys can be loaded when secure session is established or not. All keys are stored in the iCLASS SE processor. For keys other than MIFARE and iCLASS, see *Section: 5.5.1 Load Key*.

Load Keys command APDU

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0x82	Key Structure	Key Number	Key Length	Key	–

General: will work with any card type or can be sent using SCardControl().

Load Keys P1 coding (key structure)

b7	b6	b5	b4	b3	b2	b1	b0	Description
0	--		RFU	----				Card key.
1	--			----				Reader key.
--	0	---		----				Fixed to 0. Plain transmission.
--		0		----				Stored in volatile memory.
--		1		----				Stored in non-volatile memory.
---				0000				Fixed value 000.

KeyNumber: See *Section: 5.2 User key locations*

KeyLength: 6 or 8 or 16 bytes

Key: Key in plain text

Load Keys response

Data Out	SW1SW2	
–	0x9000	Operation successful.
	0x6982	Card key not supported.
	0x6983	Reader key not supported.
	0x6986	Invalid key.
	0x6988	Key number not valid.
	0x6989	Key length is not correct.

5.1.4 0x86 – General Authenticate

This command allows the user to authenticate a credential. Before using this command the correct keys must have been loaded to the relevant key slot. For iCLASS keys these keys are preloaded onto the reader, so the application must just select the correct key number for the area they are attempting to access.

General Authenticate command APDU

CLA	INS	P1	P2	Lc	Data field	Le
0xFF	0x86	0x00	0x00	0x05	Data, see below.	–

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
Version = 0x01	Address MSB	Address LSB	Key Type	Key Number

Key Types: 0x00 = PicoPass Debit Key (KD)
0x01 = PicoPass Credit Key (KC)
0x60 =MIFARE KeyA
0x61 = MIFARE KeyB

For MIFARE cards:

Address MSB = 0, Address LSB = the block number counted from 0 to [19 (MINI), 63 (1K), 127(2K) or 255(4K)].

For iCLASS the following scheme is used:

Address LSB: Page number 0 - 7

Address MSB: Book number 0 or 1, bit 0 - book number, bit 1 select flag.

Select flag 0 - authenticate without implicit select

Select flag 1 - authenticate with implicit select book page according LSB bit3:0

General Authenticate supported card addressing

Supported cards	Memory addressing
iCLASS	MSB = Book / LSB = Page.
MIFARE	Any block number in the requested sector.

Response APDU:

General Authenticate response

Data field	SW1SW2
empty	See following table.

General Authenticate return codes

Type	SW1SW2	Description
Normal	0x9000	Successful.
Warning		
Execution Error	0x6400	No Response from media (Time Out).
	0x6581	Illegal block number (out of memory space).
Checking Error	0x6700	Wrong APDU length.
	0x6982	Security status not satisfied (not authenticated).
	0x6986	Wrong key type.
	0x6988	Wrong key number.

5.1.5 0xB0 – Read Binary

The Read Binary command returns the data on a credential. For MIFARE Classic and Plus cards this requires a prior general authenticate command to succeed. For iCLASS all blocks except blocks 0-5 require the relevant page to be authenticated beforehand, but the correct book and page must be selected to avoid reading the wrong data. See *Section: 7.1 APDU commands* for an APDU command.

Read Binary command APDU

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xBO	Address MSB	Address LSB	–	–	xx

Read Binary supported cards

Supported cards	Memory addressing	Size
iCLASS	Block number	Any multiple of a block (8 bytes) less than the page size.
MIFARE 1K/4K	Block number	Any multiple of a block (16 bytes) less than the sector size.

Read Binary P1 coding for iCLASS

b7	b6	b5:0				Description
0	0	b5	b4	b3	b2:0	
		RFU	0	0	0 0 0	Read block number (P2) without SELECT.
			1	0	x x x	Read block number (P2) with SELECT book 0, page xxx.
			1	1	x x x	Read block number (P2) with SELECT book 1, page xxx.
0	1	0 0 0 0 0 0				Rread with DES decrypted.
1	0					RFU.
1	1					Read with 3-DES decrypted.

Using P1 to indicate the targeted book and page allows reading the addressed block numbers without a dedicated prior authentication command. This is only applicable for free accessible blocks e.g. block 0-2 and 5. The most significant bits 6 and 7 of P1 indicate whether the IFD is forced to either read the data in plain or to decrypt the data using DES or 3DES.

Read Binary response

Data field	SW1SW2
Media Data according Le, dependent on supported block size.	See the following table.

Note: If the media is readable then the IFD always returns the number of data bytes according to the Le value. If Le is less than block size, the data field is cut off the Le position and the return code is 6Cxx, where xx is the real block size. If Le is greater than the available block size, the IFD returns the number of available bytes and the return code 6282 (warning end of data reached before Le bytes). If the application requests a multiple of media block size in the Le field, then the IFD returns all requested bytes and the return code is 9000. This ensures a high performance particularly for media with "Read Multiple Blocks" support.

Read Binary SW1SW2 values

Type	SW1SW2	Description
Normal	0x9000	Successful.
Warning	0x6282	End of data reached before Le bytes (Le is greater than data length).
Execution Error	0x6400	No Response from media (Time Out).
	0x6F00	Unknown error.
Checking Error	0x6700	Wrong APDU length.
	0x6982	Block not authenticated (Security status not satisfied).
	0x6A82	Illegal block number (file not found).
	0x6Cxx	Wrong Le; SW2 encodes the exact number of available data bytes.

5.1.6 0xD6 – Update Binary

This command allows data to be written to a credential. For MIFARE Classic, Plus and iCLASS the relevant block must have been authenticated by a prior general authenticate command.

Update Binary command APDU

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xD6	Address MSB	Address LSB	xx	Data	–

Update Binary supported cards

Supported cards	Memory addressing	Size
iCLASS	Block number	1 block of 8 bytes.
MIFARE 1K/4K	Block number	1 block of 16 bytes.

Note: iCLASS update binary - selecting the book and page is not necessary because the write operation requires a prior authentication command. The most significant bits 6 and 7 of P1 indicate whether the IFD is forced to either write data in plain or to encryption the data using DES or 3DES.

Update Binary response

Data field	SW1SW2
empty	See the following table.

Read Binary SW1SW2 values

Type	SW1SW2	Description
Normal	0x9000	Successful.
Warning		
Execution Error	0x6400	No response from media (Time Out).
	0x6581	Not usable block number in the memory area (memory failure).
	0x6F00	Unknown error.
Checking Error	0x6700	Wrong APDU length.
	0x6982	Block not authenticated (security status not satisfied).
	0x6A82	Illegal block number (file not found).
	0x6Cxx	Wrong Lc; SW2 encodes the exact number of expected data bytes.

5.1.7 0xC2 – Increment / Decrement

This command increments or decrements the value of a designated block. This command is currently supported for MIFARE cards only. For further details, refer to *Section 5.4.2 MIFARE DESFire example*.

Increment/Decrement command APDU

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xC2	0x0	0x03	0x0B	Data	–

Where Data In is in the following format:

Value	Description
0xAx	Tag = Operation to be performed: A0 = Increment A1 = Decrement
0x09	Length to end of command = 9.
0x80	Tag = Block index.
0x01	Length of value = 1.
0xxx	Value = Index of block to be incremented or decremented (1 byte).
0x81	Tag = Value to be added or subtracted.
0x04	Length of value = 4.
xxxxxxxx	Value = Value to be added or subtracted from the source block (4 bytes, LSB first).

Increment/Decrement supported cards

Supported cards	Memory addressing	Size
MIFARE 1K/4K	Block number	Any multiple of a block (16 bytes) less than the sector size.

Increment/Decrement response

Data field	SW1SW2
empty	See the following table.

Increment/Decrement response SW1SW2 bytes

Type	SW1SW2	Description
Normal	0x9000	Successful.
Warning		
Execution Error	0x6400	No response from media (Time Out).
	0x6581	Memory failure (unsuccessful increment / decrement).
	0x6F00	Unknown error.
Checking Error	0x6700	Wrong APDU length.

Type	SW1SW2	Description
Checking Error	0x6700	Wrong APDU length.
	0x6981	Incompatible command.
	0x6982	Block not authenticated (security status not satisfied).
	0x6986	Command not allowed.
	0x6A81	Function not supported.
	0x6A82	Invalid block number.

5.2 User key locations

There are two sets of user keys; PC/SC and SE processor. The following table lists the default key slots available.

The PC/SC keys can only be used with PC/SC commands and can be changed using the PC/SC Load Keys command.

Likewise, the SE processor keys can only be used with SE processor API commands and can be changed using the `samCommandCardApiLoadKey` command. Additional keys can be created with new keys reference OIDs.

Note: MIFARE DESFIRE credentials are accessed via the iCLASS SE processor and hence refer to the `samCommandCardApiLoadKeyAPI`.

PC/SC key number		SE processor card API key reference OID	Size (bytes)	Type	Description
Decimal	Hex				
0	0x00	30 00 00	6	MIFARE	Key slot 0.
1	0x01	30 00 01	6	MIFARE	Key slot 1.
2	0x02	30 00 02	6	MIFARE	Key slot 2.
3	0x03	30 00 03	6	MIFARE	Key slot 3.
4	0x04	30 00 04	6	MIFARE	Key slot 4.
5	0x05	30 00 05	6	MIFARE	Key slot 5.
6	0x06	30 00 06	6	MIFARE	Key slot 6.
7	0x07	30 00 07	6	MIFARE	Key slot 7.
8	0x08	30 00 08	6	MIFARE	Key slot 8.
9	0x09	30 00 09	6	MIFARE	Key slot 9.
10	0x0A	30 00 0A	6	MIFARE	Key slot 10.
11	0x0B	30 00 0B	6	MIFARE	Key slot 11.
12	0x0C	30 00 0C	6	MIFARE	Key slot 12.
13	0x0D	30 00 0D	6	MIFARE	Key slot 13.
14	0x0E	30 00 0E	6	MIFARE	Key slot 14.
15	0x0F	30 00 0F	6	MIFARE	Key slot 15.
16	0x10	30 00 10	6	MIFARE	Key slot 16.

