Reference Manual for
uTrust 3720F Contactless Reader
and
uTrust 3721F Contactless Reader w/Keyboard Interface

For Part #: **905592-* and 905593-***

Document Version: 1.00, Last Revised On: 2020-04-10

Abstract

This document contains in-depth information about the hardware and software features of the uTrust 3720 F Contactless Reader and uTrust 3721 F Contactless Reader with keyboard interface.

Audience

This document is intended for system integrators and software developers.

Revision History

Rev.	Date	Description
1.00	2020-04-10	Initial version

Contact Information

For additional information, please visit http://www.identiv.com/

Table of Contents

1. Legal information

1.1. Disclaimers

The content published in this document is believed to be accurate. However, Identiv does not provide any representation or warranty regarding the accuracy or completeness of its content, or regarding the consequences of your use of the information contained herein.

Identiv reserves the right to change the content of this document without prior notice. The content of this document supersedes the content of any previous versions of the same document. This document may contain application descriptions and/or source code examples, which are for illustrative purposes only. Identiv gives no representation or warranty that such descriptions or examples are suitable for the application that you may want to use them for.

Should you notice any problems with this document, please provide your feedback to support@identiv.com.

1.2. FCC, ISED Statement

The device contains license-exempt transmitter(s)/receiver(s) that comply with Innovation, Science and Economic Development Canada's license-exempt RSS(s). Operation is subject to the following two conditions:

(1) This device may not cause harmful interference, and
(2) This device must accept any interference received, including interference that may cause undesired operation

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications.

However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Information to user: Changes or modifications not expressly approved by Identiv could void the user's authority to operate the equipment.

1.3. Licenses

If the document contains source code examples, they are provided for illustrative purposes only and subject to the following restrictions:

- You MAY at your own risk use or modify the source code provided in the document in applications you may develop. You MAY distribute those applications ONLY in form of compiled applications.
- You MAY NOT copy or distribute parts of or the entire source code without prior written consent from Identiv Inc.
- You MAY NOT combine or distribute the source code provided with Open Source Software or with software developed using Open Source Software in a manner that subjects the source code or any portion thereof to any license obligations of such Open Source Software. If the document contains technical drawings related to Identiv Inc. products, they are provided for documentation purposes only. Identiv Inc. does not grant you any license to its designs.

1.4. Trademarks

MIFARE™, DESFire™ are registered trademarks of NXP Semiconductors B.V.

FeliCa™ is a registered trademark of Sony Corporation.

Windows™ is a trademark of Microsoft Corporation.

Chrome OS™ is a trademark of Google

my-d™ is a trademark of Infineon Technology

CryptoRF™ is a registered trademark of Microchip Technology

2. Introduction to the manual

2.1. Objective of the manual

This manual provides an overview of the hardware and software features of the uTrust 372x F contactless smart card readers (uTrust 3720 F and uTrust 3721 F).

This manual describes in detail interfaces and supported commands available for developers using uTrust 372x F in their applications.

2.2. Target audience

This document describes the technical implementation of uTrust 372x F.

The manual targets software developers. It assumes knowledge about ISO 7816, 13.56 MHz contactless technologies like ISO/IEC 14443, ISO/IEC 15693 and commonly used engineering terms.

Should you have questions, you may send them to support@identiv.com.

2.3. Product version corresponding to the manual

Product Component	Version
Hardware	0.1
Firmware	1.00 and above

2.4. Definition of various terms and acronyms

Term or Acronym	Expansion
APDU	Application Protocol Data Unit
ATR	Answer to Reset, defined in ISO7816
ATS	Answer to select, defined in ISO/IEC 14443
Byte	Group of 8 bits
CCID	Chip Card Interface Device
CID	Card Identifier
DFU	Device Firmware Upgrade
DR	Divider receive: used to determine the baud rate between the reader to the card
DS	Divider send: used to determine the baud rate between the card to the reader
LED	Light emitting diode
LF	Low frequency (125 KHz)
MIFARE	The ISO14443 Type A with extensions for security (NXP)
NA	Not applicable
NAD	Node Address
Nibble	Group of 4 bits. 1 digit of the hexadecimal representation of a byte. *Example:* 0xA3 is represented in binary as (10100011)b. The least significant nibble is 0x3 or (0011)b and the most significant nibble is 0xA or (1010)b
PCD	Proximity Coupling Device
PC/SC	Personal Computer/Smart Card: software interface to communicate between a PC and a smart card
PICC	Proximity Integrated Chip Card
PID	Product ID
Proximity	Distance coverage till ~10 cm.
PUPI	Pseudo unique PICC identifier
RF	Radio Frequency
RFU	Reserved for future use
USB	Universal Serial Bus
VCD	Vicinity Coupling Device
VICC	Vicinity Integrated Circuit Card
VID	Vendor ID
(xyz)b	Binary notation of a number x, y, z $\in$ {0,1}.
0xYY	The byte value YY is represented in hexadecimal

2.5. References

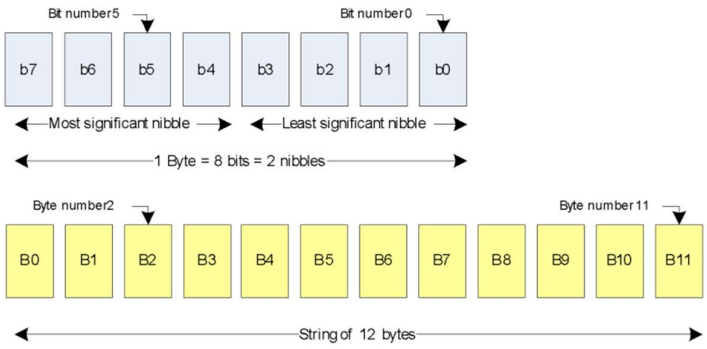
Document reference in the manual	Description of the referenced document	Document Issuer
ISO/IEC 7816-3	Identification cards — Integrated circuit cards —Part 3:Cards with contacts — Electrical interface and transmission protocols	ISO / IEC
ISO/IEC 7816-4	Identification cards - Integrated circuit(s) cards with contacts Part 4: Interindustry commands for interchange ISO/IEC 7816-4: 1995 (E)	ISO / IEC
ISO/IEC 14443-3	Identification cards — Contactless integrated circuit(s) cards — Proximity cards —Part 3:Initialization and anticollision	ISO / IEC
ISO/IEC 14443-4	Identification cards — Contactless integrated circuit(s) cards — Proximity cards Part 4: Transmission protocol ISO/IEC 14443-4:2001(E)	ISO / IEC
ISO/IEC 15693-3	Identification cards – Contactless integrated circuit(s) cards – Vicinity cards – Part 3: Anticollision and transmission protocol	ISO / IEC
ATMEL-5276	Atmel CrytoRF Specification (AT88SCxxxxCRF) Rev 5276G, 01/2014	Microchip Technology
NXP-163735	MF 1PLUSx0y1 Mainstream contactless smart card IC for fast and easy solution development Rev 3.5	NXP Semiconductors B.V.
PC/SC	Interoperability Specification for ICCs and Personal Computer Systems v2.01	PC/SC Workgroup
PCSC3	Interoperability Specification for ICCs and Personal Computer Systems Part 3. Requirements for PC-Connected Interface Devices	PC/SC Workgroup
PCSC3-AMD1	Interoperability Specification for ICCs and Personal Computer Systems Part 3. Requirements for PC-Connected Interface Devices– Amendment 1	PC/SC Workgroup
PCSC3-SUP	Interoperability Specification for ICCs and Personal Computer Systems Part 3. Supplemental Document	PC/SC Workgroup

Document reference in the manual	Description of the referenced document	Document Issuer
PCSC3-SUP2	Interoperability Specification for ICCs and Personal Computer Systems Part 3. Supplemental Document for Contactless ICCs	PC/SC Workgroup
CCID	Specification for Integrated Circuit(s) Cards Interface Devices 1.1	USB-IF
USB	Universal Serial Bus Specification 2.0	USB-IF

2.6. Conventions for Bits and Bytes

Bits are represented by lower case ‘b’ where followed by a numbering digit.

Bytes are represented by upper case ‘B’ where followed by a numbering digit.



Example:

163 decimal number is represented

- in hexadecimal as 0xA3
- in binary as (10100011)b

The least significant nibble of 0xA3 is

- 0x3 in hexadecimal
- (0011)b in binary

The most significant nibble of =xA3 is

- 0xA in hexadecimal
- (1010)b in binary

3. General information about uTrust 372x F

3.1. uTrust 372x F key benefits

With its combination of a modern slim design and its state of the art feature set, uTrust 3720 F is the perfect desktop reader choice for environments where HF/LF contactless card support is required while uTrust 3721 F perfectly fits environments where access to HF/LF contactless cards with data read as keyboard input is required.

As for all Identiv Inc. products, uTrust 372x F is designed to offer best in class interoperability.

3.2. uTrust 372x F key features

- 13.56MHz contactless reader:
  - ISO14443 Type A & B
  - ISO15693 (NFC Forum Tag Type 5)
  - MIFARE™
  - FELICA™
  - Topaz (NFC Forum Tag Type 1)
  - Type B Innovatron protocol support (Calypso)
  - my-d move
  - CryptoRF
- 125KHz contactless reader with support for popular formats
- PC/SC v2.0 compliant
- Secure in-field SmartOS™ firmware upgrade
- Unique reader serial number which enables that uTrust 372x F can be plugged into any USB slot on a PC without having to re-install the driver. Additionally, the application S/W running on the host can check for exact readers
- Communication speed up to 848 Kbps for ISO14443 (when PICCC also supports it)
- Wall mount option on reader
- Optional stand and card holder available as accessory

3.3. uTrust 372x F ordering information

Item	Part number	
uTrust 3720 F HF+LF	905592	PC/SC compatible reader that supports HF and LF
uTrust 3720 F HF	905592-1	PC/SC compatible reader that supports HF only
uTrust 3720 F LF	905592-2	PC/SC compatible reader that supports LF only
uTrust 3721 F HF+LF	905593	PC/SC compatible reader with keyboard interface that supports HF and LF
uTrust 3721 F HF	905593-1	PC/SC compatible reader with keyboard interface that supports HF only
uTrust 3721 F LF	905593-2	PC/SC compatible reader with keyboard interface that supports LF only

3.4. Available accessories

The reader has an optional stand accessory that allows the reader to be placed on level surface at 40° angle. A card holder clip accessory is available to hold the card in place. Either the stand or card holder clip accessory, or both accessories can be pre-assembled if needed.

3.5. uTrust 372x F customization options

Upon request, Identiv Inc. can consider customizing:

- The color of the casing
- The logo
- The product label
- The USB strings

Terms and conditions apply, please contact your local Identiv representative or send an email to sales@identiv.com.

3.6. Contactless communication principles and uTrust 372x F usage recommendations

uTrust 372x F is a dual interface reader capable of reading contactless user tokens. The following paragraph focuses on a few specifics of contactless communication to outline usage recommendations in order to ensure best user experience.

uTrust 372x F is a contactless reader[1] designed to communicate with user credentials.

User credentials[2] are made of a contactless integrated circuit chip connected to an antenna

User credentials can take several form factors:

- Credit card sized smart card
- Key fob
- USB token
- NFC mobile phone etc…

Communication between uTrust 372x F and user credentials uses magnetic field inductive coupling.

The magnetic field generated by uTrust 372x F has a carrier frequency of 13.56MHz for HF and 125KHz for LF.

3.6.1. Power supply

When the user credential is put in the magnetic field of the reader, its antenna couples with the reader and an induction current appears in the antenna thus providing power to the integrated circuit. The generated current is proportional to the magnetic flux going through the antenna of the user credential.

3.6.2. Data exchange

The carrier frequency of the magnetic field is used as a fundamental clock signal for the communication between the reader and the credential. It is also used as a fundamental clock input for the integrated circuit microprocessor to function.

To send data to the user credential the reader modulates the amplitude of the field. There are several amplitude modulation and data encoding rules defined in ISO/IEC 14443 and ISO/IEC 15693. The reader should refer to the standard for further details.

To answer the reader, the integrated circuit card of the user credential modulates its way of loading (impedance) the field generated by the reader. Here also further details can be found in ISO/IEC 14443 and ISO/IEC 15693.

[1] In the ISO/IEC 14443 standard, the reader is called the proximity coupling device (PCD); in the ISO/IEC 15693 standard, the reader is called the vicinity coupling device (VCD)

[2] In the ISO/IEC 14443 standard, the user credential is called proximity integrated chip card (PICC); in the ISO/IEC 15693 standard, the reader is called vicinity integrated chip card (VICC)

3.6.3. Recommendations

The communication between the reader and the user credential is sensitive to the presence of material or objects interfering with the magnetic field generated by the reader.

The presence of conductive materials like metal in the vicinity of the reader and the user credential can significantly degrade the communication and even make it impossible. The magnetic field of the reader generates Eddy or Foucault's currents in the conductive materials; the field is literally absorbed by that kind of material.

- It is recommended for proper communication to avoid placing uTrust 372x F in close proximity of conductive materials.

The presence of multiple user credentials in the field also interferes with the communication. When several user credentials are in the field of the reader, load of the field increases which implies that less energy is available for each of them and that the system is detuned. For this reason, Identiv has implemented in its driver only one slot.

- It is recommended to present only one user credential at a time in front of uTrust 372x F.

The communication between the reader and the credential is sensitive to the geometry of the system {reader, credential}. Parameters like the geometry and especially the relative size of the reader's and credential's antennas directly influence the inductive coupling and therefore the communication.

uTrust 372x F was designed and optimized to function with user credentials of various technologies and sizes.

