TWN4

Description PC/SC

DocRev10, March 5, 2021

ELATEC GmbH

Contents

1. PC/SC Specification Overview

TWN4 is the name of a powerful and versatile series of RFID readers and writers. PC/SC (short for “Personal Computer/Smart Card”) is a specification for smart card integration into computing environments.

This document describes how to setup a PC/SC environment with TWN4 and different kinds of Integrated Circuit Cards.

1.1. PC/SC Components

1.2. Integrated Circuit Card (ICC)

The ICC (commonly called a “smart card”) is a credit card sized plastic case with an embedded microprocessor chip.

1.3. Interface Device (IFD)

The IFD (commonly called a “smart card reader”) is the physical interface through which ICCs communicate with a PC. The IFD provides DC power to the microprocessor chip (DC = direct current, as opposed to alternating current). Also, the IFD provides a clock signal, which is used to step the program counter of the microprocessor, as well as an I/O line through which digital information may be passed between the IFD and the ICC.

1.4. IO Device Driver

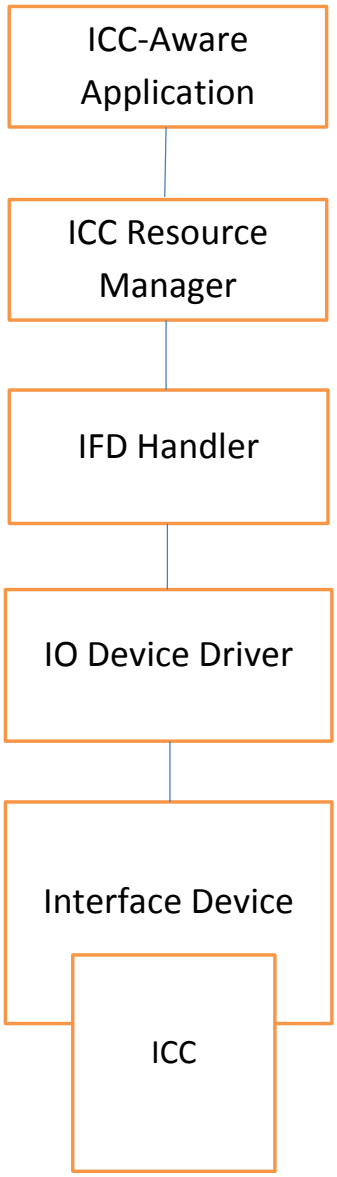
The IO Device Driver implements the low level communication between the IFD and the Host. A USB document[3] describes proposed requirements and specifications for Universal Serial Bus (USB) devices that interface with Integrated Circuit Cards or act as interfaces with Integrated Circuit Cards.

1.5. Interface Device Handler (IFD Handler)

The IFD Handler encompasses the PC software necessary to map the native capabilities of the IFD to the IFD Handler interface defined in Part 3 of this specification.

Under Linux, the corresponding software component can be for example libccid.

Figure 1.1.: PC/SC Components



1.6. ICC-Aware Application

The ICC-Aware Application (“Application”) is an arbitrary software program within the PC operating environment, which wants to make use of the functionality provided by one or more ICCs.

Under Linux, the corresponding software component can be for example opensc-tool.

1.7. Resource Manager

The ICC Resource Manager is a key component of the PC/SC Workgroup's architecture. It is responsible for managing the other ICC-relevant resources within the system and for supporting controlled access to IFDs and, through them, individual ICCs.

Under Linux, the corresponding software component can be for example pcscd.

2. Typical ICC-Aware Application Workflow

Typically, an ICC-aware application that the user implements should distinguish the type of the ICC, because different ICCs have different APDUs supported. This can be done by reading the Answer to Reset (ATR) of the ICC.

Once the ATR is read, the ICC-aware application can send APDU commands to TWN4, in order to read or write data saved on ICC.

2.1. Distinguish the Type of ICC

2.1.1. Answer to Reset (ATR)

An Answer To Reset (ATR) is a message output by a contact card conforming to ISO/IEC 7816 standards, following electrical reset of the card's chip by a card reader. The ATR conveys information about the communication parameters proposed by the card, and the card's nature and state.

How an ATR is constructed and how to parse an ATR is beyond the range of this document. Please refer to ISO 7816-3 [1] for more information.

For contactless ICCs, the IFD subsystem must construct an ATR from the fixed elements that identify the cards. Please refer to Interoperability Specification for ICCs and Personal Computer Systems Part 3 [4] for more information.

2.2. Exchange Data to ICC

2.2.1. APDU

An Application Protocol Data Unit (APDU) is the communication unit between a smart card reader and a smart card. The structure of the APDU is defined by ISO/IEC 7816-4 [2].

2.2.2. Structure of an APDU

There are two categories of APDUs: command APDUs and response APDUs. A command APDU is sent by the reader to the card. It contains a mandatory 4-byte header (CLA, INS, P1, P2) 0 to 65534 bytes of data.

A response APDU is sent by the card to the reader. It contains from 0 to 65 535 bytes of data, and 2 mandatory status bytes (SW1, SW2).

ISO 7816-4 defines interindustry commands for interchange as well as the responses for the commands. Please refer to it for more information.

