3GPP TSG-T (Terminals) Meeting #5
Kyongju, Korea, October 7 - 8, 1999

Document *TSGT#5(99)185*

3G TS 31.101 V1.0.0 (1999-10)

Technical Specification

**3rd Generation Partnership Project;
Technical Specification Group Terminals;
UICC-Terminal Interface; Physical and Logical Characteristics
(3G TS 31.101 version 1.0.0)**

Reference

DTS/TSGT-0331101U

Keywords

USIM,UICC

_

Contents

Foreword

This Technical Specification has been produced by the 3GPP.

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of this TS, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version 3.y.z

where:

x the first digit:

1 presented to TSG for information;

2 presented to TSG for approval;

3 Indicates TSG approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the specification;

Introduction

This specification defines a generic terminal/Integrated Circuit Card (ICC) interface. This specification is independent of the 3GPP USIM application and can thus be the platform for any IC card application.

The aim of this document is to ensure interoperability between an ICC and a terminal independently of the respective manufacturer, card issuer or operator. This specification does not define any aspects related to the administrative management phase of the ICC. Any internal technical realisation of either the ICC or the terminal is only specified where these are reflected over the interface.

Application specific details for applications residing on an ICC are specified in the respective application specific documents. The Universal Subscriber Identity Module (USIM)-application for 3GPP telecommunication networks is specified in document 3GPP 31.102.

1 Scope

The present document specifies the interface between the UICC and the Terminal for 3GPP telecom network operation.

This document specifies:

- the requirements for the physical characteristics of the UICC;
- the electrical interface between the UICC and the Terminal;
- the initial communication establishment and the transport protocols;
- the model which serves as a basis for the logical structure of the UICC;
- the communication commands and the procedures;
- the application independent files and protocols.

The administrative procedures and initial card management are not part of this document.

2 Normative References

The present document incorporates by dated and non-dated reference, provisions from other publications. This normative reference is cited at the appropriate places in the text and the publications are listed hereafter. For dated references, subsequent amendments to or revisions of any of these publications apply to the present document only when incorporated in it by amendment or revision. For non-dated references the latest edition of the publication referred to applies.

[1] ISO 7816-1 (1998): "Identification cards - Integrated circuit(s) cards with contacts, Part 1: Physical characteristics".

[2] ISO 7816-2 (1988): "Identification cards - Integrated circuit(s) cards with contacts, Part 2: Dimensions and locations of the contacts".

[3] ISO/IEC 7816-3 (1997): "Identification cards - Integrated circuit(s) cards with contacts, Part 3: Electronic signals and transmission protocols".

[4] ISO/IEC 7816-4 (1995): "Identification cards - Integrated circuit(s) cards with contacts, Part 4: Interindustry commands for interchange ".

[5] ISO/IEC 7816-5 (1994): "Identification cards - Integrated circuit(s) cards with contacts, Part 5: Numbering system and registration procedure for application identifiers ".

[6] ISO/IEC 7816-6 (1996): "Identification cards - Integrated circuit(s) cards with contacts, Part 6: Interindustry data elements".

[7] ISO/IEC FCD 7816-8 (1997): "Identification cards - Integrated circuit(s) cards with contacts, Part 8: Security related Interindustry commands".

[8] ISO/IEC FCD 7816-9 (1999): "Identification cards - Integrated circuit(s) cards with contacts, Part 9: Additional Interindustry commands and security attributes".

[9] ISO/IEC 7810 (1995): "Identification cards - Physical characteristics".

[10] ISO/IEC 7811-1 (1995): "Identification cards - Recording technique - Part 1: Embossing".

[11]	ISO/IEC 7811-3 (1995): "Identification cards - Recording technique - Part 3: Location of embossed characters on ID-1 cards".
[12]	GSM 03.38: "Digital cellular telecommunications system (Phase 2+); Alphabets and language-specific information".
[18]	CCITT Recommendation E.118: "The international telecommunication charge card".
[30]	ISO 639 (1988): "Code for the representation of names of languages".

3 Definitions, symbols and abbreviations

3.1 Definitions

For the purposes of the present document, the following definitions apply.

3V technology Smart Card: A Smart Card operating at 3V-± 10% and 5V ± 10%.

1.8V technology Smart Card: A Smart Card operating at 1.8V ± 10% and 3V ± 10%.

3V technology Terminal: A terminal operating the Smart Card - Terminal interface at 3V-± 10% and 5V ± 10%.

1.8V technology Terminal: A terminal operating the Smart Card - Terminal interface at 1.8V ± 10% and 3V ± 10%.

Function: A function contains a command and a response pair.

Command: TBD

Response: TBD

Application DF: An Application DF is the entry point to an application

For the purposes of the present document, the following definitions apply. For further information and definitions refer to GSM 01.02 [1].

Access conditions: A set of security attributes associated with a file.

Application: An application consists of a set of security mechanisms, files, data and protocols (excluding transmission protocols).

Application protocol: The set of procedures required by the application.

Card session: A link between the card and the external world starting with the ATR and ending with a subsequent reset or a deactivation of the card.

Current directory: The latest MF or DF selected.

Current EF: The latest EF selected.

Data field: Obsolete term for Elementary File.

Data Object: Information coded as TLV objects, i.e. consisting of a Tag, a Length and a Value part.

Dedicated File (DF): A file containing access conditions and, optionally, Elementary Files (EFs) or other Dedicated Files (DFs).

Directory: General term for MF and DF.

Elementary File (EF): A file containing access conditions and data and no other files.

file: A directory or an organized set of bytes or records in the SIM.

File identifier: The 2 bytes, which address a file in the SIM.

GSM, DCS 1800 or PCS 1900 application: Set of security mechanisms, files, data and protocols required by GSM, DCS 1800 or PCS 1900.

GSM session: That part of the card session dedicated to the GSM operation.

IC card SIM: Obsolete term for ID-1 SIM.

ID-1 SIM: The SIM having the format of an ID-1 card (see ISO 7816-1 [24]).

Master File (MF): The unique mandatory file containing access conditions and optionally DFs and/or EFs.

Normal GSM operation: Relating to general, CHV related, GSM security related and subscription related procedures.

Padding: One or more bits appended to a message in order to cause the message to contain the required number of bits or bytes.

Plug-in SIM: A Second format of SIM (specified in clause 4).

Proactive SIM: A SIM, which is capable of issuing commands to the Terminal. Part of SIM Application Toolkit (see clause 11).

Record: A string of bytes within an EF handled as a single entity (see clause 6).

Record number: The number, which identifies a record within an EF.

Record pointer: The pointer, which addresses one record in an EF.

Root directory: Obsolete term for Master File.

SIM application toolkit procedures: Defined in GSM 11.14 [27].

NOTE: Most of the definitions are “borrowed” from GSM 11.11 and have to be checked for relevance in 3GPP context.

3.2 Symbols

For the purposes of the present document, the following symbols apply.

t_F	fall time
t_R	rise time
V_{IH}	Input Voltage (high)
V_{IL}	Input Voltage (low)
V_{OH}	Output Voltage (high)
V_{OL}	Output Voltage (low)

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

For the purposes of the present document, the following abbreviations apply:

AC	Access Condition
ADF	Application Dedicated File
ADM	Access condition to an EF which is under the control of the authority which creates this file
AID	Application IDentifier
ALW	ALWays
APDU	Application Protocol Data Unit

ATR	Answer To Reset
BGT	Block Guard Time
BWT	Block Waiting Time
C-APDU	Command APDU
CLA	CLAss
CLK	Clock
C-TPDU	Command TPDU
CWI	Character Waiting Integer
CWT	Character Waiting Time
DAD	Destination ADress
DF	Dedicated File
DO	Data Object
EDC	Error Detection Code byte
EF	Elementary File
etu	elementary time unit
FCI	File Control Information
GSM	Global System for Mobile communications
ICC	Integrated Circuit Card
I-Block	Information Block
ID	IDentifier
IEC	International Electrotechnical Commission
IFS	Information Field Sizes
IFSC	Information Field Size for the UICC
IFSD	Information Field Size for the Terminal
INF	INFormation field
I/O	Input/Output
ISO	International Organization for Standardization
LEN	LENgth
LSB	Least Significant Bit
MF	Master File
MMI	Man Machine Interface
MSB	Most Significant Bit
NAD	Node Address byte
NEV	NEVer
NPI	Numbering Plan Identifier
PCB	Protocol Control Byte
PIN	Personal Identification Number
PPS	Protocol and Parameter Select (response to the ATR)
R-APDU	Response APDU
R-Block	Receive-ready Block
RFU	Reserved for Future Use
R-TPDU	Response TPDU
RST	Reset
SAD	Source ADdress
S-Block	Supervisory Block
SE	Security Environment
SFI	Short EF Identifier
SW	Status Word
TE	Terminal Equipment
TLV	Tag Length Value
TPDU	Transfer Protocol Data Unit
UICC	Universal Integrated Circuit Card
USIM	Universal Subscriber Identity Module
WTX	Waiting Time eXtenstion
WWT	Work Waiting Time

4 Physical Characteristics

Two physical types of UICCs are currently specified by the present document. One of the card sizes is to be supported by a terminal that is compliant to this specification. These are the "ID-1 UICC" and the "Plug-in UICC".

The physical characteristics of both types of UICCs shall be in accordance with ISO 7816-1,2 [24, 25] unless otherwise specified by the present document. The following additional requirements shall be applied to ensure proper operation in a Telecom environment.

Check the latest version of the 7816 series for changes.

4.1 Card

The ID-1 card physical characteristics shall conform to ISO/IEC 7816-1,2 [1,2]

The Terminal shall accept embossed ID-1 cards. The embossing shall be in accordance with ISO/IEC 7811 [7, 8]. The contacts of the ID-1 UICC shall be located on the front (embossed face, see ISO/IEC 7810 [6]) of the card.

NOTE: Card warpage and tolerances are now specified for embossed cards in ISO/IEC 7810 [6].

4.2 Card, Plug-in

The Plug-in UICC shall have a width of 25 mm, a height of 15 mm, a thickness the same as an ID-1 UICC and a feature for orientation.

Annex A of ISO 7816-2 [2] applies with the location of the reference points adapted to the smaller size. The three reference points P1, P2 and P3 measure 7,5 mm, 3,3 mm and 20,8 mm, respectively, from 0. The values in table A.1 of ISO 7816-2 [2] are replaced by the corresponding values of figure 1.

The physical characteristics of the plug-in card are defined in the present document.

NOTE: the plug-in is also referred to as ID-000.