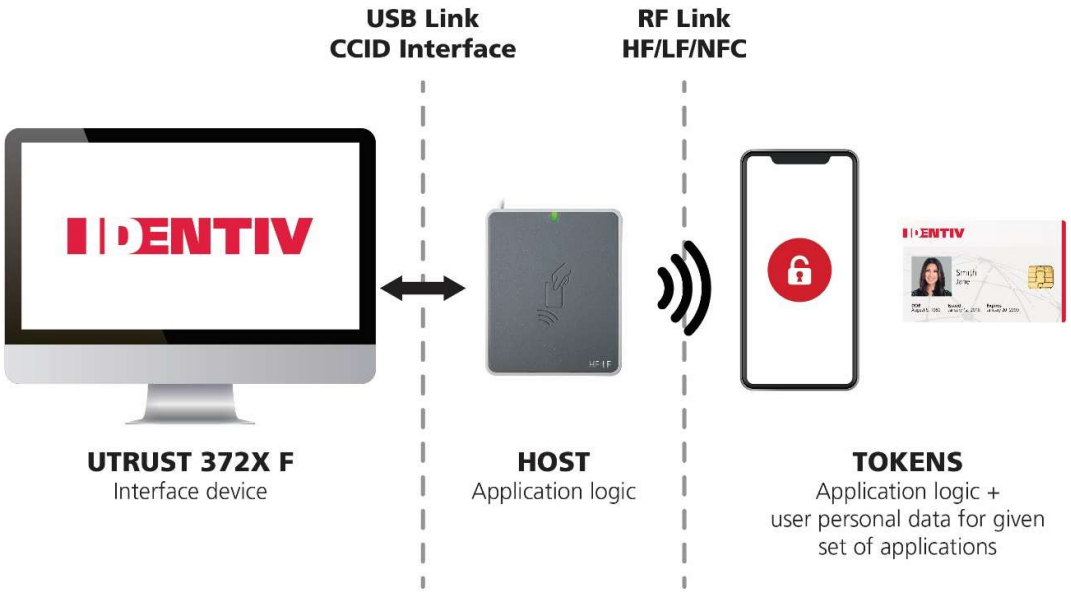
- It may happen, that uTrust 372x F is not capable of communicating with extremely large or extremely small credentials.
- In order to optimize the coupling between the reader and the credential, it is recommended to put both antennas as parallel as possible to each other
- In order to optimize transaction speed between the reader and the card it is recommended to place the credential as close as possible to the reader. This will increase the amount of energy supplied to the user credential which will then be able to use its microprocessor at higher speeds

3.7. Applications

3.7.1. General

uTrust 372x F is a transparent reader designed to interface a personal computer host supporting PC/SC interface with 13.56MHz user tokens like public transport cards, contactless banking cards, electronic identification documents – e.g. e-passports, e-ID cards, driving licenses etc. and smartcards like CAC and PKI cards and health insurance cards.

User credentials can have several form factors like credit cards, key fobs, NFC mobile phones or USB dongles like our uTrust Token products.



uTrust 3720 F itself handles the communication protocol but not the application related to the token or card. The application-specific logic has to be implemented by software developers on the host.

uTrust 3721 F handles the communication protocol related to the token or card and has some capabilities to handle some application logic like authentication, reading a specific block, data extraction, data formatting.

3.7.2. Applications provided by Identiv Inc.

Identiv Inc. does not provide payment or transport applications or PKI or CAC applications.

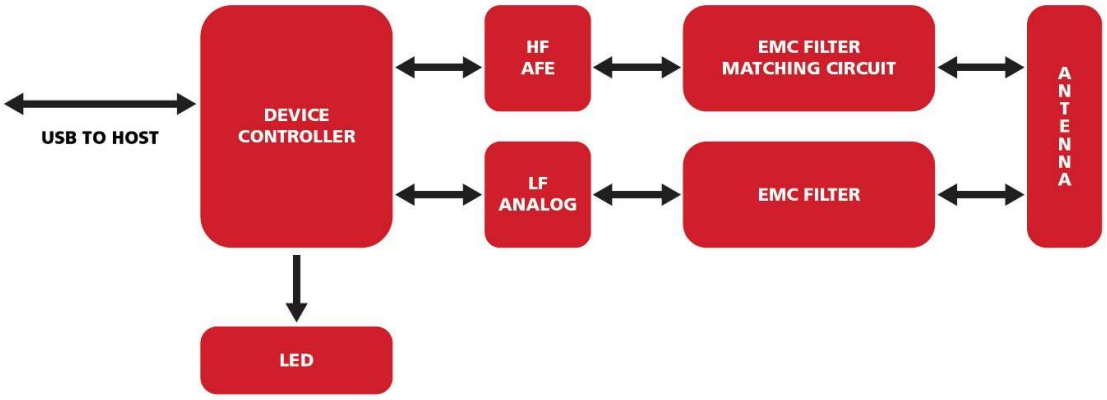
Identiv Inc. provides a few utilities for evaluation purposes that can function with uTrust 372x F. They are covered in Section 5.2.

4. uTrust 372x F characteristics

4.1. uTrust 372x F high level architecture

4.1.1. Block diagram

The link between uTrust 372x F and the host to which it is connected is the USB interface providing both the power and the communication channel.



The device controller has several interfaces available. In the uTrust 372x F implementation three peripherals are connected to the device controller:

- LED for reader status indication
- HF AFE to handle communication with HF cards; this peripheral is not present in LF only variants
- LF analog circuitry to handle communication with LF cards; this peripheral is not present in HF only variants

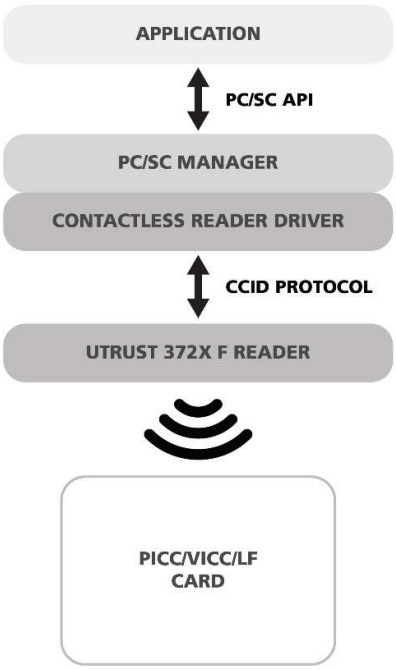
The controller embeds flash memory that contains the firmware developed by Identiv to handle all the RF communication protocols and the CCID communication protocol with the host. The flash can be upgraded once the device is deployed in the field, hence enabling firmware upgrades to add and potentially patch features.

The RF frontend or analog circuitry ensures the coding/decoding/framing modulation/demodulation required for the RF communication. It is controlled by the device controller through registers.

The matching circuitry provides the transmission and receiver paths adaptation for the antenna to function properly.

4.1.2. Software architecture

Applications can interface with the driver directly through the PC/SC interface.



The uTrust 372x F leverages a PC/SC CCID driver that is freely available for all supported operating systems (Windows, macOS X and Linux). With current Windows versions (starting with Windows Vista) and macOS, this driver is already included in the basic installation.

With the diverse Linux derivatives, there may be distribution specific drivers that should get installed using the install mechanism of the used distribution.

If there is none, the driver may always be downloaded from https://ccid.apdu.fr/.

Additionally, Identiv provides a proprietary driver for all the supported operating systems (except Chrome OS where inbox driver is sufficient).

4.2. Quick reference data

4.2.1. uTrust 372x F Physical Charteristics

Item	Characteristic	Value	
uTrust 3720 F uTrust 3721 F	Weight	905592 HF + LF	88 gms
		905592-1 HF	85 gms
		905592-2 LF	88 gms
		905593 HF + LF	88 gms
		905593-1 HF	85 gms
		905593-2 LF	88 gms
	External dimensions	91 x 75 x 12 mm (3.583 x 2.953 x 0.472 in)	
	Cable length	1.5 meter long with USB type A plug	
	Default color	White and grey	

Drawing with dimensions of the uTrust 372x F can be found in annex.

4.2.2. LED behavior

uTrust 372x F is equipped with one bicolor LED. Its default behavior is described in the table below. Note that when RED and GREEN are turned on in sync, the color would appear as AMBER.

Reader states	GREEN	RED
Just after plug-in (with drivers already installed)	ON	OFF
Just after DFU operation	ON	OFF
Suspend / standby	OFF	OFF
Reader powered, Contactless card IN field, but not powered	ON	ON
Contactless card powered / communication	500ms ON 500ms OFF	500ms ON 500ms OFF
Reader / card errors	OFF	100ms ON 100ms OFF

4.2.3. Other data

4.2.3.1. General

Parameter	Value/Description
Clock of the device controller	24MHz
API	PC/SC 2.0
Operating temperature range	-20º to 70ºC
Operating humidity range	Up to 95%RH non condensing
Certifications and compliances	USB, CE, FCC, WHQL, WEEE, RoHS3, REACH

4.2.3.2. USB

Parameter	Value/Description
DC characteristics	High bus powered (uTrust 372x F draws power from USB bus) Voltage: 5V Max Current : 200mA Suspend Current: 500µA
USB specification	USB 2.0 FS Device
USB Speed	Full Speed Device (12Mbit/s)
Device Class	CCID for uTrust 3720 F CCID & HID for uTrust 3721 F
PID	0x5612 for uTrust 3720 F 0x5613 for uTrust 3721 F
VID	0x04E6

4.2.3.3. Contactless HF interface

Parameter	Value/Description
RF carrier frequency	13.56MHz +/-50ppm
Maximum supported card baud-rate	ISO14443: 848Kbps ISO15693: 52Kbps
Cards supported	• MIFARE: Classic 1K and 4K, MIFARE mini, MIFARE Plus, DESFire, DESFire EV1, DESFire EV2, DESFire Light • Ultralight, Ultralight C, NTAG family • FeliCa™ 212 and 424 Kbps support: FeliCa Standard/Lite • NFC forum tag type 1, 2, 3, 4, 5 • iClass UID support • my-d move – SLE 66RxxP, my-d move NFC – SLE 66RxxPN, SLE 66RxxS, SLE 55RxxE • NFC enabled Smart Phones and Tablets[3]
ISO-14443A and B compliant	Yes
ISO-15693 compliant	Yes
Number of slots	1 (shared with LF)
Ejection mechanism	Manual

4.2.3.4. Contactless LF interface

Parameter	Value/Description
RF carrier frequency	125 KHz
Maximum supported card baud-rate	Up to 9600 bps
Cards supported	• ASK, FSK, PSK cards, HITAG • EM44xx • HID26, HID32, HID33, HID34 • HID35, HID36B, HID36C, HID36D • HID37, HID40 • INDALA26 • HONEYWELL • CASI (ProLite)
Number of slots	1 (shared with HF)
Ejection mechanism	Manual

[3] tested with available device during development & qualification phase

5. Software modules

5.1. Installation

On operating systems that include a generic CCID driver, no additional installation steps are necessary.

Where there's no CCID driver pre-installed – e.g. Linux systems or old Windows systems – the driver has to be installed using distribution-specific measures or using the available source packages.

Nevertheless, due to some limitations of the available CCID drivers under some circumstances, Identiv does provide a dedicated driver for this reader, as well, which is available through Windows Update or on the Identiv support pages.

5.2. Utilities

The following utilities are available:

- PCSC Diagnostics – This is a simple tool to check that the reader is installed properly and is able to read a card or tag
- TestResMan – This is a simple tool to quickly send an APDU to reader and get the response back

5.3. Driver

5.3.1. uTrust 372x F listing

uTrust 3720 F is listed by PC/SC applications as

- *Identiv uTrust 3720 Contactless Reader*

uTrust 3721 F is listed by PC/SC applications as

- *Identiv uTrust 3721 Contactless Reader*

5.3.2. Supported operating systems

- Windows Server 2012, 2016, 2019
- Windows 7 (x86, x64)
- Windows 8.1 (x86, x64)
- Windows 10 (x86, x64)
- macOS 10.12, 10.13, 10.14
- Linux 3.x, 4.x, 5.x (x86, x64)
- ChromeOS 48 and later
- Android 4.4 and later

5.3.3. PC/SC 2.0 compliant ATR for contactless interface

When a user credential is placed on the reader, initialization, anti-collision is done. The user credential is automatically activated and an ATR is built as defined in the PC/SC specification. For further information, please refer to section 3.1.3.2.3 of [PCSC3] and to [PCSC3-SUP].

5.3.3.1. ATR for contactless storage user tokens

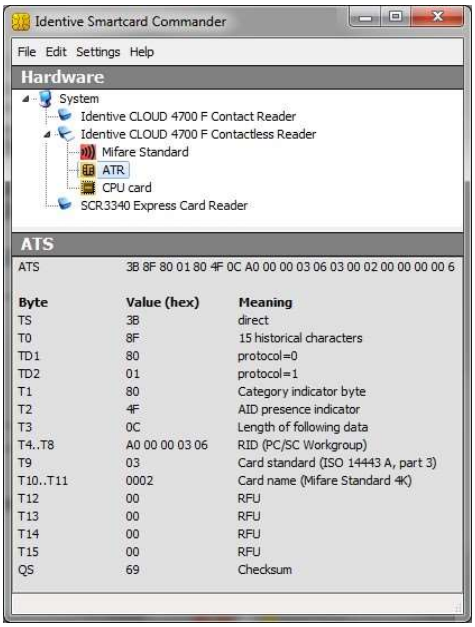
The ATR of the credential is composed as described in the table below. In order to allow the application to identify the storage card properly, it's Standard and Card name describing bytes must be interpreted according to the Part 3 Supplemental Document, maintained by PC/SC.