Command APDU		
Field name	**Lenth(bytes)**	**Description**
CLA	1	Instruction class - indicates the type of command, e.g. interindustry or proprietary
INS	1	Instruction code - indicates the specific command, e.g. “write data”
P1-P2	2	Instruction parameters for the command, e.g. offset into file at which to write the data
Lc	0, 1 or 3	Encodes the number (Nc) of bytes of command data to follow
Command data	Nc	Nc bytes of data
Le	0, 1 or 2	Encodes the maximum number (Ne) of response bytes expected
Response APDU		
Field name	**Lenth(bytes)**	**Description**
Response Data	X	Payload of the response.
SW1-SW2	2	Command processing status. For instance 0x9000 indicates success

3. Overview of ICCs and TWN4 Family

This chapter helps the user to understand different TWN4 products and different ICC types. It also helps the user to choose a suitable TWN4, depending on the ICC they want to use.

3.1. TWN4 Family

3.1.1. TWN4 SmartCard

The TWN4 SmartCard Reader includes both a contactless RFID interface as well as a slot to read contact-based cards. This exciting combination allows its users to add an additional higher level of security where required for certain applications dealing with extremely sensitive information.

It is worthy of being mentioned that the contactless RFID interface and the contact slot have the same logical slot. Contact-based ICC has priority against the contactless ICC. When a contact-based ICC is inserted, TWN4 SmartCard will close its RFID interface.

3.1.2. TWN4 Desktop

TWN4 Desktop has the same performance like a TWN4 SmartCard, when it needs to exchange data with a transponder. It has no contact slots, where the user can insert its own contact-based cards.

3.2. Overview of ICCs

Depending on the physical connection of an ICC, an ICC can be contact-based or contactless.

3.2.1. Contact-Based ICCs

Contact-based ICCs have a contact area, which provides electrical connection between ICC and TWN4.

3.2.1.1. Microprocessor Cards

TWN4 SmartCard supports the most contact-based microprocessor ICCs. Please contact the respective ICC manufacturer concerning ATRs and/or APDUs.

3.2.1.2. Storage Cards

TWN4 SmartCard supports storage cards without built-in microprocessor. When such card is inserted into the reader, TWN4 emulates a PC/SC compliant microprocessor environment in order to give access to the respective card via APDUs.

3.2.2. Contactless ICCs

3.2.2.1. Microprocessor Cards

TWN4 offers native support to most contactless microprocessor cards. In such case, the cards can be directly accessed by issuing the respective APDUs, please contact the respective ICC manufacturer for details regarding APDUs.

3.2.2.2. Storage Cards

For contactless storage cards without built-in microprocessor, TWN4 emulates a PC/SC compliant microprocessor environment in order to support unified access to them via APDUs.

3.2.3. List of supported Contactless ICCs

Transponder Type	Microprocessor Card	Storage Card	Encoding supported	TWN4 Mifare supported	TWN4 Legic supported
SmartMX	✓		✓	✓	✓
Calypso	✓		✓	✓	✓
MIFARE DESFire EV1	✓		✓	✓	✓
MIFARE Plus SL0/SL3	✓		✓	✓	✓
MIFARE Plus SE SL0/SL3	✓		✓	✓	✓
MIFARE Classic 1K		✓	✓	✓	✓
MIFARE Classic 4K		✓	✓	✓	✓
MIFARE Ultralight		✓	✓	✓	✓
MIFARE Ultralight C		✓	✓	✓	✓
MIFARE Ultralight EV1		✓	✓	✓	✓
MIFARE Plus SL1/SL2		✓	✓	✓	✓
MIFARE Plus SE SL1		✓	✓	✓	✓
Legic Prime		✓	✓		✓
Legic Advant		✓	✓		✓
Topaz		✓	✓	✓	
ICode SLI		✓	✓	✓	✓
SRF55V02P/10P		✓	✓	✓	✓
SRF55V02S/10S		✓	✓	✓	✓
TagIT		✓	✓	✓	
HID ICLASS		✓		✓	
EM4102		✓		✓	✓
EM4150		✓		✓	✓
Hitag1		✓		✓	✓
Hitag2		✓		✓	✓
HitagS		✓		✓	✓
HID Prox		✓		✓	✓
ioProx		✓		✓	✓

3.3. How to Select a Suitable TWN4

This section lists different ICCs which are supported by TWN4 SmartCard or by TWN4 Desktop.

For detailed information concerning supported APDUs please refer to section 6.2.9.

	Contact-based ICCs	Micro Processor Cards	Storage Cards
TWN4 SmartCard	✓	✓	✓
TWN4 Desktop		✓	✓

4. Contact-Based Card

In general, TWN4 works with ISO 7816-3 and ISO 7816-4 conformed cards.

An Answer To Reset (ATR) is a message output by a contact card conforming to ISO/IEC 7816 standards, following electrical reset of the card's chip by a card reader. The ATR conveys information about the communication parameters proposed by the card, and the card's nature and state.

ATRs cannot be used for user identification.

The APDUs used by contact-based card are defined in ISO 7816-4. Manufacturers may also define proprietary APDUs. Please refer to the relative documents and/or the manufacturers for more information.

5. Microprocessor Cards

The information about the ATR of a microprocessor card can be found in PC/SC specification Part 3 Requirements for PC-Connected Interface Devices [4]. Please refer to the document for more information.

ATRs cannot be used for user identification.