PC/SC key number		SE processor card API key reference OID	Size (bytes)	Type	Description
Decimal	Hex				
17	0x11	30 00 11	6	MIFARE	Key slot 17.
18	0x12	30 00 12	6	MIFARE	Key slot 18.
19	0x13	30 00 13	6	MIFARE	Key slot 19.
20	0x14	30 00 14	6	MIFARE	Key slot 20.
21	0x15	30 00 15	6	MIFARE	Key slot 21.
22	0x16	30 00 16	6	MIFARE	Key slot 22.
23	0x17	30 00 17	6	MIFARE	Key slot 23.
24	0x18	30 00 18	6	MIFARE	Key slot 24.
25	0x19	30 00 19	6	MIFARE	Key slot 25.
26	0x1A	30 00 1A	6	MIFARE	Key slot 26.
27	0x1B	30 00 1B	6	MIFARE	Key slot 27.
28	0x1C	30 00 1C	6	MIFARE	Key slot 28.
29	0x1D	30 00 1D	6	MIFARE	Key slot 29.
30	0x1E	30 00 1E	6	MIFARE	Key slot 30.
31	0x1F	30 00 1F	6	MIFARE	Key slot 31.
32	0x20	30 00 20	8	iCLASS	HID Encryption Key for HID Application (Read Only).
33	0x21	30 00 21	8	iCLASS	Book 0 Page 0 HID Master Key.
34	0x22	30 00 22	8	iCLASS	Book 0 Page 0 Elite Key (K_D).
35	0x23	30 00 23	8	iCLASS	Book 0 Page 0 App 2 (K_C).
36	0x24	30 00 24	8	iCLASS	Book 0 Page 1 App 1 (K_D).
37	0x25	30 00 25	8	iCLASS	Book 0 Page 1 App2 (K_C).
38	0x26	30 00 26	8	iCLASS	Book 0 Page 2 App 1 (K_D).
39	0x27	30 00 27	8	iCLASS	Book 0 Page 2 App 2 (K_C).
40	0x28	30 00 28	8	iCLASS	Book 0 Page 3 App 1 (K_D).
41	0x29	30 00 29	8	iCLASS	Book 0 Page 3 App 2 (K_C).
42	0x2A	30 00 2A	8	iCLASS	Book 0 Page 4 App 1 (K_D).
43	0x2B	30 00 2B	8	iCLASS	Book 0 Page 4 App 2 (K_C).
44	0x2C	30 00 2C	8	iCLASS	Book 0 Page 5 App 1 (K_D).
45	0x2D	30 00 2D	8	iCLASS	Book 0 Page 5 App 2 (K_C).
46	0x2E	30 00 2E	8	iCLASS	Book 0 Page 6 App 1 (K_D).
47	0x2F	30 00 2F	8	iCLASS	Book 0 Page 6 App 2 (K_C).
48	0x30	30 00 30	8	iCLASS	Book 0 Page 7 App 1 (K_D).
49	0x31	30 00 31	8	iCLASS	Book 0 Page 7 App 2 (K_C).

PC/SC key number		SE processor card API key reference OID	Size (bytes)	Type	Description
Decimal	Hex				
50	0x32	30 00 32	8	iCLASS	Transport App 1 (K_D).
51	0x33	30 00 33	8	iCLASS	Transport App 2 (K_C).
52	0x34	30 00 34	8	iCLASS	Book 1 Page 0 App 1 (K_D).

5.3 OMNIKEY specific commands

The card reader supports features outside the specified commands of PC/SC-3; see *Section: 1.4 Reference documents*. Vendor specific proprietary commands allow applications to control device specific features provided by the reader. Use of such a generic command prevents conflicts of reserved INS values used by certain card reader.

OMNIKEY specific command APDU format

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0x70	0x07	0x6B	xx	DER TLV coded PDU (Vendor Payload).	xx

The IFD supports the INS Byte 70 for vendor specific commands.

P1 and P2 constitute the vendor ID. For OMNIKEY products is the VID = 0x076B. The data field is constructed as ASN.1 objects/items.

Response for OMNIKEY specific commands

Data field	SW1SW2
DER TLV Response PDU	See ISO 7816-4

OIDs are organized as a leaf tree under an invisible root node. The following table shows the first root nodes.

Vendor payload command types

Vendor payload	Tag value (hex)	Description
readerInformationApi	0x02	Reader Information API.
response	0x1D	Response.
errorResponse	0x1E	Error Response.

The following description explains the DER TLV coded data field.

Note: The L field uses the definite form. For the definite form, the length octets consist of one or more octets, short form or long form. For the long form, the IFD uses the version with two subsequent octets.

5.3.1 Response APDU

For all commands encapsulated in generic 70h APDU, the IFD returns response frame constructed as follows:

Data field	SW1SW2
DER TLV coded response PDU	See ISO 7816-4.

Two last bytes of response frame are always the return code, SW1SW2.

In cases of an ISO 7816 violation, the return code is according to ISO 7816-4 and the data field may be empty.

In cases of positive processing or internal errors, the IFD returns SW1SW2 = 9000 and the data field is encapsulated in the response TAG (9Dh or BDh) or error response TAG (9Eh).

The response includes more than one leaf, depending on the request. Each leaf is encapsulated in the leaf tag.

5.3.2 Error response

The error response TAG caused by the firmware core is 9Eh (Class Context Specific) + (Primitive) + (1Eh). Length is 2 bytes. First byte is the cycle in which the error occurred and the second byte is the exception type.

9E 02 xx yy 90 00	
Value	**Description**
0x9E	Tag = Error Response. (0x0E) + (Class Context Specific) + (Primitive)
0x02	Len = 2.
cycle	Value byte 1: Cycle in which the error is occurred. See *Error Cycle*, below.
error	Value byte 2: Error code. See *Error Code*, below.
SW1	0x90
SW2	0x00

Error cycle

First value byte	
Cycle	**Description**
0	HID Proprietary Command APDU.
1	HID Proprietary Response APDU.
2	HID Read or Write EEPROM Structure.
3	RFU.
4	RFU.
5	RFU.

Error code

Second value byte		
Exception		**Description**
3	0x03	NOT_SUPPORTED.
4	0x04	TLV_NOT_FOUND.
5	0x05	TLV_MALFORMED.
6	0x06	ISO_EXCEPTION.
11	0x0B	PERSISTENT_TRANSACTION_ERROR.
12	0x0C	PERSISTENT_WRITE_ERROR.
13	0x0D	OUT_OF_PERSISTENT_MEMORY.
15	0x0F	PERSISTENT_MEMORY_OBJECT_NOT_FOUND.
17	0x11	INVALID_STORE_OPERATION.
19	0x13	TLV_INVALID_SETLENGTH.
20	0x14	TLV_INSUFFICIENT_BUFFER.
21	0x15	DATA_OBJECT_READONLY.
31	0x1F	APPLICATION_EXCEPTION (Destination Node ID mismatch).
42	0x2A	MEDIA_TRANSMIT_EXCEPTION (Destination Node ID mismatch).
43	0x2B	SAM_INSUFFICIENT_MSGHEADER (Secure Channel ID not allowed).
47	0x2F	TLV_INVALID_INDEX.
48	0x30	SECURITY_STATUS_NOT_SATISFIED.
49	0x31	TLV_INVALID_VALUE.
50	0x32	TLV_INVALID_TREE.
64	0x40	RANDOM_INVALID.
65	0x41	OBJECT_NOT_FOUND.

5.3.3 Reader Information API

This command group is reserved for GET and SET of reader specific information. See *Section: 7.2 Accessing configuration*.

5.4 Communication examples

In the examples below, the following color-coding is used for APDUs:

Color coding of examples

CLA	INS	P1, P2	Lc	Data	Le
Red	Orange	Green	Blue	Black	Purple

5.4.1 MIFARE Classic 1K/4K example

To read and write to a MIFARE card, first authenticate the card with the correct key, pre-loaded into the reader. The PC/SC Load Keys, General Authenticate, Read Binary and Update Binary APDUs can be used for these functions. An example APDU sequence is as follows:

Load a 6-byte MIFARE key of all FFs to key number 1:

FF 82 00 01 06 FF FF FF FF FF FF

Authenticate block 1 with key number 1:

FF 86 00 00 05 01 00 01 60 01

Read block1 (16 bytes):

FF B0 00 01 10

Write 16 bytes of data to block 2:

FF D6 00 02 10 00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF

5.4.2 MIFARE DESFire example

The example APDU sequence below shows how to read a standard data file, which is not protected by a key from a DESFire or DESFire EV1 (All values are LSB first):

Select application with AID = xx xx xx (that is the application which contains the file to be read):

Command: 90 5A 00 00 03 xx xx xx 00

Response: 91 00

Read 10 bytes of file xx (the file to be read), starting at byte 0:

Command: 90 BD 00 00 07 xx 00 00 00 0A 00 00 00

Response: xx xx xx xx xx xx xx xx xx xx 91 00

The xx bytes in the response are the data from the file.

For full details of all DESFire commands, refer to the NXP data sheets.

Example 1: Decrement MIFARE block 5 and restore to block 6 (backup)

FF C2 00 03 0E

```
A1 0C                 // decrement
80 01 05              // block 5 and
80 01 06              // restore to block 6
81 04 01 00 00 00     // value = 1
```

Example 2: Decrement MIFARE block 5 (value = 100) and increment block 6 (value = 2)

FF C2 00 03 16

```
A1 09                 // decrement
80 01 05              // block 5
81 04 64 00 00 00     // value = 100
A0 09                 // increment
80 01 06              // block 6
81 04 02 00 00 00     // value = 2
```

5.5 Wrapped SE processor commands

This is a specific type of OMNIKEY or vendor command that wraps an SE processor command. For full details of all the available SE processor commands, see the *iCLASS SE Processor User Guide* (6700-902). The following sections give examples of some commonly used commands for reading and writing MIFARE DESFire credentials.