Credentials using technology like MIFARE are examples of this:

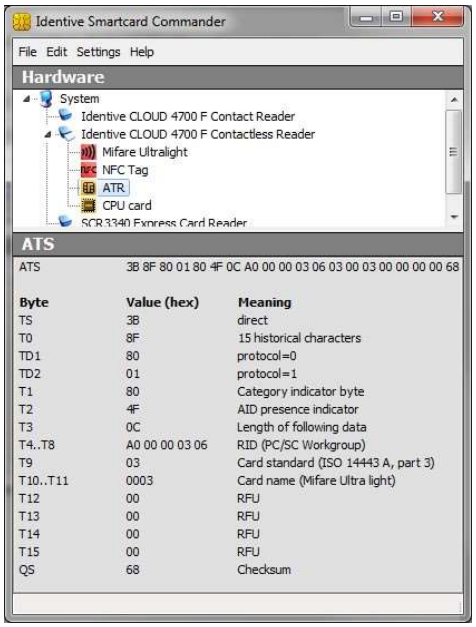
Byte#	Value	Designation	Description
0	0x3B	Initial header	
1	0x8n	T0	n indicates the number of historical bytes in following ATR
2	0x80	TD1	upper nibble 8 indicates no TA2, TB2, TC2 lower nibble 0 means T=0
3	0x01	TD2	upper nibble 0 indicates no TA3, TB3, TC3 lower nibble 1 means T=1
4...3+n	0x80		A status indicator may be present in an optional TLV data object
	0x4F	Optional TLV data object	Tag: Application identifier
	Length		1 byte
	RID		Registered identifier on 5 bytes
	PIX		Proprietary identifier extension on 3 bytes
	0x00 0x000x000x00		4 RFU bytes
4+n		TCK	XOR of all previous bytes

Example of the ATR built for contactless storage tokens:

MIFARE Classic 4K



MIFARE Ultralight



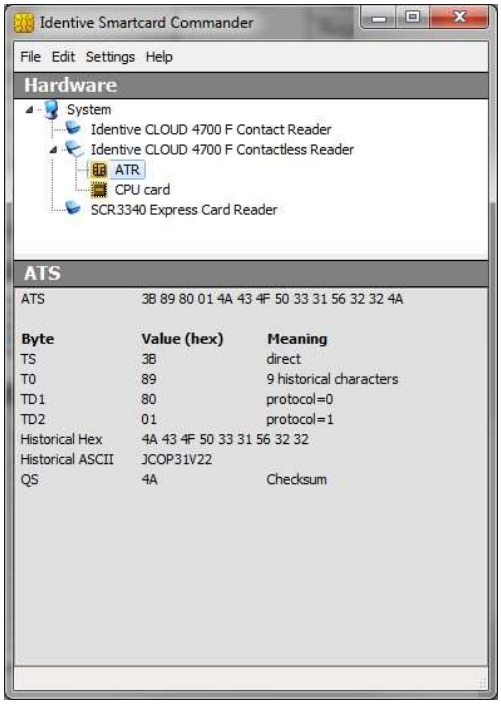
5.3.3.2. ATR for ISO/IEC 14443-4 user tokens

The credential exposes its ATS or application information which is mapped to an ATR. The table describes how this mapping is done.

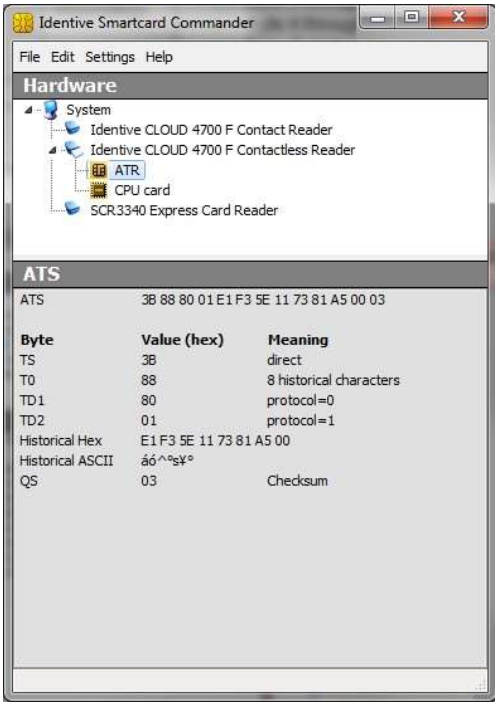
Byte#	Value	Designation	Description
0	0x3B	Initial header	
1	0x8n	T0	n indicates the number of historical bytes in following ATR
2	0x80	TD1	upper nibble 8 indicates no TA2, TB2, TC2 lower nibble 0 means T=0
3	0x01	TD2	upper nibble 0 indicates no TA3, TB3, TC3 lower nibble 1 means T=1
4...3+n		Historical bytes or application information	Type A: the historical bytes from the ATS (up to 15 bytes) Type B (8 bytes): • Byte 0 through 3: application data from ATQB, • Byte 4 through 6: protocol info byte from ATQB, • Byte 7: highest nibble is the MBLI (maximum buffer length index) from ATTRIB, lowest nibble is 0x0
4+n		TCK	XOR of all previous bytes

Example of the ATR built for an ISO14443-4 credential:

Type A



Type B



5.3.3.3. ATR for ISO/IEC 15693 tokens

The credential exposes its ATS or application information which is mapped to an ATR. The table describes how this mapping is done.

Byte#	Value	Designation	Description
0	0x3B	Initial header	
1	0x8n	T0	n indicates the number of historical bytes in following ATR
2	0x80	TD1	upper nibble 8 indicates no TA2, TB2, TC2 lower nibble 0 means T=0
3	0x01	TD2	upper nibble 0 indicates no TA3, TB3, TC3 lower nibble 1 means T=1
4...3+n	0x80		A status indicator may be present in an optional TLV data object
	0x4F	Optional TLV data object	Tag: Application identifier
	Length		1 byte
	RID (0xA000000306)		Registered identifier on 5 bytes
	PIX (0x0B 0xXX 0xXX)		Proprietary identifier extension on 3 bytes
	0x00 0x000x000x00		4 RFU bytes
4+n		TCK	XOR of all previous bytes

5.3.3.4. ATR for LF tokens

The credential exposes its ATS or application information which is mapped to an ATR. The table describes how this mapping is done.

Byte#	Value	Designation	Description
0	0x3B	Initial header	
1	0x8n	T0	n indicates the number of historical bytes in following ATR
2	0x80	TD1	upper nibble 8 indicates no TA2, TB2, TC2 lower nibble 0 means T=0
3	0x01	TD2	upper nibble 0 indicates no TA3, TB3, TC3 lower nibble 1 means T=1
4...3+n	0x80		A status indicator may be present in an optional TLV data object
	0x4F	Optional TLV data object	Tag: Application identifier
	Length		1 byte
	RID (0xA000000306)		Registered identifier on 5 bytes
	PIX (0x40 0xXX 0xXX)		Proprietary identifier extension on 3 bytes
	0x00 0x000x000x00		4 RFU bytes
4+n		TCK	XOR of all previous bytes

5.4. Firmware

5.4.1. CCID transport protocol

uTrust 372x F implements a transport protocol that is compliant with USB Device Class: *Smart Card CCID Specification for Integrated Circuit(s) Cards Interface Devices Revision 1.10*.

This paragraph describes the CCID specification features that are implemented.

5.4.1.1. CCID class requests supported

- Abort

5.4.1.2. CCID messages supported

The following CCID messages are supported for the contact interface when received through bulk-out endpoint.

- PC_to_RDR_IccPowerOn
- PC_to_RDR_IccPowerOff
- PC_to_RDR_GetSlotStatus
- PC_to_RDR_XfrBlock
- PC_to_RDR_GetParameters
- PC_to_RDR_ResetParameters
- PC_to_RDR_SetParameters
- PC_to_RDR_Escape
- PC_to_RDR_ICCClock
- PC_to_RDR_T0APDU
- PC_to_RDR_Abort
- PC_to_RDR_SetDataRateAndClockFrequency

5.4.1.3. CCID Error Codes

Extensive error codes are reported on many conditions during all CCID responses. Most of the error messages are reported by the CCID appropriately. Some of the main error codes for the contact interface are:

- HW_ERROR
- XFR_PARITY_ERROR
- ICC_PROTOCOL_NOT_SUPPORTED
- BAD_ATR_TS
- BAD_ATR_TCK
- ICC_MUTE
- CMD_ABORTED
- Command not supported

The following sub-sections discuss when and why these error codes are returned:

5.4.1.3.1. HW_ERROR

This error code is returned when a hardware short circuit condition is detected, during application of power to the card or if any other internal hardware error is detected.

5.4.1.3.2. XFR_PARITY_ERROR

This error code is returned when a parity error condition is detected. This error will be reported in the response to a PC_to_RDR_XfrBlock message.

5.4.1.3.3. ICC_PROTOCOL_NOT_SUPPORTED

This error code is returned if the card is signaling to use a protocol other than T=0 or T=1 in its ATR.

5.4.1.3.4. BAD_ATR_TS

This error code is returned if the initial character of the ATR contains invalid data.

5.4.1.3.5. BAD_ATR_TCK

This error code is returned if the check character of the ATR contains is invalid.

5.4.1.3.6. ICC_MUTE

This error code is returned when the card does not respond until the reader time out occurs. This error will be reported in the response to PC_to_RDR_XfrBlock message and PC_to_RDR_IccPowerOn messages.

5.4.1.3.7. CMD_ABORTED

This error code is returned if the command issued has been aborted by the control pipe.

5.4.1.3.8. Command not supported

This error would be returned, if the command would not be supported by the reader.

5.4.2. HID transport protocol

This is applicable to uTrust 3721 F only. uTrust 3721 F reader supports HID transport protocol besides the CCID transport protocol. The HID transport protocol allows to read information out of card, perform limited formatting and allows it to be fed into host as keystrokes.

6. Commands description

6.1. Generic APDU

6.1.1. Working with DESFire and MIFARE Plus tokens

To work with DESFire EV1 and MIFARE Plus tokens, please refer to the according application notes [AN337] and [AN338], respectively.

Please note that, since these application notes contain information available only under NDA with NXP, you'd need to sign an NDA with NXP to be allowed to receive them.

6.1.2. PAPDU_GET_UID

GET UID will retrieve the UID or SNR or PUPI of the user token. This command can be used for all supported technologies.

Command APDU:

CLA	INS	P1	P2	Lc	Data in	Le
0xFF	0xCA	0x00/0x01	0x00	-	-	XX

Response APDU:

Data	Status Word
Requested bytes of UID	SW1, SW2

Setting P1=0x00 and Le = 0x00 can be used to request the full UID or PUPI is sent back. For ISO14443A single 4 bytes or double 7 bytes or triple 10 bytes shall be returned. For ISO14443B 4 bytes PUPI shall be returned. For ISO15693 tags 8 bytes for ISO15693 tags. For LF tags, the raw data blocks shall be returned.

Setting P1=0x01 and Le=0x00 is specific to LF tags. When P1=0x01 reader interprets the raw data from card, matches it against one of the pre-defined formats and returns the processed data. Processed data would be of the form "FAC <space> XX <NUL> YYYYYY <NUL> CSN <space>" where <space> is ASCII space character (0x20) and <NUL> is ASCII NUL character (0x00).

For example, "FAC 27 40037 CSN" would be returned as

0x46 0x41 0x43 0x20 0x32 0x37 0x00 0x34 0x30 0x30 0x33 0x37 0x00 0x43 0x53 0x4E 0x20 0x90 0x00

Setting P1=0x01 for HF cards would return status word 0x6A81 indicating that this function is not supported.

6.1.3. PAPDU_ESCAPE_CMD

Usually escape commands are transmitted through SCardControl as defined in PCSC API using IOCTL_CCID_ESCAPE. But on some environments, the driver will block this IOCTL unless the registry has been edited to allow it. Hence this vendor specific APDU was defined to transmit Escape commands to the reader as below:

Command APDU:

CLA	INS	P1	P2	Lc	Data in	Le
0xFF	0xCC	0x00	0x00	Length of data	Escape Command Buffer	XX

Response APDU:

Data	Status Word
Reader Response	SW1, SW2

Example:

1) To issue the "READER_GETIFDTYPE (0x12)" escape command , this pseudo APDU would be used:

   Command APDU: FF CC 00 00 01 12
   Response : 12 56 90 00 ; if reader is uTrust 3720 F, or
   Response : 13 56 90 00 ; if reader is uTrust 3721 F

2) To issue the "READER_SETMODE (0X01)" escape command, this pseudo APDU would be used:

   Command APDU: FF CC 00 00 02 01 01 (to set to EMV mode)
   Response APDU : 90 00

Note:

1) To send Escape commands using this method, the reader should be connected in shared mode using T0 or T1 protocol. Only then would the resource manager allow SCardTransmit.
2) As the escape commands defined using "READER_GENERIC_ESCAPE" have ISO 7816 APDU format, they can be sent using SCardTransmit without having any need to prepend "FF CC 00 00 P3".

6.2. Supported Pseudo APDU (Contactless Interface)

All Pseudo APDUs specific to Contactless Interface supported in the reader are explained in this section

6.2.1. PAPDU_MIFARE_READ_BINARY

This command is used to read data from a Mifare card. Refer to section 3.2.2.1.8 of [PCSC3] for details.

Command APDU:

Command	CLA	INS	P1	P2	Lc	Data	Le
Read Binary	0xFF	0xB0	Addr MSB	Addr LSB	-	-	xx

P1 and P2 represent the block number of the block to be read, starting with 0 for sector 0, block 0, continuing with 4 for sector 1, block 0 (sector no. x 4 + block no.)

