circle

CIR315B Dual Interface Smart Card Reader Developer Guide

Version 2.00 | August 2021

Revision History

Version	Date	Details
v2.00	6 Aug 2021	Format restructured

Contents

1.0. Introduction

1.1. Purpose

The developer guide is a detailed document designed for system integrators and software developers to illustrate how to integrate the Circle's USB Dual Interface Smart Card Reader into their applications.

The following products are covered by this document:

- CIR315B Dual Interface Smart Card Reader

Should you have questions, please contact support@abcircle.com.

1.2. Glossary

Term	Description
AFI	Application Family Identifier
APDU	Application Protocol Data Unit
API	Application Programming Interface
ATQB	Answer to request, Type B
ATR	Answer to Reset
ATS	Answer to Select
ATTRIB	PICC selection command, Type B
BER-TLV	Basic Encoding Rules – Tag Length Value
CCID	Integrated Circuit(s) Card Interface Devices
CID	Card Identifier
CSN	Card Serial Number
Dd	Default Value of Baud Rate Adjustment Integer
Di	Indicated Value of Baud Rate Adjustment Integer
DSFID	Data Storage Format Identifier
ETU	Elementary Time Unit
Fd	Default Value of Clock Rate Conversion Integer
Fi	Indicated Value of Clock Rate Conversion Integer
ICC	Integrated Circuit Card, i.e. Contact Smart Card
IDm	Manufacture ID
LED	Light Emitting Diode
LSB	Least Significant Byte
MBLI	Maximum Buffer Length Index, Type B
MSB	Most Significant Byte
NAD	Node Address
NFC	Near Field Communications
PCB	Protocol Control Byte
PC/SC	Personal Computer/Smart Card
PICC	Proximity Integrated Circuit Card
PPS	Protocol Parameter Selection
PUPI	Pseudo-Unique PICC Identifier
RATS	Request for Answer to Select
RFU	Reserved for Future Use
SAK	Select Acknowledge
SAM	Secure Access Module
SW	Status Word
TPDU	Transmission Protocol Data Unit
UID	Unique Identifier
USB	Universal Serial Bus

2.0. Product Information

2.1. Key features

- Supports ISO/IEC 14443 Type A & B, ISO/IEC 18092, ISO/IEC 15693, ISO/IEC 7816 standards
- Supports ISO/IEC 14443-4 Type A & B cards, ISO/IEC 15693-3 cards, MIFARE® series, FeliCa, J-LIS cards, Calypso and NFC Forum Type 1/2/3/4/5 Tags
- Supports 1 x ISO/IEC 7816 Part 1/2/3/4 standard full-sized card slot
- Supports 1 x ISO/IEC 7816 Part 3/4 SAM slot (Optional)
- Supports full-sized ISO/IEC 7816 Class A/ B/ C cards with T=0 or T=1 protocol
- Supports extended APDU
- Complies with CCID standard
- Conforms to PC/SC specifications
- Firmware upgradable
- Works on different operating systems – Windows®, Linux®, macOS® and Android™
- Supports reader configuration (Escape commands)

2.2. Ordering information

Product	Product Code	Product Photo
CIR315B Dual Interface Smart Card Reader	CIR315B-01: USB Type A Connector, with SAM slot (Standard model)	
	CIR315B-02: USB Type A Connector, without SAM slot	
	CIR315B-03: USB Type C Connector, with SAM slot	
	CIR315B-04: USB Type C Connector, without SAM slot	

Should you have inquiries, please contact sales@abcircle.com.

2.3. Driver and tools

Circle's proprietary drivers and libraries need to be installed in order for the reader(s) to operate. Circle has developed drivers and libraries for the Windows®, Linux®, macOS® and Android™ platform. All can be downloaded from www.abcircle.com.

2.4. Host interface

The reader(s) is a USB 2.0 Full Speed device compliant with CCID 1.1 specification. CCID is a USB protocol defined for the vendors of smart card readers to standardize the communication protocol between the reader and the host. Details of the CCID specification can be found at https://www.usb.org/sites/default/files/DWG_Smart-Card_CCID_Rev110.pdf.

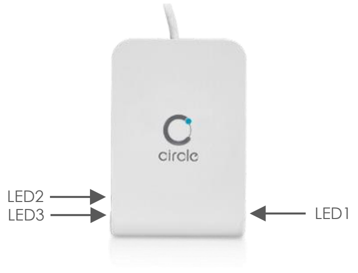
Below is a list of CCID command supported by the reader.

CCID Command	Description
PC_to_RDR_IccPowerOn	Power on the inserted ICC
PC_to_RDR_IccPowerOff	Power off the inserted ICC
PC_to_RDR_GetSlotStatus	Get the card presence status of a slot
PC_to_RDR_XfrBlock	Perform data exchange
PC_to_RDR_GetParameters	Get the current communication parameters
PC_to_RDR_ResetParameters	Reset the communication parameters to default
PC_to_RDR_SetParameters	Set the parameters for subsequent communication
PC_to_RDR_Escape	Send escape command

The reader supports short APDU exchange for both T=0 and T=1 protocols and extended APDU exchange for T=1 protocol.

2.5. Human Interface

The reader(s) has three LEDs to indicate the power and card operation status. It is also equipped with a monotone buzzer to provide additional information on the card operation status.

Product	Buzzer	LED1 – Green (Power)	LED2 – Blue (Contactless Card Operation)	LED3 – Blue (Contacts Card Operation)	Product Photo
CIR315B Contact Smart Card Reader	✓	✓	✓	✓	

LED2 and LED3 are configurable. See section 7.5 and 7.6 for more details. Below is the default LEDs behavior:

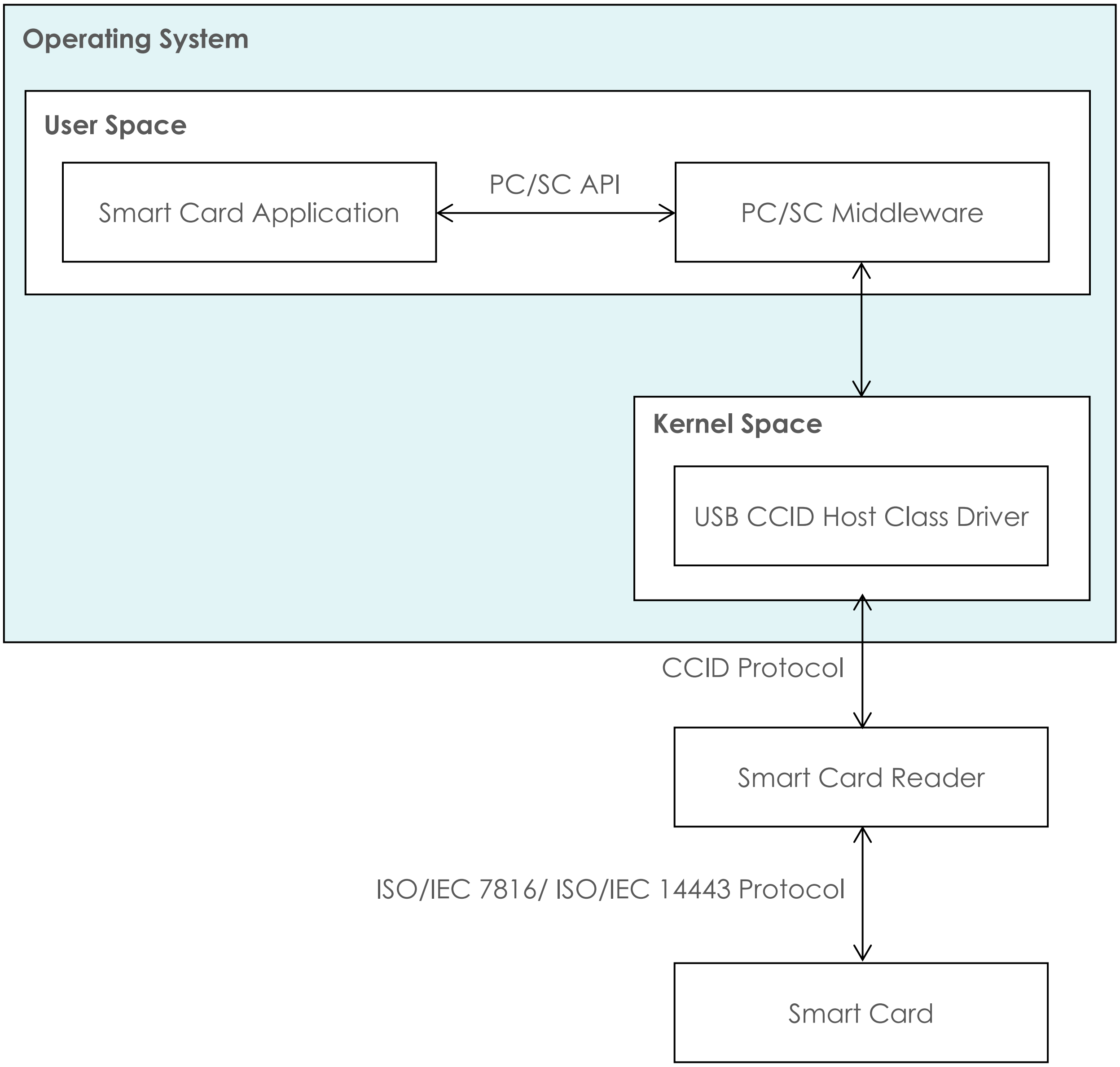
Status	LED1 – Green (Power)	LED2 – Blue (Contactless Card Operation)	LED3 – Blue (Contact Card Operation)
Reader power on	On	On for 10 sec, then off	On for 10 sec, then off
Card is absent or inactive		Off	Off
Card is present and active		On	On
Data transmission		Fast Blinking	Fast Blinking

The buzzer is also configurable. See section 7.5 and 7.6 for more details. Below is the default buzzer behavior:

Status	Buzzer
Contactless interface is initiated	1 Beep
Contactless Card is detected	1 Beep
Contactless Card is removed	1 Beep
Contact Card is detected	1 Beep
Contact Card is removed	1 Beep

3.0.Host programming API (PC/SC)

This is a specification that ensures the interoperability between smart card applications and the smart cards. This standard is implemented on most of the operating systems, including Windows®, Linux® and macOS®. Below is a simple illustration of the role of PC/SC, CCID and ISO/IEC 7816/ ISO/IEC 14443 in a typical smart card system.