Wrapped SE processor command APDU

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0x70	0x07	0x6B	xx	DER TLV coded PDU.	xx

Where the data is a DER TLV coded SE processor command. See the following sections for example commands.

Wrapped SE processor response

Data field	SW1SW2
DER TLV coded response PDU.	See the following table.

Where the data is a DER TLV coded SE processor command. See the following sections for example commands.

Wrapped SE processor response SW1SW2 bytes

Type	SW1SW2	Description
Normal	0x9000	Successful.
Warning		
Execution error	0x6400	No response from media (Time Out).
	0x6F00	Unknown error.
Checking error	0x6700	Wrong APDU length.

5.5.1 Load Key

This uses the `samCommandCardApiLoadKey` command.

Value	Description
0xA0	Tag = SE processor command.
xx	Length to end of command.
0xA5	Tag = Process card API.
xx	Length to end of command.
0xA5	Tag = Card API Load Key.
xx	Length to end of command.
0x80	Tag = Is Persistent.
0x01	Length of value = 1.
0x0x	Value = Persistent or volatile flag: 00 = Volatile 01 = Persistent
0x81	Tag = Key Reference OID.
xx....xx	Value = Key reference (one of the following): User key OID (3 bytes, first byte must be 03). Key reference OID of predefined Secure Channel key (1 byte). Full OID of user defined Secure Channel key (x bytes).
0x82	Tag = Key Value.
xx	Length of the value = x.
xx....xx	Value = Value of the key.

There are further optional parameters for Secure Session Key loading. See the *iCLASS SE Processor User Guide* (6700-902).

Load Key response

The normal response is just the SW1SW2 code. There is additional data in some error conditions. See the *iCLASS SE Processor User Guide* (6700-902).

Load Key example

Load Key OID 03 01 01:

FF 70 07 6B 20 A0 1E A5 1C A5 1A 80 01 01 81 03 03 01 01 82 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Successful response:

90 00

5.5.2 DESFire Authenticate Native

This performs an authentication using the backwards compatible Native communication mode, suitable for use with DESFire v 0.6 or EV1 cards. It uses the `samCommandCardApiDesfireAuthNative` command.

There is an alternative command, `samCommandCardApiDesfireAuthIso`, using ISO communication mode. See the *iCLASS SE Processor User Guide* (6700-902).

DESFire Authenticate Native command

Value	Description
0xA0	Tag = SE processor command.
xx	Length to end of command.
0xA5	Tag = Process card API.
xx	Length to end of command.
0xA2	Tag = Card API DESFire.
xx	Length to end of command.
0xA1	Tag = DESFire Authenticate Native.
xx	Length to end of command.
0x80	Tag = Key number.
0x01	Length of value = 1.
xx	Value = Number of the key to be used for authentication (Application Key or Master Key of the DESFire card).
0x81	Tag = Key reference OID.
0x03	Length of value = 3.
xxxxxx	Value = Reference OID of the Key (3 bytes, first byte must be 03).

There are further optional parameters for key diversification. For more details refer to the *iCLASS SE Processor User Guide* (6700-902).

DESFire Authenticate Native response

There is no data in the response to this command, just the SW1SW2 code.

DESFire Authenticate Native example

DESFire Authenticate Native:

FF 70 07 6B 10 A0 0E A5 0C A2 0A A1 08 80 01 00 81 03 03 01 01

Successful response:

90 00

5.5.3 DESFire Format Card

This uses the `samCommandCardApiDesfireFormatPICC` command.

DESFire Format Card command

Value	Description
0xA0	Tag = SE processor command.
0x6	Length to end of command.
0xA5	Tag = Process card API.
0x04	Length to end of command.
0xA2	Tag = Card API DESFire.
0x02	Length to end of command.
0x93	Tag = DESFire Format.
0x00	Length of value = 0.
	Value = NULL (no parameters).

DESFire Format Card response

There is no data in the response to this command, just the SW1SW2 code.

DESFire Format Card example

DESFire Format Card:

FF 70 07 6B 08 A0 06 A5 04 A2 02 93 00

Successful response:

90 00

5.5.4 DESFire Create Application

This uses the `samCommandCardApiDesfireCreateApplication` command.

DESFire Create Application command

Value	Description
0xA0	Tag = SE processor command.
0x11	Length to end of command.
0xA5	Tag = Process card API.
0x0F	Length to end of command.
0xA2	Tag = Card API DESFire.
0x0D	Length to end of command.
0xA6	Tag = DESFire Create Application.
0x0B	Length to end of command.
0x80	Tag = Application number.
0x03	Length of value = 3.
xxxxxx	Value = Application number (3 bytes, MSB first).
0x81	Tag = Key setting 1.
0x01	Length of value = 1.
xx	Value = Application master key settings (1 byte).
0x82	Tag = Key setting 2.
0x01	Length of value = 1.
xx	Value = Number of keys (1 byte).

DESFire Create Application response

There is no data in the response to this command, just the SW1SW2 code.

DESFire Create Application example

DESFire Create Application 1:

FF 70 07 6B 13 A0 11 A5 0F A2 0D A6 0B 80 03 00 00 01 81 01 0F 82 01 01

Successful response:

90 00

5.5.5 DESFire Select Application

This uses the `samCommandCardApiDesfireSelectApp` command.

DESFire Select Application command

Value	Description
0xA0	Tag = SE processor command.
0x0B	Length to end of command.
0xA5	Tag = Process card API.
0x09	Length to end of command.
0xA2	Tag = Card API DESFire.
0x07	Length to end of command.
0xA0	Tag = DESFire Select Application.
0x05	Length to end of command.
0x80	Tag = Application number.
0x03	Length of value = 3.
xxxxxx	Value = Application number (3 bytes, MSB first).

DESFire Select Application response

There is no data in the response to this command, just the SW1SW2 code.

DESFire Select Application example

DESFire Select Application 1:

FF 70 07 6B 0D A0 0B A5 09 A2 07 A0 05 80 03 00 00 01

Successful response:

90 00

5.5.6 DESFire Create Standard Data File

This uses the `samCommandCardApiDesfireCreateStdDataFile` command.

DESFire Create Standard Data File command

Value	Description
0xA0	Tag = SE processor command.
xx	Length to end of command.
0xA5	Tag = Process card API.
xx	Length to end of command.
0xA2	Tag = Card API DESFire.
xx	Length to end of command.
0xA8	Tag = DESFire Create Standard Data File.
xx	Length to end of command.
0x80	Tag = File number.
0x01	Length of value = 1.
xx	Value = Number of file to be created (1 byte): 00 to 0F (00 to 1F for MIFARE DESFire EV1 cards)
0x82	Tag = Communication Settings.
0x01	Length of value = 1.
0x0x	Value = Communication mode of the file (1 byte): 00 = Plain 01 = MAC 03 = Encrypt
0x83	Tag = Access Rights.
xx	Length of value = x.
xx...xx	Value = Access rights of the file (x bytes).
0x84	Tag = File Size.
0x03	Length of value = 3.
xxxxxx	Value = Size of file (3 bytes, MSB first).

DESFire Create Standard Data File response

There is no data in the response to this command, just the SW1SW2 code.

DESFire Create Standard Data File example

DESFire Create Standard Data File number 2 with 59 bytes:

FF 70 07 6B 18 A0 16 A5 14 A2 12 A8 10 80 01 00 82 01 02 83 03 00 EE EE 84 03 00 00 3B

Successful response:

90 00

5.5.7 DESFire Write Data

This uses the `samCommandCardApiDesfireWriteData` command.

DESFire Write Data command

Value	Description
0xA0	Tag = SE processor command.
xx	Length to end of command.
0xA5	Tag = Process card API.
xx	Length to end of command.
0xA2	Tag = Card API DESFire.
xx	Length to end of command.
0xA4	Tag = DESFire Write Data.
xx	Length to end of command.
0x80	Tag = File number.
0x01	Length of value = 1.
xx	Value = Number of file to be written (1 byte): 00 to 0F for Standard Data Files 00 to 07 for Backup Data Files (00 to 1F for MIFARE DESFire EV1 cards)
0x81	Tag = Offset.
0x0x	Length of value = x (1 or 2).
xx...xx	Value = Offset into file to start writing (1 or 2 bytes).
0x82	Tag = Data Length.
xx	Length of value = x.
xx...xx	Value = File data to be written.
0x83	Tag = Mode.
0x01	Length of value = 1.
xx	Value = Communication mode of the operation (1 byte): 00 = Plain 01 = MAC 03 = Encrypt
0x84	Tag = Commit.
0x01	Length of value = 1.
xx	Value = Commit flag (1 byte): 00 = No commit 01 = Commit

DESFire Write Data response

There is no data in the response to this command, just the SW1SW2 code.

DESFire Write Data example

DESFire Write 59 bytes to file number 2:

FF 70 07 6B 51 A0 4F A5 4D A2 4B A4 49 80 01 02 81 01 00 82 3B 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 83 01 00 84 01 00

Successful response:

90 00

5.5.8 DESFire Read Data

This uses the `samCommandCardApiDesfireReadData` command.