In general, TWN4 SmartCard and TWN4 Desktop support ISO 14443-4 conformed cards. The following ICCs have been tested by ELATEC:

- MIFARE® DESFire®[1] EV1
- Calypso® [2]
- SmartMX™[3]

Users may also use APDUs defined in PC/SC specification Part 3 [4], PC/SC specification Part 3 Supplemental Document 2 - Contactless ICCs [5] and Amendment 1: Requirements for PC-Connected Interface Devices [6] to get information of a micro processor card. Please refer to the corresponding documents for more information.

Manufacturers may also have defined proprietary APDUs. Please refer to the relative documents and/or the manufacturers for more information.

In general, one can use Get Data command to retrieve the UID of a microprocessor card.

Command	CLA	INS	P1	P2	Lc	Le
Get Data	FF	CA	00	00	-	00

Response data	SW1-SW2
UID of the transponder	90 00

[1] MIFARE and MIFARE DESFire are registered trademarks of NXP B.V. and are used under license.
[2] Calypso is a registered trademark of Calypso Technology.
[3] SmartMX is a trademark of NXP Semiconductors N.V.

6. Storage Cards

6.1. Answer To Reset

This chapter lists the ATRs of storage cards. If not explicitly noted, all values are in hex format.
The information about ATR of a storage card can be found in PC/SC specification Part 3 Supplemental Document 1 - Storage Card ATR [7]. Please refer to the document for more information.

6.1.1. HF Transponders

Transponder Type	ATR
MIFARE Classic 1K	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 01 00 00 00 00 6A
MIFARE Classic 1K	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 02 00 00 00 00 69
MIFARE Ultralight	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 03 00 00 00 00 68
MIFARE Ultralight C	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 3A 00 00 00 00 51
MIFARE Ultralight EV1	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 3D 00 00 00 00 56
MIFARE Plus 2K SL1	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 36 00 00 00 00 5D
MIFARE Plus 4K SL1	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 37 00 00 00 00 5C
MIFARE Plus 2K SL2	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 38 00 00 00 00 53
MIFARE Plus 4K SL2	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 39 00 00 00 00 52
Legic Prime/Advant	3B 88 80 01 45 53 43 4F 53 31 30 30 71
Topaz	3B 8F 80 01 80 4F 0C A0 00 00 03 06 03 00 30 00 00 00 00 5B
ICode SLI	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 14 00 00 00 00 77
TagIT	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 12 00 00 00 00 71
SRF55V10P	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 0E 00 00 00 00 6D
SRF55V02P	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 0F 00 00 00 00 6C
SRF55V10S	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 10 00 00 00 00 73
SRF55V02S	3B 8F 80 01 80 4F 0C A0 00 00 03 06 0B 00 11 00 00 00 00 72

6.1.2. LF Transponders

Transponder Type	ATR
EM4102/4150	3B 8F 80 01 80 4F 0C A0 00 00 03 06 40 00 00 00 00 00 00 28
Hitag 1, 2, S	3B 8F 80 01 80 4F 0C A0 00 00 03 06 40 00 00 00 00 00 00 28
ioProx	3B 86 80 01 69 6F 50 72 6F 78 34
HID Prox	Variable as the ID is transmitted with the ATR

6.1.3. Contact Based Memory Cards

Transponder Type	ATR
SLE44XX	3B 84 80 01 A0 13 10 91 37
I2C Memory Cards	3B 04 49 32 43 2E

6.2. Contactless Storage Cards

6.2.1. List of APDUs

The APDUs in this section are defined in Interoperability Specification for ICCs and Personal Computer Systems Part 3 [4].

Command	CLA	INS	P1	P2	Lc	Data	Le
Get Data	FF	CA	00	00	-	-	00
Load Key	FF	82	00	KeyNumber	KeyLength	Key	-
General Authenticate	FF	86	00	00	05	Parameters	-
Read Binary	FF	B0	AddressMSB	AddressLSB	-	Data	XX
Update Binary	FF	D6	AddressMSB	AddressLSB	XX	Data	-
WritePerso	80	A8	AddressMSB	AddressLSB	10	Data	-
CommitPerso	80	AA	00	00	-	-	00

6.2.2. Get Data

The Get Data command can be used to retrieve the UID of a transponder.

Command	CLA	INS	P1	P2	Lc	Le
Get Data	FF	CA	00	00	-	00

Response: UID of the transponder followed by SW1-SW2.

SW1-SW2	Description
90 00	Success
67 00	Lc shall not be present
6B 00	Unsupported P1-P2
6C 00	Wrong Le, Le shall be 0x00

6.2.3. Load Key

The reader is able to store up to 16 keys in its volatile memory, that can be used for authentication purposes. Each key can be 16 bytes in length. Allowed authentication methods are:

- MIFARE Classic Crypto1
- MIFARE Ultralight C 3DES
- MIFARE Plus AES

Command	CLA	INS	P1	P2	Lc	Data	Le
Load Key	FF	82	00	Key number	Key Length	Key	-

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect or absent
69 83	Reader key not supported
69 85	Secure transmission not supported
69 88	Key number not valid
69 89	Key length not valid

Example:
Load key FF FF FF FF FF FF to key number 1:

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

6.2.4. General Authenticate

Command	CLA	INS	P1	P2	Lc	Data	Le
General Authenticate	FF	86	00	00	5	Parameters	-

Parameters:

Byte1	Byte2	Byte3	Byte4	Byte5
Version 01	Address MSB	Address LSB	Key type	Key Nr.