3.1. Establish Context

The **SCardEstablishContext** function establishes the resource manager context (the scope) within which database operations are performed. [1]

```
LONG SCardEstablishContext(
  DWORD          dwScope,
  LPCVOID        pvReserved1,
  LPCVOID        pvReserved2,
  LPSCARDCONTEXT phContext
);
```

3.2. List Readers

The **SCardListReaders** function provides the list of readers within a set of named reader groups, eliminating duplicates.

The caller supplies a list of reader groups, and receives the list of readers within the named groups. Unrecognized group names are ignored. This function only returns readers within the named groups that are currently attached to the system and available for use. [2]

```
LONG SCardListReadersA(
  SCARDCONTEXT hContext,
  LPCSTR       mszGroups,
  LPSTR        mszReaders,
  LPDWORD      pcchReaders
);
```

3.3. Get Status Change

The **SCardGetStatusChange** function blocks execution until the current availability of the cards in a specific set of readers changes.

The caller supplies a list of readers to be monitored by an SCARD_READERSTATE array and the maximum amount of time (in milliseconds) that it is willing to wait for an action to occur on one of the listed readers. [3]

```
LONG SCardGetStatusChangeA(
  SCARDCONTEXT         hContext,
  DWORD                dwTimeout,
  LPSCARD_READERSTATEA rgReaderStates,
  DWORD                cReaders
);
```

3.4. Connect

The **SCardConnect** function establishes a connection (using a specific resource manager context) between the calling application and a smart card contained by a specific reader. If no card exists in the specified reader, an error is returned. [4]

```
LONG SCardConnectA(
  SCARDCONTEXT  hContext,
  LPCSTR        szReader,
  DWORD         dwShareMode,
  DWORD         dwPreferredProtocols,
  LPSCARDHANDLE phCard,
  LPDWORD       pdwActiveProtocol
);
```

3.5. Status

The **SCardStatus** function provides the current status of a smart card in a reader. You can call it any time after a successful call to **SCardConnect** and before a successful call to **SCardDisconnect**. It does not affect the state of the reader or reader driver. [5]

```
LONG SCardStatusA(
  SCARDHANDLE hCard,
  LPSTR       mszReaderNames,
  LPDWORD     pcchReaderLen,
  LPDWORD     pdwState,
  LPDWORD     pdwProtocol,
  LPBYTE      pbAtr,
  LPDWORD     pcbAtrLen
);
```

3.6. Data Exchange with Card

The **SCardTransmit** function sends a service request to the smart card and expects to receive data back from the card. [6]

```
LONG SCardTransmit(
  SCARDHANDLE         hCard,
  LPCSCARD_IO_REQUEST pioSendPci,
  LPCBYTE             pbSendBuffer,
  DWORD               cbSendLength,
  LPSCARD_IO_REQUEST  pioRecvPci,
  LPBYTE              pbRecvBuffer,
  LPDWORD             pcbRecvLength
);
```

3.7. Reader Control

The **SCardControl** function gives you direct control of the reader. This is typically used to send escape commands to reader. You can call it any time after a successful call to **SCardConnect** and before a successful call to **SCardDisconnect**. The effect on the state of the reader depends on the control code. [7]

```
LONG SCardControl(
  SCARDHANDLE hCard,
  DWORD       dwControlCode,
  LPCVOID     lpInBuffer,
  DWORD       cbInBufferSize,
  LPVOID      lpOutBuffer,
  DWORD       cbOutBufferSize,
  LPDWORD     lpBytesReturned
);
```

3.8. Disconnect

The **SCardDisconnect** function terminates a connection previously opened between the calling application and a smart card in the target reader. [8]

```
LONG SCardDisconnect(
  SCARDHANDLE hCard,
  DWORD       dwDisposition
);
```

3.9. Release Context

The **SCardReleaseContext** function closes an established resource manager context, freeing any resources allocated under that context, including SCARDHANDLE objects and memory allocated using the SCARD_AUTOALLOCATE length designator. [9]

```
LONG SCardReleaseContext(
  SCARDCONTEXT hContext
);
```

4.0.APDU Definition

APDU the communication unit between the reader and the smart card. Its format follows ISO/IEC 7816-4 standard. There are two types of APDUs: short APDU and extended APDU.

4.1. Short APDU

In a short APDU, each field occupies one byte. Lc and Le are the write and read data length respectively. The maximum number of data bytes that can be exchanged by a short APDU is no larger than 256 bytes.

Command APDU definition

CLA	INS	P1	P2	Lc	Command Data	Le

CLA – Instruction class
INS – Instruction code
P1, P2 – Command parameters
Lc – Number of command data bytes
Command Data – Lc number bytes of data
Le – Maximum number of expected response bytes
If the length of command data is 0, Lc must be omitted.

Response APDU definition

Data field	SW1	SW2

Data field – The data returns from the card
SW1 SW2 – Status bytes for the response

4.2. Extended APDU

For extended APDU, the data length field, Lc and Le, expands to 3 bytes and that increases the maximum data length to 65535 bytes. Due to the limited amount of memory a reader has, an extended APDU has to be divided into multiple chunks of smaller packets for the reader to handle. This process requires works from both the host and the reader. The host divides the extended APDU into chunks of smaller packets and feed them to the reader through CCID interface. Reader then handles the chaining of these packets and feed them to the smart card accordingly. Since T=1 protocol supports chaining natively, extended APDU is supported by the reader with T=1 protocol. With T=0 protocol, the heavy lifting will need to be done by the application software however.

Command APDU definition

CLA	INS	P1	P2	Lc1	Lc2	Lc3	Command Data	Le1	Le2	Le3

CLA – Instruction class
INS – Instruction code
P1, P2 – Command parameters
Lc1, Lc2, Lc3 – Number of command data bytes (where Lc1 must be 00)
Command Data – Lc number bytes of data
Le1, Le2, Le3 – Maximum number of expected response bytes (where Le1 must be 00)
If the length of command data is 0, Lc must be omitted.

Response APDU definition

Data field	SW1	SW2

Data field – The data returns from the card
SW1 SW2 – Status bytes for the response

5.0.Contactless Card

5.1. Contactless Card Interface

The reader supports single polling mode with polling period 250 ms as default.

The 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 the protocol.
2. When a card is detected, anti-collision procedure is executed to select a single card. When the reader and the card (ISO/IEC 14443-4 Type A & B cards) support higher bit rate, the speed exchange is performed automatically.
3. The reader informs the host card is found through CCID protocol and the card details (ATR) is obtained through CCID protocol.
4. APDU layer communication is now possible through the CCID interface.
5. The reader continues to poll for card removal.
6. Once the card is removed, the reader informs the host card is removed through CCID protocol.

5.2. ATR Generation

Unlike contact cards, contactless cards do not generate an ATR. Instead, they generate an ATS. To make contactless cards available within the PC/SC framework, the reader generates a PC/SC compliant ATR according to PC/SC v2.01.

<table>
<tr><th>Byte Nr</th><th>Value</th><th>Designation</th><th colspan="3">Description</th></tr>
<tr><td>0</td><td>0x3B</td><td>Initial Header</td><td colspan="3"></td></tr>
<tr><td>1</td><td>0x8n</td><td>T0</td><td colspan="3">‘n’ is the number of historical bytes in the ATR.</td></tr>
<tr><td>2</td><td>0x80</td><td>TD1</td><td colspan="3">T=0 protocol.</td></tr>
<tr><td>3</td><td>0x01</td><td>TD2</td><td colspan="3">T=1 protocol.</td></tr>
<tr><td rowspan="6">4 to 3+n</td><td rowspan="6">XX …</td><td rowspan="6">T1 … Tk</td><td colspan="3">Historical bytes (optional):</td></tr>
<tr><td colspan="3"><i>ISO/IEC 14443-Type A:</i>
The historical bytes from ATS response.</td></tr>
<tr><td rowspan="4"><i>ISO/IEC 14443-Type B:</i></td><td>Byte Nr</td><td>Description</td></tr>
<tr><td>1 to 4</td><td>Application Data from ATQB.</td></tr>
<tr><td>5 to 7</td><td>Protocol Info Byte from ATQB.</td></tr>
<tr><td>8</td><td>0xa0
‘a’ = MBLI from ATTRIB command.</td></tr>
<tr><td>4+n</td><td>XX</td><td>TCK</td><td colspan="3">XOR value of bytes from T0 to Tk</td></tr>
</table>

Table 1 - ATR Returned for Contactless Smart Card

Byte Nr	Value	Designation	Description		
0	0x3B	Initial Header			
1	0x8F	T0	15 historical bytes in the ATR.		
2	0x80	TD1	T=0 protocol.		
3	0x01	TD2	T=1 protocol.		
4 to 18	0x80	T1 … Tk	Category indicator byte, must be 0x80.		
	0x4F		Application identifier Presence indicator, must be 0x4F.		
	0x0C		Length, must be 0x0C.		
	0xA0		Registered application provider identifier (RID), must be 0xA0 0x00 0x00 0x03 0x06 – This is the RID of PC/SC Workgroup.		
	0x00				
	0x00				
	0x03				
	0x06				
	XX		PIX	SS	Byte For Standard. (See Table 3 - SS Byte definition for standard)
	XX			NN	Bytes for Card Name. (See Table 4 - NN Byte definition for card name)
	XX			NN	
	0x00			RR	RFU: Set to zero.
	0x00			RR	
	0x00			RR	
	0x00			RR	
19	XX	TCK	XOR value of bytes from T0 to Tk		