DESFire Read Data command

Value	Description
0xA0	Tag = SE processor command.
xx	Length to end of command.
0xA5	Tag = Process card API.
xx	Length to end of command.
0xA2	Tag = Card API DESFire.
xx	Length to end of command.
0xA3	Tag = DESFire Read Data.
xx	Length to end of command.
0x80	Tag = File number.
0x01	Length of value = 1.
xx	Value = Number of file to be read (1 byte): 00 to 0F for Standard Data Files 00 to 07 for Backup Data Files (00 to 1F for MIFARE DESFire EV1 cards)
0x81	Tag = Offset.
0x0x	Length of value = x (1 or 2).
xx...xx	Value = Offset into file to start reading (1 or 2 bytes).
0x82	Tag = Data Length.
0x01	Length of value = 1.
xx...xx	Value = File data to be read.
0x83	Tag = Mode.
0x01	Length of value = 1.
xx	Value = Communication mode of the operation (1 byte): 00 = Plain 01 = MAC 03 = Encrypt

DESFire Read Data response

Value	Description
xx...xx	Value = File data to be read (255 bytes max)

DESFire Read Data example

DESFire Read 59 bytes from file number 2:

FF 70 07 6B 51 A0 12 A5 10 A2 0E A3 0C 80 01 02 81 01 00 82 01 3B 83 01 00 00

Successful response:

00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F 20 21 22 23 24 25 26 27 28 29 2A 2B 2C 2D 2E 2F 30 31 32 33 34 35 36 37 38 39 3A 90 00

5.5.9 Read PACS Data

This uses the `samCommandGetContentElement2` command.

Read PACS Data command

The following table gives the basic use of the `samCommandGetContentElement2` command. For more details refer to the *iCLASS SE Processor User Guide* (6700-902).

Value	Description
0xA0	Tag = SE processor command.
xx	Length to end of command.
0xBE	Tag = Get Content Element 2.
xx	Length to end of command.
0x80	Tag = Content Element.
0x01	Length of value = 1.
0x04	Value = Physical Access Bits.

Read PACS Data response

Value	Description
0xA0	Tag = Get Content Element Data.
xx	Length to end of command.
0x80	Tag = Get Element Data.
xx	Length of value = xx.
xx...xx	Value = Content Element Data (e.g. PACS).
0x81	Tag = Secure Object OID.
xx	Length of value = xx.
xx...xx	Value = OID of the Secure Object containing the returned Content Element.
0x82	Tag = Secure Object Media Edge Type.
0x01	Length of value = 1.
xx	Value = Type of media containing the Secure Object: 0 = Unknown 1 = DESFire 2 = MIFARE 3 = iCLASS (PicoPass) 4 = ISO14443AL4 6 = MIFARE Plus 7 = Seos®

Read PACS Data example

Reading PACS data from a Seos credential:

FF 70 07 6B 07 A0 05 BE 03 80 01 04 00

Successful response:

A0 1C 80 05 06 80 80 0A 40 81 10 2B 06 01 04 01 81 E4 38 01 01 02 04 01 8F 63 13 82 01 07 90 00

Where: PACS data = 06 80 80 0A 40

SO OID = 2B 06 01 04 01 81 E4 38 01 01 02 04 01 8F 63 13

Media type = 07 (Seos)

Section 6

6 Secure session model

The secure session model provides a secure way of communicating with the iCLASS SE® processor for loading keys and reading credentials, etc. As the commands are encrypted, this prevents any snooping of messages between the host application and the reader. In addition to providing a secure way of communicating with the device, the secure session manages access to data objects. The secure session model allows for access to certain objects to be conditional, based on the access rights of the user using the secure session. These access rights are established based on the key used to establish the secure session and on the access conditions for the particular object being accessed. Although at present the access conditions are predefined, it is planned that in future firmware versions the user will change these conditions provided they have administration rights.

Note that the state of the secure session is independent of the CCID state. If a card is removed, powered off or reset, the secure session will not be reset. Similarly when talking to the reader via PC/SC service, the secure session will not be closed when the host application closes the connection to the reader.

6.1 Using a secure session

To establish a secure session, the user must use the two OMNIKEY® specific functions to initialize and continue authentication, described in detail in the following sections. Once the secure session has been established, then data is exchanged with the device using the OMNIKEY specific APDU for data exchange described below. This will allow access to the data objects that require a secure session to access. The secure session will terminate following an error in the data exchange or encryption or if the user sends the terminate session commands described below.

6.1.1 Establish and manage a secure session

The endpoint of the secure session when using the OMNIKEY 5023 is always the iCLASS SE processor. The system supports a scalable security. The security level (access right) depends on the key used for the secure session authentication. The highest security level session is one that is authenticated using the “HID Admin” key. In this case, the SAM / reader will allow a HID data object to be updated. Access rights are managed for data objects such as the HID application in a MIFARE card or iCLASS card. New data objects can be added without impact to existing data objects.

For a secure channel transmission, `SCardConnect` should be used with a ShareMode of `SCARD_SHARE_EXCLUSIVE`. The client (host application) must ensure the correct termination of the secure channel after the last transaction.

The procedure to establish a secured channel is achieved in two phases, AUTH1 and AUTH2.

In the following, “Client” is the host application and “Server” is the reader.

6.1.1.1 Initialize Authentication (AUTH1)

To initialize the secured channel, the client must send an 8 byte RND.A and the key number. By the choice of key number, the client can establish a secured session as either read only or read/write.

DER TLV PDU:

```
A1 12                                   // CHOICE ManageSECS
   A0 10                                // CHOICE EstablishAUTH1
    80 01 00                            // VersionSECCH (Currently SAM ignores this value)
    81 01 yy                            // Key Number (OID)
    82 08 xx xx xx xx xx xx xx xx       // RND.A
```

Note: Currently iCLASS SE processor ignores the value of version Tag (RFU). Code RFU s as 0.

Response APDU:

Initialize Authentication response APDU

Data field	SW1SW2
9D 20 uu uu uu uu uu uu uu uu // 8 byte UID rr rr rr rr rr rr rr rr // 8 byte RND.B xx xx xx xx xx xx xx xx // 16 byte Reader Cryptogram xx xx xx xx xx xx xx xx	See *Section: 6.1.1.5 Secure channel return codes*.

The complete APDU is:

FF 70 07 6B 14 A1 12 A0 10 80 01 00 81 01 yy 82 08 xx xx xx xx xx xx xx xx 00

6.1.1.2 Continue Authentication (AUTH2)

The second authentication stage completes the establishment of the secured channel.

DER TLV PDU:

```
A1 26                                   // CHOICE ManageSECS
   A1 24                                // CHOICE EstablishAUTH2
      80 10 xx xx xx xx xx xx xx xx      // xx = ClientCryptogram
            xx xx xx xx xx xx xx xx
      81 10 yy yy yy yy yy yy yy yy      // yy = C-MAC
            yy yy yy yy yy yy yy yy
```

Response APDU:

Continue Authentication response APDU

Data field	SW1SW2
9D 10 yy yy yy yy yy yy yy yy // 16 byte R-MAC yy yy yy yy yy yy yy yy	See *Section: 6.1.1.5 Secure channel return codes*.

The complete APDU is:

FF 70 07 6B 28 A1 26 A1 24 80 10 xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx 81 10 yy yy yy yy yy yy yy yy yy yy yy yy yy yy yy yy 00.

6.1.1.3 Data Exchange in a secure session

To exchange data during a secure session the complete PC/SC command should be encrypted, except the transport protocol header. The secure message is wrapped in an APDU of the form FF 70 07 6B + Lc + message + Le, because the resource manager and Microsoft CCID driver do not accept messages without CLA INS P1 P2 and Lc. Note that the message should use an Le value of 00. The plain text message body is padded according to ISO 9797-1 padding method 2 to a whole multiple of the block length (multiple of 16 bytes). Next, the client encrypts the padded body of the message according to message encryption algorithm. Finally, the client computes the MAC on the encrypted body and appends the MAC to the encrypted body. The iCLASS SE processor has its own PCSC command handler, and during a secure session commands are sent directly to this with no processing by the reader firmware.

As an example, if the plain message is a READ BINARY command for block 6, it should be padded as follows:

Data Exchange APDU in a secure session

APDU	Padding
FF B0 00 06 08	80 00 00 00 00 00 00 00 00 00 00

The complete message to send through `SCardTransmit` is then:

```
FF 70 07 6B
20
xx xx xx xx xx xx xx xx        // xx = Enc(APDU+PADDING, S-ENC)
xx xx xx xx xx xx xx xx        //
yy yy yy yy yy yy yy yy        // yy = C-MAC
yy yy yy yy yy yy yy yy
00
```

Response APDU:

Data Exchange response APDU in a secure session

Data field	SW1SW2
9D 20 xx xx xx xx xx xx xx xx // n x 16 byte encrypted response xx xx xx xx xx xx xx xx // Enc(response + padding, S-(ENC) yy yy yy yy yy yy yy yy // 16 byte R-MAC yy yy yy yy yy yy yy yy 90 00	See *Section: 6.1.1.5 Secure channel return codes*.

6.1.1.4 Terminate Secure Session

The session is terminated if an error occurs (bad client cryptogram) or if the client terminates the session. In both cases, the IFD deletes the session keys S-MAC1, S-MAC2 and S-ENC. Following termination of the session, the card will lose its security state. The coding of the Terminate Secure Session command is:

DER TLV PDU:

```
A1 02                                   // CHOICE ManageSECS
   A2 00                                // CHOICE terminateSecuredSession
```

This message is always encrypted in the secure channel and is never sent plain.

Terminate Secure Session APDU

APDU	Padding
FF 70 07 6B 04 A1 02 A2 00	80 00 00 00 00 00 00

Encrypted message:

```
FF 70 07 6B
20
xx xx xx xx xx xx xx xx                  // xx = Enc(APDU+PADDING, S-ENC)
xx xx xx xx xx xx xx xx
yy yy yy yy yy yy yy yy                 // yy = C-MAC
yy yy yy yy yy yy yy yy
00
```

Response APDU:

Terminate Secure Session response APDU

Data field	SW1SW2
9D 00	See *Section: 6.1.1.5 Secure channel return codes*.

Note: Up to version 1.0, the iCLASS SE processor does not support the PC/SC command; it works only with the iCLASS SE processor internal SAM command `samCommandSecureChannelTerminate`.

iCLASS SE processor internal secure channel terminate command

SAM command	Padding
91 00	80 00 00 00 00 00 00 00 00 00 00 00 00 00

```
xx = Enc(SAM Command + PADDING, S-ENC)
```

6.1.1.5 Secure channel return codes

Return codes for managing the secure channel

Type	SW1SW2	Description
Normal	0x9000	Successful.
Execution Error	0x6400	No response from endpoint.
Checking Error	0x6700	Wrong APDU length.
	0x6982	Security status not satisfied.