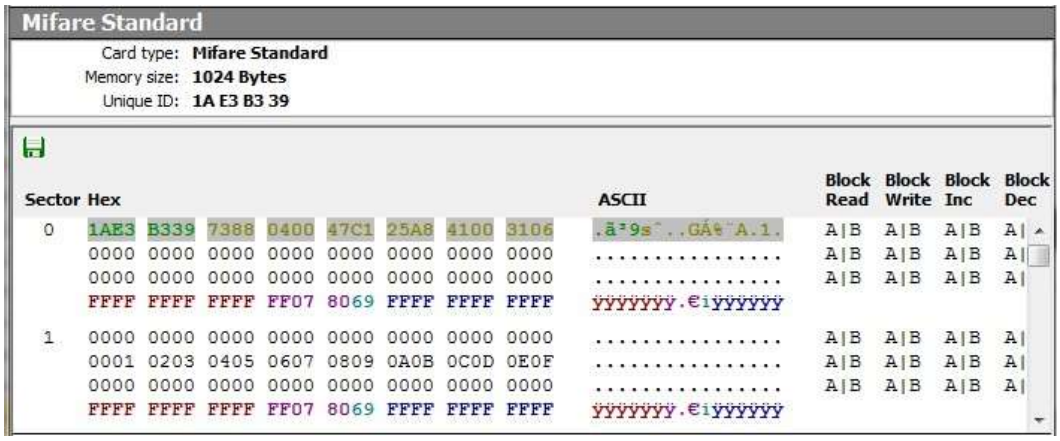
Regardless of the value given in Le, this command will always return the entire block content:
16 bytes for Mifare Classic 4
bytes for Mifare UL and UL C

Response APDU:

Data	Status Word
N bytes of block data	SW1, SW2

Example:

For a Mifare Classic 1K card with the following content:



The following command will read the sixth block and yield the mentioned output:

```
APDU: FF B0 00 05 02
SW12: 9000 (OK)
DataOut: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F (16 bytes)
```

6.2.2. PAPDU_MIFARE_UPDATE_BINARY

This command is used to update the non-volatile memory of a Mifare card. Refer to section 3.2.2.1.9 of [PCSC3] for further details.

Command APDU:

Command	CLA	INS	P1	P2	Lc	Data	Le
Update Binary	0xFF	0xD6	Addr MSB	Addr LSB	xx	data	-

For a description of P1 and P2, see PAPDU_MIFARE_READ_BINARY
Lc has got to match the block size of the used card
16 bytes for Mifare Classic 4
bytes for Mifare UL and UL C

Response APDU:

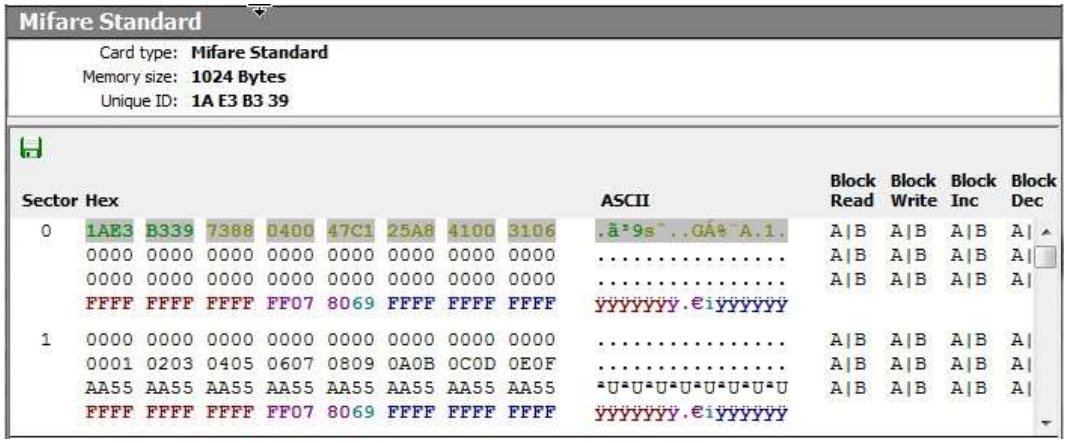
Data	Status Word
-	SW1, SW2

Example:

To write the bytes `AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55` to block 7 of a Mifare Classic 1K, the following command has got to be issued:

```
APDU: FF D6 00 06 10 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55
SW12: 9000 (OK)
```

Resulting in this content on the card:



6.2.3. PAPDU_MIFARE_LOAD_KEYS

This command is used to load the key to the volatile memory of the reader. It can be used for all kinds of contactless cards. Refer to section 3.2.2.1.4 of [PCSC3] for further details.

Command APDU:

Command	CLA	INS	P1	P2	Lc	Data	Le
Load Keys	0xFF	0x82	0x00	Key Num	Key data	Key	-

Response APDU:

Data	Status Word
-	SW1, SW2

Examples

Load Keys

The command to load Mifare key A "`FF FFFFFFFFFF`" is

```
FF82006006 FFFFFFFFFFFF
```

6.2.4. PAPDU_MIFARE_AUTHENTICATE

This command is used to authenticate using the key number. Refer to section 3.2.2.1.6 of [PCSC3] for further details.

Command APDU:

Command	CLA	INS	P1	P2	Lc	Data	Le
General Authenticate	0xFF	0x86	0x00	0x00	0x05	data	xx

The data structure is defined as follows:

Byte #	Value	Description
B0	0x01	Version
B1		Block Number MSB (always 0x00 for Mifare Classic cards)
B2		Block Number LSB
B3	0x60	Mifare Classic Key A
	0x61	Mifare Classic Key B
	0x00	Use key from non-volatile storage
B4	0x01	when B3=0x60 or B3=0x61
		Key number of key from non-volatile storage when B3=0

For authentication with MIFARE ULC using a key loaded into non-volatile storage use all 0x00 for data.

Response APDU:

Data	Status Word
-	SW1, SW2

Example:

1. Load Key A unencrypted and authenticate for block 6 (sector 1, actually) with that key:

```
APDU: FF 82 00 60 06 FF FF FF FF FF FF
SW12: 9000 (OK)
APDU: FF 86 00 00 05 01 00 06 60 01
SW12: 9000 (OK)
```

2. Authenticate with MIFARE ULC using key loaded into non-volatile storage:

```
APDU: FF 86 00 00 05 00 00 00 00 00
SW12: 9000 (OK)
```

6.2.5. PAPDU_MIFARE_READ_SECTOR

This command reads the specified sector from a Mifare Classic card (first 3 blocks of the sector, excluding the Key block) or the entire content of Mifare UL/UL C cards.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
Read Sector	FF	B1	Addr MSB	Addr LSB	0	-

Response APDU:

Data	Status Word
Mifare classic - 48 bytes of sector data read from card / Mifare UL – Entire card data is returned (64 bytes)	SW1, SW2

Example:

Read sector 1 of a Mifare Classic 1K

```
APDU: FF B1 00 01 00
SW12: 9000 (OK)
DataOut: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 (48 bytes)
```

Read entire content of a Mifare UL:

```
APDU: FF B1 00 01 10
SW12: 9000 (OK)
DataOut: 04 6B 5D BA 09 F8 01 80 70 48 00 00 E1 10 06 00
00 01 02 03 1D 6E 6F 6B 69 61 2E 63 6F 6D 3A 62
74 01 00 11 67 9F 5F B6 04 06 80 30 30 30 30 00
00 00 00 00 00 00 00 00 00 00 00 02 42 54 FE 00 (64 bytes)
```

6.2.6. PAPDU_MIFARE_READ_SECTOR_EX

This command read the specified sector from a Mifare Classic card (all the 4 blocks of the sector, including the Key block) or the entire content of Mifare UL/UL C cards.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
Read Sector Extended	FF	B3	Addr MSB	Addr LSB	0	-

Response APDU:

Data	Status Word
Mifare classic - 64 bytes of sector data read from card / Mifare UL – Entire card data is returned (64 bytes)	SW1, SW2

Example:

Read sector 1 of a Mifare Classic 1K

```
APDU: FF B3 00 01 10
SW12: 9000 (OK)
DataOut: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55 AA 55
00 00 00 00 00 00 FF 07 80 69 FF FF FF FF FF FF (64 bytes)
```

6.2.7. PAPDU_MIFARE_WRITE_SECTOR

This command writes the contained data to the specified sector of a Mifare classic or Mifare UL/UL C card (first blocks of the sector, excluding the Key block are written in case of Mifare Classic).

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
Write Sector	FF	D7	AddrMsb	AddrLsb	Lc	Data

Lc (P3) has got to be 0x30 when writing to the small sectors of a Mifare Classic and 0xF0 when writing to the large sectors of a Mifare Classic 4K.
Lc has got to be 0x30 for Mifare UL and the data will get written from block 4 till the end of the memory.

Response APDU:

Data	Status Word
-	SW1, SW2

6.2.8. PAPDU_MIFARE_VALUE_BLK_OLD

This command increments or decrements the data in a Value Block on a Mifare Classic card.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
Increment / Decrement OLD	FF	F0	00	Block Num	Lc	Data

where P2 codes the block number.

The data field is structured as follows

Byte #	Value	Description
B0	0xC0	Decrement
	0xC1	Increment
B1		Block number
B2-B5		Value (LSB first)

Response APDU:

Data	Status Word
-	SW1, SW2

Example: decrement block 4 by 1 (key loading and authentication not shown)
(block 4 has got to be set up as value block prior to executing this command, see datasheet for Mifare Classic cards)

```
APDU: FF B0 00 04 00 // Read Block 4
SW12: 9000 (OK)
DataOut: A9 AA AA AA 56 55 55 55 A9 AA AA AA 05 FA 05 FA (16 bytes)

APDU: FF F0 00 04 06 C0 04 01 00 00 00 // decrement block 4 by 1
SW12: 9000 (OK)

APDU: FF B0 00 04 00 // Read Block 4
SW12: 9000 (OK)
DataOut: A8 AA AA AA 57 55 55 55 A8 AA AA AA 05 FA 05 FA (16 bytes)
```

6.2.9. PAPDU_MIFARE_VALUE_BLK_NEW

This command increments or decrements the value of a data object if the card supports it. Refer to section 3.2.2.1.10 of [PCSC3-AMD1] for further details.

Command APDU:

Command	CLA	INS	P1	P2	Lc	Data	Le
Increment/ Decrement	FF	C2	00	03	xx	BERTLV	00

The data object consists of a TLV structure that defines, which action should be performed, which block the actions pertain to (the destination(s)) and which value should be applied for the action.

Tags for the action include:
0xA0: Increment
0xA1: Decrement

The Tag to define the destination is:
0x80: Destination

The Tag to define the value is:
0x81: value to increment or decrement Destination by, LSB first

Example:
Increment block 5 by 100

```
FF C2 00 03 0B
A0 09                    increment
80 01 05    block 5
81 04 64 00 00 00 by 100                 00
```

This command returns a Response APDU according to section 2.2 of [PCSC3-SUP2].

Response APDU:

Data	Status Word
C0 03 Error status, see below	SW1, SW2 (card itself will send SW1, SW2)

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 vale
XX 64 00	Data Object XX execution error (no response from IFD)
XX 64 01	Data Object XX execution error (no response from ICC)
XX 6F 00	Data object XX failed, no precise diagnosis

6.2.10. PAPDU_TCL_PASS_THRU (T=CL Pass Thru)

This command can be used to send raw data using T=CL protocol to a card. Please refer to the status words defined by the PICC manufacturer for a description of the status words

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
Pass-through	FF	FE	00	00	Lc	Data

Response APDU:

Data	Status Word
PICC response data	SW1, SW2 (card itself will send SW1, SW2)

6.2.11. PAPDU_ISO14443_PART3_PASS_THRU (Mifare Pass Thru)

This command is used to send raw data using Type A standard framing to a card. CRC bytes will be appended automatically. The reader will not add transport protocol data to the raw data – e.g. PCB, NAD, CID etc.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
Part 3 Pass-through	FF	EF	Transmit CRC	00	Lc	Data

P1 = 0x00 will transmit the CRC bytes form the card as is to the application. P1 = 0x01 will discard the CRC bytes.

Response APDU:

Data	Status Word
Data returned by card	SW1, SW2

6.2.12. PAPDU_ISO14443_PART4_PART3_SWITCH (TCL – Mifare Switch)

This command switches the card state between TCL and MIFARE modes

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
Part 4-Part 3 Switch	FF	F8	P1	00	00	-

P1 = 0x00 switches from MIFARE mode to TCL mode
P1 = 0x01 switches from TCL mode to MIFARE mode

Response APDU:

Data	Status Word
-	SW1, SW2

NOTE: This command is mainly targeted at Mifare plus S0 cards. Mifare plus card at S0 level get detected as Mifare memory card. In order to personalize these cards first it needs to be switched to Part 4 mode. For this purpose this user command needs to be issued using SCardTransmit function.

6.2.13. PAPDU_FELICA_REQC

This command Issues REQC as defined in JIS 7.5.1. It is used to detect the presence of a NFC Forum tag type 3 in the field

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
FeliCa REQC	FF	40	00	00	04	2 bytes of system code, 1 byte RFU, 1 byte TSN

Response APDU:

Data	Status Word
16 bytes of NFCID2 + 2 bytes of System Code (sent only if the RFU byte is 0x01)	SW1, SW2

6.2.14. PAPDU_FELICA_REQ_SERVICE

This command issues a REQ SERVICE as defined in JIS 9.6.2. P1. On receiving this command an NFC Forum tag type 3 will respond with the area key version of the specified area and the service key version of the specified service.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
FeliCa REQ Service	FF	42	Number of services/areas	00	2 * P1	Service Code List / Area Code List

Response APDU:

Data	Status Word
8 bytes IDm + No. of Service or areas(n) + Service version or area version list (2*n)	SW1, SW2

6.2.15. PAPDU_FELICA_REQ_RESPONSE

This command issues a REQ RESPONSE as defined in JIS 9.6.1. When an NFC Forum tag type 3 receives this command, it responds with its current mode (0/1/2).