Key type:

- 0x00: General purpose AES/3DES key, the block address is interpreted as an absolute key reference. Use this key type for AES authentication different form normal sector authentication, e.g. for additional MIFARE Plus SL1 authentication or for switching the security levels.
- 0x60: Crypto1 key type A
- 0x61: Crypto1 key type B
- 0x62: AES sector key type A
- 0x63: AES sector key type B

Response: SW1-SW2 will be returned.

Example 1:
Authenticate at MIFARE Classic block 02 with key number 1, key type A.

Command	FF 86 00 00 05 01 00 02 60 01
Response	90 00

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect
6B 00	Unsupported P1-P2
69 83	Authentication failed
69 86	Unsupported key type
69 88	Invalid key number

Example 2:
Perform MIFARE Plus AES authentication in SL1 with key number 2, use general purpose key type 0x00.

Command FF 86 00 00 05 01 90 04 00 02

Response 90 00

6.2.5. Read Binary

Use this command to read data from a contactbased or contactless storage card. As the memory of most contactless transponders is organized page- or blockwise, hence this block size limits the amount of data, that can be read at one time.

Command	CLA	INS	P1	P2	Lc	Data	Le
Read Binary	FF	B0	AddressMSB	AddressLSB	-	-	XX

Parameters:

- AddressMSB: Block or byte address, high byte
- AddressLSB: Block or byte address, low byte
- Le: Expected length. If 0x00 is specified the whole block is read

Response: Data followed by SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect, shall be absent
6C XX	Wrong expected length, XX denotes the maximum allowed length
69 81	Error during reading
69 82	Authentication status not satisfied

6.2.6. Update Binary

Use this command to write data to a contactbased or contactless storage card. The Update Binary command always writes entire blocks/pages to a contactless storage card, this means i.e. in case of Mifare

Classic 16 bytes of payload data is required.

Command	CLA	INS	P1	P2	Lc	Data	Le
Update Binary	FF	D6	AddressMSB	AddressLSB	XX	Data	-

Parameters:

- AddressMSB: Block or byte address, high byte
- AddressLSB: Block or byte address, low byte
- Lc: Length of data to be written
- Data: Data to be written

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect or absent
6B 00	Unsupported P1-P2
6C XX	Wrong length, XX denotes the expected length
65 81	Error during writing
69 82	Authentication status not satisfied
69 86	Card type not supported

Example: Write data block number 4

Command FF D6 00 04 10 00 11 22 33 44 55 66 77 88 99 AA BB CC DD EE FF

6.2.7. Value Functions

Use this command to perform a increment or decrement operation on a MIFARE Classic or MIFARE Plus transponder. The functionality is implemented according to PC/SC specification Part 3 [4], PC/SC specification Part 3 Supplemental Document 2 - Contactless ICCs [5] and Amendment 1: Requirements for PC-Connected Interface Devices [6], so please refer to this document for additional information.

Command	CLA	INS	P1	P2	Lc	Data	Le
Envelope	FF	C2	00	03	XX	BER-TLV	-

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect or absent
6B 00	Unsupported P1-P2
65 81	Inc/Dec unsuccessful
69 81	Incompatible Lc
69 86	Card type not supported

6.2.8. MIFARE Plus

6.2.8.1. Write Personalisation Data

Use this command to write personalisation data to a MIFARE Plus transponder running in SL0.

Command	CLA	INS	P1	P2	Lc	Data	Le
WritePerso	80	A8	AddressMSB	AddressLSB	10	Data	-

Parameters:

- AddressMSB: Block address, high byte
- AddressLSB: Block address, low byte
- Lc: Length of data to be written, must be always 16 bytes
- Data: Data to be written

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect or absent
6C 10	Wrong length, Lc shall be 0x10
65 81	Write personalisation failed
69 86	Card type not supported

6.2.8.2. Commit Personalisation

Use this command to commit personalisation of a MIFARE Plus transponder and switch from SL0 to SL1.

Command	CLA	INS	P1	P2	Lc	Data	Le
CommitPerso	80	AA	00	00	-	-	00

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect, shall be absent
6B 00	Unsupported P1-P2
65 81	Commit personalisation failed
69 86	Card type not supported

6.2.9. Supported APDUs by Different Transponders

	Get Data	Load Key	General Authenticate	Read Binary	Update Binary	Envelope	Write Perso	Commit Perso
MIFARE Classic	✓	✓	✓	✓	✓	✓		
MIFARE Ultralight C	✓	✓	✓	✓	✓			
MIFARE Plus	✓	✓	✓	✓	✓	✓	✓	✓
MIFARE Ultralight	✓			✓	✓			
MIFARE Ultralight EV1	✓			✓	✓			
Topaz	✓			✓	✓			
ICode SLI	✓			✓	✓			
TagIT	✓			✓	✓			
SRF55V02P	✓			✓	✓			
SRF55V10P	✓			✓	✓			
SRF55V02S	✓			✓	✓			
SRF55V10S	✓			✓	✓			
EM4102	✓							
EM4150	✓							
Hitag 1	✓							
Hitag 2	✓							
Hitag S	✓							
ioProx	✓							
HID Prox	✓							

6.3. Contactbased Storage Cards

6.3.1. List of APDUs

Command	CLA	INS	P1	P2	Lc	Data	Le
Read Binary	FF	B0	AddressMSB	AddressLSB	-	-	XX
Update Binary	FF	D6	AddressMSB	AddressLSB	XX	Data	-
Verify	FF	20	00	00	03	Pin	00
ChangeCHV	FF	24	00	00	03	Pin	00
ReadRecord	FF	B2	00	00	-	-	00
WriteRecord	FF	D2	00	00	04	Data	00

6.3.2. Read Binary

Use this command to read data from a contactbased storage card.