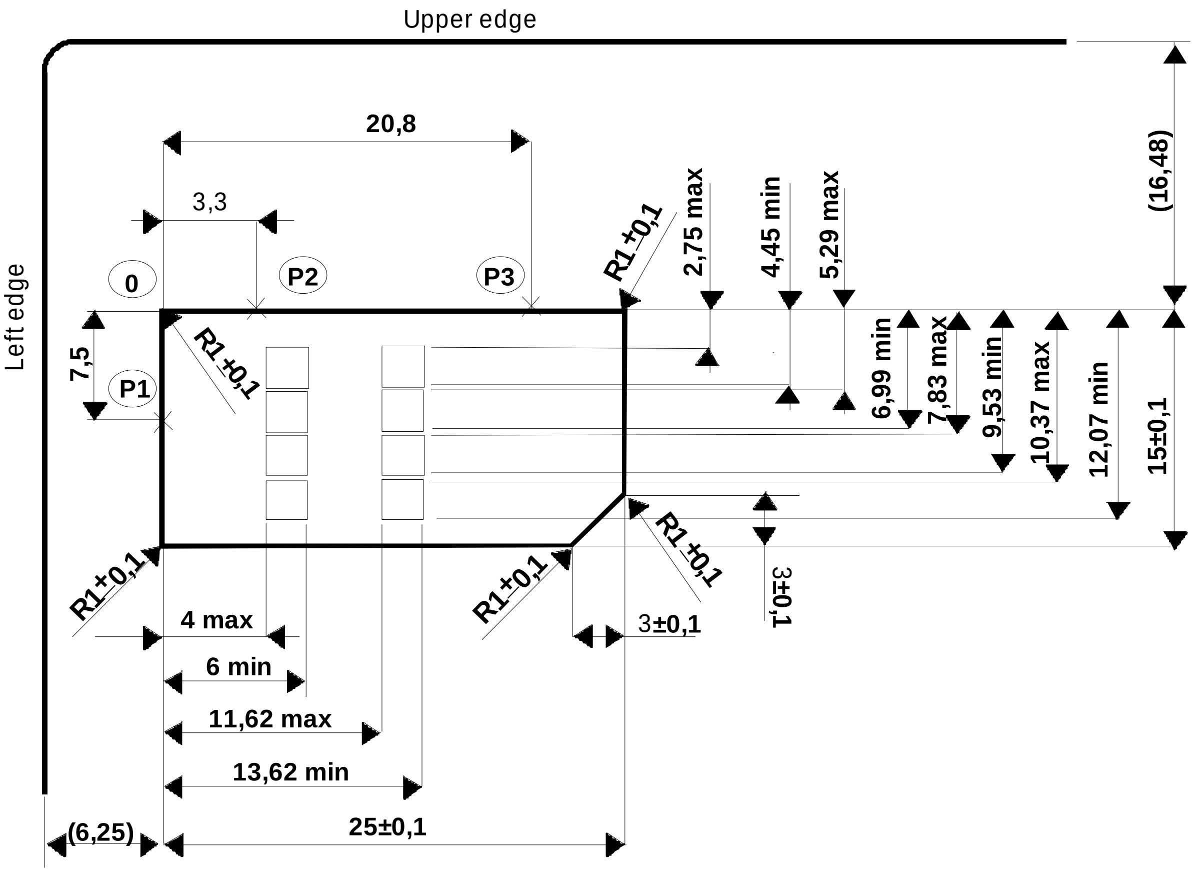


Figure 1: Plug-in UICC

4.3 Temperature range for card operation

The temperature range for full operational use shall be between -25°C and +70°C with occasional peaks of up to +85°C. "Occasional" means not more than 4 hours each time and not over 100 times during the life time of the card.

4.4 Contacts

4.4.1 Provision of contacts

Terminal: Contacting elements in the Terminal in positions C4 and C8 are optional, and are not used by a Telecom application complying to the present document. They shall present high impedance to the UICC. If it is determined that the UICC is a multi-application UICC, then these contacts may be used. Contact C6 need not be provided for Plug-in s.

UICC: Contacts C4 and C8 need not be provided by the UICC, but if they are provided, then they shall not be connected internally in the UICC if the UICC only contains a Telecom application. Contact C6 shall not be bonded in the UICC for any function other than supplying Vpp.

4.4.2 Contact activation and deactivation

The Terminal shall connect, activate and deactivate the UICC in accordance with the Operating Procedures specified in ISO/IEC 7816-3 [3].

For any voltage level, monitored during the activation sequence, or during the deactivation sequence following normal power-down, the order of the contact activation/deactivation shall be respected.

It is recommended that whenever possible, the deactivation sequence defined in ISO/IEC 7816-3 [3] should be followed by the Terminal on all occasions when the Terminal is powered down.

If the UICC clock is already stopped and is not restarted, the Terminal may deactivate all the contacts in any order, provided that all signals reach low level before Vcc leaves high level. If the UICC clock is already stopped and is restarted before the deactivation sequence, then the deactivation sequence specified in ISO/IEC 7816-3 [3] subclause 5.4 shall be followed.

When Vpp is connected to Vcc, as allowed in this specification, then Vpp shall be activated and deactivated with Vcc, at the time of the Vcc activation/deactivation, as specified in the sequences of ISO/IEC 7816-3 [3] subclauses 5.2 and 5.4.

4.4.3 Inactive contacts

The voltages on contacts C1, C2, C3, C6 and C7 of the Terminal shall be between 0 and ± 0,4 volts referenced to ground (C5) when the Terminal is switched off with the power source connected to the Terminal. The measurement equipment shall have a resistance of 50 kohms when measuring the voltage on C2, C3, C6 and C7. The resistance shall be 10 kohms when measuring the voltage on C1.

4.4.4 Contact pressure

The contact pressure shall be large enough to ensure reliable and continuous contact (e.g. to overcome oxidisation and to prevent interruption caused by vibration). The radius of any curvature of the contacting elements shall be greater than or equal to 0,8 mm over the contact area.

Under no circumstances shall the contact force exceed 0,5 N per contact.

Care shall be taken to avoid undue point pressure to the area of the UICC opposite to the contact area. Such pressure is potentially damaging to the components within the UICC.

5 Electrical specifications of the UICC – Terminal interface

Editor´s note: Add a comment that this document is a generic document. Each specific application e.g. possible voltage levels etc is specified in a separate document.

The electrical specification in the present document covers the supply voltage range from 4.5V to 5.5V, 2.7V to 3.3V and 1.62V to 1.98V. For each state (V_{OH}, V_{IH}, V_{IL} and V_{OL}), a positive current is defined as flowing out of the entity (Terminal or UICC) in that state. Vpp shall not be supported by the 3V and 1.8V technology Terminal or the 3V and 1.8V technology UICC.

When the UICC is in idle state, the current consumption shall not exceed 200 µA at 1 MHz at +25°C. When the UICC is in clock stop mode, the current consumption shall not exceed 100 µA at +25 °C.

The clock duty cycle shall be between 40 % and 60 % of the period during stable operation. A clock cycle is defined at 50% of Vcc from rising to rising edge or falling to falling edge. When switching clock frequencies Terminals shall ensure that no pulse is shorter than 100 ns which is 40 % of the shortest allowed period.

The Terminal supporting 3V or 1.8V need not provide contact C6 (Vpp). Contact C6 shall not be connected in the Terminal if provided.

5.1 Electrical Specification of the 5V UICC – Terminal Interface

5.1.1 Supply voltage Vcc (contact C1)

The UICC shall be operated within the following limits:

Table 5.1 Electrical characteristics of Vcc under normal operating conditions

Symbol	Minimum	Maximum	Unit
Vcc	4,5	5,5	V

The current consumption of the UICC shall not exceed the value given in table 5.4 during the ATR (including activation and deactivation).

When the UICC is in idle state (see below) the current consumption of the card shall not exceed 200 µA at 1 MHz and 25°C. If clock stop mode is enabled, then the current consumption shall also not exceed 200 µA while the clock is stopped.

The Terminal shall source the maximum current requirements defined above. It shall also be able to counteract spikes in the current consumption of the card up to a maximum charge of 40 nAs with no more than 400 ns duration and amplitude of at most 200 mA, ensuring that the supply voltage stays in the specified range.

NOTE: A possible solution would be to place a capacitor (e.g. 100 nF, ceramic) as close as possible to the contacting elements.

5.1.2 Reset (RST) (contact C2)

The Terminal shall operate the UICC within the following limits:

Table 5.2 Electrical characteristics of RST under normal operating conditions

Symbol	Conditions	Minimum	Maximum
V_{OH}	I_{OHmax} = +20 µA	Vcc-0,7	Vcc (note)
V_{OL}	I_{OLmax} = -200 µA	0V (note)	0,6 V
t_R t_F	C_{out} = C_{in} = 30 pF		400 µs
NOTE: To allow for overshoot the voltage on RST shall remain between -0,3 V and Vcc+0,3 V during dynamic operation.			

5.1.3 Programming voltage Vpp (contact C6)

The UICC shall not require any programming voltage on Vpp. The Terminal need not provide contact C6. If the Terminal provides contact C6, then, in the case of the ID-1 UICC the same voltage shall be supplied on Vpp as on Vcc, while in the case of Plug-in UICC the Terminal need not provide any voltage on C6. Contact C6 may be connected to Vcc in any Terminal but shall not be connected to ground.

5.1.4 Clock CLK (contact C3)

The Terminal shall support 1 to 5 MHz. The Terminal shall supply the clock. No "internal clock" UICC shall be used.

The duty cycle shall be between 40 % and 60 % of the period during stable operation.

The Terminal shall operate the UICC within the following limits:

Table 5.3 Electrical characteristics of CLK under normal operating conditions

Symbol	Conditions	Minimum	Maximum
V_{OH}	I_{OHmax} = +20 µA	0,7xVcc	Vcc (note)
V_{OL}	I_{OLmax} = -200 µA	0 V (note)	0,5 V
t_R t_F	$C_{out} = C_{in}$ = 30 pF		9 % of period with a maximum of 0,5 µs
NOTE: To allow for overshoot the voltage on CLK shall remain between -0,3 V and Vcc+0,3 V during dynamic operation.			

5.1.5 I/O (contact C7)

Table 5.4 defines the electrical characteristics of the I/O (contact C7). The values given in the table allow the derivation of the values of the pull-up resistor in the Terminal and the impedance of the drivers and receivers in the Terminal and UICC.

Table 5.4: Electrical characteristics of I/O under normal operating conditions

Symbol	Conditions	Minimum	Maximum
V_{IH}	I_{IHmax} = ± 20 µA (note 2)	0,7xVcc	Vcc+0,3 V
V_{IL}	I_{ILmax} = +1 mA	-0,3 V	0,8 V
V_{OH} (note 1)	I_{OHmax} = + 20µA	3,8 V	Vcc (note 3)
V_{OL}	I_{OLmax} = -1 mA	0 V (note 3)	0,4 V
t_R t_F	$C_{out} = C_{in}$ = 30 pF		1 µs
NOTE 1: It is assumed that a pull-up resistor is used in the interface device (recommended value: 20 kohms).			
NOTE 2: During static conditions (idle state) only the positive value can apply. Under dynamic operating conditions (transmission) short term voltage spikes on the I/O line may cause a current reversal.			
NOTE 3: To allow for overshoot the voltage on I/O shall remain between -0,3 V and Vcc+0,3 V during dynamic operation.			

5.2 Electrical Specifications of the 3V UICC – Terminal Interface

5.2.1 Supply voltage Vcc (contact C1)

Table 5.5: Electrical characteristics of Vcc under normal operating conditions

Symbol	Minimum	Maximum	Unit
Vcc	2,7	3,3	V

The Terminal shall be capable of sourcing the maximum current as defined in table 6.4. It shall also be able to counteract spikes in the current consumption of the card up to a maximum charge of 12 nAs with no more than 400 ns duration and an amplitude of at most 60 mA, ensuring that the supply voltage stays in the specified range.

5.2.2 Reset (RST) (contact C2)

Table 5.6: Electrical characteristics of RESET (RST) under normal operating conditions

Symbol	Conditions	Minimum	Maximum	Unit
V_{OH}	I_{OHmax} = + 20 µA	0,8 x Vcc	Vcc (Note)	V
V_{OL}	I_{OLmax} = -200 µA	0 (Note)	0,2 x Vcc	V
T_R t_F	C_{in} = C_{out} = 30 pF		400	µs
NOTE: To allow for overshoot the voltage on RST should remain between -0,3V and Vcc +0,3V during dynamic operations.				

5.2.3 Clock CLK (contact C3)

Table 5.7: Electrical characteristics of Clock (CLK) under normal operating conditions

Symbol	Conditions	Minimum	Maximum	Unit
V_{OH}	I_{OHmax} = + 20 µA	0,7 x Vcc	Vcc (Note)	V
V_{OL}	I_{OLmax} = - 20 µA	0 (Note)	0,2 x Vcc	V
T_R t_F	C_{in} = C_{out} = 30 pF		50	ns
NOTE: To allow for overshoot the voltage on CLK should remain between -0,3V and Vcc+0,3V during dynamic operations.				