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
FeliCa REQ Response	FF	44	00	00	00	-

Response APDU:

Data	Status Word
8 bytes IDm + Mode	SW1, SW2

6.2.16. PAPDU_FELICA_READ_BLK

This command issues a READ as defined in JIS 9.6.3

- P1 specifies the number of service
- P2 specifies the number of blocks
- Data buffer specifies the service code and block list

When an NFC Forum tag type 3 receives this command, it responds with the record value of the specified service.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
FeliCa REQ Response	FF	46	Number of service	Number of blocks	2*(P1 + P2)	Service Code List, Block List

Response APDU:

Data	Status Word
8 bytes IDm + Status Flag 1 + Status Flag 2 + No. of blocks(n) + Block data (n*16)	SW1, SW2

6.2.17. PAPDU_FELICA_WRITE_BLK

This command issues a WRITE as defined in JIS 9.6.4

- P1 specifies the number of service
- P2 specifies the number of blocks

When an NFC Forum tag type 3 receives this command, it writes the records of the specified service.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
FeliCa Write Block	0xFF	0x48	Number of service	Number of blocks	2*(P1 + P2) + (16 * P2)	Service Code List, Block List, Block Data

Response APDU:

Data	Status Word
8 bytes IDm + Status Flag 1 + Status Flag 2	SW1, SW2

6.2.18. PAPDU_FELICA_SYS_CODE

This command issues a REQ SYSTEM CODE as defined in RC-S850 / 860 Command-Ref-Manual Section 6.1.7

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
FeliCa REQ SYSTEM CODE	FF	4A	00	00	00	-

Response APDU:

Data	Status Word
8 bytes IDm + No. of System Codes (n) + System Code List (2n)	SW1, SW2

6.2.19. PAPDU_NFC_TYPE1_TAG_RID

This command issues a RID to get the tag's identification data.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
TYPE1 Tag RID	FF	50	00	00	00	-

Response APDU:

Data	Status Word
HR0 HR1 UID0 UID1 UID2 UID3	SW1, SW2

Where

- HR0 and HR1 are the 2 bytes Header ROM which identify the tag
- UID0 through UID3 are the first 3 bytes of the tag's UID.

Topaz tags have a 7 bytes long UID which can be fully fetched using the GET_UID APDU described earlier in this manual.

6.2.20. PAPDU_NFC_TYPE1_TAG_RALL

This command issues a RALL to read the two header ROM bytes and the whole of the static memory blocks 0x0-0xE.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
TYPE1 Tag RALL	FF	52	00	00	00	-

Response APDU:

Data	Status Word
HR0 HR1 120 bytes (Blocks 0 – E)	SW1, SW2

6.2.21. PAPDU_NFC_TYPE1_TAG_READ

This command issues a READ to read a single EEPROM memory byte within the static memory model area of blocks 0x0-0xE.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
TYPE1 Tag READ	FF	54	00	Byte Addr	00	-

Where P2 codes the address of the memory byte in the following way:

Bit numbers	Description
b7 – b3	Block # (value between 0x0 and 0xE)
b2 – b0	Byte # within the block (value between 0 and 7)

Response APDU:

Data	Status Word
Data returned by card	SW1, SW2

6.2.22. PAPDU_NFC_TYPE1_TAG_WRITE_E

This command issues a WRITE to erase and then write the value of 1 memory byte within the static memory model area of blocks 0x0-0xE.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
TYPE1 Tag WRITE ERASE	FF	56	00	Byte Addr	01	Data

Where P2 codes the address of the memory byte in the following way:

Bit numbers	Description
b7 – b3	Block # (value between 0x0 and 0xE)
b2 – b0	Byte # within the block (value between 0 and 7)

Response APDU:

Data	Status Word
Data returned by card	SW1, SW2

6.2.23. PAPDU_NFC_TYPE1_TAG_WRITE_NE

This command issues a WRITE-NE to write a byte value to one byte within the static memory model area of blocks 0x0-0xE. It does not erase the value of the targeted byte before writing the new data. Execution time of this command for NFC Forum tags type 1 is approximately half that of the normal write command (WRITE-E). Using this command, EEPROM bits can only be set, not reset.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
TYPE1 Tag WRITE No ERASE	FF	58	00	Byte Addr	01	Data

Where P2 codes the address of the memory byte in the following way:

Bit numbers	Description
b7 – b3	Block # (value between 0x0 and 0xE)
b2 – b0	Byte # within the block (value between 0 and 7)

Response APDU:

Data	Status Word
Data returned by card	SW1, SW2

6.2.24. PAPDU_NFC_TYPE1_TAG_RSEG

This command issues a RSEG to read out a complete segment (or block) of the memory within dynamic memory model.
Please note that this command works only on specific Topaz tags in the dynamic memory model.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
TYPE1 Tag READ SEGMENT	FF	5A	00	SegAddr	00	-

Where P2 Segment Address is:

Bit numbers	Description
b7 – b4	Segment (0x0 – 0xF)
b2 – b0	0

Response APDU:

Data	Status Word
128 bytes of data	SW1, SW2

6.2.25. PAPDU_NFC_TYPE1_TAG_READ8

This command issues a READ8 to read out a block of eight bytes.
Please note that this command only works on Topaz tags in dynamic memory model.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
TYPE1 Tag READ BLOCK	FF	5C	00	Block Addr	00	-

Where P2 Block Address is:

Bit numbers	Description
b7 – b0	General block (0x00 -0xFF)

Response APDU:

Data	Status Word
8 bytes of data	SW1, SW2

6.2.26. PAPDU_NFC_TYPE1_TAG_WRITE_E8

This command issues a WRITE8 to erase and then write a block of eight bytes.
Please note that this command only works on Topaz tags in dynamic memory model.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
TYPE1 Tag WRITE and ERASE BLOCK	FF	5E	00	Block Addr	08	Data

Where P2 Block Address is:

Bit numbers	Description
b7 – b0	General block (0x00 -0xFF)

Response APDU:

Data	Status Word
8 bytes of data that have been written	SW1, SW2

6.2.27. PAPDU_NFC_TYPE1_TAG_WRITE_NE8

This command issues a WRITE8 to write a block of eight bytes. It does not erase the value of the targeted byte before writing the new data. Using this command, EEPROM bits can be set but not reset.
Please note that this command only works on Topaz tags in dynamic memory model.

Command APDU:

Command	CLA	INS	P1	P2	P3	Data
TYPE1 Tag WRITE and NO ERASE BLOCK	FF	60	00	Block Addr	08	Data

Where P2 Block Address is:

Bit numbers	Description
b7 – b0	General block (0x00 -0xFF)

Response APDU:

Data	Status Word
8 bytes of data	SW1, SW2

6.3. Escape commands for the uTrust 372x F

With Amendment 1 of the PC/SC specification, Part 3, a method to define vendor specific commands has been introduced.

uTrust 372x F provides the command READER_GENERIC_ESCAPE to send commands using this method. However, most of the escape commands listed here are not defined according to this method because of backward compatibility reasons.

All newly defined commands will adhere to this new standard. See the command CONTACT_READ_INSERTION_COUNTER as an example.

6.3.1. Sending Escape commands to uTrust 372x F

A developer can use the following methods to send Escape commands to uTrust 372x F

- SCardControl method defined in PC/SC API
- SCardTransmit method defined in PC/SC API in conjunction with the Escape command APDU defined in this manual
  Please note, that SCardTransmit will only work when connected to a card.

In Windows, in order to be able to send Escape commands to the uTrust 372x F, the feature has got to be enabled by setting a REG_DWORD value named 'EscapeCommandEnable' in the registry to a value of '1'.

When using the Identiv supplied driver, this will not be necessary.

For Windows XP and Windows Vista, the key to hold the value for uTrust 3720 F would be HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USB\VID_04E6&PID_5612\ Device-Instance-xxxx \Device Parameters that for uTrust 3721 F would be HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USB\VID_04E6&PID_5613&MI_01\ Device-Instance-xxxx \Device Parameters

For Windows 7 and Windows 8, the value for uTrust 3720 F, contact part, would be HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USB\VID_04E6&PID_5612\Device-Instance-xxxx \Device Parameters\WUDFUsbccidDriver and that for uTrust 3721 F would be

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Enum\USB\VID_04E6&PID_5613&MI_01\Device-Instance-xxxx \Device Parameters\WUDFUsbccidDriver

Device-Instance-xxxx will be an auto-generated combination of four hexadecimal numbers, separated by '&', so this modification has got to be made for every physical reader/slot intended to be used on the machine in question. The reader has got to be plugged in at least once for the mentioned keys to exist and the driver has got to be restarted for this setting to take effect. (Unplug and re-plug the reader).

See appendix B for some sample code sending Escape commands to the reader.

6.3.2. Escape command codes

Escape commands can be used by an application to configure uTrust 372x F to function in a mode that is not its default configured mode or to get specific information. To put the uTrust 372x F back into its default mode, it either has to be unplugged and plugged again or the application can send the same Escape command again.

The following Escape commands are supported by uTrust 372x F:

6.3.3. Generic Commands Common to uTrust Contactless Interfaces

ESCAPE COMMAND	ESCAPE CODE
READER_GETIFDTYPE	0x12
READER_LED_CONTROL	0x19
READER_GETINFO_EXTENDED	0x1E
READER_LED_CONTROL_BY_FW	0xB2
READER_GENERIC_ESCAPE	FF 70 04 E6 XX

6.3.3.1. READER_GET_IFDTYPE

This Escape command is used to get the current IFD type from the reader.

Input:

The first byte of the input buffer contains the escape code.

Byte0
Escape code(0x12)

Output:

The reader returns its PID LSB first.

PID value		Description
B0	B1	
0x12	0x56	Identiv uTrust 3720 F
0x13	0x56	Identiv uTrust 3721 F

6.3.3.2. READER_LED_CONTROL

This Escape command is used to toggle the LED state. LED control by firmware should be disabled using the escape command READER_LED_CONTROL_BY_FW to see proper LED change when using this IOCTL.

Input:

The first byte of the input buffer contains the escape code, followed by LED number (if more than one LED is present, else set to 0) and then desired LED state. This will be required for production purpose.

Byte0	Byte 1	Byte2
Escape code(0x19)	LED number (0-RED,1-GREEN)	LED state (0-OFF, 1-ON)

Output:

Output buffer
NULL

6.3.3.3. READER_GET_INFO_EXTENDED

This Escape command is used to get the firmware version, reader capabilities, and Unicode serial number of the reader.

Input:

The first byte of the input buffer contains the escape code.

Byte0
Escape code(0x1E)

Output:

The firmware will return data as per structure SCARD_READER_GETINFO_PARAMS_EX mentioned below.

Field Size in Bytes	Field Name	Field Description	Value/Default
1	byMajorVersion	Major Version in BCD	Based on current firmware version
1	byMinorVersion	Minor Version in BCD	
1	bySupportedModes	0x00	ISO mode
2	wSupportedProtocols	Protocols supported by the Reader Bit 0 – T0 Bit 1 – T1	0x0003 Received as LSB first
2	winputDevice	IO_DEV_NONE 0x00 IO_DEV_KEYPAD 0x01 IO_DEV_BIOMETRIC 0x02	0x0000 Received as LSB first
1	byPersonality	Reader Personality (Not Used)	0x00
1	byMaxSlots	Maximum number of slots	0x01
1	bySerialNoLength	Serial number length (0x1C)	0x1C
28	abySerialNumber	Unicode serial number	Reader serial number Received as MSB first

6.3.3.4. READER_LED_CONTROL_BY_FW

This Escape command is used to enable/disable LED control by firmware.

Input:
The first byte of the input buffer contains the escape code. The second byte specifies if LED control by firmware should be disabled or enabled. The output buffer is NULL.

Byte0	Byte1	
	Value	Description
Escape code(0xB2)	0	Enable LED Control by firmware
	1	Disable LED Control by firmware
	FF	Get State: 0 -- LED control by firmware enabled 1 -- LED control by firmware disabled

Output:
No response is returned for set state. For Get State 1 byte response is received.

Output buffer
NULL or current state

6.3.3.5. READER_GENERIC_ESCAPE

This Escape command is used to invoke newly defined escape functions and send proprietary commands to the reader. It is defined in line with vendor specific generic command defined in [PCSC3AMD1].

Input:
The first five bytes of the input buffer shall follow APDU structure as per [PCSC3-AMD1]. 6TH byte shall be the command code used to identify the specific command.

Byte0	Byte1	Byte2	Byte3	Byte4	From Byte5 (up to Lc bytes)		Byte Lc+5
					Byte 5	Byte 6 onwards	
0xFF	0x70	0x04	0xE6	Lc (always > 0)	Cmd Opcode	Command parameters or data	Le (optional)

Output:
Depending on the command, the output shall be Le bytes of data + SW1 + SW2 or SW1+ SW2. The escape message shall at least return 2 bytes status word SW1, SW2. In case of success, SW1=0x90 and SW2=0x00 shall be returned. In error scenario, appropriate error status shall be returned (as defined in Error Code section 8.0).

6.3.4. Specific for Contactless Interface

ESCAPE COMMAND	ESCAPE CODE
CNTLESS_GETCARDINFO	0x11
CNTLESS_GET_ATS_ATQB	0x93
CNTLESS_GET_TYPE	0x94
CNTLESS_SET_TYPE	0x95
CNTLESS_CONTROL_PPS	0x99
CNTLESS_RF_SWITCH	0x96
CNTLESS_CONTROL_848	0x9D
CNTLESS_GET_BAUDRATE	0x9E
CNTLESS_CONTROL_RETRIES	0xA7
CNTLESS_CONTROL_POLLING	0xAC
CNTLESS_FORCE_BAUDRATE	0xAD
CNTLESS_GET_CARD_DETAILS	0xDA
CNTLESS_IS_COLLISION_DETECTED	0xE4
CNTLESS_FELICA_PASS_THRU	0xF3

6.3.4.1. CNTLESS_GET_CARD_INFO

This Escape command is used to get information about the contactless card placed in the field of the reader.

Input:

The first byte of input buffer contains the escape code.

Byte0
Escape code(0x11)

Output:

Byte0	Byte1	Byte2
Contactless card present (0x01)	Card to Reader communication baud rate (0xNN - see table below for details)	Card Type Info (Upper nibble indicates memory card/T=CL/dual mode card; Lower nibble indicates Type A/ Type B card See Table below for values)

Card to Reader communication baud rate BYTE is defined as follows:
b0 – 212kbps supported (direction reader to card)
b1 – 424kbps supported (direction reader to card)
b2 – 848kbps supported (direction reader to card)
b3 – always 0
b4 – 212kbps supported (direction card to reader)
b5 – 424kbps supported (direction card to reader)
b6 – 848kbps supported (direction card to reader)
b7– 1 – indicates same baud rate in both directions
0 – indicates different baud rates in both directions

Example:
If 0xNN = 0x77, the card supports all baud rates namely 106, 212, 424 and 848 kbps in both directions.
If 0xNN = 0xB3, the card supports 106, 212 and 424 kbps in both directions.