Command	CLA	INS	P1	P2	Lc	Data	Le
Read Binary	FF	B0	AddressMSB	AddressLSB	-	-	XX

Parameters:

- AddressMSB: Byte address, high byte
- AddressLSB: Byte address, low byte
- Le: Expected length.

Response: Data followed by SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect, shall be absent
68 00	CLA not supported
6B 00	Unsupported P1-P2
6F 00	Read Binary failed
62 82	Wrong expected length

6.3.3. Update Binary

Use this command to write data to a contactbased storage card.

Command	CLA	INS	P1	P2	Lc	Data	Le
Update Binary	FF	D6	AddressMSB	AddressLSB	XX	Data	-

Parameters:

- AddressMSB: Byte address, high byte
- AddressLSB: Byte address, low byte
- Lc: Length of data to be written
- Data: Data to be written

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect or absent
6B 00	Unsupported P1-P2
6F 00	Update Binary failed
62 82	Write operation exceeds memory boundaries

6.3.4. Verify PIN

Use this command to present a PIN code to a contactbased SLE memory card.

Command	CLA	INS	P1	P2	Lc	Data	Le
Verify	FF	20	00	00	03	PIN	00

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc absent, shall be present
6C 03	Lc incorrect, shall be 0x03
6F 00	Verify failed
63 CX	Wrong PIN presented, X denotes the number of remaining retries

6.3.5. Change PIN

Use this command to change the PIN of a contactbased SLE memory card.

Command	CLA	INS	P1	P2	Lc	Data	Le
ChangeCHV	FF	24	00	00	03	PIN	00

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc absent, shall be present
6C 03	Lc incorrect, shall be 0x03
6F 00	Change PIN failed

6.3.6. Read Protection Memory

Use this command to read the protection memory of a contactbased SLE memory card.

Command	CLA	INS	P1	P2	Lc	Data	Le
Read Record	FF	B2	00	00	-	-	00

Response: Protection Memory Data followed by SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc present, shall be absent
6B 00	Unsupported P1-P2
6F 00	Read Protection Memory failed

6.3.7. Write Protection Memory

Use this command to write the protection memory of a contactbased SLE memory card.

Command	CLA	INS	P1	P2	Lc	Data	Le
Write Record	FF	D2	00	00	04	Protection Memory	00

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc absent, shall be present
6B 00	Unsupported P1-P2
6C 04	Lc incorrect, shall be 0x04
6F 00	Write Protection Memory failed

7. Legic Cards

When a Legic transponder is presented, the IFD emulates an ISO7816-4 compatible MF/DF/EF file structure. This means every accessible data segment located on the transponder is mapped as a DF (Directory File) under the MF (Master File), the respective DF ID is a 16 bit number starting from 0001. The respective segment stamp becomes the DF name, so each segment can be accessed either by its DF ID or the stamp. As a DF cannot contain any payload data, each DF has an EF (Elementary File), containing the segment data.
In order to grant access to the segment data, TWN4 provides the necessary ISO7816-4 interindustry commands.

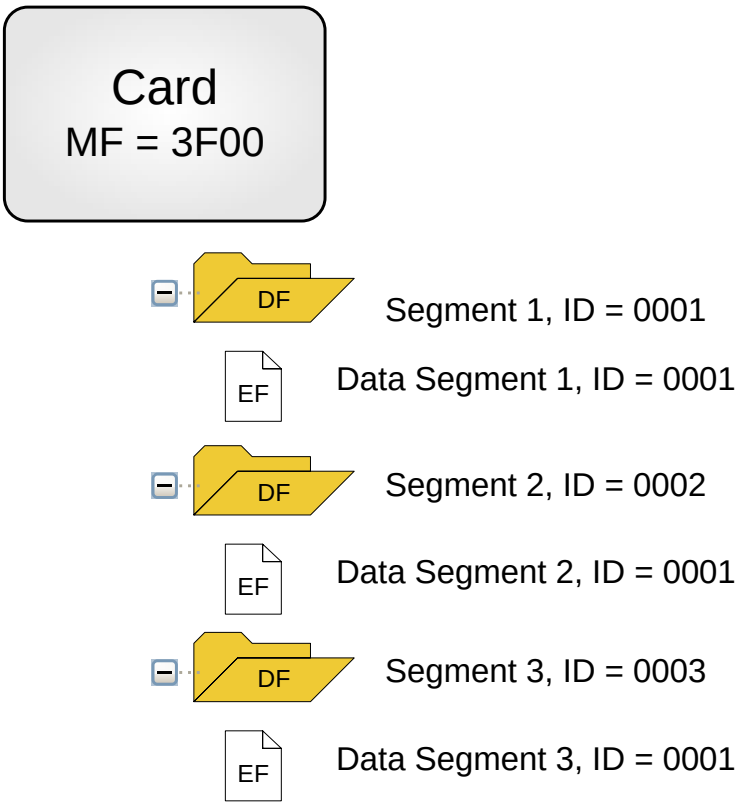


Figure 7.1.: MF/DF/EF Structure

Example:
Data Segment 3 (Stamp 2700010203) can either be accessed by specifying the file IDs 3F00/0003/0001 or by specifying the DF name 3F00/2700010203/0001.