5.2.4 I/O (contact C7)

Table 5.8: Electrical characteristics of I/O under normal operating conditions

Symbol	Conditions	Minimum	Maximum	Unit
V_{IH}	I_{IHmax} = ± 20 µA (Note 2)	0,7 x Vcc	Vcc+0,3	V
V_{IL}	I_{ILmax} = + 1 mA	- 0,3	0,2 x Vcc	V
V_{OH} (Note 1)	I_{OHmax} = + 20 µA	0,7 x Vcc	Vcc (Note 3)	V
V_{OL}	I_{OLmax} = - 1mA	0 (Note 3)	0,4	V
T_R t_F	C_{in} = C_{out} = 30 pF		1	µs
NOTE 1: It is assumed that a pull-up resistor is used on the interface device (recommended value: 20 k Ω). NOTE 2: During static conditions (idle state) only the positive value can apply. Under dynamic operating conditions (transmissions) short term voltage spikes on the I/O line may cause a current reversal. NOTE 3: To allow for overshoot the voltage on I/O shall remain between -0,3V and Vcc+0,3V during dynamic operation.				

5.3 Electrical Specifications of the 1.8V UICC – Terminal Interface

5.3.1 Supply voltage Vcc (contact C1)

Table 5.9: Electrical characteristics of Vcc under normal operating conditions

Symbol	Minimum	Maximum	Unit
Vcc	1,62	1,98	V

The Terminal shall be capable of sourcing the maximum current as defined in table 6.4. It shall also be able to counteract spikes in the current consumption of the card up to a maximum charge of 12 nAs with no more than 400 ns duration and an amplitude of at most 60 mA, ensuring that the supply voltage stays in the specified range.

5.3.2 Reset (RST) (contact C2)

Table 5.10: Electrical characteristics of RESET (RST) under normal operating conditions

Symbol	Conditions	Minimum	Maximum	Unit
V_{OH}	I_{OHmax} = + 20 µA	0,8 x Vcc	Vcc (Note)	V
V_{OL}	I_{OLmax} = -200 µA	0 (Note)	0,2 x Vcc	V
T_R t_F	C_{in} = C_{out} = 30 pF		400	µs
NOTE: To allow for overshoot the voltage on RST should remain between -0,3V and Vcc +0,3V during dynamic operations.				

5.3.3 Clock CLK (contact C3)

Table 17: Electrical characteristics of Clock (CLK) under normal operating conditions

Symbol	Conditions	Minimum	Maximum	Unit
V_{OH}	I_{OHmax} = + 20 µA	0,7 x Vcc	Vcc (Note)	V
V_{OL}	I_{OLmax} = - 20 µA	0 (Note)	0,2 x Vcc	V
T_R t_F	C_{in} = C_{out} = 30 pF		50	ns
NOTE: To allow for overshoot the voltage on CLK should remain between -0,3V and Vcc+0,3V during dynamic operations.				

5.3.4 I/O (contact C7)

Table 18: Electrical characteristics of I/O under normal operating conditions

Symbol	Conditions	Minimum	Maximum	Unit
V_{IH}	I_{IHmax} = ± 20 µA (Note 2)	0,7 x Vcc	Vcc+0,3	V
V_{IL}	I_{ILmax} = + 1 mA	- 0,3	0,2 x Vcc	V
V_{OH} (Note 1)	I_{OHmax} = + 20 µA	0,7 x Vcc	Vcc (Note 3)	V
V_{OL}	I_{OLmax} = - 1mA	0 (Note 3)	0,4	V
T_R t_F	C_{in} = C_{out} = 30 pF		1	µs

NOTE 1: It is assumed that a pull-up resistor is used on the interface device (recommended value: 20 k Ω).
NOTE 2: During static conditions (idle state) only the positive value can apply. Under dynamic operating conditions (transmissions) short term voltage spikes on the I/O line may cause a current reversal.
NOTE 3: To allow for overshoot the voltage on I/O shall remain between -0,3V and Vcc+0,3V during dynamic operation.

6 Initial communication establishment procedures

The Terminal and the UICC shall support all the procedures and codings defined in this clause. The Terminal shall support T=0 and T=1. The UICC shall support the protocol T=0 and optionally T=1.

6.1 UICC activation and deactivation

The Terminal shall activate and deactivate the contacts of the UICC according to clause 4.4.2. During activation supply voltage switching, as defined in clause 6.2, shall take place prior to any further activity not related to the supply voltage switching.

6.2 Supply voltage switching

The Terminal shall initially activate the UICC with the lowest voltage class available. If no ATR is received, the UICC shall be deactivated and activated with the next higher class, if supported by the Terminal. If an ATR is received at the first applied voltage class, the contents of the ATR shall be analysed by the Terminal. If the operating class used by the Terminal is not supported by the UICC, the Terminal shall deactivate the UICC and activate it with a supply voltage class indicated by the UICC. If the ATR is corrupted, the Terminal shall repeat the procedure at least 3 times using the same operating class before rejecting the UICC. In case of 3 consecutive corrupted ATRs, the Terminal may activate the UICC with the next higher class. The Terminal is restricted not to use but the next higher class in the retrial attempt in this case.

6.2.1 Supply Voltage Classes

The supply voltage class shall be indicated in the ATR by the UICC (TA_i i>2). Alternatively, for reasons of backward compatibility with existing applications, it may be indicated instead in the status response when the MF or DF is selected.

If supply voltage class indication is not present in the ATR, the Terminal shall use the alternative method for determine the supply voltage class. This method is based on the response to a STATUS command when the MF or a ADF is selected. If the supply voltage class is not indicated in the ATR the Terminal shall select the MF or a ADF and perform a STATUS command. The supply voltage class is indicated in the response data. The STATUS command voltage class indication shall for compatibility reasons be supported by a terminal supporting an existing application based on this voltage class indication.

Table 6.1: Supply Voltage Classes indicated in ATR

Symbol	Minimum	Maximum	Unit	Class	Encoding (Binary)
Vcc	4,5	5,5	V	A	xx xxx1
Vcc	2,7	3,3	V	B	xx xx1x
Vcc	1,62	1,98	V	C	xx x1xx
Vcc	RFU	RFU	V	D	xx 1xxx
Vcc	RFU	RFU	V	E	x1 xxxx
Note: Class A and B values are according to ISO/IEC 7816-3. Class C and D is a further evolution of value specified in ISO/IEC 7816-3. It is possible to support a range of classes. The support must be consecutive e.g. AB, BC etc. A combination like AC is not allowed. Note 2: x indicates '0'.					

Table 6.2: Supply Voltage Classes indicated in STATUS response

UICC Supply Voltage	Bit 7	Bit 6	Bit 5
5V	0 (RFU) [1]	0 (RFU) [1]	0 (RFU) [1]
3V	0 (RFU) [1]	0 (RFU) [1]	1
1.8V	0 (RFU) [1]	1	1
Future Class	1	1	1

6.2.2 Power Consumption Classes

The power consumption of the UICC during ATR is specified in table 6.3. Power consumption Class I is for supply voltage class A, class II is for supply voltage class B etc. The UICC power consumption during the ATR shall conform to the power supply class indicated in the ATR. If the UICC supports several supply voltage classes, each class shall conform to the corresponding ATR power consumption class. This is required because the terminal is not aware of the power consumption of the UICC until the ATR is received and an application is selected.

Table 6.3: Power Consumption classes that applies during ATR

Symbol	Voltage Class	Maximum	Unit	ATR Class
Icc	A	10	MA	I
Icc	B	6	MA	II
Icc	C	4	MA	III
Icc	D	RFU	MA	IV
Icc	E	RFU	MA	V

6.2.3 Application Related Electrical Parameters

The power consumption of an UICC depends upon the supply voltage class and the application it is running. The power consumption of the UICC is restricted to the values indicated in 6.3 until an application is selected. An application is considered selected when the access condition is successfully verified. If no access condition is required for the application, the application is considered selected when an application related command is executed within the selected application. Selecting the application and performing a STATUS command is not considered to be the execution of an application command for these purposes.

The Terminal retrieves the application power consumption requirements by selecting the application and performing a STATUS command within the application. The power consumption parameters are returned by the card in the response to a STATUS command. This information will be located in the response to a STATUS command issued at a DF or EF level.

If no power consumption indication is available in the card, the terminal shall assume the application power consumption as specified in table 6.4.

Table 6.4: Power Consumption during the Application Session

Symbol	Voltage Class	Maximum	Unit	Remark
Icc	A	60	mA	
Icc	A	10	mA	USIM Application
Icc	B	50	mA	
Icc	B	6	mA	USIM Application
Icc	C	20	mA	
Icc	C	4	mA	USIM Application
Icc	D	RFU	mA	
Icc	E	RFU	mA	

6.3 Answer To Reset content

The ATR is the first string of bytes sent from the UICC to the Terminal after a reset has been performed.

The following table gives an explanation of the ATR characters specified in ISO/IEC 7816-3 [3] and the requirements for their use in an UICC. The Terminal shall be able to receive interface characters for transmission protocols other than T=0 and T=1, historical bytes and a check byte, even if only T=0 and T=1 are used by the Terminal.

Table 6.5: General ATR format

Character	Contents	sent by the card	a) evaluation by the Terminal b) reaction by the Terminal
1. Initial character TS	coding convention for all subsequent characters (direct or inverse convention)	always	a) always b) using appropriate convention
2. Format character T0	subsequent interface characters, number of historical bytes	always	a) always b) identifying the subsequent bytes accordingly
3. Interface character (global) TA1	parameters to calculate the work etu	optional	a) always if present b) if TA1 is not '11' or '01', PPS procedure shall be used
4. Interface character (global) TB1	parameters to calculate the programming voltage and current	optional	a) always if present b) if PI1 is not 0, then reject the UICC
5. Interface character (global) TC1	parameters to calculate the extra guardtime requested by the card; no extra guardtime is used to send characters from the card to the Terminal	optional	a) always if present b) if TC1 is neither 0 nor 255, then reject the UICC; see the note after the table
6. Interface character TD1	protocol type; indicator for the presence of interface characters, specifying rules to be used for transmissions with the given protocol type	always, if T=1 is indicated in TD2 or T=15 is indicated in TDi, i>1	a) always if present b) identifying the subsequent characters accordingly.
7. Interface character (specific) TA2	not used for protocols T=0 and T=1	optional	a) optional b) --------
8. Interface character (global) TB2	parameter to calculate the programming voltage	never	the allowed value of TB1 above defines that an external programming voltage is not applicable
9. Interface character (specific) TC2	parameters to calculate the work waiting time	Optional	a) always if present b) using the work waiting time accordingly
10. Interface character TDi (i>1)	protocol type; indicator for the presence of interface characters, specifying rules to be used for transmissions with the given protocol type	Optional/Always if T=1 is supported (TD2)	a) always if present b)identifying the subsequent characters accordingly

(continued)

Table 6.5 (concluded): General ATR format

Character	Contents	sent by the card	a) evaluation by the Terminal b) reaction by the Terminal
11. Interface character TAi, TBi, TCi (I>2)	Characters that contain interface characters for other transmission protocols. If TD(i-1) indicates T=15 TAi is interpreted as global interface character	optional/Always if TD(i-1) indicates T=15	a) always b) if T=1 is indicated in TD2: TA3 indicates the IFSC TB3 indicates the CWI (b4 to b1) BWI (b8 to b5) TC3 indicates CRC is used if b1=1 LRC is used if b1=0 If T=15 is indicated in TD(i-1). TAi indicates: XI clock stop indicator over b8 b7 UI class indicator over b6 to b1
12. Historical characters T1,...,TK	contents not specified in ISO/IEC	optional	a) optional b) --------
13. Check character TCK	check byte (exclusive -ORing)	not sent if only T=0 is indicated in the ATR; If T=0 and T=15 are present and in all other cases TCK shall be sent	a) optional b) --------
NOTE: According to ISO/IEC 7816-3:1997 [3] N=255 indicates that the minimum delay is 12 etu for the asynchronous half-duplex character transmission protocol.			