Card Type Info:

Upper Nibble Value	Description
0	Memory card
1	T=CL card
2	Dual mode card
Lower Nibble Value	
0	Type A card
1	Type B card

6.3.4.2. CNTLESS_GET_ATS_ATQB

This Escape command retrieves the ATS for Type A T= CL or the ATQB for Type B cards.

Input:
The first byte of input buffer contains the escape code.

Byte0
Escape code(0x93)

Output:
The output buffer contains the ATS bytes or the ATQB bytes depending on the type of PICC placed on the reader

6.3.4.3. READER_CNTLESS_GET_TYPE

This escape command retrieves the type of cards which the reader is configured to poll for. The input buffer shall contain the escape command code in the first byte and an optional extension specifier 0xFF in the second byte.

Input:

Byte0	Byte1
0x94	Empty or 0xFF

The output buffer shall point to a BYTE buffer in case the extension specifier is not given and will contain the type value coded as

Value	Description
0x00	Type A
0x01	Type B
0x02	Type A + type B

The output buffer shall point to a WORD buffer in case the extension specifier is given and will contain the type value coded as bitmask as

Cards-Type-Bit Mask (Lo Byte)								
Bit-7	**Bit-6**	**Bit-5**	**Bit-4**	**Bit-3**	**Bit-2**	**Bit-1**	**Bit-0**	**Card Type**
-	-	-	-	-	-	-	1	Type-A
-	-	-	-	-	-	1	-	Type-B
-	-	-	-	-	1	-	-	B-prime
-	-	-	-	1	-	-	-	B-prime-Sof
-	-	-	1	-	-	-	-	i-Class
-	-	1	-	-	-	-	-	FeliCa 212
-	1	-	-	-	-	-	-	FeliCa 424
1	-	-	-	-	-	-	-	Topaz

The Hi Byte will always be 0x00 (RFU).

6.3.4.4. READER_CNTLESS_SET_TYPE

This escape command configures the type of cards the reader will poll for.

Using this command can improve the polling efficiency for applications where only specific types of cards are expected.

This escape command needs to be used with care.

For example, we should not disable the polling for a given type of the card, after having placed that card on the reader.

Since reader would immediately stop polling for the card, it would never detect that card is removed.

The input buffer shall contain two or three bytes

Byte0	Byte1	Byte3	Description
Escape code(0x95)	0x00	-	Type A
	0x01	-	Type B
	0x02	-	Type A + type B
	0xFF	Bitmask	See the following table

Cards-Type-Bit Mask (Lo Byte)								
Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0	Card Type
-	-	-	-	-	-	-	1	Type-A
-	-	-	-	-	-	1	-	Type-B
-	-	-	-	-	1	-	-	B-prime
-	-	-	-	1	-	-	-	B-prime-Sof
-	-	-	1	-	-	-	-	i-Class
-	-	1	-	-	-	-	-	FeliCa 212
-	1	-	-	-	-	-	-	FeliCa 424
1	-	-	-	-	-	-	-	Topaz

The Hi Byte will always be 0x00 (RFU).

The output buffer is

Output buffer
NULL

6.3.4.5. CNTLESS_CONTROL_PPS

This Escape command disables the automatic PPS done by the firmware/device for contactless cards (HF only).

Input:

The first byte of input buffer contains the escape code.

The second byte either sets the mode or contains a code to retrieve the setting.

Input			Output
Byte0	Byte1 - PPS control byte		Byte0
Escape code(0x99)	0	Enable	No Output
	1	Disable	No Output
	FF	Get current status	0 – PPS is enabled 1 – PPS is disabled

Output:
No response is returned for set state. For Get State 1 byte response is received.

Output buffer
NULL or current state

6.3.4.6. CNTLESS_RF_SWITCH

This Escape command can be used to switch the RF field ON or OFF.

Input:
The first byte of input buffer contains the escape code.
The second byte either sets the mode or contains a code to retrieve the setting.

Byte0	Byte1		Output
	Value	Description	Byte0
Escape code(0x96)	0x00	Switch RF Field OFF	No Output
	0x01	Switch RF Field ON	No Output
	0xFF	Get current field state	0 – RF field is ON 1 – RF field is OFF

Output:
No response is returned for set state. For Get State 1 byte response is received.

Output buffer
NULL or current state

6.3.4.7. CNTLESS_CONTROL_848

This Escape command can be used to enable/disable 848kbps support as well as query whether 848kbps is currently enabled or disabled.
The RF communication with a user token will only switch to 848Kbps if the user token supports this baud rate and provided automatic PPS is ON.

The input buffer shall contain 2 bytes

Byte0	Byte1	Description
0x9D	0x00	Disable 848Kbps support
	0x01	Enable 848Kbps support
	0xFF	Get current status on 848Kbps support

If B1 of the input buffer is 0x00 or 0x01 then the output buffer is

Output buffer
NULL

If B1 of the input buffer is 0xFF, the output buffer is a BYTE buffer with following possible values

Output buffer	Description
0x00	848Kbps is disabled
0x01	848Kbps is enabled

6.3.4.8. CNTLESS_GET_BAUDRATE

This Escape command is used to get the current baud rate of card-reader communication.

Input:

The first byte of input buffer contains the escape code.

Byte0
Escape code(0x9E)

Output:

The output contains a byte with the following possible values

Byte0	Description
0x00	106Kbps in both directions
0x01	106Kbps from PICC to PCD, 212Kbps from PCD to PICC
0x02	106Kbps from PICC to PCD, 424Kbps from PCD to PICC
0x03	106Kbps from PICC to PCD, 848Kbps from PCD to PICC
0x10	212Kbps from PICC to PCD, 106Kbps from PCD to PICC
0x11	212Kbps in both directions
0x12	212Kbps from PICC to PCD, 424Kbps from PCD to PICC
0x13	212Kbps from PICC to PCD, 848Kbps from PCD to PICC
0x20	424Kbps from PICC to PCD, 106Kbps from PCD to PICC
0x21	424Kbps from PICC to PCD, 212Kbps from PCD to PICC
0x22	424Kbps in both directions
0x23	424Kbps from PICC to PCD, 848Kbps from PCD to PICC
0x30	848Kbps from PICC to PCD, 106Kbps from PCD to PICC
0x31	848Kbps from PICC to PCD, 212Kbps from PCD to PICC
0x32	848Kbps from PICC to PCD, 424Kbps from PCD to PICC
0x33	848Kbps in both directions

6.3.4.9. CNTLESS_CONTROL_RETRIES

This Escape command is used to enable/disable CRC/PROTOCOL/TIMEOUT error retries which are enabled by default for contactless cards.

Input:
The first byte of input buffer contains the escape code.
The second byte either sets the mode or contains a code to retrieve the setting.

Input			Output
Byte0	**Byte1- Description**		**Byte 0**
Escape code(0xA7)	0x00	Enable RNAK retries	No Output
	0x01	Disable RNAK retries	No Output
	0xFF	Get current state of retries	0x00 - Retries are enabled 0x01 - Retries are disabled

Output:
No response is returned for set state. For Get State 1 byte response is received.

Output buffer
NULL or current state

6.3.4.10. CNTLESS_CONTROL_POLLING

This Escape command is used to enable/disable firmware polling for contactless cards.

Input:
The first byte of input buffer contains the escape code.
The second byte either sets the mode or contains a code to retrieve the setting.

Input			Output
Byte0	**Byte1 - Description**		**Byte 0**
Escape code(0xAC)	0x00	Enable polling	No output
	0x01	Disable polling	No output
	0XFF	Get current state of polling	0x00 – Polling enabled 0x01 – Polling disabled

Output:
No response is returned for set state. For Get State 1 byte response is received.

Output buffer
NULL or current state

6.3.4.11. CNTLESS_FORCE_BAUDRATE

This escape command can be used to restrict the baud rate for contactless cards to certain values. The input buffer is

Byte #	Value	Description
B0	0xAD	Escape command code
B1	0x00	Use the baud rate specified by the card
	0x01	Only allow baud rates specified in B2
B2	b0- DR=2 supported, if bit is set to 1 b1- DR=4 supported, if bit is set to 1 b2- b3- DR=8 supported, if bit is set to 1 b4- b5- shall be set to 0, 1 is RFU b6- b7- DS=2 supported, if bit is set to 1 DS=4 supported, if bit is set to 1 b7- e DS=8 supported, if bit is set to 1 1 if the same D is required for both ıication directions 0 if different D is supported for ach ɹnication direction	Encoding of the baud rate to be allowed if B1 value is 0x01. No need to send this byte in case B1 has the value =x00
	NULL	If B1=0x00

The output buffer is

Output buffer
NULL

6.3.4.12. CNTLESS_GET_CARD_DETAILS

This Escape command is used to get details about the PICC placed in the field of the reader.

Input:

The first byte of input buffer contains the escape code.

Byte0
Escape code(0xDA)

Output:

Byte #	Value	Description
B0	0x00	Type A card
	0x01	Type B card
	0x04	FeliCa 212
	0x08	FeliCa 424
B1	0x00	Memory card
	0x01	T-CL card
	0x02	Dual interface card
	0x43	FeliCa
	0x44	Topaz
	0x45	B-prime
	0x46	i-Class
B2	'xx'	'xx' is the PUPI / UID Length
	0x08	For FeliCa cards
THEN EITHER		
B3-B12		PUPI/UID bytes 0x00 byte padding used if length smaller than 10
B13	0x00	CID not supported
	0x01	CID supported
B14	0x00	NAD not supported
	0x01	NAD supported
B15		Bit Rate Capability

Byte #	Value	Description
B16		FWI
B17		IFSC
B18		MBLI
B19		SAK
B20		SFGI
OR		
B3–B10		8 Bytes NFCID2
B11		Request service command response time parameter (see JIS6319 specification)
B12		Request response command response time parameter
B13		Authentication command response time parameter
B14		Read command response time parameter
B15		Write command response time parameter

6.3.4.13. CNTLESS_IS_COLLISION_DETECTED

This Escape command is used to identify if multiple Type A cards are detected in the field.

Input:

The first byte of input buffer contains the escape code.

Byte0
Escape code(0xE4)

Output:

Byte0	
Value	**Description**
0x00	Collision is not detected
0x01	Collision is detected

6.3.4.14. CNTLESS_FELICA_PASS_THRU

This Escape command is used as a pass through to send FeliCa commands to FeliCa cards.

Input:
The first byte of input buffer contains the escape code followed by FeliCa command to be sent to the card. At least 1 byte of command is required to be sent to the card. Otherwise an error will be reported.

Byte0	Byte1 onwards
Escape code (0xF3)	FeliCa command bytes

Output:
The response received from the FeliCa card is sent as output for this escape command.

6.3.5. Specific for Keyboard Interface

6.3.5.1. READER_CONTROL_KEYBOARD_SLOT

To enable or disable status of keyboard interface in SRAM, or to get current status of keyboard interface.

Command APDU:

Byte0 CLA	Byte1 INS	Byte2 P1	Byte3 P2	Byte4 Lc	Byte5 opcode	Byte6	Byte7	Le
0xFF	0x70	0x04	0xE6	0x03	0x11	0x01	0x00 – enable	0x00
0xFF	0x70	0x04	0xE6	0x03	0x11	0x01	0x01 – disable	0x00
0xFF	0x70	0x04	0xE6	0x02	0x11	0x00 – get current status		0x00

Response:

If the command is successful, 5 bytes are returned indicating the status of keyboard slot.

Byte Offset	Keyboard Description
Byte 0	0x01: Disabled in SRAM 0x00: Enabled in SRAM
Byte 1	0x01: Reader does not include keyboard interface 0x00: Reader includes keyboard interface
Byte 2	0x01: Enabled in OS-Configuration 0x00: Disabled in OS-Configuration
Byte 3	RFU
Byte 4	RFU

- Byte 1 would be 0x01 for readers that do not include keyboard interface, like, uTrust 3720 F. In this case the HID/keyboard interface shall not be enumerated.
- If the HID/keyboard interface is enumerated but disabled in OS-Configuration or SRAM, then keyboard output shall not be emitted until configuration is changed using suitable tool.

6.3.6. my-d Move Specific Commands

6.3.6.1. Access

This APDU performs password verification with my-d card.

Command APDU:

CLA	INS	P1	P2	Lc	opcode	Password	Le
0xFF	0xFD	0x04	0x01	0x05	0xB2	4 bytes	0x00

Response:

Data	Status Word	
-	SW1	SW2

6.3.6.2. Set Password

This APDU changes password of my-d move card with the new password value provided.

Command APDU:

CLA	INS	P1	P2	Lc	opcode	Password	Le
0xFF	0xFD	0x04	0x01	0x05	0xB1	4 bytes	0x00

Response:

Data	Status Word	
-	SW1	SW2

6.3.6.3. Compatibility Write

This APDU writes 16 bytes of data to my-d move at block specified.

Command APDU:

CLA	INS	P1	P2	Lc	opcode	Block No	Data	Le
0xFF	0xFD	0x06	0x01	0x12	0xA0	1 byte	16 bytes	0x00

Response:

Data	Status Word	
-	SW1	SW2

6.3.6.4. Write 2 Blocks (8 bytes)

This APDU writes 8 bytes of data to my-d move at block specified.

Command APDU:

CLA	INS	P1	P2	Lc	opcode	Block No	Data	Le
0xFF	0xFD	0x06	0x01	0x0A	0xA1	1 byte	8 bytes	0x00

Respons:

Data	Status Word	
-	SW1	SW2

6.3.6.5. Write 1 Block (4 bytes)

This APDU writes 4 bytes of data to my-d move at block specified.

Command APDU:

CLA	INS	P1	P2	Lc	opcode	Block No	Data	Le
0xFF	0xFD	0x04	0x01	0x06	0xA2	1 byte	4 bytes	0x00

Response:

Data	Status Word	
-	SW1	SW2

6.3.6.6. Read 4 Blocks (16 bytes)

This APDU reads 16 bytes of data from my-d move at block specified.