Table 2 - ATR Returned for Contactless Storage Card

SS Byte	Description
0x00	No information given
0x01	ISO/IEC 14443 Type A, part 1
0x02	ISO/IEC 14443 Type A, part 2
0x03	ISO/IEC 14443 Type A, part 3
0x04	RFU
0x05	ISO/IEC 14443 Type B, part 1
0x06	ISO/IEC 14443 Type B, part 2
0x07	ISO/IEC 14443 Type B, part 3
0x08	RFU
0x09	ISO/IEC 15693, part 1
0x0A	ISO/IEC 15693, part 2
0x0B	ISO/IEC 15693, part 3
0x0C	ISO/IEC 15693, part 4
0x11	FeliCa
…	RFU
0xFF	RFU

Table 3 - SS Byte definition for standard

Card Name	Two Byte – Identifier
MIFARE® Standard 1K	0x00 0x01
MIFARE® Standard 4K	0x00 0x02
MIFARE Ultralight®	0x00 0x03
SRF55V10P	0x00 0x0E
SRF55V02P	0x00 0x0F
TAG_IT	0x00 0x12
LRI512	0x00 0x13
ICODESLI	0x00 0x14
PicoPass 2K	0x00 0x17
PicoPass 2KS	0x00 0x18
PicoPass 16K	0x00 0x19
PicoPass 16KS	0x00 0x1A
PicoPass 16K(8x2)	0x00 0x1B
PicoPass 16KS(8x2)	0x00 0x1C
PicoPass 32KS(16+16)	0x00 0x1D
PicoPass 32KS(16+8x2)	0x00 0x1E
PicoPass 32KS(8x2+16)	0x00 0x1F
PicoPass 32KS(8x2+8x2)	0x00 0x20
MIFARE Mini®	0x00 0x26
Topaz (NFC Forum Type 1 Tag)	0x00 0x30
MIFARE Ultralight C®	0x00 0x3A
FeliCa	0x00 0x3B

Table 4 - NN Byte definition for card name

5.3. Contactless Card Communications

5.3.1. Get Card Data

This command retrieves specific information, such as the UID or CSN or ATS of the card, for all card supported.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xCA	**P1**	0x00	-	-	XX

P1	Description
0x00	Get UID of the card
0x01	Get ATS of ISO/IEC 14443 Type A card

Le = 0x00, this means Return full length of the data.

Response:

Data field	SW1 SW2
Data	See the following table

P1	Card type	**Data**
0x00	MIFARE®	4, 7-byte UID
	ISO/IEC 14443 Type A	4, 7 or 10-byte UID
	ISO/IEC 14443 Type B	4-byte PUPI
	FeliCa	8-byte IDm
	Topaz (NFC Forum Type 1 Tag)	7-byte UID
	ISO/IEC 15693	8-byte UID
0x01	ISO/IEC 14443 Type A	ATS

SW1	SW2	Description
0x90	0x00	Operation successful
0x62	0x82	Wrong length, Le is greater than data length
0x6A	0x81	Function not supported
0x6C	XX	Wrong length, Le is less than the available UID length. 'XX' encodes the exact number.

5.3.2. MIFARE® Card

MIFARE® Card command APDU

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	**INS**	P1	P2	XX	XX	XX

INS	Description	MIFARE Classic® 1K/4K	MIFARE Ultralight®
0x82	Load Keys command	✓	×
0x86	General Authenticate command	✓	×
0xB0	Read Binary command	✓	✓
0xD6	Update Binary command	✓	✓
0xB1	Read Value Block command	✓	×
0xD7	Update Value Block command	✓	×
	Copy Value Block command	✓	×

5.3.2.1. Load Keys

This command loads the authentication keys into the reader.

Command:

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

Key Structure	**Key Number**	**Key Length**	**Key**
0x00	0x00: Key slot 0	0x06	6-bytes length of Key for MIFARE® authentication
	0x01: Key slot 1		

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.2.2. General Authenticate

This command authenticates MIFARE® 1K/4K card. Before using this command, the correct keys must have been loaded to the relevant key slot.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0x86	0x00	0x00	0x05	**Data**	-

Data				
Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
Version	Address MSB	Address LSB	Key Type	Key Number
0x01	0x00	Block number to be authenticated	0x60: Key A	0x00: Key slot 0
			0x61: Key B	0x01: Key slot 1

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.2.3. Read Binary

This command retrieves the block data from the card. (*For MIFARE® Classic 1K/4K card, this requires a prior general authenticate command to succeed.*)

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xB0	**Address MSB**	**Address LSB**	-	-	**Le**

	Address MSB	**Address LSB**	**Le**
MIFARE Classic® 1K/4K	0x00	Block number	0x10
MIFARE Ultralight®			0x04/0x08/0x0C/0x10

Response:

Data field	SW1 SW2
Data	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.2.4. Update Binary

This command writes the block data into the card. (*For MIFARE Classic® 1K/4K card, this requires a prior general authenticate command to succeed.*)

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xD6	**Address MSB**	**Address LSB**	**Lc**	**Data**	-

	Address MSB	**Address LSB**	**Lc**	**Data**
MIFARE Classic® 1K/4K	0x00	Block number	0x10	16-byte data
MIFARE Ultralight®			0x04	4-byte data

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.2.5. Read Value Block

This command retrieves the value of Value block from the card. (*For MIFARE Classic® 1K/4K card, this requires a prior general authenticate command to succeed and the block data is in the fixed data format of Value block.*)

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xB1	**Address MSB**	**Address LSB**	-	-	**Le**

Address MSB	**Address LSB**	**Le**
0x00	Block number	0x04

Response:

Data field	SW1 SW2
Value	See the following table

Value (signed 4-byte 2'complement value)			
Byte 1	Byte 2	Byte 3	Byte 4
MSB	XX	XX	LSB

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.2.6. Update Value Block

This command writes the value of Value block into the card. (*For MIFARE Classic® 1K/4K card, this requires a prior general authenticate command to succeed and the block data is in the fixed data format of Value block.*)

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xD7	**Address MSB**	**Address LSB**	**Lc**	**Data**	-

Address MSB	**Address LSB**	**Lc**	**Data**				
0x00	Block number	0x05	Operation	Value (signed 4-byte 2'complement value)			
			Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
			0x00: Write	MSB	XX	XX	LSB
			0x01: Increment				
			0x02: Decrement				

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.2.7. Copy Value Block

This command copies the value of Value block from source address to destination address. (*For MIFARE Classic® 1K/4K card, this requires a prior general authenticate command to succeed and the block data is in the fixed data format of Value block.*)

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xD7	**Address MSB**	**Address LSB**	**Lc**	**Data**	-

Address MSB	**Address LSB**	**Lc**	**Data**	
0x00	Source Block number	0x02	Operation	Address LSB
			Byte 1	Byte 2
			0x03	Destination Block number

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.3. ISO/IEC 15693 Memory Card

ISO/IEC 15693 Memory Card command APDU

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	**INS**	P1	P2	XX	XX	XX

INS	Description
0x30	Get Data command
	Put Data command
	Lock command
	Get Security Status command
0xB0	Read Binary command
0xD6	Update Binary command

5.3.3.1. Get Data

This command retrieves information from the card.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0x30	**Data Type**	0x00	-	-	0x00

Data Type	Description
0x02	AFI of ISO/IEC 15693 card is returned if supported
0x03	DSFID of ISO/IEC 15693 card is returned if supported
0x04	PICC memory size is returned if supported
0x05	IC reference is returned if supported

Response:

Data field	SW1 SW2
Information	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails
0x64	XX	Operation fails 'XX' is the error status from the card

5.3.3.2. Put Data

This command writes the data into the system information of the card.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0x30	0x00	0x01	0x04	**Information**	-

Information			
Byte 1	Byte 2	Byte 3	Byte 4
Version	Flag1	Flag2	Data
0x01	0x02: AFI 0x03: DSFID	0x00	Value

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails
0x64	XX	Operation fails 'XX' is the error status from the card

5.3.3.3. Lock

This command locks the memory area of the card.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0x30	0x00	0x02	**Lc**	**Information**	-

Lc	**Information**				
	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
0x05	0x01	0x00: Data Block	0x00	Block Address MSB	Block Address LSB
0x03		0x02: AFI		-	-
0x03		0x03: DSFID		-	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails
0x64	XX	Operation fails 'XX' is the error status from the card

5.3.3.4. Get Security Status

This command retrieves the Block Security Status of the memory area of the card.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0x30	0x00	0x03	**Lc**	**Information**	-

Lc	**Information**				
	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
0x05	0x01	0x00	0x00	0x00	Block Address

Response:

Data field	SW1 SW2
Block Security Status	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails
0x64	XX	Operation fails 'XX' is the error status from the card

5.3.3.5. Read Binary

This command retrieves the block data from the card.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xB0	**Address MSB**	**Address LSB**	-	-	**Le**

Address MSB	**Address LSB**	**Le**
0x00	Block number	Length of the data to be read (Multiple of Block size)

Response:

Data field	SW1 SW2
Data	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails
0x64	XX	Operation fails 'XX' is the error status from the card

5.3.3.6. Update Binary

This command writes the block data into the card.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xD6	**Address MSB**	**Address LSB**	**Lc**	**Data**	-

Address MSB	**Address LSB**	**Lc**	**Data**
0x00	Block number	Length of the data to be written (Multiple of Block size)	1 block of data

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.4. FeliCa Card

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xAB	0x00	0x00	**Lc**	**Data**	-