6.2 Secure session access rights and secure session key

The OMNIKEY 5x27CK supports one secure session key and this key occupies slot 0x80.

For obtaining the pre-configured secure session key, contact HID Tech Support, www.hidglobal.com/support.

The user should change this default value setting. See *Section 6.3 Changing the secure session keys*. The secure session key has read access to the HID area of HID credentials and reads the PACS data of HID cards. At present, you cannot change the secure session key access rights. The following table summarizes the access rights.

Secure session access rights

Object	Write	Read
Keys 0x1F to 0x34 and 0x80	Secure Session Key.	Not allowed.
All other keys	Free access.	Not allowed.
HID Application Area (MIFARE Classic, DESFire and iCLASS)	Not allowed.	Secure Session Key.
PACS Data (MIFARE Classic, DESFire and iCLASS)	Not allowed.	Secure Session Key.
Non-HID Application Areas (MIFARE Classic, DESFire and iCLASS)	Free access.	Free access.
Prox PACS data	Not supported.	Free access.

6.3 Changing the secure session keys

You should change the secure session keys default values used by the iCLASS SE processor. The default values are the same for every customer and are therefore insecure. Note that it is the responsibility of the customer to maintain the new keys. If the key values are lost, they cannot be recovered by HID. To change the keys use the following procedure.

- Establish a secure session using the HID Admin Key, as previously described.
- Send each key to the iCLASS SE Processor Card API load key command through the secured session. See the *iCLASS SE Processor User Guide* (6700-902) for details. The full unencrypted iCLASS SE processor command to load key value XXXX XXXX XXXX XXXX to slot NN is:

  ```
  A0DA0263000028440a44000000A01EA51CA51A80010181030300 NN 8210  XXXX XXXX XXXX XXXX
  0000
  ```

  As the command header is not sent when using a secure session, the part that must be encrypted and sent through the secure session is:

  ```
  A01EA51CA51A80010181030300 NN 8210  XXXX XXXX XXXX XXXX
  ```

- Terminate the secure session.

The keys are now loaded.

This page is intentionally left blank.

Section 7

7 Reader configuration

All OMNIKEY® 5023 configurable items are identified by a unique ASN.1 leaf. A full description is given below, including default values and example APDU commands to get and set.

7.1 APDU commands

If the attached host implements a PC/SC environment, the OMNIKEY 5023 ASN.1 leafs are accessible using proprietary APDU commands sent through the CCID USB device class. The APDU commands are used to set and get the configuration items and to control the reader – to apply, store or reset the changes.

APDUs supported by the OMNIKEY 5023 reader exist in the following groups:

- Standard inter-industry commands as defined in ISO/IEC 7816-4:2005(E). These commands are passed transparently to the contactless card related to the CCID slot.
- PC/SC commands as defined in "*Interoperability Specification for ICCs and Personal Computer Systems - Part 3*".
- ICAO (International Civil Aviation Organization) test commands as defined in "*RF Protocol and Application Test Standard for e-Passport - Part 4*", Appendix C.
- OMNIKEY 5023 Specific Commands. These include APDUs to manage the reader, to directly access the configuration items.

7.2 Accessing configuration

OMNIKEY Specific Commands include the Reader Information API command group that provides access to reader configuration and allows control of the reader.

Configuration structure

Root	Request	Branch
readerInformationApi (0x02)	Get (0x00)	readerCapabilities (0x02) tlvVersion (0x00) deviceID (0x01) productName (0x02) productPlatform (0x03) enabledCLFeatures (0x04) firmwareVersion (0x05) hfControllerVersion (0x08) hardwareVersion (0x09) hostInterfacesFlags (0x0A) numberOfContactSlots (0x0B) numberOfContactlessSlots (0x0C) numberOfAntennas (0x01) vendorName (0x01) exchangeLevel (0x01) serialNumber (0x01) hfControllerType (0x01) sizeOfUserEEProm (0x01) firmwareLabel (0x01)
	Get (0x00) Set (0x01)	contactlessSlotConfiguration (0x04) contactlessCommon (0x00) pollingSearchOrder (0x09) emdSuppresionEnable (0x07) iso14443aConfig (0x02) iso14443aEnable (0x00) iso14443aRxTxBaudRate (0x01) mifareKeyCache (0x03) mifarePreferred (0x04) iso14443bConfig (0x03) iso14443bEnable(0x00) iso14443bRxTxBaudRate (0x01) felicaConfig (0x05) felicaEnable (0x00) felicaRxTxBaudRate (0x01) iClassConfig (0x06) iClass15693Enable (0x03) iClass15693DelayTime (0x04) iClass15693Timeout (0x05) iClassActallTimeout (0x06)
	Get (0x00) Set (0x01)	readerEEPROM (0x07) eepromOffset (0x01) eepromRead (0x02) eepromWrite (0x03)
	Set (0x01)	readerConfigurationControl (0x09) applySettings (0x00) restoreFactoryDefaults (0x01) rebootDevice (0x03)

7.2.1 Example: Get Reader Information

With Get Reader Information the host application gets specific information about the reader. The following example shows how to read a single item:

DER TLV PDU for retrieve single IFD information (productName):

```
A2 06             // CHOICE ReaderInformationAPI
   A0 04          // CHOICE GetReaderInformation
      A0 02       // CHOICE ReaderCapabilites
      82 00       // SEQUENCE productName
```

The reply of single information is TLV coded:

BD 0F 82 0D **4F 4D 4E 49 4B 45 59 20 35 30 32 32 00** 90 00 // 'OMNIKEY 5023' + return code

For a Reader Information GET Request the Response Tag (1D) is always CONSTRUCTED. The response can include more than one leaf, depending on the request.

DER TLV PDU for retrieve single IFD information (productPlatform):

```
A2 06             // CHOICE ReaderInformationAPI
   A0 04          // CHOICE GetReaderInformation
      A0 02       // CHOICE ReaderCapabilites
      83 00       // SEQUENCE productplatform
```

The reply of single information is TLV coded:

BD 0A 83 08 **41 56 69 61 74 6F 52 00** 90 00 // 'AViatoR' + return code success

7.3 Reader Capabilities

Reader Capabilities structure:

Root	Branch
readerCapabilities (0x02)	tlvVersion (0x00)
	deviceID (0x01)
	productName (0x02)
	productPlatform (0x03)
	enabledCLFeatures (0x04)
	firmwareVersion (0x05)
	hfControllerVersion (0x08)
	hardwareVersion (0x09)
	hostInterfacesFlags (0x0A)
	numberOfContactSlots (0x0B)
	numberOfContactlessSlots (0x0C)
	numberOfAntennas (0x01)
	vendorName (0x01)
	exchangeLevel (0x01)
	serialNumber (0x01)
	hfControllerType (0x01)
	sizeOfUserEEProm (0x01)
	firmwareLabel (0x01)

7.3.1 tlvVersion

Tag	0x00
Access	Read-only.
Type	INTEGER.
Length	1 byte.
Value	0x00 - 0xFF
Description	The version of the TLV encoding used by APDUs.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 80 00 00
Sample response	BD 03 80 01 01 90 00

7.3.2 deviceID

Tag	0x01
Access	Read-only.
Type	OCTET STRING.
Length	2 bytes.
Value	0x0000 - 0xFFFF
Description	Product ID, 0x0006 for OMNIKEY 5023.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 81 00 00
Sample response	BD 04 81 02 00 06 90 00

7.3.3 productName

Tag	0x02
Access	Read-only.
Type	OCTET STRING.
Length	Variable.
Value	Null terminated string.
Description	The name of the reader.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 82 00 00
Sample response	BD 0F 82 0D 4F 4D 4E 49 4B 45 59 20 35 30 32 33 00 90 00

7.3.4 productPlatform

Tag	0x03
Access	Read-only.
Type	OCTET STRING.
Length	Variable.
Value	Null terminated string.
Description	The name of the platform on which the product is based, “AViatoR” for OMNIKEY 5023.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 83 00 00
Sample response	BD 0A 83 08 41 56 69 61 74 6F 52 00 90 00

7.3.5 enabledCLFeatures

Tag	0x04
Access	Read-only.
Type	OCTET STRING.
Length	2 bytes.
Value	Bit mask, 0x0B91 for OMNIKEY 5023.
Description	Provides information about what contactless protocols are supported.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 84 00 00
Sample response	BD 04 84 02 0B 91 90 00

CL features:

0x0001 – FeliCa support.

0x0002 – EMVCo support.

0x0004 – Calypso support.

0x0008 – NFC P2P support.

0x0010 – SIO processor available.

0x0020 – SDR (LF processor) available.

0x0040 – Native FW Secure Engine.

0x0080 – T=CL support.

0x0100 – ISO 14443 A support.

0x0200 – ISO 14443 B support.

0x0800 – PicoPass 15693-2 support.

0x1000 – PicoPass 14443B-2 support.

0x2000 – PicoPass 14443A-3 support.

0x4000 – RFU.

0x8000 – RFU.

7.3.6 firmwareVersion

Tag	0x05
Access	Read-only.
Type	OCTET STRING.
Length	3 bytes.
Value	Null terminated string.
Description	The version number of the reader's firmware. 1st byte is Major, 2nd byte is Minor, 3rd byte is revision number.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 85 00 00
Sample response	BD 05 85 03 01 00 00 90 00

7.3.7 hfControllerVersion

Tag	0x08
Access	Read-only.
Type	OCTET STRING.
Length	1 byte.
Value	Version number.
Description	The version of the HF front end used for controlling high frequency credentials.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 88 00 00
Sample response	BD 03 88 01 18 90 00

7.3.8 hardwareVersion

Tag	0x09
Access	Read-only.
Type	OCTET STRING.
Length	Variable.
Value	Null terminated string.
Description	The version of the reader hardware used.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 89 00 00
Sample response	BD 11 89 0F 50 43 42 2D 30 30 30 34 34 20 52 45 56 32 00 90 00

7.3.9 hostInterfaceFlags

Tag	0x0A
Access	Read-only.
Type	OCTET STRING.
Length	1 byte.
Value	Bit mask.
Description	Provides information on the interfaces supported by the reader for communication with the host. Bit 1 (0x02) = USB interface.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 8A 00 00
Sample response	BD 03 8A 01 02 90 00

Host Interface flags:

0x01 – Ethernet available.