6.3.1 Coding of historical bytes

The historical bytes indicate to the external world how to use the card. The information carried by the historical bytes of the UICC follows the clause 8 of ISO/IEC 7816-3 [3].

The category indicator is the first byte sent by the UICC. Its value shall be ‘80’ which means that the historical bytes are coded in COMPACT-TLV data objects.

The first information sent by the card shall be the “card data service” data object. This data object is introduced by ‘31’.

The second information sent by the card shall be the “card capabilities” data object. This data object is introduced by ‘73’.

The others data objects are optional.

6.3.2 Speed enhancement

If speed enhancement is implemented, the Terminal and the UICC shall at least support F=512 and D=8 in addition to F=372 and D=1 (default values). However, other values may also be supported (see table below). If the Terminal requests PPS using values other than those above then the PPS procedure shall be initiated accordingly. The value of the transmission factors F and D is given by the UICC in TA_1 at the ATR.

Table 6.6: Bit/s for the transmission factors F and D

F	D	f_{MAX} (MHz)	f_{CLK} (MHz)	Bit/s	M/O
372	1	5	3.5712	9600	M
372	12	5	3.5712	115200	O

512	8	5	4.9152	76800	M
512	16	5	4.9152	153600	O
512	32	5	4.9152	307200	O

6.4 PPS procedure

The Terminal and the UICC shall support the PPS procedure in order to use other transmission parameters than default values. The alternative parameters are indicated in the ATR. The interpretation of these parameters is according to ISO/IEC 7816-3 [3].

6.5 Reset

After a cold reset or a warm reset, the ATR sent by the UICC shall be identical. The cold reset and the warm reset are defined in ISO/IEC 7816-3 [3].

When a cold reset or a warm reset has been performed, the UICC shall be in the same status.

6.6 Clock stop mode

The Terminal shall wait at least 1860 clock cycles after having received the last character, including the guard time (2 etu), of the response before it switches off the clock (if it is allowed to do so). It shall wait at least 744 clock cycles before it sends the first command after having started the clock.

6.7 Bit/character duration and sampling time

The bit/character duration and sampling time specified in ISO/IEC 7816-3 [3], subclause 5.3.2 are valid for all communications.

6.8 Error handling

Following receipt of an ATR, which is not in accordance with this specification, e.g. because of forbidden ATR characters or too few bytes being transmitted, the Terminal shall perform a Reset. The Terminal shall not reject the UICC until at least three consecutive wrong ATRs are received.

During the transmission of the ATR and the PPS, the error detection and character repetition procedure specified in ISO/IEC 7816—3 [3], subclause 5.3.3, is optional for the Terminal. For the subsequent transmission on the basis of T=0 this procedure is mandatory for the Terminal.

For the UICC the error detection and character repetition procedure is mandatory for all communications.

6.9 Compatibility

Terminals operating at 5V only and that are compliant to the electrical parameters defined in section 5.1 and subsections can operate with a 3V Technology Smart Card according to this specification

For compatibility with existing terminals, UICCs that are used in applications where the supply voltage class detection is based on the STATUS response procedure (see section 4.4.1) shall support this procedure in addition to the supply voltage class indication in the ATR as defined in this standard.

In case the UICC do not support any supply voltage indication, the UICC shall be treated as a 5V only card by the Terminal.

NOTE: Do we need this clause at all – perhaps it should be moved to chapter 5????

7 Transmission Protocols

NOTE: This chapter needs to be updated and renumbered – to be done after the Korea meeting.

This section defines the transmission protocols used to exchange data between the terminal and the UICC. The structure and processing of commands initiated by the terminal for transmission control and for specific control in asynchronous half duplex transmission protocols will be described.

Two different protocols are defined, the character based protocol T=0 and the block based protocol T=1.

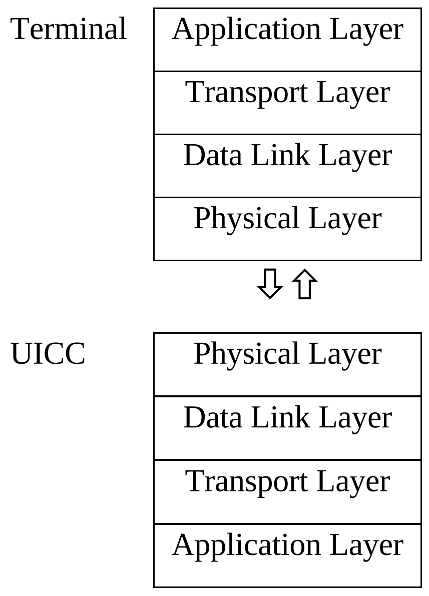
The UICC shall support T=0 or T=1 as well. Terminals shall support both protocols T=0 and T=1.

The protocol to be used for subsequent communication between the UICC and terminal is indicated in TD1, and shall be T=0 or T=1. If TD1 is absent in the ATR, T=0 is assumed.

The protocol starts after either the answer to reset or a successful PPS exchange. Other parameters provided in the ATR and relevant to a specific protocol are defined in the respective parts of this section.

The protocol applies a layering principle of the OSI-reference model. Four layers are considered. The layers are:

- The physical layer. The contained definitions are valid for T=0 and T=1
- The data link layer, which consists of:
  a character component
  a block component
  block identification
  send blocks
  detect transmission and sequence errors
  handle errors
  synchronise the protocol
- The transport layer , which defines the transmission of application-oriented messages specific to each protocol
- The application layer, which defines the exchange of messages according to an application protocol that is common to both protocols



7.1 Physical Layer

Both protocols T=0 and T=1 use the physical layer and character frame as defined in 7.2.1

7.2 Data Link Layer

This section describes the timing, specific options and error handling for T=0 and T=1 protocols.

7.2.1 Character Frame

A character which will be transmitted over the I/O line is embedded in a character frame.

Before the transmission of a character, the I/O line shall be in state H.

A character consists of 10 consecutive bits (see Figure 7.1):

- 1 start bit in state L
- 8 bits, which comprise the data byte
- 1 even parity checking bit

The parity bit is set, in a way, that there is an even number of bits set to ‘1’ including the parity bit in the character frame.

The time origin is fixed as the mean between the last observation of state H and the first observation of state L. The receiver shall confirm the existence of a start bit before 0.7 etu (receiver time). Then the subsequent bits shall be received at intervals of (n + 0.5 ± 0.2) etu (n being the rank of the bit). The start bit is bit 1.

Within a character, the time from the leading edge of the start bit to the trailing edge of the nth bit is (n ± 0.2) etu.

The interval between the leading edges of the start bits of two consecutive characters is comprised of the character duration (10 ± 0.2) etu, plus a guardtime. Under error free transmission, during the guardtime both the ICC and the terminal shall be in reception mode (I/O line in state H).

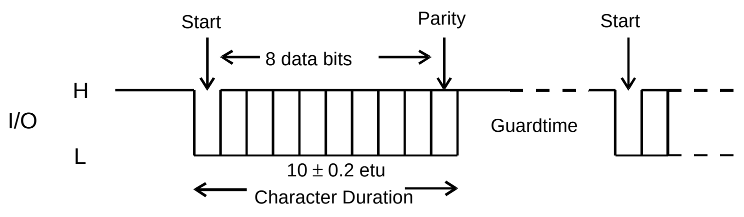


Figure 7.1: Character Frame

The data shall always be passed over the I/O line with the most significant bit (msb) first. The order of bits within a byte (that is, whether the least significant (l.s.) or m.s. bit is transferred first) is specified in character TS returned in the answer to reset.

7.2.2 Transmission Protocol T=0

The T=0 is a half-duplex asynchronous character based transmission protocol.

All commands using the T=0 are initiated from the terminal by sending a five byte header, which informs the UICC what to do. The terminal will always act as master and the UICC as a slave. The direction of the transmission is assumed to be known to both the UICC and the terminal.

7.2.2.1 Timing and specific options for characters in T=0

The minimum interval between the leading edge of the start bits of two consecutive characters shall be at least 12 etus.

The maximum interval between the start leading edge of any character sent by the UICC and the start leading edge of the previous character sent by either by the UICC or the terminal i.e. the WWT. The value of the WWT shall not exceed $960 \times WI \times Fi/f$. WI is an integer received in the specific interface byte TC2. If no TC2 is available, default value of WI is 10. The clock rate conversion factor, Fi, may be indicated in TA1. If TA1 is absent the default value 372 is used.

If the WWT is exceeded, the terminal shall initiate a deactivation within 960 etus.

Note (Thomas): Why is it 960 etus. ISO: On the work waiting time overflow, VPP shall be set to or maintained at pause state

The minimum interval between the leading edges of the start bits of two consecutive characters sent in opposite direction shall be 16 etus.

7.2.2.2 Command Header

Note (Thomas): Changed Headline because it is a Card's and not a Terminal Command. We should also go for terms which are already introduced by ISO and EMV.

A command is always initiated by the terminal which sends an instruction to the UICC in the form of a five byte header called the command header. The command header is comprised of five consecutive bytes, CLA, INS, P1, P2, and P3.

These bytes together with any data to be sent with the command form the command transport protocol data unit (C-TPDU) for T=0. The mapping of the command Application Protocol Data Unit (C-APDU) onto the C-TPDU is described in clause 12.5.

The terminal transmits the header to the UICC and waits for a procedure byte.

7.2.2.3 Command Processing

Note (Thomas): Changed Headline because it Command Processing seems more precise in order to show that the card is processing the card inside as well as to show that there is a process to get the data from the card.

When the UICC has received the command header, a response containing a procedure byte shall be sent to the terminal. Both the terminal and the UICC shall be able to keep track of the direction of the data flow and who has the access to the I/O-line.

7.2.2.3.1 Procedure bytes

The procedure byte indicates to the terminal what action it shall take next. They are used to keep up the communication between the terminal and the UICC. Procedure bytes will never be transmitted to the Application Layer.

The coding of the byte and the action that shall be taken by the terminal is shown in table 7.1.

NOTE: A better wording is needed for the paragraph above.

Table 7.1: Procedure Byte coding

Byte	Value	Action
ACK	Equal to INS byte	All remaining data bytes shall be transferred by the terminal, or the terminal shall be ready to receive all remaining data bytes from the UICC
	Equal to complement of INS byte ($\overline{\text{INS}}$)	The next data byte shall be transferred by the terminal, or the terminal shall be ready to receive the next data byte from the UICC
NULL	‘60’	The NULL-byte requests no further data transfer and the terminal shall only wait for a character conveying a procedure byte. This behaviour provides additional work waiting time as defined in this section.
SW1	‘61’	The terminal shall wait for a second procedure byte then send a GET RESPONSE command header to the UICC with a maximum length of ‘xx’, where ‘xx’ is the value of the second procedure byte (SW2)
	‘6C’	The terminal shall wait for a second procedure byte then immediately repeat the previous command header to the UICC using a length of ‘xx’, where ‘xx’ is the value of the second procedure byte (SW2)

After these actions, the terminal shall wait for a further procedure byte or status word

7.2.2.3.2 Status bytes

The status bytes SW1 SW2 form an end sequence indicating the status of the UICC at the end of a command. A normal ending of a command is indicating by SW1 SW2 = '90 00'

Table 7.2: Status Byte Coding

Byte	Value	Action
SW1	‘6x’ or ‘9x’ (except ‘60’, ‘61’, and ‘6C’)	The terminal shall return the status words (together with any appropriate data) to the Application Layer and shall wait for another C-APDU

7.2.2.4 Error detection and correction

The error detection and correction procedure is mandatory for the T=0 protocol except for during the ATR-procedure.

An error, from the receiver’s point of view, is defined by an incorrect parity. The error is indicated on the I/O-line, which is set to state L after (10.5 ±0.2) etus after the leading edge of the start bit for the character. The I/O line shall be in state L for a maximum of 2 etus and a minimum of 1 etu.

If the UICC or terminal as receiver detects a parity error within 11±0.2 etus starting from the leading edge of the start bit, in a character just received, it shall set I/O to state L to indicate the error to the sender – see figure 2 “character frame”.