	Lc	**Data**						
		FeliCa Command Structure						
	Command Length	Command code	Command information					
Command Example	Length	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	…	Byte N
Polling	0x05	0x00	SysCode		ReqCode	TimeSlot	-	-
Read	Variable	0x06	Payload					
Write	Variable	0x08	Payload					

Response:

Data field	SW1 SW2
Data	See the following table

Data				
Byte 1	Byte 2	Byte 3 – Byte 10	…	Byte N
Response Length	Response code	IDm	XX	XX

SW1	SW2	Description
0x90	0x00	Operation successful
0x64	0x01	Operation fails

5.3.5. Topaz (NFC Forum Type 1 Tag)

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xAB	0x00	0x00	**Lc**	**Data**	-

	Lc	**Data**			
		Topaz Command Structure			
	Command Length	Command code	Command information		
Command Example	Length	Byte 1	Byte 2	Byte 3	Byte 4 – Byte 7
RALL	0x07	0x00	0x00	0x00	UID
READ	0x07	0x01	Address	0x00	UID
WRITE-E	0x07	0x53	Address	Data	UID

Response:

Data field	SW1 SW2
Data	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.6. Picopass

5.3.6.1. Read Binary

This command retrieves the block data from the card.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xB0	**Address MSB**	**Address LSB**	-	-	**Le**

Address MSB	**Address LSB**	**Le**
0x00	Block number	0x08/0x20

Response:

Data field	SW1 SW2
Data	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.6.2. Update Binary

This command writes the block data into the card.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xD6	**Address MSB**	**Address LSB**	**Lc**	**Data**	-

Address MSB	**Address LSB**	**Lc**	**Data**
0x00	Block number	0x08	8-byte data

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation successful
0x63	0x00	Operation fails

5.3.7. PC/SC Transparent command

Transparent command passes data from an application to a contactless card transparently, and returns data transparently to the application.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xC2	0x00	**Function**	Variable	Command Data Object (BER-TLV encoded)	--

Function	Description	Remark
0x00	Manage Session command	Uses to manage the environment, capabilities and contexts
0x01	Transparent Exchange command	Uses to exchange bit/bytes between card and reader
0x02	Switch Protocol command	Uses to switch the protocol between different layers

5.3.7.1. Manage Session command

Manage session command starts and ends a transparent session, manages the environment and extracts or sets the capabilities of the reader for the following communication (transparent session).

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xC2	0x00	0x00	Variable	**Command Data Object** (BER-TLV encoded)	--

The data filed shall contain at least one Data Object.

Description	**Command Data Object** (BER-TLV encoded)					
	Tag	Length	Value			
Version	0x80	0x03	Major	Minor	Build	
Start Transparent Session	0x81	0x00	-			
End Transparent Session	0x82	0x00	-			
Turn Off RF Field	0x83	0x00	-			
Turn On RF Field	0x84	0x00	-			
Timer	0x5F46	0x04	Timer (ULONG) 32-bit unsigned long value in microseconds			
			Byte 1	Byte 2	Byte 3	Byte 4
			LSB	XX	XX	MSB
Get Parameters	0xFF6D	Variable	Tag	Length	Value	
			See Table 5			
Set Parameters	0xFF6E	Variable	Tag	Length	Value	
			See Table 6			

Response:

Data field		SW1 SW2
Data Field	**Response Data Object** (BER-TLV encoded)	See ISO/IEC 7816

The response data field can be empty or can contain more than one Data Object. This depends on the Tag(s) of the command data field.

Data Field		
Tag	Length	Value
0xC0	0x03	Error Status (See Table 7)

Description	**Response Data Object** (BER-TLV encoded)				
	Tag	Length	Value		
No response for data object	-	-	-		
Version	0x80	0x03	Major	Minor	Build
Reader parameter	0xFF6D	Variable	Tag	Length	Value
			See Table 5		

Parameters requested	Tag	Length	Value
Frame size for Reader Integer (FSDI)	0x01	0x00	-
Frame size for Card Integer (FSCI)	0x02	0x00	-
Frame waiting Time Integer (FWTI) of the current Card	0x03	0x00	-
Maximum communication speed supported by the Reader	0x04	0x00	-
Communication speed of the current Card	0x05	0x00	-
Modulation index currently	0x06	0x00	-
PCB for ISO/IEC 14443	0x07	0x00	-
CID for ISO/IEC 14443	0x08	0x00	-
NAD for ISO/IEC 14443	0x09	0x00	-
Param 1 to Param 4 for ISO/IEC 14443 Type B	0x0A	0x00	-

Table 5 - Tags for Get Parameters data object

Tag	Length	Value
0x01	0x01	Frame size for Reader Integer (FSDI)
0x02	0x01	Frame size for Card Integer (FSCI)
0x03	0x01	Frame waiting Time Integer (FWTI) of the current Card Time = 302.07 x 2^{FWTI} microseconds
0x04	0x01	Maximum communication speed supported by the Reader
0x05	0x01	Communication speed is set or will be set for the current Card
0x06	0x01	Modulation index (can be used for all Cards, where modulation index is required)
0x07	0x01	PCB for ISO/IEC 14443
0x08	0x01	CID for ISO/IEC 14443
0x09	0x01	NAD for ISO/IEC 14443
0x0A	0x04	Param 1 to Param 4 for ISO/IEC 14443 Type B

Table 6 - Tags for Set Parameters data object

Error Status	Description
XX SW1 SW2	XX = number of the bad data object in the APDU; 00 = general error of APDU 01 = error in the 1st data object 02 = error in the 2nd data object; etc.
00 90 00	No error occurred
XX 62 82	Data object XX warning, requested information not available
XX 63 00	No information
XX 63 01	Execution stooped due to failure in other data object
XX 6A 81	Data object XX not supported
XX 67 00	Data object XX with unexpected length
XX 6A 80	Data object XX with unexpected value
XX 64 00	Data Object XX execution error (no response from reader)
XX 64 01	Data Object XX execution error (no response from card)
XX 6F 00	Data object XX failed, no precise diagnosis

The first value byte indicates the number of the erroneous data object XX and the last two bytes indicate the explanation of the error. Further SW1 SW2 values, according to ISO/IEC 7816, are allowed.

Table 7 - Error Status

5.3.7.2. Transparent Exchange command

Transparent exchange command transmits and receives any bit or bytes from Card.

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xC2	0x00	0x01	Variable	**Command Data Object** (BER-TLV encoded)	--

The data filed shall contain at least one Data Object.

<table>
<tr><th rowspan="2">Description</th><th colspan="3"><b>Command Data Object</b> (BER-TLV encoded)</th></tr>
<tr><th>Tag</th><th>Length</th><th>Value</th></tr>
<tr><td>Transmission and Reception Flag</td><td>0x90</td><td>0x02</td><td>See Table 8</td></tr>
<tr><td>Transmission Bit framing</td><td>0x91</td><td>0x01</td><td>See Table 9</td></tr>
<tr><td>Reception Bit framing</td><td>0x92</td><td>0x01</td><td>See Table 10</td></tr>
<tr><td>Transmit</td><td>0x93</td><td>Variable</td><td>Data to be transmitted</td></tr>
<tr><td>Receive</td><td>0x94</td><td>0x00</td><td>-</td></tr>
<tr><td>Transceive - Transmit and Receive</td><td>0x95</td><td>Variable</td><td>Data to be transmitted</td></tr>
<tr><td>Timer</td><td>0x5F46</td><td>0x04</td><td>Timer (ULONG)
32-bit unsigned long value in microseconds<table><tr><th>Byte 1</th><th>Byte 2</th><th>Byte 3</th><th>Byte 4</th></tr><tr><td>LSB</td><td>XX</td><td>XX</td><td>MSB</td></tr></table></td></tr>
<tr><td>Get Parameters</td><td>0xFF6D</td><td>Variable</td><td><table><tr><th>Tag</th><th>Length</th><th>Value</th></tr><tr><td colspan="3">See Table 5</td></tr></table></td></tr>
<tr><td>Set Parameters</td><td>0xFF6E</td><td>Variable</td><td><table><tr><th>Tag</th><th>Length</th><th>Value</th></tr><tr><td colspan="3">See Table 6</td></tr></table></td></tr>
</table>

Response:

Data field		SW1 SW2
Data Field	**Response Data Object** (BER-TLV encoded)	See ISO/IEC 7816

The response data field can be empty or can contain more than one Data Object. This depends on the Tag(s) of the command data field.

Data Field		
Tag	Length	Value
0xC0	0x03	Error Status (See Table 7)

Description	**Response Data Object** (BER-TLV encoded)		
	Tag	Length	Value
No response for data object	-	-	-
Number of valid bits in the last byte of received data	0x92	0x01	See Table 10
Response Status	0x96	0x02	See Table 11
Response	0x97	Variable	See Table 12
Reader parameter	0xFF6D	Variable	See Table 5

Tag	Length	Value	
		Bit	Description
0x90	0x02	0	0 – Append CRC in the transmit data according to existing protocol 1 – Do NOT append CRC in the transmit data
		1	0 – Discard CRC from the received data according to existing protocol 1 – Do NOT discard CRC from the received data
		2	0 – Insert parity in transmit data in existing protocol 1 – Do NOT insert parity
		3	0 – Expect parity in received data in existing protocol 1 – Do NOT expect parity
		4	0 – Append protocol prologue in the transmit data or discard from the response if any 1 – Do NOT append or discard protocol prologue if any (e.g., for ISO/IEC 14443 PCB and CID, NAD)
		5-15	RFU

Table 8 - Transmission and Reception Flag Data Object

Tag	Length	Value	
		Bit	Description
0x91	0x01	0-2	Number of valid bits of the last byte (0 means all bits are valid)
		3-7	RFU

Table 9 - Transmission Bit Framing Data Object

Tag	Length	Value	
		Bit	Description
0x92	0x01	0-2	Number of valid bits of the last byte (0 means all bits are valid)
		3-7	RFU

Table 10 - Reception Bit Framing Data Object

Tag	Length	Value		
		Byte 0		Byte 1
0x96	0x02	Bit 0	0 – CRC is OK (or not checked) 1 – CRC check failed	RFU
		Bit 1	0 – No Collision detected 1 – Collision detected	Collision position
		Bit 2	0 – no parity error detected 1 – parity error detected	RFU
		Bit 3	0 – no framing error detected 1 – framing error detected	RFU
		Bit 4-7	RFU	RFU

Table 11 - Response Status Data Object

Tag	Length	Value
0x97	Variable	Reply from Card

Table 12 - Response Data Object

5.3.7.3. Switch Protocol command

This command switches the specific protocol and different layers of the standard. Switch Protocol command is only allowed in a transparent session (after calling "Start Transparent Session" data object and before calling "End Transparent Session" data object using Manage Session command).