0x02 – USB available.

0x04 – Serial RS232 available.

0x08 – SPI available.

0x10 – I^2C available.

7.3.10 numberOfContactSlots

Tag	0x0B
Access	Read-only.
Type	OCTET STRING.
Length	1 byte.
Value	Number of contact slots.
Description	Number of contact slots supported by the reader. 0 for OMNIKEY 5023.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 8B 00 00
Sample response	BD 03 8B 01 00 90 00

7.3.11 numberOfContactlessSlots

Tag	0x0C
Access	Read-only.
Type	OCTET STRING.
Length	1 byte.
Value	Number of contactless slots.
Description	The number of contactless PCSC slots supported by the reader. 1 for OMNIKEY 5023.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 8C 00 00
Sample response	BD 03 8C 01 01 90 00

7.3.12 numberOfAntennas

Tag	0x0D
Access	Read-only.
Type	OCTET STRING.
Length	1 byte.
Value	Number of antennas.
Description	The number of antennas the reader contains. 1 for OMNIKEY 5023.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 8D 00 00
Sample response	BD 03 8D 01 01 90 00

7.3.13 vendorName

Tag	0x0F
Access	Read-only.
Type	OCTET STRING.
Length	1 byte.
Value	Null terminated string.
Description	The vendor of the reader, “HID Global” for OMNIKEY 5023.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 8F 00 00
Sample response	BD 0D 8F 0B 48 49 44 20 47 6C 6F 62 61 6C 00 90 00

7.3.14 exchangeLevel

Tag	0x11
Access	Read-only.
Type	OCTET STRING.
Length	1 byte.
Value	Bit mask. 0x01 – TPDU, 0x02 – APDU, 0x04 – Extended APDU.
Description	Provides information about the different APDU levels supported by the reader. 0x04 for OMNIKEY 5023 (Extended APDU).
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 91 00 00
Sample response	BD 03 91 01 02 90 00

7.3.15 serialNumber

Tag	0x12
Access	Read-only.
Type	OCTET STRING.
Length	Variable, max 32 bytes.
Value	Serial number.
Description	The serial number of the reader.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 92 00 00
Sample response	BD 19 92 17 4B 54 2D 30 38 36 33 30 30 33 30 2D 31 36 31 30 2D 30 30 30 31 31 34 90 00

7.3.16 hfControllerType

Tag	0x13
Access	Read-only.
Type	OCTET STRING.
Length	Variable, max 32 bytes.
Value	Null terminated chip name.
Description	The IC used for control of HF credentials. “RC663” for OMNIKEY 5023.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 93 00 00
Sample response	BD 08 93 06 52 43 36 36 33 00 90 00

7.3.17 sizeOfUserEEPROM

Tag	0x14
Access	Read-only.
Type	OCTET STRING.
Length	2 bytes.
Value	Size in bytes.
Description	The amount of user EEPROM memory available. For OMNIKEY 5023 1024 bytes.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 94 00 00
Sample response	BD 04 94 02 04 00 90 00

7.3.18 firmwareLabel

Tag	0x16
Access	Read-only.
Type	OCTET STRING.
Length	Variable.
Value	Firmware unique ID as string.
Description	Detailed information about the firmware version.
Get APDU	FF 70 07 6B 08 A2 06 A0 04 A0 02 96 00 00
Sample response	BD 35 96 33 4F 4B 35 30 32 32 2D 31 2E 30 2E 30 2E 32 37 30 2D 32 30 31 36 30 31 32 36 54 31 35 32 30 32 34 2D 30 36 43 41 35 34 31 46 30 38 32 41 2D 46 4C 41 53 48 90 00

7.4 Contactless configuration

Contactless Slot Configuration structure

Root	Branch	
contactlessSlotConfiguration (0x04)	contactlessCommon (0x00)	pollingSearchOrder (0x09)
		emdSuppresionEnable (0x07)
	iso14443aConfig (0x02)	iso14443aEnable (0x00)
		iso14443aRxTxBaudRate (0x01)
		mifareKeyCache (0x03)
		mifarePreferred (0x04)
	iso14443bConfig (0x03)	iso14443bEnable(0x00)
		iso14bRxTxBaudRate (0x01)
	felicaConfig (0x05)	felicaEnable (0x00)
		felicaRxTxBaudRate (0x01)
	iClassConfig (0x06)	iClass15693Enable (0x03)
		iClass15693DelayTime (0x04)
		iClass15693Timeout (0x05)
		iClassActallTimeout (0x06)

7.4.1 Baud rates

OMNIKEY 5023 allows setting maximum baud rate to and from a card for ISO/IEC 14443 Type A, ISO/IEC 14443 Type B and FeliCa protocols.

Commands: `iso14443aRxTxBaudRate`, `iso14443bRxTxBaudRate` and `felicaRxTxBaudRate` use the same format. A one-byte argument defines separately the baud rate for receiving (Rx) and transmitting (Tx) data. The first 4 bits are used to set Rx baud rate, the other for Tx baud rate. The resulting value is combination of bits:

Bit 0 (0x01) – 212 kbps
Bit 1 (0x02) – 424 kbps
Bit 2 (0x04) – 848 kbps

The reader always supports 106 kbps regardless of bit settings. If a card does not support a specific transmission speed the reader would use the other value.

For example. 0x77 means the reader supports 106, 212, 424, 848 kbps for Rx and Tx. If a card supports only 106 kbps and 424 kbps the reader would use 424 kbps or 106 kbps (in case card activation at 424 kbps fails).

Note: Doubling baud rate does not double transmission speed. In an extreme example, changing the baud rate from 424 kbps to 848 kbps increases transmission speed by less than 10%. The number may vary depending on the amount of data transmitted. The worst ratio is for short packets. Increasing maximum baud rate may cause transmission problems and shorten maximum effective distance between a card and the reader.

7.4.1.1 Examples

0x00 – 106 kbps for Rx and Tx

0x23 – 106 and 424 kbps for Rx and 106, 212, 424 kbps for Tx

0x71 – 106, 212, 424, 848 kbps for Rx and 106, 212 kbps for Tx

7.4.1.2 Default values

ISO/IEC 14443 Type A: 0x33 – 106, 212, 424 kbps for Rx and Tx

ISO/IEC 14443 Type B: 0x33 – 106, 212, 424 kbps for Rx and Tx

FeliCa: 0x11 – 106, 212 kbps for Rx and Tx

7.4.2 Common parameters

pollingSearchOrder

Tag	0x09
Access	Read/write.
Type	OCTET STRING.
Length	5 bytes.
Value	See description below.
Description	Sets polling order.
Set APDU	FF 70 07 6B 0F A2 0D A1 0B A4 09 A0 07 89 05 xx xx xx xx xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A0 02 89 00 00
Sample response	BD 07 89 05 xx xx xx xx xx 90 00

The command expects 5 bytes which indicate polling order. Byte position sets priority of the protocol. Protocol on the first byte is checked first and protocol on 5th byte as the last one. Values assigned to protocols:

0x00 – none
0x02 – ISO/IEC 14443 Type A
0x03 – ISO/IEC 14443 Type B
0x04 – iCLASS® ISO/IEC 15693
0x06 – FeliCa

For example 02 03 04 06 means order: ISO/IEC 14443 Type A, ISO/IEC 14443 Type B, iCLASS ISO/IEC 15693, FeliCa. To support only ISO/IEC 14443 Type A protocol use: 02 00 00 00 00.

Note: If a protocol is not included in the search order table, the card will not be recognized even if the specific protocol is enabled.

emdSupressionEnable

Tag	0x07
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	0x01 enable, 0x00 disable.
Description	Enables or disables EMD suppression.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A0 03 87 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A0 02 87 00 00
Sample response	BD 03 87 01 xx 90 00

7.4.3 ISO/IEC 14443 Type A

iso14443aEnable

Tag	0x00
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	0x01 enable, 0x00 disable.
Description	Enables or disables support for ISO/IEC 14443 Type A cards.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A2 03 80 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A2 02 80 00 00
Sample response	BD 03 80 01 xx 90 00

iso14443aRxTxBaudRate

Tag	0x01
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	See *Section: 7.4.1 Baud rates*.
Description	Sets supported baud rates for ISO/IEC 14443 Type A cards.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A2 03 81 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A2 02 81 00 00
Sample response	BD 03 81 01 xx 90 00

mifareKeyCache

Tag	0x03
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	0x01 enable, 0x00 disable.
Description	Enables or disables key cache for MIFARE cards.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A2 03 83 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A2 02 83 00 00
Sample response	BD 03 83 01 xx 90 00

mifarePreferred

Tag	0x04
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	0x01 enable, 0x00 disable.
Description	Enables or disables MIFARE preferred mode.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A2 03 84 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A2 02 84 00 00
Sample response	BD 03 84 01 xx 90 00

7.4.4 ISO/IEC 14443 Type B

iso14443bEnable

Tag	0x00
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	0x01 enable, 0x00 disable.
Description	Enables or disables support for ISO/IEC 14443 Type B cards.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A3 03 80 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A3 02 80 00 00
Sample response	BD 03 80 01 xx 90 00

iso14443bRxTxBaudRate

Tag	0x01
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	See *Section: 7.4 Contactless configuration*.
Description	Sets supported baud rates for ISO/IEC 14443 Type B cards.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A3 03 81 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A3 02 81 00 00
Sample response	BD 03 81 01 xx 90 00