7.1. Command Set

7.1.1. Get Data

Use this APDU to retrieve either information about the emulated card operating system or the card UID.

Command	CLA	INS	P1	P2	Lc	Le
Get Data OS	00	CA	01	82/83	-	00
Get Data UID	FF	CA	00	00	-	00

P2 Value	Meaning
82	Query OS version
83	Query current lifecycle phase of current DF (0x10 is always returned, means "operational")

Response: Data followed by SW1-SW2

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect, shall be absent
6C XX	Wrong length, XX denotes the expected length
6A 86	Unsupported P1
6A 81	Unsupported P2

7.1.2. List Directory

Use this command to list the content of the currently selected MF or DF.

Command	CLA	INS	P1	P2	Lc	Le
List Directory	80	16	00	00	-	00

Response: TLV coded FCI followed by SW1-SW2

SW1-SW2	Description
90 00	Success
6E 00	CLA not supported
6A 81	Unsupported P1

7.1.3. Select File

Use this command to select a MF, DF or EF for subsequent operations.

Command	CLA	INS	P1	P2	Lc	Data	Le
Select File	00	A4	Selection Control	Selection Control	Empty or Length of data field		00

P1 Value	Meaning
00	Select DF or EF directly below the current DF using the file ID. If no file ID is transmitted, the MF 3F00 is selected.
04	Select a DF using its name.
08	Select DF or EF using the path starting from the MF.

P2 Value	Meaning
00	Send FCI data
0C	Only return SW1-SW2

Response: Data followed by SW1-SW2

SW1-SW2	Description
90 00	Success
67 00	Lc absent, shall be present
6E 00	CLA not supported
6A 81	Unsupported P1
6A 82	File not found
6A 86	Unsupported P2
6A 87	Lc inconsistent with P1-P2

7.1.4. Read Binary

Use this command to read binary data from a EF. The file must have been selected prior starting this operation. Please note that reading is limited to a maximum byte count of 200 bytes, if a larger amount is required, reading shall be done by issuing a sequence of Read Binary APDUs.

Command	CLA	INS	P1	P2	Lc	Data	Le
Read Binary	00	B0	AddressMSB	AddressLSB	-	-	XX

Response: Data followed by SW1-SW2

SW1-SW2	Description
90 00	Success
67 00	Lc present, shall be absent
6B 00	Unsupported P1-P2
6C XX	Wrong length, XX denotes the expected length
6E 00	CLA not supported
6F 00	Read Binary failed
6A 81	Short EF ID not supported
6A 82	No DF/EF selected

7.1.5. Update Binary

Use this APDU to write binary data to a EF. The file must have been selected prior starting this operation. Please note that writing is limited to a maximum byte count of 200 bytes, if a larger amount is required, writing shall be done by issuing a sequence of Update Binary APDUs.

Command	CLA	INS	P1	P2	Lc	Data	Le
Update Binary	FF	D6	AddressMSB	AddressLSB	Length of data field	Data	-

Response: SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect or absent
6B 00	Unsupported P1-P2
6E 00	CLA not supported
6F 00	Update Binary failed
6A 81	Short EF ID not supported
6A 82	No DF/EF selected

8. Virtual Slot

Depending on installed firmware, TWN4 offers an additional Virtual Slot. Within this slot, the host system will find a emulated virtual smartcard which is permanently present. This smartcard can be accessed using APDUs. This makes it possible to take additional control on the behavior of the reader on APDU level.

8.1. Answer To Reset

The virtual smartcard can be identified by its ATR:
3B 86 81 11 FE 81 43 69 6E 64 79 30

8.2. Simple Protocol

The virtual smartcard supports calling functions of the Simple Protocol. The respective call is mapped into the payload of a specified APDU. For details regarding Simple Protocol, please refer to the respective documentation.

Command	CLA	INS	P1	P2	Lc	Data	Le
Simple Protocol	FF	9B	00	00	Length	Simple Protocol Message	-

Response: Simple Protocol Response followed by SW1-SW2 will be returned.

SW1-SW2	Description
90 00	Success
67 00	Lc incorrect or absent
6E 00	Invalid CLA
6F 00	Error during execution of function call
69 00	Unsupported P1-P2

9. Test PC/SC with TWN4 under Linux

9.1. Prepare the Hardware

In order to test PCSC under Linux with TWN4, it is necessary to program a correct Firmware/App to TWN4. Please refer to ELATEC support for the latest Firmware/App image.

9.2. Prepare the Software

A complete PC/SC software solution contains an ICC-aware software(e.g. opensc), an ICC resource manager(e.g. pcscd) and an IFD handler(e.g. libccid). All these components can be compiled from source code. The source code can be found in Internet [13, 14, 15, 16].

If the OS has a package manager available, these components can also be installed with the package manager.

9.2.1. ICC Resource Manager

The ICC Resource Manager under Linux is called PCSC-Lite. The project can be found under:

https://pcsclite.alioth.debian.org/pcsclite.html.

Under Debian, it can also be installed with

```
$sudo apt-get install pcscd
```

9.2.2. IFD Handler

The IFD Handler under Linux is called CCID Driver. The project can be found under:

https://pcsclite.alioth.debian.org/ccid.html.

Under Debian, the package will be installed automatically by the system, when the user installs pcscd. It can be also explicitly installed by

```
$sudo apt-get install libccid
```

In order to make pcscd work under Linux, it is necessary to know the Vendor ID (VID) and Product ID (PID) of the reader.