Command:

CLA	INS	P1	P2	Lc	Data In	Le
0xFF	0xC2	0x00	0x02	Variable	**Command Data Object** (BER-TLV encoded)	--

The data filed shall contain at least one Data Object.

Description	**Command Data Object** (BER-TLV encoded)				
	Tag	Length	Value		
Switch Protocol	0x8F	0x02	See Table 13		
Get Parameters	0xFF6D	Variable	Tag	Length	Value
			See Table 5		
Set Parameters	0xFF6E	Variable	Tag	Length	Value
			See Table 6		

Response:

Data field		SW1 SW2
Data Field	**Response Data Object** (BER-TLV encoded)	See ISO/IEC 7816

The response data field can be empty or can contain more than one Data Object. This depends on the Tag(s) of the command data field.

Data Field		
Tag	Length	Value
0xC0	0x03	Error Status (See Table 7)

Description	**Response Data Object** (BER-TLV encoded)				
	Tag	Length	Value		
No response for data object	-	-	-		
Reader parameter	0xFF6D	Variable	Tag	Length	Value
			See Table 5		
Response for Switch Protocol	0x8F	Variable	See Table 14		
	0x5F51				

Tag	Length	Value		
		Byte 0 – Defines the standard type 0x00 - ISO/IEC 14443 Type A 0x01 - ISO/IEC 14443 Type B 0x02 - ISO/IEC 15693 0x03 - FeliCa	Byte 1 – Define the layer to switch 0x00 - If no layer separation 0x01 - Switch to layer 2 0x02 - Switch or activate to layer 3 0x03 - Activate to layer 4	Protocol
0x8F	0x02	0x00	0x02	ISO/IEC 14443 Type A, Layer 2
		0x00	0x03	ISO/IEC 14443 Type A, Layer 3
		0x00	0x04	ISO/IEC 14443 Type A, Layer 4
		0x01	0x02	ISO/IEC 14443 Type B, Layer 2
		0x01	0x03	ISO/IEC 14443 Type B, Layer 3
		0x01	0x04	ISO/IEC 14443 Type B, Layer 4
		0x02	0x02	ISO/IEC 15693, Layer 2
		0x02	0x03	ISO/IEC 15693, Layer 3
		0x03	0x00	FeliCa, no layer separation
		RFU	RFU	RFU

Table 13 - Switch Protocol Data Object

Protocol	Tag	Length	Value
ISO/IEC 14443 Type A, Layer 2	-	-	-
ISO/IEC 14443 Type A, Layer 3	0x8F	0x01	Final SAK (1 byte)
ISO/IEC 14443 Type A, Layer 4	0x5F51	Variable	ATR
ISO/IEC 14443 Type B, Layer 2	-	-	-
ISO/IEC 14443 Type B, Layer 3	0x8F	0x03	Protocol info (3 bytes)
ISO/IEC 14443 Type B, Layer 4	0x5F51	Variable	ATR
ISO/IEC 15693, Layer 2	-	-	-
ISO/IEC 15693, Layer 3	-	-	-
FeliCa, no layer separation	-	-	-

Table 14 - Response for Switch Protocol Data Object

5.3.8. Operations Examples

5.3.8.1. MIFARE Classic® 1K/4K example

To read and write a MIFARE® card, first authenticate the card with the correct key.

An example APDU sequence is as follows:

1. Load a 6-byte MIFARE® key of all FFs to Key Slot 0

```
Command: FF 82 00 01 06 FF FF FF FF FF FF
Response: 90 00
```

2. Authenticate Block 4 with Key Slot 0 of Key A

```
Command: FF 86 00 00 05 01 00 04 60 00
Response: 90 00
```

3. Read Block 4 data

```
Command: FF B0 00 04 10
Response: XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX 90 00
```

4. Write 16 bytes of data to Block 5

```
Command: FF D6 00 05 10 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF 00
Response: 90 00
```

5.3.8.2. MIFARE® DESFire® example

MIFARE® DESFire® supports two kinds of APDU command. One is native MIFARE® DESFire® APDU command, another is ISO/IEC 7816-4 Wrapping APDU command. Once the first APDU command is sent to MIFARE® DESFire® card, the corresponding command format cannot be changed until the card is power off.

An example APDU is as follows: (ISO/IEC 7816-4 Wrapping APDU Format)

1. Get Version (Part 1)

```
Command: 90 60 00 00 00
Response: XX XX XX XX XX XX XX XX AF
```

2. Get Version (Part 2)

```
Command: 90 AF 00 00 00
Response: XX XX XX XX XX XX XX XX AF
```

3. Get Version (Part 3)

```
Command: 90 AF 00 00 00
Response: XX XX XX XX XX XX XX XX XX XX XX XX XX XX 91 00
```

4. Select application with AID = yy yy yy

```
Command: 90 5A 00 00 03 yy yy yy 00
Response: 91 00
```

5. Get File ID

```
Command: 90 6F 00 00 00
Response: zz 91 00
```

6. Read 10 bytes data of file zz, starting at byte 0

```
Command: 90 BD 00 00 07 zz 00 00 00 0A 00 00 00
Response: XX XX XX XX XX XX XX XX XX XX 91 00
```

For full details of the MIFARE® DESFire® commands, refer to NXP document.

5.3.8.3. ISO/IEC 14443-4 Type A or ISO/IEC 14443-4 Type B example

An example APDU is as follows:

1. Get Challenge

```
Command: 00 84 00 00 08
Response: XX XX XX XX XX XX XX XX 90 00
```

5.3.8.4. FeliCa example

An example APDU is as follows:

1. Polling Command

```
Command: FF AB 00 00 05 00 FF FF 01 00
Response: 14 XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX 90
00
```

2. Read Block 0 data through Read Without Encryption command

```
(IDm = zz zz zz zz zz zz zz zz, Service Code = yy yy, Block number = 1,
Block List = 00 00 00)

Command: FF AB 00 00 10 06 zz zz zz zz zz zz zz zz 01 yy yy 01 00 00 00
Response: 1D 07 zz zz zz zz zz zz zz zz 00 00 01 XX XX XX XX XX XX XX XX
XX XX XX XX XX XX XX XX 90 00
```

3. Write Block 0 data through Write Without Encryption command

```
(IDm = zz zz zz zz zz zz zz zz, Service Code = yy yy, Block number = 1,
Block List = 00 00 00)

Command: FF AB 00 00 20 08 zz zz zz zz zz zz zz zz 01 yy yy 01 00 00 00 11
22 33 44 55 66 77 88 99 AA BB CC DD EE FF 00
Response: 0C 09 zz zz zz zz zz zz zz zz 00 00 90 00
```

5.3.8.5. MIFARE Plus® through PC/SC Transparent command example

An example APDU is as follows:

1. Start transparent session

```
Command: FF C2 00 00 02 81 00
Response: C0 03 00 90 00 90 00
```

2. Turn on antenna

```
Command: FF C2 00 00 02 84 00
Response: C0 03 00 90 00 90 00
```

3. Switch to ISO/IEC 14443 Type A layer 4

```
Command: FF C2 00 02 04 8F 02 00 04
Response: C0 03 00 90 00 5F 51 0C 3B 87 80 01 C1 05 2F 2F 01 BC D6 A9 90
00
```

4. AES Authentication with Section key Part 1

```
Command: FF C2 00 01 13 5F 46 04 00 80 00 00 90 02 10 00 95 06 0A 00 70 00
40 00
Response: C0 03 00 90 00 92 01 00 96 02 00 00 97 13 0A 00 90 XX XX XX XX
XX XX XX XX XX XX XX XX XX XX XX XX 90 00
```

5. AES Authentication Part 2

```
Command: FF C2 00 01 30 5F 46 04 00 80 00 00 90 02 10 00 95 23 0B 00 72 XX
XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
XX XX XX XX XX XX
Response: C0 03 00 90 00 92 01 00 96 02 00 00 97 23 0B 00 90 XX XX XX XX
XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX XX
XX XX XX 90 00
```

6. End transparent session

```
Command: FF C2 00 00 02 82 00
Response: C0 03 00 90 00 90 00
```

5.3.8.6. ISO/IEC 15693 ICODE SLIX 2

An example APDU is as follows:

1. Write Binary

```
Command: FF D6 00 00 04 00 01 02 03
Response: 90 00
```

2. Write Binary

```
Command: FF D6 00 01 04 04 05 06 07
Response: 90 00
```

3. Read Binary

```
Command: FF B0 00 00 08
Response 00 01 02 03 04 05 06 07 90 00
```

5.3.8.7. ISO/IEC 15693 through PC/SC Transparent command example

An example APDU is as follows:

1. Start transparent session

```
Command: FF C2 00 00 02 81 00
Response: C0 03 00 90 00 90 00
```

2. Turn on antenna

```
Command: FF C2 00 00 02 84 00
Response: C0 03 00 90 00 90 00
```

3. Switch to ISO/IEC 15693 layer 3

```
Command: FF C2 00 02 04 8F 02 02 03
Response: C0 03 00 90 00 90 00
```

4. Send Reset-to-ready command

```
Command: FF C2 00 01 0C 5F 46 04 40 42 0F 00 95 03 26 01 00
Response: C0 03 00 90 00 92 01 00 96 02 00 00 97 0A 00 00 2A A5 1E 16 00
00 07 E0 90 00
```

5. End transparent session

```
Command: FF C2 00 00 02 82 00
Response: C0 03 00 90 00 90 00
```

6.0.Contact Card Communications

6.1. Contact Card Interface

The smart card reader interface is compliant with the ISO/IEC 7816 standard and CCID specification. In general terms, ISO/IEC 7816 governs the communication protocol between the reader(s) and the card and CCID specifies the communication protocol between the reader(s) and the host (usually PC). The reader(s) is generally a passive device and has to wait for CCID commands from the host before it performs any actions. Notification of card state change is an exception as reader(s) notifies the host of any changes to card state.