Command APDU:

CLA	INS	P1	P2	Lc	opcode	Block No	Le
0xFF	0xFD	0x01	0x01	0x02	0x30	1 byte	0x00

Response:

Data	Status Word	
16 bytes of data	SW1	SW2

6.3.6.7. Read 2 Blocks (8 bytes)

This APDU reads 8 bytes of data from my-d move at block specified.

Command APDU:

CLA	INS	P1	P2	Lc	opcode	Block No	Le
0xFF	0xFD	0x01	0x01	0x02	0x31	1 byte	0x00

Response:

Data	Status Word	
8 bytes of data	SW1	SW2

6.3.6.8. Decrement

This APDU decrements counter value of my-d move by the value specified.

Command APDU:

CLA	INS	P1	P2	Lc	opcode	Decrement Value	Le
0xFF	0xFD	0x06	0x01	0x03	0xD0	2 bytes	0x00

Response:

Data	Status Word	
-	SW1	SW2

6.3.7. ISO15693 Specific Commands

This sections explains the APDU that are supported for ISO15693 cards.

Command APDU:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0xFC	0x00	0x00	Number of bytes in Data field	Command as described in ISO15693-3	Expected number of bytes from card

Response:

Data	Status Word	
Data from card as described in ISO15693-3	SW1	SW2

In all the 15693 card inputs (commands), the optional flags byte and the optional UID field must be omitted. In all the 15693 card outputs (responses), the flags byte will be omitted and error code (if any) shall be sent as SW2.

6.3.7.1. Read Single Block

This APDU reads 4 bytes of data from block number specified.

Command APDU:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0xFC	0x00	0x00	0x02	0x20	Block number to read (1 byte)	0x00

Response:

Data	Status Word	
4 bytes of data from card	SW1	SW2

Example to read block number 0x0F

APDU: FF FC 00 00 02 20 0F 00

RESPONSE: xx xx xx xx 90 00

6.3.7.1. Write Single Block

This APDU writes 4 bytes of data to block number specified.

Command APDU:

CLA	INS	P1	P2	Lc	Data			Le
0xFF	0xFC	0x00	0x00	0x06	0x21	Block number to write (1 byte)	4 bytes of data	-

Response:

Data	Status Word	
-	SW1	SW2

Example to write data bytes “01 02 03 04” to block number 0x0F

APDU: FF FC 00 00 06 21 0F 01 02 03 04

RESPONSE: 90 00

6.3.7.3. Lock Block

This APDU locks the block number specified. Once successfully locked, the block shall become read only.

Command APDU:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0xFC	0x00	0x00	0x02	0x22	Block number to lock (1 byte)	-

Response:

Data	Status Word	
-	SW1	SW2

Example to lock block number 0x0F

APDU: FF FC 00 00 02 22 0F

RESPONSE: 90 00

6.3.7.4. Read Multiple Blocks

This APDU reads 4 bytes of data from each consecutive block, for number of blocks, starting from block number specified.

Command APDU:

CLA	INS	P1	P2	Lc	Data			Le
0xFF	0xFC	0x00	0x00	0x03	0x23	Start Block (1 byte)	Num blocks – 1 (1 byte)	0x00

Response:

Data	Status Word	
Data from card (number blocks * 4 bytes)	SW1	SW2

Example to read 04 consecutive blocks starting at block number 0x10

APDU: FF FC 00 00 03 23 10 03 00

RESPONSE: xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx 90 00

6.3.7.5. Write Multiple Blocks

This APDU writes 4 bytes of data to each consecutive block, for number of blocks, starting from block number specified.

Command APDU:

CLA	INS	P1	P2	Lc	Data				Le
0xFF	0xFC	0x00	0x00	3 + (4 * number of blocks)	0x24	Start Block (1 byte)	Number of blocks – 1 (1 byte)	Data to be written to each block; (4 * number of blocks) bytes	-

Response:

Data	Status Word	
-	SW1	SW2

Example to write "AA AA AA AA BB BB BB BB" to 02 consecutive blocks starting at block number 0x10

APDU: FF FC 00 00 0B 24 10 01 AA AA AA AA BB BB BB BB

RESPONSE: 90 00

6.3.7.6. Write AFI

This APDU writes the AFI value specified into card's memory.

Command APDU:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0xFC	0x00	0x00	0x02	0x27	AFI value (1 byte)	-

Response:

Data	Status Word	
-	SW1	SW2

Example to write AFI value 0xF0

APDU: FF FC 00 00 02 27 F0

RESPONSE: 90 00

6.3.7.7. Lock AFI

This APDU locks the AFI value on the card. Upon successful execution of this command, the AFI can no longer be updated.

Command APDU:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0xFC	0x00	0x00	0x01	0x28	-

Response:

Data	Status Word	
-	SW1	SW2

Example to lock AFI

APDU: FF FC 00 00 01 28

RESPONSE: 90 00

6.3.7.8. Write DSFID

This APDU writes the DSFID value specified into card's memory.

Command APDU:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0xFC	0x00	0x00	0x02	0x29	DSFID value (1 byte)	-

Response:

Data	Status Word	
-	SW1	SW2

Example to write DSFID value 0xF0

APDU: FF FC 00 00 02 29 F0

RESPONSE: 90 00

6.3.7.9. Lock DSFID

This APDU locks the DSFID value on the card. Upon successful execution of this command, the DSFID can no longer be updated.

Command APDU:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0xFC	0x00	0x00	0x01	0x2A	-

Response:

Data	Status Word	
-	SW1	SW2

Example to lock DSFID

APDU: FF FC 00 00 01 2A

RESPONSE: 90 00

6.3.7.10. Get System Info

This APDU retrieves the system information from the card. The system information includes UID, DSFID, AFI, memory size and manufacturer information.

Command APDU:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0xFC	0x00	0x00	0x02	0x2B 0x00	0x00

Response:

Data	Status Word	
System Information Data	SW1	SW2

Note that the size of system information returned can vary between cards depending on the number of field that are available.

System Information Data is formatted as below:

System Information Data					
Info flags 1 byte	UID 8 bytes	DSFID 1 byte	AFI 1 byte	Memory Size 2 bytes	IC Reference 1 byte

Bits in Info flags byte provide information about the presence or absence of other fields. Info flags is formatted as below:

Bit	Value	Description
b1	0	DSFID not present
	1	DSFID present
b2	0	AFI not present
	1	AFI present
b3	0	Memory size not present
	1	Memory size present
b4	0	IC reference not present
	1	IC reference present
b5-b8	0	RFU

Memory size is interpreted as below:
MSB gives the block size in bytes – 1 (add 1 to get number of bytes in block)
LSB gives the number of blocks – 1 (add 1 to get number of physical blocks)

Example to lock AFI

APDU: FF FC 00 00 02 2B 00

RESPONSE: xx xx xx xx xx xx xx xx xx xx xx xx xx xx xx 90 00

6.3.7.11. Read Multiple Block Security Status

This APDU retrieves the block security status of each consecutive block, for number of blocks, starting from block number specified.

Command APDU:

CLA	INS	P1	P2	Lc	Data			Le
0xFF	0xFC	0x00	0x00	0x03	0x2C	Start Block (1 byte)	Num blocks – 1 (1 byte)	0x00

Response:

Data	Status Word	
Security status bytes (number blocks * 1 bytes)	SW1	SW2

Example to read security status of 04 consecutive blocks starting at block number 0x10

APDU: FF FC 00 00 03 2C 10 03 00

RESPONSE: xx xx xx xx 90 00

6.3.7.12. Traverse

Traverse APDU is used to send the "Raw Card Command" in data field to the card without any card specific processing by the reader and returns the raw response data from the card. The reader only takes care of protocol specific processing (like CRC, Prologue field,…). This command is intended for sending ISO15693 custom commands defined by card manufacturer.

The reader uses the Frame Type specified in the P2 parameter field and the Frame Waiting Time (FWT) specified in the P1 parameter field while transmitting command and receiving response respectively.

Command APDU:

CLA	INS	P1	P2	Lc	Data	Le
0xFF	0xFD	FWT code	Frame Type	Number of data bytes	Raw Card Command	0x00

The following table provides the FWT codes for P1 parameter and the corresponding wait time.

FWT Code	FWT in uSeconds	FWT Code	FWT in uSeconds
0x00	500	0x0B	750000
0x01	1000	0x0C	1000000
0x02	2000	0x0D	1250000
0x03	5000	0x0E	1500000
0x04	10000	0x0F	1750000
0x05	25000	0x10	2000000
0x06	50000	0x11	2500000
0x07	75000	0x12	3000000
0x08	100000	0x13	4000000
0x09	250000	0x14	5000000
0x0A	500000		

The following table provides the Frame Types for P2 parameter and their description:

Frame Type	Description
0x00	FRAMETYPE_SHORT
0x01	FRAMETYPE_STD
0x02	FRAMETYPE_ACBITORIENTED

Response:

Data	Status Word	
Response from card	SW1	SW2

The response from card is returned raw without any processing. Reception of any response from the card is considered as success irrespective of the content of the response. The calling application needs to process the specific response from card.

6.4. Reader Key Management

The uTrust 372x F reader provides provision to store card keys in its non-volatile memory. The reader can store custom/user card keys. An authenticated user can later refer to them during card communication using key numbers. This section describes the command used to achieve this functionality in detail.

6.4.1. Reader Authenticate

The Reader Authenticate command is used to authenticate with uTrust 372x F. The PIN specified in the command is verified with the PIN stored in the uTrust 372x F.
Only after a successful Reader Authenticate, the user can use the Reader Load Keys command to store card specific keys or modify the Reader PIN in non-volatile area. The default PIN is "00 00 00 00 00 00 00 00".

This command ensures that a malicious user does not gain access to modify the keys stored in reader.

Command APDU:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0x00	0x00	0x00	0x09	0x09	Reader PIN (8 bytes)	-

Response:

Data	Status Word	
-	SW1	SW2

The authentication state shall be reset immediately after the first Reader Load Keys command that follows the Reader Authenticate command; this reset happens irrespective of whether the Reader Load Keys command is successful or not. So user shall have to authenticate with reader every time before issuing Load Keys command.

6.4.2. Reader Load Keys

The Reader Load Keys command is used to store card authentication keys and reader PIN in the non-volatile area of uTrust 372x F.

A successful Reader Authenticate command should have been executed before using this command.

The reader has provision to store

- 1 Reader PIN
- 16 MIFARE keys along with key type
- 4 DESFire keys (1 PICC Master key and 3 Application keys) along with AID, PCD key number and PICC key number
- 6 MIFARE Plus AES sector keys and 10 special keys with key block number
- 1 authentication key for MIFARE Ultralight C cards

When a card specific Authenticate APDU is received from host, the appropriate keys are fetched from non-volatile memory of reader and used for authentication. This command does not require the presence of a card over the reader.

Command APDU:

CLA	INS	P1	P2	Lc	Data		Le
0xFF	0x00	0x00	0x00	Number of bytes in Data field	0x07	Key Data	-

Response:

Data	Status Word	
-	SW1	SW2

Reader Load Keys command should be used to store the appropriate keys before attempting to issue Authenticate APDU to respective card type.

6.4.2.1. Load Reader Authentication PIN into Reader

The reader PIN can be any 8 byte numeric value. The following command can be used to change reader PIN.

Command APDU:

CLA	NS	P1	P2	Lc	Data			Le
0xFF	0x00	0x00	0x00	0x0A	0x07	0xFF	Reader PIN (8 bytes)	-

Response:

Data	Status Word	
-	SW1	SW2

Example to change PIN to “01 02 03 04 05 06 07 08”

APDU: FF 00 00 00 0A 07 FF 01 02 03 04 05 06 07 08
RESPONSE: 90 00

6.4.2.2. Load MIFARE Authentication Keys into Reader

The following command can be used to load MIFARE authentication keys into the reader.

Command APDU:

CLA	NS	P1	P2	Lc	Data			Le
0xFF	0x00	0x00	0x00	0x0A	0x07	0x00	MIFARE key data (as shown below)	-

MIFARE key data		
Key Number (1 byte)	Key Type (1 byte)	Key (6 bytes)

Where,

Key Number – any value from 0x00 to 0x4F
Key Type - 0x60 (Key Type A) for 0x61 (Key Type B)

Response:

Data	Status Word	
-	SW1	SW2

Example to load MIFARE keys with PCD Key Number = 0x00, Key Type = Key A

APDU: FF 00 00 00 0A 07 00 00 60 FF FF FF FF FF FF
RESPONSE: 90 00

6.4.2.3. Load DESFire Authentication Keys into Reader

The following command can be used to load DESFire authentication keys into the reader.

Command APDU:

CLA	NS	P1	P2	Lc	Data			Le
0xFF	0x00	0x00	0x00	0x1F	0x07	0x01	DESFire key data (as shown below)	-

DESFire key data					
PCD Key Number (1 byte)	AID (3 bytes)	PICC Key Number (1 byte)	Key 1 (8 bytes)	Key 2 (8 bytes)	Key 3 (8 bytes)

Where,

PCD Key Number – any value from 0x00 to 0x04
- Key number 0x00 refers to PICC master key
- Key numbers 0x01 to 0x03 refer to Application keys

AID - Application identifier in the card to which the key belongs
- Must be 000000 for PICC master key

PICC Key Number - Key number to be used in the DESFire Authenticate command
Key (1-2-3) - Can be DES/TDES or AES key
- DES key: (Key 1 = Key 2)
- TDES key: (Key 1 ≠ Key 2); (Key 1 = Key 3)
- AES key: (Key 3 = all zeros)

- 3KTDES: (Key 1, Key 2, Key 3 can be any value)

Reader Load Keys shall fail if any of the command parameters is invalid.