7.4.5 FeliCa

felicaEnable

Tag	0x00
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	0x01 enable, 0x00 disable.
Description	Enables or disables support for FeliCa cards.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A5 03 80 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A5 02 80 00 00
Sample response	BD 03 80 01 xx 90 00

felicaRxTxBaudRate

Tag	0x01
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	See *Section: 7.4 Contactless configuration*.
Description	Sets supported baud rates for FeliCa cards.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A5 03 81 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A5 02 81 00 00
Sample response	BD 03 81 01 xx 90 00

7.4.6 iCLASS

iCLASS15693Enable

Tag	0x03
Access	Read/write.
Type	INTEGER.
Length	1 byte.
Value	0x01 enable, 0x00 disable.
Description	Enables or disables support for iCLASS cards.
Set APDU	FF 70 07 6B 0B A2 09 A1 07 A4 05 A6 03 83 01 xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A6 02 83 00 00
Sample response	BD 03 83 01 xx 90 00

iCLASS15693DelayTime

Tag	0x04
Access	Read/write.
Type	INTEGER.
Length	4 bytes.
Value	Timeout.
Description	Sets or gets minimum chip response to reader command delay.
Set APDU	70 07 6B 0E A2 0C A1 0A A4 08 A6 06 84 04 xx xx xx xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A6 02 84 00 00
Sample response	BD 06 84 04 xx xx xx xx 90 00

iCLASS15693Timeout

Tag	0x05
Access	Read/write.
Type	INTEGER.
Length	4 bytes.
Value	Timeout.
Description	Sets or gets time to wait for response to a command.
Set APDU	70 07 6B 0E A2 0C A1 0A A4 08 A6 06 85 04 xx xx xx xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A6 02 85 00 00
Sample response	BD 06 85 04 xx xx xx xx 90 00

iCLASSActallTimeout

Tag	0x06
Access	Read/write.
Type	INTEGER.
Length	4 bytes.
Value	Timeout.
Description	Sets or gets time to wait for response to ACT/ACTALL.
Set APDU	70 07 6B 0E A2 0C A1 0A A4 08 A6 06 86 04 xx xx xx xx 00
Sample response	BD 00 90 00
Get APDU	FF 70 07 6B 0A A2 08 A0 06 A4 04 A6 02 86 00 00
Sample response	BD 06 86 04 xx xx xx xx 90 00

7.5 Reader EEPROM

OMNIKEY 5023 provides user available area (1024 bytes) in internal EEPROM memory. The content of this memory is preserved even when the power is off.

When specifying command to read or write data offset must be specified (Tag 0x01; 2 bytes).

Root	Branch
readerEEPROM (0x07)	eepromOffset (0x01)
	eepromRead (0x02)
	eepromWrite (0x03)

7.5.1 EEPROM read

Tag	0x02
Access	Read-only.
Type	OCTET STRING.
Length	Variable.
Value	yyyy – address (0x0000–0x03FF), nn – number of bytes to read.
Description	The version of the TLV encoding used by APDUs.
Get APDU	FF 70 07 6B 0D A2 0B A0 09 A7 07 81 02 yy yy 82 01 nn 00
Sample response	9D ss xx xx xx xx xx 90 00

7.5.2 EEPROM write

Tag	0x03
Access	Write-only.
Type	OCTET STRING.
Length	Variable.
Value	yyyy – address (0x0000–0x03FF), ss – number of bytes to write, xx – data to write
Description	Writes data to user EEPROM area.
Get APDU	FF 70 07 6B 11 A2 0F A1 0D A7 0B 81 02 yy yy 83 ss xx xx xx xx xx 00
Sample response	9D 00 90 00

7.6 Reader Configuration Control

Commands to apply, reset and store configuration changes.

Reader Configuration Control structure

Root	Branch
readerConfigurationControl (0x09)	applySettings (0x00)
	restoreFactoryDefaults (0x01)
	rebootDevice (0x03)

7.6.1 applySettings

Tag	0x00
Access	Write-only.
Type	
Length	
Value	None.
Description	Apply settings. This command must be used to accept changes in the reader configuration. The only settings that takes changes immediately are: iso14443aRxTxBaudRate, iso14443bRxTxBaudRate, felicaRxTxBaudRate. The commands resets the device.
Get APDU	FF 70 07 6B 08 A2 06 A1 04 A9 02 80 00 00
Sample response	9D 00 90 00

7.6.2 restoreFactoryDefaults

Tag	0x01
Access	Write-only.
Type	
Length	
Value	None.
Description	Sets reader configuration to factory defaults. The command resets the device.
Get APDU	FF 70 07 6B 08 A2 06 A1 04 A9 02 81 00 00
Sample response	9D 00 90 00

7.6.3 rebootDevice

Tag	0x03
Access	Write-only.
Type	
Length	
Value	None.
Description	Reboots the reader.
Get APDU	FF 70 07 6B 08 A2 06 A1 04 A9 02 83 00 00
Sample response	9D 00 90 00

Section 8

8 ICAO test commands

8.1 Command set

The International Civil Aviation Organization (ICAO) has defined a set APDUs for testing e-Passport readers. These are defined in Annex C of the technical report "*RF Protocol and Application Test Standard for e-Passport - Part 4*", available from the ICAO website www.icao.int. The standard APDU syntax and standard SCardTransmit API are used with the reserved value of the CLA byte of "FF" and the values of the INS byte are also reserved (in the range of 0x9x).

The commands supported by this reader are as follows:

8.1.1 ICAO commands

Instruction	Description	Comments
0x92	ISO/IEC 14443-2	Partially supported.
0x94	Transmit Pattern	Partially supported.
0x96	ISO/IEC 14443-3	Partially supported.
0x98	ISO/IEC 14443-4	Not supported.
0x9A	Miscellaneous	Partially supported.

All of the ICAO test commands are attempted regardless of card presence or type.

8.1.2 0x92 - ISO/IEC 14443-2: ISO/IEC 14443-2 command APDU

CLA	INS	P1	P2	Pc	Data In	Le
0xFF	0x92	XX	RFU	XX	Lc bytes	–

General: Any data received back from the card is ignored in this test.

8.1.3 ISO/IEC 14443-2 P1 coding

b7	b6	b5	b4	b3	b2	b1	b0	Description	Supported
0	0	------						Turn off RF field FF 92 00 00.	Yes
0	1	------						Turn on RF field with no sub-carrier.	Yes
1	0	------						Turn on RF field and transmit Lc bytes.	No
1	1	------						RFU.	No
--		0	0	------				ISO/IEC 14443 Type A transmission.	No
--		0	1	------				ISO/IEC 14443 Type B transmission.	No
--		1	0	------				ISO/IEC 15693 transmission (proprietary).	No
--		1	1	------				iCLASS® 15693 transmission (proprietary).	No
----				RFU		0	0	106 kbps.	No
----						0	1	212 kbps.	No
----						1	0	424 kbps.	No
----						1	1	848 kbps.	No

8.1.4 ISO/IEC 14443-2 response

Data Out	SW1SW2	
–	0x9000	Operation successful.
	0x6700	Wrong length (e.g. Lc absent when P1 b7 =1).
	0x6401	Internal error (e.g. protocol setup failed).

8.1.5 0x94 - Transmit Pattern command APDU

CLA	INS	P1	P2	Pc	Data In	Le
0xFF	0x94	XX	XX	XX	Lc bytes	XX

General: This test can be used to transmit and/or receive data to/from the card. No parity bit or CRC bytes are added, but framing (that is, start/stop bits, SOF/EOF) WILL be added. This is NOT fully compliant with the ICAO test standard.

8.1.6 ICAO Transmit Pattern P1 coding

b7	b6	b5	b4	b3	b2	b1	b0	Description	Supported
RFU				---			0	ISO/IEC 14443 Type A transmission.	Yes
				---			1	ISO/IEC 14443 Type B transmission.	Yes
				xxx			–	Number of bits in last byte to be transmitted.	Yes

8.1.7 ICAO Transmit Pattern P2 coding

b7	b6	b5	b4	b3	b2	b1	b0	Description	Supported
RFU		---			RFU	0	0	Tx - 106 kbps.	Yes
		---				0	1	Tx - 212 kbps.	Yes
		---				1	0	Tx - 424 kbps.	Yes
		---				1	1	Tx - 848 kbps.	Yes
		RFU	0	0	---			Rx - 106 kbps.	Yes
			0	1	---			Rx - 212 kbps.	Yes
			1	0	---			Rx - 242 kbps.	Yes
			1	1	---			Rx - 848 kbps.	Yes

8.1.8 ICAO Transmit Pattern SW1SW2 response bytes

Data Out	SW1SW2	
XX bytes (if Le present)	0x9000	Operation successful.
–	0x6700	Wrong length (e.g. Lc and Le are both absent).
	0x6A8A	Modulation index not supported (P1 b7:b4).
	0x6401	Internal error (e.g. protocol setup failed or transceive failed).