The reader supports the following features:

- Multiple voltage support for card activation
- PPS exchange for faster communication speed
- APDU exchange level
- Extended APDU exchange for T=1 protocol

Multiple Voltage Support for Card Activation

Smart cards can operate at one of the following voltages, 1.8V, 3V and 5V. For Smart card reader(s) that support multiple card voltages, the following card activation sequence applies:

1. Upon detection of a smart card, the host instructs the reader to activate the card.
2. Then the reader(s) will activate the card in the following order 5V, 3V, 1.8V.
3. If the card responds at a specific voltage, the activation sequence is completed and an ATR is returned.

With escape command, the order of card activation voltages can be configured to work in reverse order, i.e. from 1.8V to 5V instead of from 5V to 1.8V, or to use only specific voltage level, e.g. only use 5V, to activate the card.

PPS Exchange

PPS is the mechanism defined in ISO/IEC 7816 standard to switch the default communication speed and protocol of the smart card to a higher one. Depending on the returned ATR, the host determines if the card can run at a higher speed. If the card supports higher speed, then the host sends a CCID command to the reader to handle PPS exchange accordingly.

APDU Exchange Level

In ISO/IEC 7816, two communication units are defined, APDU and TPDU. APDU is the communication unit filled out by smart card application while TPDU is the unit accepted by the smart card. A conversion, though rather straightforward, from APDU to TPDU is needed before sending the data to the card. The reader supports APDU exchange level. With APDU exchange level support, the reader handles both the APDU into TPDU conversion as well as the communication timing with the card.

6.2. Contact Card Communications

6.2.1. ISO/IEC 7816 Card

ISO/IEC 7816 compliant cards are microprocessor cards with enhanced security features. This type of cards can be used in various applications, like personal identification, authentication, financial application and healthcare, etc. Due to the versatility of microprocessor cards, they represent a major part of the smart card market.

APDU structure	Command header	Command body		
Case 1	Command header	-		
Case 2	Command header	-		Le
Case 3	Command header	Lc	Data	-
Case 4	Command header	Lc	Data	Le

6.2.2. Operations Example

6.2.2.1. Master/Visa Card (T=0)

An example APDU sequence is as follows:

1. Select PSE payment file

```
Command: 00 A4 04 00 0E 31 50 41 59 2E 53 59 53 2E 44 44 46 30 31 00
Response: 61 XX
```

2. Get Response

```
Command: 00 C0 00 00 XX
Response: 6F XX XX XX XX XX XX XX XX XX XX XX XX XX XX 90 00
```

3. Select associated application file

```
Command: 00 B2 01 0C 00
Response: 6C yy
```

4. Re-send command by setting Le with SW2 (yy) by previous reponse

```
Command: 00 B2 01 0C yy
Response: 70 XX XX XX XX XX XX XX XX XX XX XX XX XX XX 90 00
```

7.0. Escape Commands

Escape commands (a.k.a. Vendor-specific commands) give applications access to additional functions provided by the device. These commands are identified by a command code followed by a parameter byte and multiple data bytes, if any.

Escape command is encoded in APDU format with CLA = 0xFF and INS = 0x00. The command code is encoded in P1 and parameter is encoded in P2. If the command writes data to device, the length of data is encoded in Lc followed by data bytes. Le is always 0.

7.1. Get Firmware Version

This command returns the firmware version in string format to the application.

Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x01	0x00	-	-	0x00

Response:

Data field	SW1 SW2
Firmware version	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully. Firmware version is returned in Data field.
0x63	0x00	Operation fails. No firmware version is returned.

7.2. Get Reader Serial Number

This command returns the unique serial number of the device.

Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x02	0x00	-	-	0x00

Response:

Data field	SW1 SW2
SN BYTE 1 … N	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully. Unique serial number is returned in Data field.
0x63	0x00	Operation fails.

7.3. LED Control

This command sets the behavior of LED2 and LED3.

Command:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0x00	0x03	0x00	0x02	**Index**	**Enable**	-

Index								Description
b7	b6	b5	b4	b3	b2	b1	b0	
0	0	0	0	0	0	0	1	LED 3
0	0	0	0	0	0	1	0	LED 2

Enable	Description
0x00	LED off
0x01	LED on

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

7.4. Buzzer Control

This command sets the buzzer behavior, if any.

Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x04	**Unit**	0x03	**Information**	-

Unit	Description
0x00	One duration unit = 1 second
0x01	One duration unit = 10 milliseconds

Information		
Byte 1	Byte 2	Byte 3
On duration	Off duration	Repeat count

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

7.5. LED and Buzzer Feedback Mode for Contact Interface

This command configures the ways LED2, LED3 and buzzer of the device behave when handling a contact card.

To set the feedback mode,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x12	**Memory**	0x01	**Mode**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the feedback mode,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x12	**Memory**	-	-	0x00

Response:

Data field	SW1 SW2
Mode	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Mode								Description	Value of X
b7	b6	b5	b4	b3	b2	b1	b0		
0	0	0	0	0	0	0	X	Change LED state when card is detected	0 = Disable 1 = Enable
0	0	0	0	0	0	X	0	Flash LED during data exchange	0 = Disable 1 = Enable
0	0	0	0	0	X	0	0	Beep when card is detected	0 = Disable 1 = Enable
0	0	0	0	X	0	0	0	Beep when card is removed	0 = Disable 1 = Enable
0	0	0	X	0	0	0	0	Beep when contact function in firmware has been initialized	0 = Disable 1 = Enable
0	0	X	0	0	0	0	0	RFU	0
0	X	0	0	0	0	0	0	RFU	0
X	0	0	0	0	0	0	0	Specify LED to apply the above configuration	0 = LED 3 1 = LED 2

7.6. LED and Buzzer Feedback Mode for Contactless Interface

This command configures the ways the card operation LED2, LED3 and buzzer, if any, of the device behave when handling a contactless card.

To set the feedback mode,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x13	**Memory**	0x01	**Mode**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the feedback mode,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x13	Memory	-	-	0x00

Response:

Data field	SW1 SW2
Mode	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Mode								Description	Value of X
b7	b6	b5	b4	b3	b2	b1	b0		
0	0	0	0	0	0	0	X	Change LED state when card is detected	0 = Disable 1 = Enable
0	0	0	0	0	0	X	0	Flash LED during data exchange	0 = Disable 1 = Enable
0	0	0	0	0	X	0	0	Beep when card is detected	0 = Disable 1 = Enable
0	0	0	0	X	0	0	0	Beep when card is removed	0 = Disable 1 = Enable
0	0	0	X	0	0	0	0	Beep when contactless function in firmware has been initialized	0 = Disable 1 = Enable
0	0	X	0	0	0	0	0	RFU	0
0	X	0	0	0	0	0	0	RFU	0
X	0	0	0	0	0	0	0	Specify LED to apply the above configuration	0 = LED 3 1 = LED 2

7.7. Shared Mode

This command controls whether the contact and contactless interfaces can operate concurrently. In shared mode, contact and contactless cards can work at the same time. While in exclusive mode, contactless card will be cut off once the device detects the presence of a contact card.

To set the shared mode,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x14	**Memory**	0x01	**Mode**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the shared mode,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x14	**Memory**	-	-	0x00

Response:

Data field	SW1 SW2
Mode	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Mode	Description
0x00	Shared mode
0x01	Exclusive mode (default)

7.8. User Guard Time

This command sets the user-defined extra guard time, which is added to the default guard time (12 ETUs), for contact and SAM interface. The unit is guard time is ETU.

To set the guard time,
Command:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0x00	0x20	**Memory**	0x02	**Contact Guard Time**	**SAM Guard Time**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the guard time,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x20	**Memory**	-	-	0x00

Response:

Data field		SW1 SW2
Contact Guard Time	**SAM Guard Time**	See table below

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Contact Guard Time	Description
0x00 – 0xFF	Number of extra ETUs on top of default guard time for contact slot

SAM Guard Time	Description
0x00 – 0xFF	Number of extra ETUs on top of default guard time for SAM slot

7.9. Automatic Response for T=0

This command configures the handling of 61XX (GET RESPONSE) and 6CXX (WRONG LENGTH LE) response from T=0 card. In both cases, XX indicates the correct value of the Le parameter for retrieving response data. With this feature enabled, the reader can issue the correct command to the card to get back the response data, saving the application from sending a separate command to get the data. In case of 61XX, the reader will send the GET_RESPONSE command. While with 6CXX, the reader will re-send the last command with Le value adjusted to XX to get the response data.