If the transmitter detects an error, the character shall be sent again after a minimum delay of 2 etus.

7.2.3 Transmission protocol T=1

Editors note: This section describes the low level protocol and parameters related to the T=1 protocol such as error handling, character timing, block timing and interface parameters.

The T=1 protocol is a half-duplex asynchronous block based transmission protocol. The protocol may be initiated after ATR or a successful PPS exchange.
The communication starts with a block sent by the terminal to the UICC. The right to send a block keeps alternating between the terminal and the UICC. A block is the smallest data unit, which can be sent and can contain either application data or transmission control data. A check of the received data might be performed before further processing of the received data.

7.2.3.1 Timing and specific options for blocks sent with T=1

This chapter defines options regarding timing, information file sizes and error detection parameters for blocks sent with T=1.

7.2.3.1.1 Information field size

The IFSC defines the maximum length of the information field of blocks that can be received by the UICC. The default value of the IFSC is 32 bytes another value may be indicated in TA(3) of the ATR.

The IFSD defines the maximum length of the information field of blocks that the terminal can receive. IFSD is initiated to 254 after ATR and shall not be adjusted during the rest of the card session.

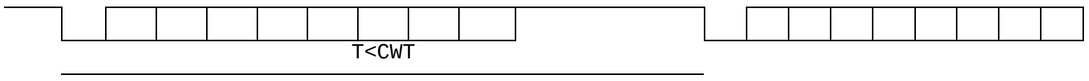
Editor's Note (Thomas): The initial IFSD is 32 bytes. In present day terminals it should not be a problem to have 254 Bytes. But most cards, working with T=1, have implemented a 32 Byte default (conforming to ISO). The first block which should be sent by the terminal shall be a IFSD-Request with 254 Bytes.

7.2.3.1.2 Character waiting integer

CWI is used to calculate CWT and shall be 0. The value is set in bits b4 to b1 in TB3.

7.2.3.1.3 Character waiting time

CWT is defined as the maximum delay between the leading edges of two consecutive characters in the block.

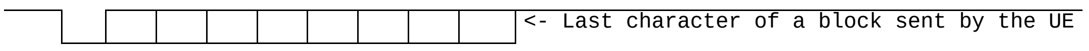


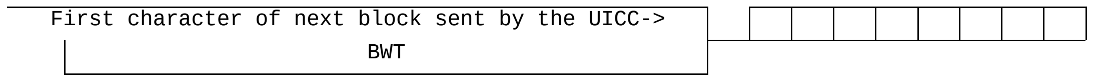
The value of CWT is 12 etus according to the formula below with CWI = 0.

$CWT = (11 + 2^{CWI})$ etu

7.2.3.1.4 Block waiting time

BWT is defined as the maximum delay between the leading edge of the last character of the block received by the card and the leading edge of the first character of the next block sent by the card.

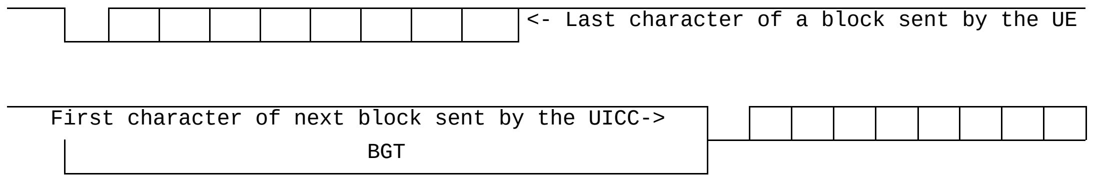




BWT is used to detect an unresponsive card.

7.2.3.1.5 Block guard time

BGT is the minimum delay between the leading edge of two consecutive characters sent in opposite directions is defined as the block guard time BGT. BGT is set to 22 etu.


<- Last character of a block sent by the UE
First character of next block sent by the UICC->
BGT

The delay between the last character of a block received by the UICC and the first character of the next block sent from the UICC shall be in the interval:

BGT < delay < BWT

7.2.3.1.6 Waiting time extension

WTX is a parameter used to ask for more time to process a command.

7.2.3.1.7 Error detection code

The parameter TC(i) in the ATR is used to define which error detection code to use. If b1=0 LRC will be used. All other bits in TC(i) are reserved for future usage and shall be set to 0.

Editor’s note: Support for CRC as well?

7.2.3.2 Block frame structure

The protocol consists of blocks, which are transmitted between the TERMINAL and the UICC. Each block has the following structure:

Prologue field			**Information field**	**Epilogue field**
NAD	PCB	Len	INF	EDC
1 byte	1 byte	1 byte	0-254 bytes	1 or 2 bytes

Where the prologue field and the epilogue field are mandatory and the Information field is optional.

7.2.3.2.1 Prologue field

The prologue field is divided into the following three mandatory fields:

- Node address byte (NAD), 1 byte
- Protocol Control byte (PCB), 1 byte
- Length (Len), 1 byte

7.2.3.2.1.1 Node address byte

The NAD-byte identifies the source and the intended destination of the block. The NAD may also be used to distinguish between different logical connections if they coexist. Below is the structure of the NAD-byte:

b8	b7	b6	b5	B4	b3	b2	b1
Unused	DAD			Unused	SAD		

In the first block sent from the UE, a logical connection will be set up based on the addresses in SAD and DAD. Subsequent blocks with an NAD containing the same pair of addresses are associated with the same logical connection. Other logical connections with other pairs of addresses shall be possible to establish during information exchange.

In UMTS phase 1, only the default value SAD=DAD=0 shall be supported. All other combinations are not allowed.

7.2.3.2.1.2 Protocol Control Byte

All information needed to control the transmission is transferred in the protocol control byte PCB. The coding of the PCB-byte specifies the type on the block. In the T=1 protocol the following three different types of block are supported:

- Information block, I-block which is used to transfer command and response APDU´s
- Receive-ready block, R-block, which is used to transfer acknowledgements
- Supervisory block, S-block, which is used to send control information

The tables below present the coding of the PCB-byte for each block-type, starting with the I-block.

Table 7.3: Coding of PCB for an I-block

b8	b7	b6	b5	B4	b3	b2	b1
0	Sequence number, N(S)	Chaining, more-data bit, M	Reserved for future usage				

Table 7.4: Coding of PCB for an R-block

b8	b7	b6	b5	B4	b3	b2	b1
1	0	0	Sequence number N(R)	See table 7.5			

Table 7.5: Bit b4-b1 in the PCB-byte for the R-block

b4	b3	b2	B1	Value	Meaning
0	0	0	0	‘0’	Error free
0	0	0	1	‘1’	EDC and/or parity error
0	0	1	0	‘2’	Other error
x	x	x	X	‘X’	Other values reserved for future usage

Table 7.6 Coding of PCB for an S-block

b8	b7	b6	b5	B4	b3	B2	b1
1	1	X	See table 7.7				

Table 7.7 Bits b5-b1 of PCB for an S-block

b5	b4	b3	b2	b1	Value	Meaning
0	0	0	0	1	‘1’	Information field
0	0	0	1	0	‘2’	Abortion
0	0	0	1	1	‘3’	Extension of BWT
X	x	x	x	x	‘X’	Other values reserved for future usage

The combination of b6 and b5-b1 contains information if there is a request (b6=0) or if there is a response (b6=1).

7.2.3.2.1.3 Length

The length byte codes number of bytes in the Information field in the block. Number of bytes in the information field may vary in the range of 0 to 254 bytes, depending on the type of block.

The value '00' on the LEN-byte indicates that the information field is absent and the value 'FF' is reserved for future usage.

7.2.3.2.1.4 Information field

The information field, INF is optional and it depends on the type of the block what the field will be used for.

Type of block	INF used for
I-block	Transfer command and response APDU’s.
R-block	Not used, shall be absent do we not use this??
S-block	Transfers non application related information: • INF shall be present (single byte) to adjust IFS with WTX • INF shall be absent to signal error on VPP, or managing chain abortion or resynchronisation

7.2.3.2.2 Epilogue field

The epilogue field contains the error detection code-byte (EDC), which transfers the error detection code of the transmitted block. Possible code is longitudinal redundancy check (LRC).

The LRC consists of one byte and the value is such that an exclusive-OR of all bytes starting with the NAD ending with the last byte of the INF, if present, shall be null.

7.2.3.2.3 Block notations

7.2.3.2.3.1 I-block

The I-blocks are denoted as follows: I(N(S), M) where:

- N(S) is the send-sequence number of the block
- M is the more-data bit used in the chaining function

7.2.3.2.3.2 R-block

The R-block is denoted as R(N(R), where

- N(R) is the number of the expected I-block

7.2.3.2.3.3 S-block

S-blocks are always used in pairs. A S(request) is always followed by a S(response) block. The S-block is denoted as:

- S(IFS request), an offering of a maximum size of the information field
- S(IFS response), an acknowledge on the information field
- S(ABORT request), a request to abort the chain function, not required in EMV
- S(ABORT response), an acknowledge of the abortion of the chain function not required in EMV
- S(WTX request), a request for an extension of the waiting time.
- S(WTX response), an acknowledge of the extension of the waiting time

7.2.3.3 Error free operation

This chapter describes the rules for error free operation with T=1.

- The terminal sends the first block, which should be either an I-block with N(S)=0 or an S-block.
- If a sender S sends I(N_s (S), 0), the block is acknowledged by the receiver R with a I(N_r (S), M). The contents of I(N_r (S) indicates data transfer data and that the receiver is ready to receive the next block from the sender.
- If a sender S sends an I(N_s(S), 1) it should be acknowledged by the receiver R with R(N_r(R)), where N_s(S) N_r(R), to indicate that the received block was correct and that the receiver is ready to receive the next block.
- The UICC might need more than BWT to process the previously received block, a S(WTX request) is sent by the UICC. The terminal shall acknowledge with a S(WTX response). The new allocated time starts at the leading edge of the last character of the S(WTX response).
- To change the initial value on IFSD, which was indicated during ATR the terminal sends an S(IFS request). The request shall be acknowledged by the UICC with an S(IFS response) with the same INF. The new IFSD is assumed to be valid as long as no new S(IFS request) has been received by the UICC.
- When the receiver has received the number of characters as indicated in the value of the LEN and EDC the receiver returns the right to send.
- S-blocks are always used in pairs. A S(request) is always followed by a S(response) block.

7.2.3.4 Error handling for T=1

This chapter contains a description of the rules used to control the error handling for the T=1 protocol.

The block component shall of the data link layer shall be able to handle errors like: BWT time-out, receive an invalid block, i.e. a block with parity errors, EDC error, invalid PDC, invalid length, lost synchronisation or failure to receive relevant S(… response) after a S(… request).

7.2.3.4.1 Protocol initialisation

After a ATR or successful PPS procedure the communication between the terminal and the UICC can be initiated. But if the terminal fails to receive an error-free block, in the beginning of the protocol, a maximum of two more successive attempts to receive the block is allowed before resetting or a deactivation of the card takes place.

If the response on the first block sent by the terminal not is sent within BWT, the terminal will send a R(0).

When the protocol has been initiated and the first block received by the UICC is invalid, the UICC responses with a R(0).

If the terminal fails to receive an error-free block during a card-session, a maximum of two further attempts is allowed before a S(RESYNCH request) is sent.

7.2.3.4.1 Block dependent errors

When an I-block has been sent and a BWT time-out occurs or an invalid block has been received in the terminal, a R-block is sent, which requests with its N(R) for the expected I-block with N(S)=N(R).

If an R-block was sent and an invalid block is received or BWT time-out, the R-block will be resent.

When an S(… request) has been sent and either a BWT time-out occurs (in the terminal) or the received response not is a S(… response), the S(… response) will be resent. But if an S(… response) has been sent and either an invalid block is received or a BWT time-out occurs (in the terminal), an R-block will be sent.