Response:

Data	Status Word	
-	SW1	SW2

Examples:

1. Load TDES PICC Master Key = “11 22 33 44 55 66 77 88 12 34 56 78 12 34 56 78”, with implicit AID = 000000, PCD Key Number = 00, PICC Key Number = 00

   APDU: FF 00 00 00 1F 07 01 00 00 00 00 00 11 22 33 44 55 66 77 88 12 34 56 78 12 34 56 78 11 22 33 44 55 66 77 88
   RESPONSE: 90 00

2. Load TDES PICC Application Key = “11 22 33 44 55 66 77 88 12 34 56 78 12 34 56 78” for application with AID = C1B1A1, PCD Key Number = 01, PICC Key Number = 00

   APDU: FF 00 00 00 1F 07 01 01 A1 B1 C1 00 11 22 33 44 55 66 77 88 12 34 56 78 12 34 56 78 11 22 33 44 55 66 77 88
   RESPONSE: 90 00

3. Load AES PICC Application Key = “AA AA AA AA BB BB BB BB CC CC CC CC EE EE EE EE” for application with AID = C3C2C1, PCD Key Number = 02, PICC Key Number = 01

   APDU: FF 00 00 00 1F 07 01 02 C1 C2 C3 01 AA AA AA AA BB BB BB BB CC CC CC CC EE EE EE EE 00 00 00 00 00 00 00 00
   RESPONSE: 90 00

4. Load 3KTDES Application Key = “AA AA AA AA BB BB BB BB CC CC CC CC EE EE EE EE 11 22 33 44 55 66 77 88” for application with AID = C3C2C1, PCD Key Number = 03, PICC Key Number = 01

   APDU: FF 00 00 00 1F 07 01 03 C1 C2 C3 01 AA AA AA AA BB BB BB BB CC CC CC CC EE EE EE EE 11 22 33 44 55 66 77 88
   RESPONSE: 90 00

6.4.2.4. Load MIFARE Plus Authentication Keys into Reader

The following command can be used to load MIFARE Plus AES authentication keys into the reader.

Command APDU:

CLA	NS	P1	P2	Lc	Data			Le
0xFF	0x00	0x00	0x00	0x15	0x07	0x03	MIFARE Plus key data (as shown below)	-

MIFARE Plus key data			
PCD Key Number (1 byte – see below)	PICC Key Block (LSB) Refer NXP-163735	PICC Key Block (MSB) Refer NXP-163735	AES Key (16 bytes)

PCD Key Number determines the location where the key is being stored in non-volatile memory of reader.

PCD Key Number	Key Description
0x00 – 0x0D	SL3 AES Sector Keys
0x00 – 0x09	Special Keys (Refer to NXP-163735 for details)

PICC Key Block number shall be used to identify the type of key being loaded.

Up to 14 AES sector keys and all special keys can be store in non-volatile memory of reader using this command. Care should be taken to load keys at different locations by changing PCD Key Number; otherwise, keys shall be overwritten and only the last loaded key shall be available. AES sector keys and special keys do not share the same memory space.

Reader Load Keys shall fail if any of the command parameters is invalid.

Response:

Data	Status Word	
-	SW1	SW2

Examples:

1. Load SL3 AES Sector Key = “11 22 33 44 55 66 77 88 12 34 56 78 12 34 56 78” for Sector 01, Key A, PCD Key Number 01

   APDU: FF 00 00 00 15 07 03 01 02 40 11 22 33 44 55 66 77 88 12 34 56 78 12 34 56 78
   RESPONSE: 90 00

2. Load Card Master Key (a special key) = “11 22 33 44 55 66 77 88 88 77 66 55 44 33 22 11” at PCD Key Number 01

   APDU: FF 00 00 00 15 07 03 01 00 90 11 22 33 44 55 66 77 88 88 77 66 55 44 33 22 11
   RESPONSE: 90 00

6.4.2.5. Load MIFARE ULC Authentication Keys into Reader

The following command can be used to load MIFARE ULC authentication keys into the reader.

Command APDU:

CLA	NS	P1	P2	Lc	Data			Le
0xFF	0x00	0x00	0x00	0x12	0x07	0x04	MIFARE ULC key data (16 bytes)	-

Reader Load Keys shall fail if any of the command parameters is invalid.

Response:

Data	Status Word	
-	SW1	SW2

Example for loading MIFARE ULC keys into reader, where key = “49 45 4D 4B 41 45 52 42 21 4E 41 43 55 4F 59 46”

APDU: FF 00 00 00 12 07 04 49 45 4D 4B 41 45 52 42 21 4E 41 43 55 4F 59 46
RESPONSE: 90 00

7. Annexes

7.1. Annex A – Status words table

SW1	SW2	Description
0x90	0x00	NO ERROR
0x63	0x00	NO INFORMATION GIVEN
0x65	0x81	MEMORY FAILURE
0x67	0x00	LENGTH INCORRECT
0x68	0x00	CLASS BYTE INCORRECT
0x69	0x82	SECURITY STATUS NOT SATISFIED
0x69	0x83	AUTHENTICATION IS REQUIRED
0x69	0x88	WRONG KEY NUMBER IN AUTHENTICATE
0x6A	0x81	FUNCTION NOT SUPPORTED
0x6B	0x00	WRONG PARAMETER P1-P2
0x6C	0xXX	WRONG Le VALUE; CORRECT Le IS GIVEN BY XX
0x6D	0x00	INVALID INSTRUCTION BYTE
0x6E	0x00	CLASS NOT SUPPORTED
0x6F	0x00	UNKNOWN COMMAND

7.2. Annex B – Sample code using escape commands

```
// File Name: uTrust 372x F Escape.h

#ifndef _uTrust_372xF_ESCAPE_H_
#define _uTrust_372xF_ESCAPE_H_

#ifdef __cplusplus
extern"C" {
#endif

#pragma pack(1)
typedef struct
{
   BYTE byMajorVersion;
   BYTE byMinorVersion;
   BYTE bySupportedModes;
   WORD wSupportedProtocols;
   WORD winputDevice;
   BYTE byPersonality;
   BYTE byMaxSlots;
   BYTE bySerialNoLength;
   BYTE abySerialNumber [28];
} ReaderInfoExtended;
#pragma pack()

#define IOCTL_CCID_ESCAPE     SCARD_CTL_CODE (0xDAC)

#define READER_GETIFDTYPE           0x12
#define READER_LED_CONTROL          0x19
#define READER_LED_CONTROL_BY_FW    0xB2
#define READER_GETINFO_EXTENDED     0x1E
#define CNTLESS_GETCARDINFO                 0x11
#define CNTLESS_GET_ATS_ATQB                0x93
#define CNTLESS_CONTROL_PPS                 0x99
#define CNTLESS_RF_SWITCH                   0x96
#define CNTLESS_GET_BAUDRATE                0x9E
#define CNTLESS_CONTROL_RETRIES             0xA7
#define CNTLESS_CONTROL_POLLING             0xAC
#define CNTLESS_GET_CARD_DETAILS            0xDA
#define CNTLESS_IS_COLLISION_DETECTED       0xE4
#define CNTLESS_FELICA_PASS_THRU            0xF3

#ifdef __cplusplus
}
#endif

#endif
// EOF
```

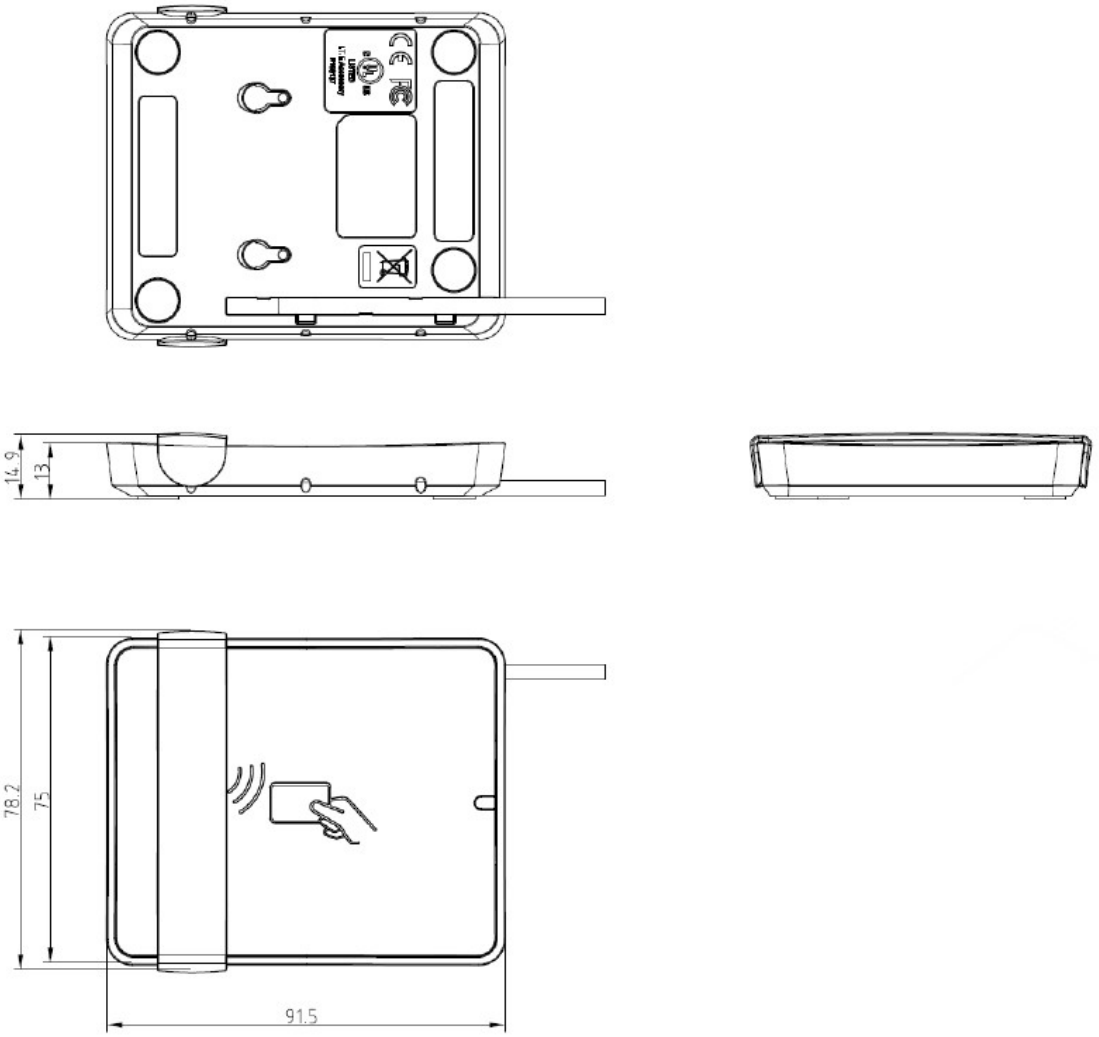
```
// File Name: uTrust 372x F Escape.c

#include <windows.h>
#include <winbase.h>
#include <stdio.h>
#include <conio.h>
#include "winscard.h"
#include "winerror.h"
#include "uTrust 372xF Escape.h"

VOID main(VOID)
{
      SCARDCONTEXT      ContextHandle = -1;
      SCARDHANDLE       CardHandle;
      ReaderInfoExtended      strReaderInfo;
      BYTE        InByte = 0, I = 0;
      DWORD       BytesRead = 0, ActiveProtocol = 0;
      ULONG       ret = 0;
      char        *s = NULL;
      char        *ReaderName[] = {
                        "Identiv uTrust 3720 Contactless Reader 0",
                        "Identiv uTrust 3721 Contactless Reader 0",
                        NULL};

/*****************************************************************************
**************/

      ContextHandle = -1;

      ret = SCardEstablishContext(SCARD_SCOPE_USER, NULL,
                  NULL,&ContextHandle);

      if (SCARD_S_SUCCESS == ret) {
            s = ReaderName[0];
            printf("Connecting to reader %s\n", s);
            ret = SCardConnect( ContextHandle,
                        s,
                        SCARD_SHARE_DIRECT,
                        SCARD_PROTOCOL_UNDEFINED,
                        &CardHandle,
                        &ActiveProtocol);

            if (SCARD_S_SUCCESS == ret) {
                  InByte = READER_GETINFO_EXTENDED;
                  ret = SCardControl( CardHandle,
                              IOCTL_CCID_ESCAPE,
                              &InByte,
                              1,
                              &strReaderInfo,
                              sizeof(strReaderInfo),
                              &BytesRead);
                  if (SCARD_S_SUCCESS == ret) {
                        printf("Version:\t\t%d%d\n",
      (strReaderInfo.byMajorVersion& 0xF0)>> 4,
      (strReaderInfo.byMinorVersion & 0x0F));
                        printf("modes:\t\t\t%d\n",
       strReaderInfo.bySupportedModes);
```

```
                        printf("protocols:\t\t%04x\n",
        strReaderInfo.wSupportedProtocols);
                        printf("input device:\t\t%04x\n",
        strReaderInfo.winputDevice);
                        printf("personality:\t\t%d\n",
        strReaderInfo.byPersonality);
                        printf("maxslots:\t\t%d\n",
        strReaderInfo.byMaxSlots);
                        printf("serial no length:\t%d\n",
        strReaderInfo.bySerialNoLength);
                        printf("serial no:\t\t");
                        for (i = 0; i <strReaderInfo.bySerialNoLength; i++) {
                              if (strReaderInfo.abySerialNumber[i] != 0) {
                                    printf("%c",
                                    strReaderInfo.abySerialNumber[i]);
                              }
                        }
                  } else {
                        printf("SCardControl failed: %08X\n", ret);
                  }
            } else {
                  printf("SCardConnect failed: %08X\n", ret);
            }
            ret = SCardReleaseContext(ContextHandle);
      }
      else {
            printf("\n SCardEstablishContext failed with %.8lX",ret);
      }

      printf("\npress any key to close the test tool\n");  getch();
}

// EOF
```

7.3. Annex C – Mechanical drawings

7.3.1. Reader (without stand)

NOTE: All dimensions on these mechanical drawings are in millimeters.



7.3.2. Reader with Stand