8.1.9 0x96 - ISO/IEC 14443-3 command APDU

CLA	INS	P1	P2	Pc	Data In	Le
0xFF	0x96	XX	XX	XX	Lc bytes	XX

8.1.10 ISO/IEC 14443-3 P1 coding

b7	b6	b5	b4	b3	b2	b1	b0	Description	Supported
–		--	ISO/IEC 14443 Type A commands						
			0	0	0	0	1	REQA.	Yes
			0	0	0	1	0	WUPA.	Yes
			0	0	0	1	1	HLTA.	No
			0	0	1	0	0	Full ISO-14443A Part 3 (that is, REQA + ANTI-COLLISION + SELECT).	No
			0	1	0	0	1	ANTI-COLLISION CL1.	No
			0	1	0	1	0	ANTI-COLLISION CL2.	No
			0	1	0	1	1	ANTI-COLLISION CL3.	No
			0	1	1	0	0	SELECT (0x70 + UID + BCC in Data In).	No
			ISO/IEC 14443 Type B commands						No
			1	0	0	0	1	REQB (Number of slots in P2).	Yes
			1	0	0	1	0	WUPB (Number of slots in P2).	Yes
			1	0	0	1	1	HLTB (PUPI may be in Data In).	No
			1	0	1	0	0	Slot-MARKER (Slot no in P2).	No
			1	0	1	0	1	ATTRIB (Bit rate in P2 or PUPI + PARAM in Data In)	No
–		xx	-----					No of repetitions of the command.	No
0		--	-----					P2 has other parameters (all others are defaults).	No
1		--	-----					Data In contains command data, P2 not used.	No

8.1.11 ISO/IEC 14443-3 P2 coding

b7	b6	b5	b4	b3	b2	b1	b0	Description	Supported
Number of slots for REQB/WUPB command									
RFU					xxx			N = $2^{(b2b1b0)}$ (that is, for b2b1b0 = 0, N = 1).	Yes
Slot no for Slot-MARKER command									
RFU				xxxx				Slot number (0001 = 2, 1111 = 16).	No
Bit rate for ATTRIB command									
--		---			RFU	0	0	Tx - 106 kbps.	No
		---				0	1	Tx - 212 kbps.	No
		---				1	0	Tx - 424 kbps.	No
		---				1	1	Tx - 848 kbps.	No
---		RFU	0	0	---			Rx - 106 kbps.	No
			0	1	---			Rx - 212 kbps.	No
			1	0	---			Rx - 424 kbps.	No
			1	1	---			Rx - 848 kbps.	No
CID		------						Card Identifier.	No

8.1.12 ISO/IEC 14443-3 SW1SW2 response bytes

Data Out	SW1SW2	
XX bytes (if Le present)	0x9000	Operation successful.
–	0x6700	Wrong length (e.g. Lc absent when P1 b7 is set).
	0x6400	Execution error (e.g. command timeout).
	0x6401	Internal error (e.g. protocol setup failed or transceive failed).
	0x6A88	Requested buffer size too big.

8.1.13 Cases for which data out is command dependent

Command	Data Out
REQA	ATQA (2 bytes).
WUPA	ATQA (2 bytes).
HLTA	–
REQA + ANTI-COLLISION + SELECT	UID (4,7 or 10 bytes) + SAK (1 byte).
ANTI-COLLISION CL1	Cascade UID (4 bytes) + BCC (1 byte).
ANTI-COLLISION CL2	Cascade UID (4 bytes) + BCC (1 byte).
ANTI-COLLISION CL3	Cascade UID (4 bytes) + BCC (1 byte).
SELECT	SAK (1 byte).
REQB	ATQB (14 bytes).
WUPB	ATQB (14 bytes).
HLTB	0x00 + CRCB (2 bytes).
Slot-MARKER	ATQB (14 bytes).
ATTRIB	MBLI+CID (1 byte) + CRCB (2 bytes).

Note: ATQB comprises: 0x50 + PUPI (4 bytes) + APP (4 bytes) + PROTO (3 bytes) +CRCB (2 bytes).

8.1.14 0x98 - ISO/IEC 14443-4 command APDU

CLA	INS	P1	P2	Pc	Data In	Le
0xFF	0x98	XX	XX	XX	Lc bytes	XX

8.1.15 ISO/IEC 14443-4 P1 coding

b7	b6	b5	b4	b3	b2	b1	b0	Description	Supported
RFU						0	0	ISO/IEC 14443 Type A RATS (FSDI+CID in P2).	No
						0	1	ISO/IEC 14443 Type A PPS (Bit rate in P2).	No
						1	0	T=CL transmit - Data In contains data to be sent, including the PCB and CID bytes.	No
						1	1	T=CL transmit - Data In contains data to be sent (I-Block only), PCB and CID bytes handled by reader.	No

8.1.16 ISO/IEC 14443-4 P2 coding

b7	b6	b5	b4	b3	b2	b1	b0	Description	Supported
FSDI+CID for RATS command									
FSDI				CID				FSDI codes FSD as in ISO/IEC 14443-4.	No
Bit rate for PPS command									
RFU		---			RFU	0	0	Tx - 106 kbps.	No
		---				0	1	Tx - 212 kbps.	No
		---				1	0	Tx - 424 kbps.	No
		---				1	1	Tx - 848 kbps.	No
		RFU	0	0	---			Rx - 106 kbps.	No
			0	1	---			Rx - 212 kbps.	No
			1	0	---			Rx - 424 kbps.	No
			1	1	---			Rx - 848 kbps.	No

8.1.17 ISO/IEC 14443-4 response bytes

Data Out	SW1SW2	
XX bytes (if Le present)	0x9000	Operation successful.
–	0x6700	Wrong length (e.g. Lc absent for transmit commands).
	0x6400	Execution error (e.g. command timeout).
	0x6A88	Requested buffer size too big.

Note: Data Out may also contain an SW1SW2 from the card.

8.1.18 0x9A: ICAO Miscellaneous command APDU

CLA	INS	P1	P2	Pc	Data In	Le
0xFF	0x9A	XX	XX	XX	Lc bytes	XX

8.1.19 ICAO Miscellaneous P1 coding

b7	b6	b5	b4	b3	b2	b1	b0	Description	Supported
0	0	0	0	0	0	0	1	Reader information (coded in P2).	No
0	0	0	0	0	0	1	0	ISO/IEC 14443 Type A trigger signal - NOT SUPPORTED.	No
0	0	0	0	0	0	1	1	ISO/IEC 14443 Type B trigger signal - NOT SUPPORTED.	No
0	0	0	0	0	1	0	0	Reader control (coded in P2).	Yes

Note: All other values of P1 are RFU.

8.1.20 ICAO Miscellaneous P2 coding

b7	b6	b5	b4	b3	b2	b1	b0	Description	Supported
Coding for Reader information (Lc absent, Le present)									
0	0	0	0	0	0	0	1	Vendor name.	No
0	0	0	0	0	0	1	0	Vendor ID.	No
0	0	0	0	0	0	1	1	Product name.	No
0	0	0	0	0	1	0	0	Product ID.	No
0	0	0	0	0	1	0	1	Product serial number.	No
0	0	0	0	0	1	1	0	Product firmware version.	No
Coding for Reader control (Lc and Le both absent)									
0	0	0	0	0	0	0	0	Turn off polling for card (enter test mode).	Yes
0	0	0	0	0	0	0	1	Turn on polling for card (exit text mode).	Yes

Note: All other values of P2 are either RFU or not supported.

8.1.21 ICAO Miscellaneous response

Data Out	SW1SW2	
XX bytes (if Le present)	0x9000	Operation successful.
–	0x6700	Wrong length (e.g. Le absent for reader information).
	0x6A82	Function not supported.
	0x6A89	Information not available.
	0x6A90	Trigger signal not available.

Appendix A

A Using PC_to_RDR_Escape command

The PC/SC layer does not allow the use of the SCardTransmit API unless the reader has previously signalled the presence and activation of a card. This prevents the use of commands such as the ICAO test commands or the HID commands without being able to properly recognize and activate a card. In order to be able to use these commands even without a previous card activation, the same functionality of pseudo-APDUs (CLA = 'FF') is provided through the `PC_to_RDR_Escape` command.

To use the `PC_to_RDR_Escape` command with the default Microsoft CCID driver, the functionality must be first enabled in the Windows registry.

To issue the `PC_to_RDR_Escape` command without a card being present, the reader must be first opened with the `SCardConnect` function with the following settings:

```
dwShareMode = SCARD_SHARE_DIRECT
dwPreferredProtocols = 0
```

Then the vendor IOCTL for the Escape command is defined as follows:

```
#define IOCTL_CCID_ESCAPE SCARD_CTL_CODE(3500)
```

The following is an example of the call:

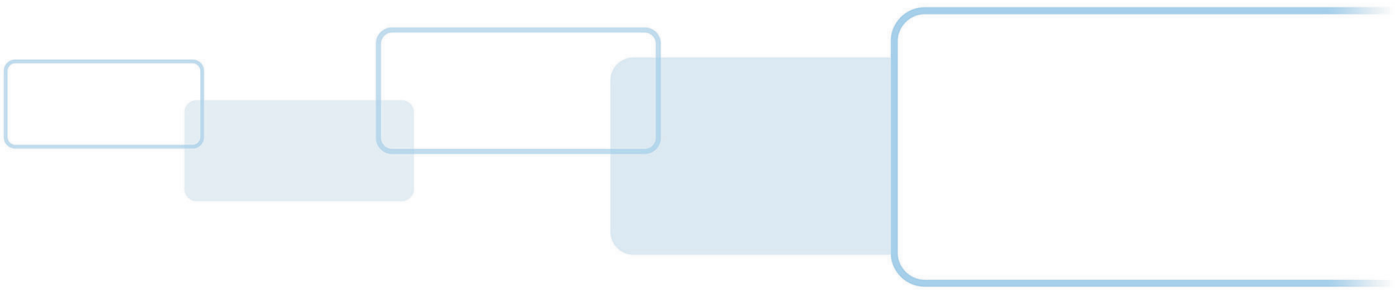
```
SCardControl(hCard, IOCTL_CCID_ESCAPE, …)
```

or:

```
SCardControl(hCard, SCARD_CTL_CODE(3500), …)
```

The data in the *lpInBuffer* parameter of the length given in *nInBufferSize* are copied to the abData field of the `PC_to_RDR_Escape` command and all the data in the response in `RDR_to_PC_Escape` *abData* field are copied back to the *lpOutBuffer*.

The abData field of the `PC_to_RDR_Escape` must contain the pseudo-APDU to be executed (typically, an ICAO test command or reader configuration). The maximum allowed size of *abData* in `PC_to_RDR_Escape` is currently 262 bytes and the maximum size of the response data in the abData field in the `RDR_to_PC_Escape` response is 464 bytes. The `PC_to_RDR_Escape` and `RDR_to_PC_Escape` do not support any form of chaining to extend the lengths of the supported parameters.



hidglobal.com