To set the automatic response,
Command:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0x00	0x21	**Memory**	0x02	**Contact Automatic Response**	**SAM Automatic Response**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the automatic response configuration,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x21	**Memory**	-	-	0x00

Response:

Data field		SW1 SW2
Contact Automatic Response	**SAM Automatic Response**	See table below

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Contact Automatic Response	Description
0x00	Disable automatic response feature for contact slot
0x01	Enable automatic response feature for contact slot

SAM Automatic Response	Description
0x00	Disable automatic response feature for SAM slot
0x01	Enable automatic response feature for SAM slot

7.10. Custom Default Speed

This command sets the default communication speed parameter, *Fd* and *Dd*, to a value other than that defined by ISO/IEC 7816 standard. Per ISO/IEC 7816 standard, default communication speed after activation shall be 0x11. However, there are cards that run at a different default speed. In order to communicate with these cards, the escape command can be used to change the default.

To set the custom default speed parameter,
Command:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0x00	0x22	**Memory**	0x02	**Contact Default Speed**	**SAM Default Speed**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the custom default speed parameter,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x22	**Memory**	-	-	0x00

Response:

Data field		SW1 SW2
Contact Default Speed Parameter	**SAM Default Speed Parameter**	See table below

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Contact Default Speed Parameter	Description
See Table 7 of ISO/IEC 7816-3 for Fi and Di values	This shall be set to the specific FiDi value used by the card in contact slot

SAM Default Speed Parameter	Description
See Table 7 of ISO/IEC 7816-3 for Fi and Di values	This shall be set to the specific FiDi value used by the card in SAM slot

7.11. Alternative Speed Parameter

The reader has its limit on the maximum communication speed that it can handle. This command sets the next highest communication speed that the reader shall use when the card indicates a speed above the reader's maximum, which are 848 kbps for contact slot at 4.8 MHz and 125 kbps for SAM slot at 4 MHz.

To set the alternative speed,

Command:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0x00	0x23	**Memory**	0x02	**Contact Alternative Speed**	**SAM Alternative Speed**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the alternative speed,

Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x23	**Memory**	-	-	0x00

Response:

Data field		SW1 SW2
Contact alternative speed	**SAM alternative speed**	See table below

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Contact Alternative Speed Parameter	Description
See Table 7 of ISO/IEC 7816-3 for Fi and Di values	The calculated speed using the Fi and Di value must not exceed 848 kbps

SAM Alternative Speed Parameter	Description
See Table 7 of ISO/IEC 7816-3 for Fi and Di values	The calculated speed using the Fi and Di value must not exceed 125 kbps

Example of calculating the communication speed from Fi and Di values:
Given a TA1 = 0x94, the corresponding Fi and Di values are 512 and 8 (from Table 7 of ISO/IEC 7816-3)
Contact slot clock frequency = 4.8 MHz = 4800000 Hz

$$Speed = \frac{D \times f}{F} bps = \frac{8 \times 4800000}{512} bps = 75\ kbps$$

7.12. Power On Sequence

This command changes card activation sequence for the contact slot. The reader can either try powering on the card with a range of voltages or only at specific voltage.

To set the power on sequence,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x24	**Memory**	0x01	**Power on sequence**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the power on sequence,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x24	**Memory**	-	-	0x00

Response:

Data field	SW1 SW2
Power on sequence	See table below

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Power On Sequence	Description
0x00	Power on from: 1.8V → 3V → 5V
0x01	Power on at 5V
0x02	Power on at 3V
0x03	Power on at 1.8V
0x04	Power on from: 5V → 3V → 1.8V

7.13. Contactless Polling Type

This command sets types of contactless cards to be polled by the reader. By limiting the number of polling card types, the turnaround time of contactless card detection can be improved.

To set the contactless polling type,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x41	**Memory**	0x01	**Card Type**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the contactless polling type,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x41	**Memory**	-	-	0x00

Response:

Data field	SW1 SW2
Card Type	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Card Type								Description	Value of X
b7	b6	b5	b4	b3	b2	b1	b0		
0	0	0	0	0	0	0	X	ISO/IEC 14443 Type A	0 = Disable 1 = Enable
0	0	0	0	0	0	X	0	ISO/IEC 14443 Type B	0 = Disable 1 = Enable
0	0	0	0	0	X	0	0	FeliCa 212 kbps	0 = Disable 1 = Enable
0	0	0	0	X	0	0	0	FeliCa 424 kbps	0 = Disable 1 = Enable
0	0	0	X	0	0	0	0	Topaz	0 = Disable 1 = Enable
0	0	X	0	0	0	0	0	RFU	0
0	X	0	0	0	0	0	0	Picopass	0 = Disable 1 = Enable
X	0	0	0	0	0	0	0	Automatic generation of RATS in case of ISO/IEC 14443-4 compliant	0 = Disable 1 = Enable

7.14. Contactless Polling Type Extended

This command is an extension to Contactless Polling Type escape command. Use this command to set the polling option for additional card types.

To set the contactless polling type extended,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x41	**Memory**	0x01	**Card Type**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the contactless polling type extended,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x41	**Memory**	-	-	0x00

Response:

Data field	SW1 SW2
Card Type	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x80	Get/Set the value in volatile memory
0x81	Get/Set the value in EEPROM

Card Type								Description	Value
b7	b6	b5	b4	b3	b2	b1	b0		
0	0	0	0	0	0	0	X	ISO/IEC 15693	0 = Disable 1 = Enable
0	0	0	0	0	0	X	0	RFU	0
0	0	0	0	0	X	0	0	RFU	0
0	0	0	0	X	0	0	0	RFU	0
0	0	0	X	0	0	0	0	RFU	0
0	0	X	0	0	0	0	0	RFU	0
0	X	0	0	0	0	0	0	RFU	0
X	0	0	0	0	0	0	0	RFU	0

7.15. Polling Parameter

This command sets the contactless card polling parameters.

To set the contactless polling parameters,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x40	**Memory**	0x01	**Parameter**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the contactless polling parameters,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x40	**Memory**	-	-	0x00

Response:

Data field	SW1 SW2
Parameter	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x80	Get/Set the value in volatile memory
0x81	Get/Set the value in EEPROM

Parameter								Description	Value of X
b7	b6	b5	b4	b3	b2	b1	b0		
0	0	0	0	0	0	0	X	Auto polling	0 = Disable 1 = Enable
0	0	0	0	0	0	X	0	Antenna off when no tag is detected	0 = Disable 1 = Enable
0	0	0	0	0	X	0	0	Antenna off when CCID powers off card	0 = Disable 1 = Enable
0	0	0	0	X	0	0	0	RFU	0
0	0	X	X	0	0	0	0	Polling period	00b = 250 ms 01b = 500 ms 02b = 1000 ms 03b = 100 ms
0	0	X	X	0	0	0	0		
0	X	0	0	0	0	0	0	RFU	0
X	0	0	0	0	0	0	0	RFU	0

7.16. Contactless Speed

This command sets the PPS setting for contactless interface (ONLY applicable to ISO/IEC 14443-4 Type A and ISO/IEC 14443-4 Type B).

To set the contactless speed,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x42	**Memory**	0x01	**Maximum Speed**	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the contactless speed,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x42	**Memory**	-	-	0x00

Response:

Data field		SW1 SW2
Maximum Speed	**Current Speed**	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

Memory	Description
0x00	Get/Set the value in volatile memory
0x01	Get/Set the value in EEPROM

Maximum Speed / Current Speed	Description
0x00	106 kbps
0x01	212 kbps
0x02	424 kbps (default)
0x03	848 kbps

7.17. FeliCa System Code

This command sets the FeliCa system code to use when polling for FeliCa.

To set the FeliCa system code,
Command:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0x00	0x50	0x00	0x02	System Code (MSB)	System Code (LSB)	-

Response:

Data field	SW1 SW2
-	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

To get the FeliCa system code,
Command:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0x00	0x50	0x00	-	-	0x00

Response:

Data field		SW1 SW2
System code (MSB)	System code (LSB)	See the following table

SW1	SW2	Description
0x90	0x00	Operation completes successfully.
0x63	0x00	Operation fails.

7.18. Operations Example

7.18.1. Get Firmware Version

To get the firmware version of the reader. An example escape command is as follows:

1. Get current firmware version of the reader

```
Command: FF 00 01 00 00
Response: XX XX XX XX XX XX XX XX XX XX XX XX XX 90 00
```

7.18.2. Play a beep sound

To play a beep sound. An example escape command is as follows:

1. Play a beep sound with 50 ms duration once

```
Command: FF 00 04 01 03 05 05 01
Response: 90 00
```

2. Play a beep sound with 250 ms duration twice

```
Command: FF 00 04 01 03 19 19 02
Response: 90 00
```

7.18.3. Configure the buzzer behavior

To configure the buzzer behavior. An example escape command is as follows:

1. Set the reader with Beep sound when card is detected and without Beep sound when card is removed

```
Command: FF 00 13 01 01 17
Response: 90 00
```

7.18.4. Configure the contactless card polling type

To configure the contactless card polling type. An example escape command is as follows:

1. Set to poll all supported contactless cards

   a. Enable ISO/IEC 14443 Type A, ISO/IEC 14443 Type B, FeliCa, Topaz, Picopass

   ```
   Command: FF 00 41 01 01 DF
   Response: 90 00
   ```

   b. Enable ISO/IEC 15693

   ```
   Command: FF 00 41 81 01 01
   Response: 90 00
   ```

2. Set to poll FeliCa only

   a. Enable FeliCa only

   ```
   Command: FF 00 41 01 01 0C
   Response: 90 00
   ```

   b. Disable ISO/IEC 15693

   ```
   Command: FF 00 41 81 01 00
   Response: 90 00
   ```

3. Set to poll ISO/IEC 14443 Type A and ISO/IEC 14443 Type B only

   a. Enable ISO/IEC 14443 Type A and ISO/IEC 14443 Type B

   ```
   Command: FF 00 41 01 01 03
   Response: 90 00
   ```

   b. Disable ISO/IEC 15693

   ```
   Command: FF 00 41 81 01 00
   Response: 90 00
   ```

Appendix A. Sample code

This sample shows how to connect to the reader and send command.

```
#include <stdio.h>
#include <windows.h>
#include <winscard.h>

#define IOCTL_CCID_ESCAPE SCARD_CTL_CODE(3500)

int main(int argc, char *argv[])
{
    LONG lResult;

    SCARDCONTEXT hContext;
    LPTSTR pmszReaders = NULL;
    LPTSTR pReader;
    DWORD cchReaders = SCARD_AUTOALLOCATE;

    SCARDHANDLE hCardHandle;
    DWORD dwActiveProtocol;
    SCARD_IO_REQUEST pioSendPci;

    DWORD dwReaderLen = 256;
    DWORD dwState, dwProtocol;
    DWORD dwAtrLen = 32;
    DWORD dwSend = 0, dwRecv = 0;

    BYTE pbAtr[32];
    BYTE pbSendAPDU[] = {0xFF, 0xCA, 0x00, 0x00, 0x00};
    BYTE pbSendEscape[] = {0xFF, 0x00, 0x01, 0x00, 0x00};
    BYTE pbRecv[258];
    DWORD i;

    // Establish context
    lResult = SCardEstablishContext(SCARD_SCOPE_USER, NULL, NULL, &hContext);
    if (SCARD_S_SUCCESS != lResult)
    {
        printf("Failed SCardEstablishContext. Error code: 0x%08X\n", lResult);
        return 1;
    }

    // Retrieve the list the readers.
    // hContext was set by a previous call to SCardEstablishContext.
```

```
lResult = SCardListReaders(hContext,
                           NULL,
                           (LPTSTR)&pmszReaders,
                           &cchReaders);
if (SCARD_S_SUCCESS != lResult)
{
    // Release context
    SCardReleaseContext(hContext);
    printf("Failed SCardListReaders. Error code: 0x%08X\n", lResult);
    return 1;
}
pReader = pmszReaders;
while ('\0' != *pReader)
{
    printf("Reader: %S\n", pReader);
    pReader = pReader + wcslen((wchar_t *)pReader) + 1;
}
pReader = pmszReaders;

// Send escape command to reader
dwSend = sizeof(pbSendEscape);
dwRecv = sizeof(pbRecv);
lResult = SCardConnect(hContext,
                       pReader,
                       SCARD_SHARE_DIRECT,
                       0,
                       &hCardHandle,
                       &dwActiveProtocol);
if (SCARD_S_SUCCESS != lResult)
{
    // Release context
    SCardReleaseContext(hContext);
    printf("Failed SCardConnect. Error code: 0x%08X\n", lResult);
    return 1;
}
lResult = SCardControl(hCardHandle,
                       IOCTL_CCID_ESCAPE,
                       pbSendEscape,
                       dwSend,
                       pbRecv,
                       dwRecv,
                       &dwRecv);
if (SCARD_S_SUCCESS != lResult)
{
    // Disconnect card
```

```
        SCardDisconnect(hCardHandle, SCARD_LEAVE_CARD);
        // Release context
        SCardReleaseContext(hContext);
        printf("Failed SCardControl. Error code: 0x%08X\n", lResult);
        return 1;
    }
    printf("\nEscape command:\n");
    printf(">");
    for (i = 0; i < dwSend; i++)
    {
        printf(" %02X", pbSendEscape[i]);
    }
    printf("\n");
    printf("<");
    for (i = 0; i < dwRecv; i++)
    {
        printf(" %02X", pbRecv[i]);
    }
    printf("\n");
    // Disconnect card
    SCardDisconnect(hCardHandle, SCARD_LEAVE_CARD);

    // Connect to card
    lResult = SCardConnect(hContext,
                           (LPCTSTR)pReader,
                           SCARD_SHARE_SHARED,
                           SCARD_PROTOCOL_T0 | SCARD_PROTOCOL_T1,
                           &hCardHandle,
                           &dwActiveProtocol);
    if (SCARD_S_SUCCESS != lResult)
    {
        // Release context
        SCardReleaseContext(hContext);
        printf("Failed SCardConnect. Error code: 0x%08X\n", lResult);
        return 1;
    }
    pioSendPci = *SCARD_PCI_T0;
    switch (dwActiveProtocol)
    {
    case SCARD_PROTOCOL_T0:
        pioSendPci = *SCARD_PCI_T0;
        break;
    case SCARD_PROTOCOL_T1:
        pioSendPci = *SCARD_PCI_T1;
        break;
```

```
}

// Get ATR
lResult = SCardStatus(hCardHandle,
                      pReader,
                      &dwReaderLen,
                      &dwState,
                      &dwProtocol,
                      pbAtr,
                      &dwAtrLen);
if (SCARD_S_SUCCESS != lResult)
{
    // Disconnect card
    SCardDisconnect(hCardHandle, SCARD_LEAVE_CARD);
    // Release context
    SCardReleaseContext(hContext);
    printf("Failed SCardStatus. Error code: 0x%08X\n", lResult);
    return 1;
}
printf("\nATR :");
for (i = 0; i < dwAtrLen; i++)
{
    printf(" %02X", pbAtr[i]);
}
printf("\n");

// Send command to card
dwSend = sizeof(pbSendAPDU);
dwRecv = sizeof(pbRecv);
lResult = SCardTransmit(hCardHandle,
                        &pioSendPci,
                        pbSendAPDU,
                        dwSend,
                        NULL,
                        pbRecv,
                        &dwRecv);
if (SCARD_S_SUCCESS != lResult)
{
    // Disconnect card
    SCardDisconnect(hCardHandle, SCARD_LEAVE_CARD);
    // Release context
    SCardReleaseContext(hContext);
    printf("Failed SCardTransmit. Error code: 0x%08X\n", lResult);
    return 1;
}
```

```
    printf("APDU Command:\n");
    printf(">");
    for (i = 0; i < dwSend; i++)
    {
        printf(" %02X", pbSendAPDU[i]);
    }
    printf("\n");
    printf("<");
    for (i = 0; i < dwRecv; i++)
    {
        printf(" %02X", pbRecv[i]);
    }
    printf("\n");

    // Disconnect card
    SCardDisconnect(hCardHandle, SCARD_LEAVE_CARD);

    // Release context
    SCardReleaseContext(hContext);

    return 0;
}
```

Appendix B. References

[1] [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winscard/nf-winscard-scardestablishcontext.

[2] [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winscard/nf-winscard-scardlistreadersa.

[3] [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winscard/nf-winscard-scardgetstatuschangea.

[4] [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winscard/nf-winscard-scardconnecta.

[5] [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winscard/nf-winscard-scardstatusa.

[6] [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winscard/nf-winscard-scardtransmit.

[7] [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winscard/nf-winscard-scardcontrol.

[8] [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winscard/nf-winscard-scarddisconnect.

[9] [Online]. Available: https://docs.microsoft.com/en-us/windows/win32/api/winscard/nf-winscard-scardreleasecontext.

[10] *[ISO/IEC 7816-1] ISO/IEC 7816-1:2011: Identification cards — Integrated circuit cards — Part 1: Cards with contacts — Physical characteristics.*

[11] *[ISO/IEC 7816-2] ISO/IEC 7816-2:2007: Identification cards — Integrated circuit cards — Part 2: Cards with contacts — Dimensions and location of the contacts.*

[12] *[ISO/IEC 7816-3] ISO/IEC 7816-3:2006: Identification cards — Integrated circuit cards — Part 3: Cards with contacts — Electrical interface and transmission protocols.*

[13] *[ISO/IEC 7816-4] ISO/IEC 7816-4:2020: Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange.*

[14] *[ISO/IEC 14443-2] ISO/IEC 14443-2:2010: Identification cards – Contactless integrated circuit cards – Proximity cards – Part 2: Radio frequency power and signal interface.*

[15] *[ISO/IEC 14443-3] ISO/IEC 14443-3:2011: Identification cards – Contactless integrated circuit cards – Proximity cards – Part 3: Initialization and anticollision.*

[16] *[ISO/IEC 14443-4] ISO/IEC 14443-4:2000: Identification cards – Contactless integrated circuit cards – Proximity cards – Part 4: Transmission protocol.*

[17] "Smart Card CCID version 1.1," [Online]. Available: https://www.usb.org/sites/default/files/DWG_Smart-Card_CCID_Rev110.pdf.

[18] "Interoperability Specification for ICCs and Personal Computer Systems - Part 3. Requirements for PC-Connected Interface Devices," [Online]. Available: http://pcscworkgroup.com/Download/Specifications/pcsc3_v2.01.09.pdf.

[19] "Interoperability Specification for ICCs and Personal Computer Systems - Part 3. Supplemental Document," [Online]. Available: http://pcscworkgroup.com/Download/Specifications/pcsc3_v2.01.09_sup.pdf.

[20] "Interoperability Specification for ICCs and Personal Computer Systems - Part 3. Supplemental Document for Contactless ICCs," [Online]. Available: http://pcscworkgroup.com/Download/Specifications/pcsc3_v2.02.00_sup2.pdf.

Appendix C. Trademarks

Android is a trademark of Google LLC.

Calypso is a registered trademark of Calypso Technology, Inc. in various country jurisdictions.

FeliCa is a registered trademark or a trademark of Sony Group Corporation or its affiliates.

Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.

macOS are trademarks of Apple Inc., registered in the U.S. and other countries and regions.

MIFARE, MIFARE Classic, MIFARE DESFire, MIFARE Plus and MIFARE Ultralight are registered trademarks of NXP B.V.

Windows is registered trademarks of Microsoft Corporation in the United States and other countries.