To find the VID/PID of the connected TWN4, one can type

```
$lsusb
```

The output of lsusb is like:

```
Bus 001 Device 003: ID 09d8:0426
```

Depending on the USB stack running on TWN4, the Product ID (PID) is different. All TWN4 readers have Vendor ID (VID) 0x09d8.

USB Stack	Vendor ID	Product ID
CCID + HID	0x09D8	0x0425
CCID + CDC	0x09D8	0x0427
CCID	0x09D8	0x0428

By adding the VID, the PID and the name of the reader to Info.plist, one can make TWN4 work under Linux. The Info.plist file is located under for instance /usr/lib/pcsc/drivers/ifd-ccid.bundle/Contents/

9.2.3. OpenSC

The simplest implementation of the ICC-aware Software is opensc

(https://github.com/OpenSC/OpenSC).

After the resource manager and the IFD handler are ready, one can call opensc to start the normal operation. Note that slot 0 is reserved for either RFID transponder or contact-based ICC, reader 0 is selected in the following examples.

9.2.3.1. List the Readers

```
$opensc-tool -l
```

9.2.3.2. Read the ATR of an ICC

```
$opensc-tool -r 0 -a
```

9.2.3.3. Read the UID of a Card

```
$opensc-tool -r 0 -s FF:CA:00:00:00
```

9.2.3.4. Send APDU to the ICC

```
$opensc-tool -r 0 -s <APDU> # the bytes should be seperated with ':'
```

9.2.4. Customized ICC-aware Software

It is also possible to write a customized program. More information regarding the sample code can be found under:

http://ludovicrousseau.blogspot.de/2010/04/pcsc-sample-in-c.html.

9.3. Known Issue with libccid

9.3.1. Communication with TWN4

Depending on the Linux distribution, it is possible that libccid is unable to recognize TWN4[1].

[1] A patch for the bug has been sent to the author of libccid. The bug will be therefore fixed in libccid version 1.4.28.

The cause for the bug is, libccid sends often GetSlotStatus command to TWN4 and requires an response from TWN4 within 100 milliseconds. On the other hand, TWN4 needs around 300 milliseconds to search a transponder. Since TWN4 fails to respond the request on-time, libccid believes TWN4 is defect and therefore refuses to work further.

The solution to the problem is to make the timeout to 1 second. This can be don by applying the sample patch[2] in appendix A. Here the reader can find further information.

9.3.2. Communication with NXP SAM AV2 Card

When an NXP SAM AV2 card is connected, ccid mistakenly sets up a wrong parameter of the SAM AV2 card. As a result TWN4 is unable to communicate with this card.

A solution for the problem is to delete the following line in ifdhandler.c:

```
extra_egt(&atr, ccid_desc, Protocol);
```

[2] Depending on the version of libccid, the lines where the changes must be applied can be different.

10. Disclaimer

ELATEC reserves the right to change any information or data in this document without prior notice. The distribution and the update of this document is not controlled. ELATEC declines all responsibility for the use of product with any other specifications but the ones mentioned above. Any additional requirement for a specific custom application has to be validated by the customer himself at his own responsibility. Where application information is given, it is only advisory and does not form part of the specification.

All referenced brands, product names, service names and trademarks mentioned in this document are the property of their respective owners.