If the UICC sends an S(IFS request) and receives an invalid block, the S(IFS request) will be resent maximum one extra time to receive an S(IFS response). After the second failure to receive an S(IFS response), the UICC stays in reception mode.

7.2.3.5 Chaining

Chaining allows the terminal or the UICC to transfer information, which is longer than IFSC or IFSD. If information longer than IFSC or IFSD is transferred the information should be divided into pieces, which each has a length =< IFSC or IFSD respectively. Each piece should be sent in an I-block using the chaining function.

The value of the M-bit in the PCB–byte of the I-block controls the chaining function according to:

- M = 0, the block is not chained to the next block
- M = 1, the block is chained to the next block, which shall be an I-block

When a receiver receives a more-data I-block, a R(N(R)) shall be sent. N(R)= N(S) of the expected I-block. At least one chained block should follow.

A physical error, e.g. buffer overrun, in the UICC can cause an error in a chaining process. To abort a chain an S(ABORT request) can be sent by either the sender or the receiver. The request shall be answered with an S(ABORT response). When the S(ABORT response) has been received an R-block may be sent to either the terminal or the UICC to give back the right to send to either.

7.2.3.5.1 Rules for chaining

- When the terminal is the receiver, the terminal shall accept a sequence of chained I-blocks sent from the UICC. The length of each block is =< IFSD
- When the UICC is the receiver, the UICC shall accept a sequence of chained I-blocks sent from the terminal. The length of each block shall be equal to the value of IFSC except for the last block whose length can be any value in the range of 1 to IFSC.
- When the terminal is the sender all I-blocks of a chain shall have LEN = IFSC bytes except for the last, which could have a value in the range of 0 to IFSC – see NOTE.
- When the UICC is the sender all I-blocks of a chain shall have LEN $\leq$ IFSC bytes per block
- If the UICC is the receiver and receives block with LEN> IFSC, the block shall be rejected and acknowledged with a R-block with bits b1-b4 in the PCB-byte having a value of 2.

NOTE: This would be used to force an acknowledgement from the UICC.

7.3 Transport Layer

This section describes how the APDU are transported between the terminal and the UICC.

7.3.1 Transportation of a APDU using T=0

This section describes the mapping of C-APDU's and R-APDU's for T=0 Protocol. The APDU exchange and the use of the GET RESPONSE command for case 2 and case 4 will be shown.

7.3.1.1 Mapping of C-APDU's and R-APDU's and Data Exchange

The mapping of the C-APDU onto the T=0 command header is dependent upon the case of the command. The mapping of the data (if present) and status returned by the ICC onto the R-APDU is dependent upon the length of the data returned.

Procedure bytes SW1 SW2 = '61xx' and SW1 SW2 = '6Cxx' are returned by the UICC to control exchanges between the Transport Layer of the Terminal and the UICC, and should never be returned to the Application Layer of the Terminal. Command processing in the UICC is not complete if it has returned procedure bytes SW1 SW2 = '61xx' or SW1 SW2 = '6Cxx'.

Normal status on completion of processing a command is indicated if the UICC returns status words SW1 SW2 = '9000' to the Transport Layer of the Terminal. Any other value of status words returned by the UICC indicates that the UICC has terminated processing of the command, and that the processing was unsuccessful for the reasons indicated in the status bytes. The Transport Layer of the Terminal shall discontinue processing of a command on receipt of any status words (but not on receipt of procedure bytes '61xx' and '6Cxx') from the UICC, irrespective of whether they indicate a normal, warning, or error condition.

The following descriptions of the mapping of data and status returned by the UICC onto the R-APDU apply only after the UICC has completed processing of the command, successfully or otherwise, and all data (if present) has been returned by the UICC under the control of '61xx' and '6Cxx' procedure bytes. Detailed use of the INS, $\overline{\text{INS}}$, and '60' procedure bytes is not described.

7.3.1.1.1 Case 1

The C- APDU is mapped onto the C-TPDU by assigning the value ´00´ to the body part (B = '00').

	Terminal		**UICC**
Application Layer (C-APDU)r	CLA INS P1 P2		
Transport Layer (T-APDU)r	CLA INS P1 P2 B = '00'	➔	Command Processing
			SW1||SW2
Transport Layer (R-TPDU)	SW1||SW2	🡐	SW1||SW2
Application Layer (R-APDU)	SW1||SW2		

See Annex C for details of the exchanges between the Transport Layer of the Terminal and the UICC.

The status words returned to the Transport Layer of the Terminal from the UICC after completion of processing of the command are mapped onto the mandatory trailer of the R-APDU without change.

Note: The ICC shall analyse the T=0 command header to determine whether this is a case 1 command or a case 2 command requesting response data of maximum length.

7.3.1.1.2 Case 2

The command APDU is mapped onto the command TPDU without any change.

	Terminal		UICC
Application Layer (C-APDU)r	CLA INS P1 P2 Le		
Transport Layer (T-APDU)r	CLA INS P1 P2 Le	→	Command Processing
			Le byte of DATA||SW1||SW2
Transport Layer (R-TPDU)	Le byte of DATA||SW1||SW2	←	Le byte of DATA||SW1||SW2
Application Layer (R-APDU)	Le byte of DATA||SW1||SW2		

Note: All case 2 commands issued according to this specification shall have Le set to '00'.

See Annex C for details of the exchanges between the Transport Layer of the Terminal and the UICC, including use of the ‘61xx’ and ‘6Cxx’ procedure bytes.

The R-TPDU is mapped onto the R-APDU without any change

The data (if present) and status returned to the Transport Layer of the Terminal from the UICC after completion of processing of the command are mapped onto the R-APDU as follows:

7.3.1.1.3 Case 3

The command APDU is mapped onto the command TPDU without any change. Lc is a value between 1 and 255.

	Terminal		UICC
Application Layer (C-APDU)r	CLA INS P1 P2 Lc [Lc bytes of DATA]		
Transport Layer (T-APDU)r	CLA INS P1 P2 Lc [Lc bytes of DATA]	→	Command Processing
			SW1||SW2
Transport Layer (R-TPDU)	SW1||SW2	←	SW1||SW2
Application Layer (R-APDU)	SW1||SW2		

See Annex C for details of the exchanges between the Transport Layer of the Terminal and the UICC.

The status words returned to the Transport Layer of the Terminal from the UICC after completion of processing of the command, or the status words returned by the UICC that caused the Transport Layer of the Terminal to discontinue processing of the command, are mapped onto the R-APDU without change.

7.3.1.1.4 Case 4

The command APDU is mapped onto the command TPDU by cutting off the last byte (Le) of the body .

	Terminal		UICC
Application Layer (C-APDU)r	CLA INS P1 P2 Lc [Lc bytes of DATA] Le		
Transport Layer (T-APDU)r	CLA INS P1 P2 Lc [Lc bytes of DATA]	→	Command Processing
			Luicc byte of DATA‖SW1‖SW2
Transport Layer (R-TPDU)	Luicc byte of DATA‖SW1‖SW2	←	Luicc byte of DATA‖SW1‖SW2
Application Layer (R-APDU)	Luicc byte of DATA‖SW1‖SW2		

The first response TPDU from the card indicates that the card performed the command correct and that the card has more data of length Luicc bytes to transfer. The first response TPDU is mapped without any changes onto the response APDU.

Note: All case 4 commands issued according to this specification shall have Le set to '00' in the C-APDU.

See Annex C for details of the exchanges between the Transport Layer of the Terminal and the UICC, including use of the '61xx' and '6Cxx' procedure bytes.

Editor's note: there may be differences in the definition of the GET RESPONSE command/response between GSM 11.11 and ISO – which should be used considering the (possible) backwards compatibility.

For cases 3 and 5, when SW1/SW2 indicates there is response data (i.e. SW1/SW2 = '9FXX'), then, if the Terminal requires to get this response data, it shall send a GET RESPONSE command as described in the relevant case above.

For case 5, in case of an ENVELOPE for SIM data download, SW1/SW2 can also indicate that there is response data with the value '9EXX', and the Terminal shall then send a GET RESPONSE command to get this response data.

NOTE: should these cases be depicted as well?

7.3.2 Transportation of a APDU using T=1

A C-APDU is sent from the Application Layer of the Terminal to the Transport Layer of the Terminal. The Transport Layer maps the C-APDU onto the INF field of an I-block without change. The I-block is sent to the UICC. The Response data (if present) and the status is returned from the UICC to the Transport Layer of the Terminal in the INF field of an I-block. If the UICC returns a status which indicates:

- normal processing ('61xx')
- a warning ('62xx' or '63xx')
- an application condition ('9xxx')
- or a successful execution of the command ('9000')

it shall also return data (if available) associated with the processing of the command. No data shall be returned with any other status.

The contents of the INF field of the I-block are mapped onto the R-APDU without change and returned to the Application Layer of the Terminal. The transportation of APDU messages with T=1 is mapped to the information field of an I-block according to the four different cases described below. Each case is described in detail in the following chapters.

7.3.2.1 Case 1

Command APDU is mapped to the information field of the I-block without any changes

	Terminal		UICC
Application Layer (C-APDU)r	CLA INS P1 P2		
Transport Layer (T-APDU, Information field)	CLA INS P1 P2	➔	Command Processing
			SW1||SW2
Transport Layer (R-TPDU, Information field)	SW1||SW2	🡐	SW1||SW2
Application Layer (R-APDU)	SW1||SW2		

The response received from the information field in the I-block is mapped unchanged to the response APDU. Information field:

7.3.2.2 Case 2

The C-APDU is mapped to the information field of an I-block without any changes.

	Terminal		UICC
Application Layer (C-APDU)r	CLA INS P1 P2 Le		
Transport Layer (T-APDU, Information field)	CLA INS P1 P2 Le	➔	Command Processing
			DATA||SW1||SW2
Transport Layer (R-TPDU, Information field)	DATA||SW1||SW2	🡐	DATA||SW1||SW2
Application Layer (R-APDU)	DATA||SW1||SW2		

The response of the APDU consists of either the information field of the I-block or the concatenation of the information field of successive I-blocks all received in the same response. All these blocks shall be chained.

7.3.2.3 Case 3

The C-APDU is mapped without any changes to either an information field or is concatenated onto several successive I-blocks, which all shall be chained.

	Terminal		UICC
Application Layer (C-APDU)r	CLA INS P1 P2 Lc [Lc bytes of DATA]		
Transport Layer (T-APDU, Information field)	CLA INS P1 P2 Lc [Lc bytes of DATA]	➔	Command Processing
			DATA||SW1||SW2
Transport Layer (R-TPDU, Information field)	DATA||SW1||SW2	🡐	DATA||SW1||SW2
Application Layer (R-APDU)	DATA||SW1||SW2		

The information field of the I-block is mapped to the R-APDU without any changes

7.3.2.4 Case 4

The C-APDU is mapped without any changes to either an information field or is concatenated onto several successive I-blocks, which all shall be chained.

	Terminal		UICC
Application Layer (C-APDU)r	CLA INS P1 P2 Lc [Lc bytes of DATA] Le		
Transport Layer (T-APDU, Information field)	CLA INS P1 P2 Lc [Lc bytes of DATA] Le	→	Command Processing
			Le byte of DATA||SW1||SW2
Transport Layer (R-TPDU, Information field)	Le byte of DATA||SW1||SW2	←	Le byte of DATA||SW1||SW2
Application Layer (R-APDU)	Le byte of DATA||SW1||SW2		

The response consists of either the information field of an I-block received in the response or the concatenation of information fields of successive I-blocks in response. All these blocks shall be chained.

7.4 Application Layer

The application protocol consists of an ordered set of exchanges between the Application Layer and the Transport Layer of the Terminal. Application protocols are defined in subsequent parts of this specification.