Appendices

A. Sample Patch for libccid

```
commit b19c87ec71689c4aed606c4c91d83e2e8c0e99f1
Author: Ming Lu <m.lu@elatec.com>
Date:   Wed Jul 5 09:38:09 2017 +0200

The reader answers after up to 1 second when no card is present in the
field. So a 100ms timeoutx was too short.

diff --git a/src/Info.plist b/src/Info.plist
index 2a43dc5..aed0c10 100644
--- a/src/Info.plist
+++ b/src/Info.plist
@@ -251,6 +251,7 @@
                <string>0x1DB2</string>
                <string>0x257B</string>
                <string>0x09D8</string>
+               <string>0x09D8</string>
                <string>0x2CE4</string>
                <string>0x096E</string>
                <string>0x096E</string>
@@ -644,6 +645,7 @@
                <string>0x088B</string>
                <string>0xD205</string>
                <string>0x0427</string>
+               <string>0x0428</string>
                <string>0x7479</string>
                <string>0x0619</string>
                <string>0x061A</string>
@@ -1037,6 +1039,7 @@
                <string>DUALi DRAGON NFC READER</string>
                <string>eID_R6 001 X8</string>
                <string>ELATEC TWN4 SmartCard NFC</string>
+               <string>ELATEC TWN4 SmartCard NFC</string>
                <string>ESMART Token GOST</string>
                <string>FEITIAN iR301</string>
                <string>FEITIAN bR301</string>
diff --git a/src/ccid.c b/src/ccid.c
index 3dc0456..577fa27 100644
--- a/src/ccid.c
+++ b/src/ccid.c
@@ -87,10 +87,13 @@ int ccid_open_hack_pre(unsigned int reader_index)
                        ccid_descriptor->dwMaxDataRate = 9600;
                        break;

-               case ElatecTWN4:
-                       /* use a timeout of 400 ms instead of 100 ms in CmdGetSlotStatus()
+               case ElatecTWN4_CCIDCDC:
+               case ElatecTWN4_CCID:
+                       /* use a timeout of 1000 ms instead of 100 ms in CmdGetSlotStatus()
                         * used by CreateChannelByNameOrChannel()
                         * The reader answers after 280 ms if no tag is present */
```

```
+                       ccid_descriptor->readTimeout = DEFAULT_COM_READ_TIMEOUT * 10;
+                       break;
                case SCM_SCL011:
                        /* The SCM SCL011 reader needs 350 ms to answer */
                        ccid_descriptor->readTimeout = DEFAULT_COM_READ_TIMEOUT * 4;
@@ -543,7 +546,8 @@ int ccid_open_hack_post(unsigned int reader_index)
                        }
                        break;

-               case ElatecTWN4:
+               case ElatecTWN4_CCID:
+               case ElatecTWN4_CCIDCDC:
                case SCM_SCL011:
                        /* restore default timeout (modified in ccid_open_hack_pre()) */
                        ccid_descriptor->readTimeout = DEFAULT_COM_READ_TIMEOUT;
diff --git a/src/ccid.h b/src/ccid.h
index b219884..e09c1da 100644
--- a/src/ccid.h
+++ b/src/ccid.h
@@ -218,7 +218,8 @@ typedef struct
 #define FEITIANR502DUAL 0x096E060D
 #define MICROCHIP_SEC1100 0x04241104
 #define CHERRY_KC1000SC 0x046A00A1
-#define ElatecTWN4     0x09D80427
+#define ElatecTWN4_CCID        0x09D80428
+#define ElatecTWN4_CCIDCDC     0x09D80427
 #define SCM_SCL011 0x04E65293
 #define HID_AVIATOR    0x076B3A21
 #define HID_OMNIKEY_5422 0x076B5422
diff --git a/src/ifdhandler.c b/src/ifdhandler.c
index 665681e..7a2c4b5 100644
--- a/src/ifdhandler.c
+++ b/src/ifdhandler.c
@@ -717,9 +717,6 @@ EXTERNAL RESPONSECODE IFDHSetProtocolParameters(DWORD Lun, D
        if (ATR_MALFORMED == atr_ret)
                return IFD_PROTOCOL_NOT_SUPPORTED;

-       /* Apply Extra EGT patch for bogus cards */
-       extra_egt(&atr, ccid_desc, Protocol);
-
        if (SCARD_PROTOCOL_T0 == Protocol)
                pps[1] |= ATR_PROTOCOL_TYPE_T0;
        else
```

Bibliography

[1] Cards with Contacts. Part 3: Electrical interface and transmission protocols. http://www.cardwerk.com/smartcards/smartcard_standard_ISO7816-3.aspx, 2002.

[2] Cards with Contacts. Part 4: Interindustry Commands for Interchange. http://www.cardwerk.com/smartcards/smartcard_standard_ISO7816-4.aspx, 2005.

[3] Specification for Integrated Circuit(s) Cards Interface Devices. Universal Serial Bus Device Class: Smart Card CCID. https://www.usb.org/sites/default/files/DWG_Smart-Card_CCID_Rev110.pdf, April 2005.

[4] PC/SC Work Group. Interoperability Specification for ICCs and Personal Computer Systems, Part 3 Requirements for PC-Connected Interface Devices. http://www.pcscworkgroup.com/specifications/files/pcsc3_v2.01.09.pdf, June 2007.

[5] PC/SC Work Group. Interoperability Specification for ICCs and Personal Computer Systems, Part 3. Supplemental Document 2 - Contactless ICCs. http://www.pcscworkgroup.com/specifications/files/pcsc3_v2.02.00_sup2.pdf, April 2010.

[6] PC/SC Work Group. Interoperability Specification for ICCs and Personal Computer Systems, Part 3. Amendment 1: Requirements for PC-Connected Interface Devices. http://pcscworkgroup.com/Download/Specifications/pcsc3_v2.01.09_amd1.pdf, August 2011.

[7] PC/SC Work Group. Interoperability Specification for ICCs and Personal Computer Systems, Part 3. Supplemental Document 1 - Storage Card ATR. http://www.pcscworkgroup.com/specifications/files/pcsc3_v2.01.09_sup.pdf, June 2013.

[8] Identification Cards – Contactless Integrated Circuit Cards – Proximity Cards. Part 1: Physical Characteristics, 2008. http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=39693.

[9] Identification Cards – Contactless Integrated Circuit Cards – Proximity Cards. Part 4: Transmission protocol. http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=50648, 2008.

[10] Identification Cards – Contactless Integrated Circuit Cards – Proximity Cards. Part 2: Radio Frequency Power and Signal Interface. http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=50941, 2010.

[11] Identification Cards – Contactless Integrated Circuit Cards – Proximity Cards. Part 3: Initialization and Anticollision. http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=50942, 2011.

[12] Jakuje. OpenSC Git Repository. https://github.com/OpenSC/OpenSC.

[13] Ludovic Rousseau. PCSC sample in C. http://ludovicrousseau.blogspot.de/2010/04/pcsc-sample-in-c.html, 2010.

[14] Ludovic Rousseau. CCID free software driver. https://pcsclite.alioth.debian.org/ccid.html, 2015.

[15] Ludovic Rousseau. Movement for the Use of Smart Cards in a Linux Environment. https://pcsclite.alioth.debian.org/, 2015.

[16] Ludovic Rousseau. PCSC lite project(Middleware to access a smart card using SCard API (PC/SC).). https://pcsclite.alioth.debian.org/pcsclite.html, 2015.