Each step in an application layer exchange consists of a command-response pair, where the Application Layer of the Terminal sends a command to the UICC via the Transport Layer of the Terminal, and the UICC processes it and sends a response to Application Layer of Terminal using the Transport Layer of the UICC and the Transport Layer of Terminal. Each specific command (C-APDU) has a specific response (R-APDU). The commands and responses are called command messages and response messages. The structure of the C-APDU can be found in clause 10.2. The structure of the R-APDU can be found in clause 10.3

Both command and response messages may contain data. Thus, four cases shall be managed by the transmission protocols via the transport layer, as shown in table 7.8.

Table 7.8: Definition of Cases for Data in APDU's

Case	Command Data	Response Data
1	Absent	Absent
2	Absent	Present
3	Present	Absent
4	Present	Present

7.4.1 Exchange of C-APDU and R-APDU

The following figure shows the principle exchange of command/response pairs.

Terminal		UICC
C-APDU	→	
	←	R-APDU

C-APDU	→	
	←	R-APDU

8 Application and File structure

This clause describes the application and logical structure for the UICC.

8.1 UICC Application structure

An example of organisation of applications in the UICC is listed in figure 8.1:

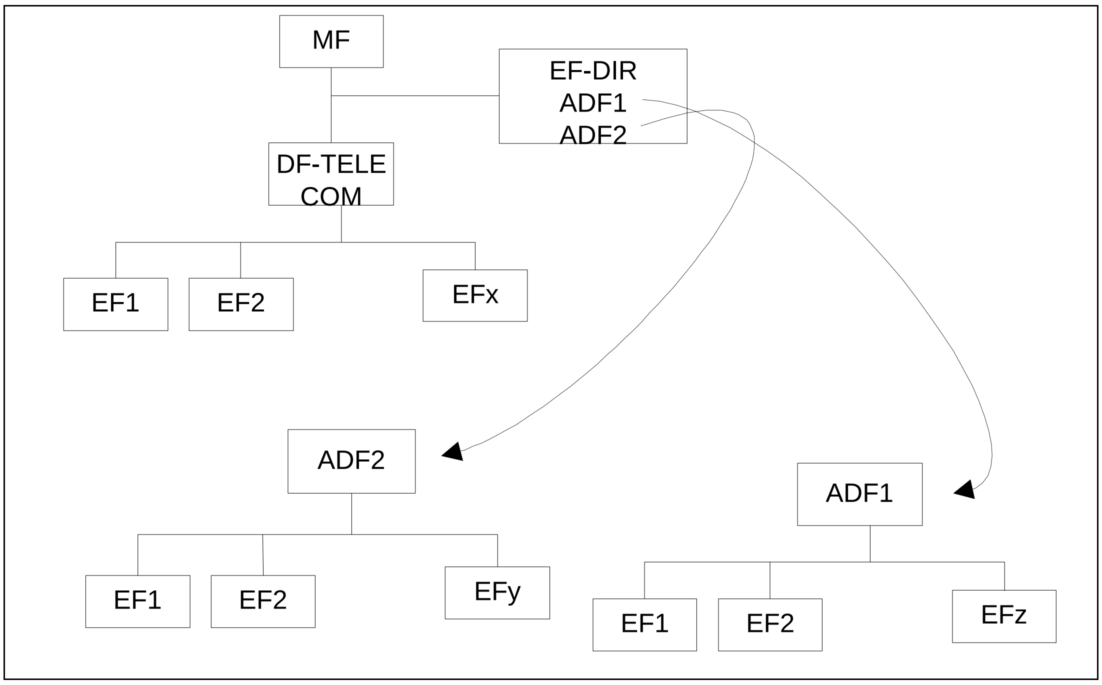


Figure 8.1: Example of an application structure

This standard does not impose any restrictions on the location of applications. All applications are uniquely identified by application identifiers that are obtained from EF_{DIR}. These application identifiers are used to select the application.

EF_{DIR} is mandatory and resides directly under the Master File. $DF_{TELECOM}$ is mandatory and resides directly under the Master File. $DF_{TELECOM}$ contains application independent information.

8.3 File types

This subclause defines the file types that applies to applications complying to this standard

8.3.1 Dedicated files

A Dedicated File (DF) allows for a functional grouping of files. It can be the parent of DFs and/or EFs. DFs are referenced by file identifiers An Application DF (ADF) is a particular DF that can be referenced by an AID. ADF contains all the DFs and EFs of an application.

8.3.2 Elementary files

8.3.2.1 Transparent EF

An EF with a transparent structure consists of a sequence of bytes. When reading or updating, the sequence of bytes to be acted upon is referenced by a relative address (offset), which indicates the start position (in bytes), and the number of bytes to be read or updated. The first byte of a transparent EF has the relative address '00 00'. The total data length is indicated in the SELECT response of the EF.

8.3.2.2 Linear fixed EF

An EF with linear fixed structure consists of a sequence of records all having the same (fixed) length. The first record is record number 1. The length of a record as well as this value multiplied by the number of records are indicated in the SELECT response of the EF.

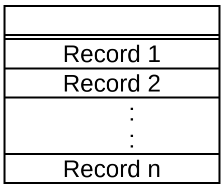


Figure 8.4: Structure of a linear fixed file

There are several methods to access records within an EF of this type:

- absolutely using the record number;
- when the record pointer is not set it shall be possible to perform an action on the first or the last record by using the NEXT or PREVIOUS mode;
- when the record pointer is set it shall be possible to perform an action on this record, the next record (unless the record pointer is set to the last record) or the previous record (unless the record pointer is set to the first record);
- by identifying a record using pattern seek starting:
  - forwards from the beginning of the file;
  - forwards from the record following the one at which the record pointer is set (unless the record pointer is set to the last record);
  - backwards from the end of the file;
  - backwards from the record preceding the one at which the record pointer is set (unless the record pointer is set to the first record).

If an action following selection of a record is aborted, then the record pointer shall remain set at the record at which it was set prior to the action.

NOTE : It is not possible, at present, to have more than 254 records in a file of this type, and each record cannot be greater than 255 bytes.

8.3.2.4 Cyclic EF

Cyclic files are used for storing records in chronological order. When all records have been used for storage, then the next storage of data shall overwrite the oldest information.

An EF with a cyclic structure consists of a fixed number of records with the same (fixed) length. In this file structure there is a link between the last record (n) and the first record. When the record pointer is set to the last record n, then the next record is record 1. Similarly, when the record pointer is set to record 1, then the previous record is record n. The last updated record containing the newest data is record number 1, and the oldest data is held in record number n.

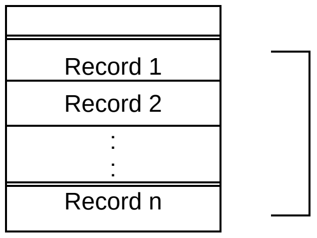


Figure 8.5: Structure of a cyclic file

For update operations only PREVIOUS record shall be used. For reading operations, the methods of addressing are Next, Previous, Current and Record Number.

After selection of a cyclic file (for either operation), the record pointer shall address the record updated or increased last. If an action following selection of a record is aborted, then the record pointer shall remain set at the record at which it was set prior to the action.

DLR NOTE: 7816-4 does not make any difference between linear and cyclic EFs regarding the record pointer after SELECT.

NOTE: It is not possible, at present, to have more than 254 records in a file of this type, and each record cannot be greater than 254 bytes.

8.4 File referencing

A file identifier (ID) is used to address or identify a specific file. The file ID consists of two bytes and shall be coded in hexadecimal notation.

File IDs shall be subject to the following conditions:

- the file ID shall be assigned at the time of creation of the file concerned;
- no two files under the same parent shall have the same ID;
- a child and any parent, either immediate or remote in the hierarchy, e.g. grandparent, shall never have the same file ID.

A path is a concatenation of file identifiers. The path begins with the identifier of the MF or of the current DF, and ends with the identifier of the file itself. If the identifier of the current DF is unknown, the reserved value '3FFF' shall be used at the beginning of the path.

A short file identifier is coded as 5 bits valued in the range from 1 to 30. No two files under the same parent shall have the same short file identifier.

A DF name is coded on 1 to 16 bytes. The DF name is the AID and shall be unique within a card.

8.5 Methods for selecting a file

After the UICC activation (as defined in clause 6.1) and the Answer To Reset (ATR), the Master File (MF) is implicitly selected and becomes the Current Directory. Each file may then be selected by using the SELECT function, using one of the 4 file referencing methods defined in sub-clause 8.4, as defined in the following sub-clauses.

8.5.1 SELECT by File Identifier Referencing

Selecting a DF, an ADF or the MF sets the Current Directory. After such a selection there is no current EF. Selecting an EF sets the current EF and the Current Directory remains the DF, ADF or MF, which is the parent of this EF. The current EF is always a child of the Current Directory.

Any application specific command shall only be operable if it is specific to the Current Directory.

The following files may be selected, by file identifier referencing, from the last selected file:

- any file which is an immediate child of the Current Directory;
- any DF or ADF which is an immediate child of the parent of the current DF;
- the parent of the Current Directory;
- the current DF or ADF;
- the MF.

This means in particular that a DF or ADF shall be selected prior to the selection of any of its EFs. All selections are made using the file ID. Figure 8.6 is an example of the logical structure for an applications conforming to this document.

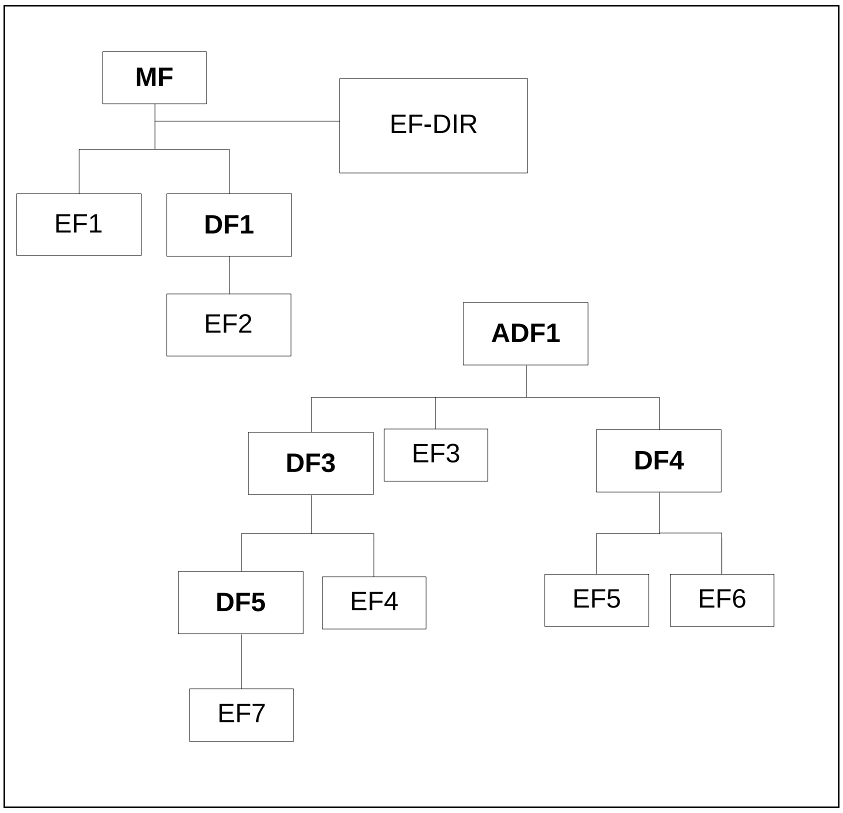


Figure 8.6: Example of a logical structure

The following table gives the valid selections for an application complying to this document for the logical structure in figure 8.6, if the file identifier referencing is used. Reselection of the last selected file is also allowed but not shown.

Table 8.1: File selection

Last selected file	Valid Selections
MF	DF1, EF1, EF-DIR
DF1	MF, EF2,
ADF1	MF, DF3, DF4, EF3
DF3	MF, ADF1, DF4, DF5, EF4
DF4	MF, ADF1, DF3, EF5, EF6
DF5	MF, DF3, EF7
EF1	MF, DF1, EF-DIR
EF2	MF, DF1
EF3	MF, ADF1, DF3, DF4
EF4	MF, ADF1, DF3, DF5,
EF5	MF, DF4, ADF1, EF6
EF6	MF, DF4, ADF1, EF5
EF7	MF, DF3, DF5

8.5.2 SELECT by Path Referencing

A file, DF or EF, may be referenced by path, as defined in clause 8.4. Table 8.2 contains examples of selection by path from figure 8.6.

It is not possible to select an ADF by path.

Table 8.2: Examples of file selection by path

Last selected file	Example Selections
MF	'MF||EF1', 'MF||EF-DIR', 'MF||DF1', 'MF||DF1||EF2'
DF3	'DF5||EF7'
DF4	'MF||DF1||EF2', 'ADF1-fID||DF3||DF5||EF7'

NOTE: at the plenary in Bonn we decided that an ADF may have a file ID. If an ADF has an ID then it is possible to make the last selection in table 8.2 otherwise it is not except if we define a default value for the ADF file ID – like '3F00' for the MF.

8.5.3 SELECT by DF Name

An Application, represented in the UICC by the AID, shall be referenced by a DF name coded on 1 to 16 bytes. Each name shall be unique within a UICC. A DF name can be used in the SELECT command to select an application.

8.5.4 Short EF Identifier

Any EF within a DF can be implicitly selected without giving a SELECT command by applying one of the following commands at the DF or ADF level and giving a short EF identifier as a part of the command:

- READ BINARY,
- UPDATE BINARY,
- READ RECORD, or
- UPDATE RECORD,
- UPDATE RECORD,
- INCREASE, or
- SEARCH BINARY.

Short EF identifier cannot be used in a path or as a file identifier e.g. in SELECT command.

8.6 Reservation of file IDs

NOTE: Did we decide to remove this chapter completely in Bonn?

In addition to the identifiers used for the files specified in the present document, the following file IDs are reserved for use by Telecom applications:

Dedicated Files:
- administrative use:
  '7F 4X', '5F1X', '5F2X'
- operational use:

'7F 10' ($DF_{TELECOM}$), '7F 20' (DF_{GSM}), '7F 21' ($DF_{DCS1800}$), '7F 22' (DF_{IS-41}), ‘7F 80’ DF_{PDC} , '7F 23' (DF_{FP-CTS}) (see GSM 11.19 [34]),and '7F 2X', where X ranges from '4' to 'F'.

- reserved under '7F10':
  '5F50' ($DF_{GRAPHICS}$)
- reserved under '7F20':
  '5F30' ($DF_{IRIDIUM}$), '5F31' ($DF_{Globalstar}$), '5F32' (DF_{ICO}), '5F33' (DF_{ACeS}), '5F3X', where X ranges from '4' to 'F' for other MSS.
  '5F40'($DF_{PCS-1900}$), '5F4Y' where Y ranges from '1' to 'F';
  '5F5X' where X ranges from '0' to 'F';
  '5F60'(DF_{CTS}), '5F6Y' where Y ranges from '1' to 'F';
  '5F70' (DF_{SoLSA}), '5F7Y' where Y ranges from '1' to 'F';
  '5FYX' where Y ranges from '8' to 'F' and X from '0' to 'F'.

Elementary files:

- administrative use:
  '6F XX' in the DFs '7F 4X'; '4F XX' in the DFs '5F 1X', '5F2X'
  '6F 1X' in the DFs '7F 10', '7F 20', '7F 21';
  '4F 1X' in all 2nd level DFs
  '2F 01', '2F EX' in the MF '3F 00';
- operational use:
  '6F 2X', '6F 3X', '6F 4X' in '7F 10' and '7F 2X';
  '4F YX', where Y ranges from '2' to 'F' in all 2nd level DFs.
  '2F 1X' in the MF '3F 00'.

In all the above, X ranges, unless otherwise stated, from '0' to 'F'.

9 Security features

Every application that conforms to this document may define security features in addition to the mandatory features defined in this clause.

9.1 File access conditions

Every EF has its own specific access condition for commands requiring access restrictions. The relevant access condition shall be fulfilled before the requested action can take place.

For each EF the access conditions for the commands READ and SEEK are identical.

The access condition levels are defined in the following table:

Table 9.1: Access condition level coding

Level	Access Condition
0	ALWays
1	PIN
2	see NOTE1
3	Reserved for Future Use
4 to 14	see NOTE2
15	NEVer

NOTE1: This level is reserved for a second PIN that may be defined by an application.
NOTE2: Allocation of these levels and the respective requirements for their fulfilment are the responsibility of the appropriate administrative authority.

The meaning of the file access conditions is as follows:

ALWAYS: The action can be performed without any restriction;

PIN: The action shall only be possible if one of the following three conditions is fulfilled:
- a correct value for the relevant PIN has been presented to the UICC during the current session;
- the PIN enabled/disabled indicator is set to "disabled";

- UNBLOCK PIN procedure has been successfully performed during the current session;

Second PIN: The action shall only be possible if one of the following two conditions is fulfilled:
- a correct second PIN value has already been presented to the application during the current session;
- UNBLOCK of the second PIN has been successfully performed during the current session;

NEVER: The action cannot be performed over the UICC/Terminal interface. The application may perform the action internally.

Condition levels are not hierarchical for a EF. For instance, correct presentation of a second PIN does not allow actions to be performed which require presentation of the PIN. A condition level which has been satisfied remains valid until the end of an application session as long as the corresponding secret code remains unblocked. That means, that after three consecutive wrong attempts, not necessarily in the same card session, the access rights previously granted by this secret code are lost immediately. A satisfied PIN condition level applies to both the application DF and $DF_{TELECOM}$.

NOTE: The wording of the third sentence depends on the decisions taken with regards to the PIN approach application or UICC session – this applies also to the last sentence.

The Terminal shall determine whether a second PIN is available by using the response to the STATUS command. If a second PIN is "not initialized" then the commands related to the second PIN, e.g. VERIFY PIN, shall not be executable.

9.2 PIN access condition

NOTE: This clause shall be updated after the T3 meeting in Korea.

The MF and every ADF shall have a child EF – EF_{PIN} - that contains a PIN for the MF or ADF. The PIN can either be enabled, disabled, blocked or unblocked.

Note: Maybe there is no need to have the EF_{PIN} as this could be interpret as implementation details.

NOTE: This ensures that each application can have it’s own PIN AC and that a master PIN AC may exist.

If the PIN at one level is enabled and the verification procedure was successful then all applications and EFs residing in the sub-tree of may be considered as having their access condition PIN fulfilled if there is no enabled PINs in the sub-tree.

It is the responsibility of the UICC to store and remember the status of the PIN for each application and file.

Table 9.2: Example of PIN Access control

PINs	EFs with PIN AC satisfied
MFPIN	EF1, EF2, EF3, EF4, EF5 and EF6
PIN1	EF2, EF3, and EF4
PIN2	EF3
PIN3	EF4
PIN4	EF5 and EF6
PIN5	EF6
MFPIN and !PIN1	EF1, EF5 and EF6

PIN1 and !PIN2	EF2 and EF4
NOTE1: ! means the PIN is enabled but not verified. NOTE2: All other than the listed PINs are enabled but not verified.	

NOTE: exchange e.g. PIN1 with PINa.

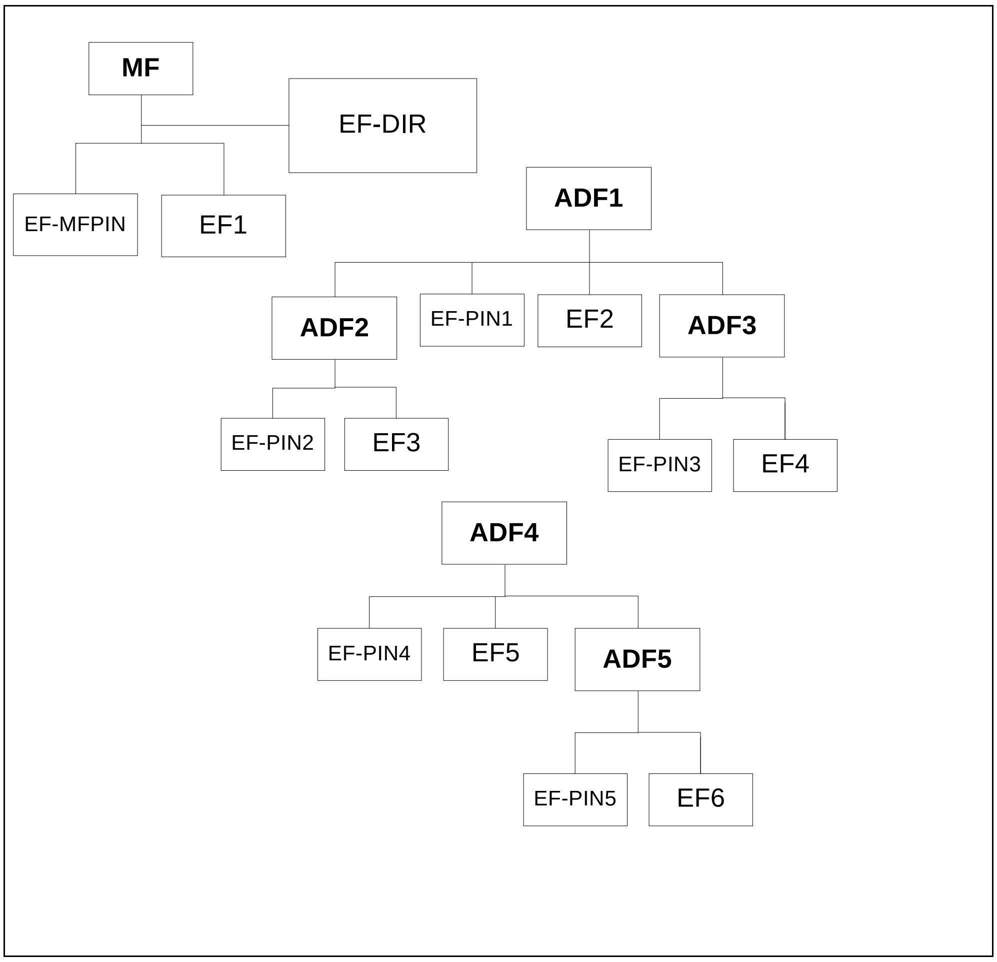


Figure 9.1: Example of the PIN access control structure

10 Structure of commands and responses

This clause defines the command and response APDU's supported by the UICC.

A function consists of a command and the associated response.

10.1 Definitions

The following definitions for the command and response APDU's applies:

Coding

All lengths are presented in bytes, unless otherwise stated. Each byte is represented by bit b8 to b1, where b8 is the most significant bit (MSB) and b1 is the least significant bit (LSB). In each representation the leftmost bit is the MSB.

RFU

In UICC all bytes which are RFU shall be set to '00' and RFU bits to 0. Where the GSM and/or USIM application exists on an UICC or is built on a generic telecommunications card (e.g. TE9) then other values may apply for other than GSM or UMTS applications. The values will be defined in the appropriate specifications for such cards and applications. These bytes and bits shall not be interpreted by a Terminal in a GSM or UMTS session.

10.2 Command APDU Structure

This clause states a generic structure of an application protocol data unit – APDU – that is used by the application protocol on the top of the transmission protocol for sending a command to the card.

A command APDU consists of a header and a body part. The contents of the command APDU are depicted in table12.1 where the header consists of the CLA, INS, P1 and P2 bytes that are mandatory for a command APDU and an optional body part that can contain the Lc, Data and Le. Parameters are further explained in the following subclauses.

Table 10.1: Contents of Command APDU

Code	Length	Description	Grouping
CLA	1	Class of instruction	Header
INS	1	Instruction code	
P1	1	Instruction parameter 1	
P2	1	Instruction parameter 2	
Lc	0 or 1	Number of bytes in the command data field	Body
Data	Lc	Command data string	
Le	0 or 1	Maximum number of data bytes expected in response of the command	

Four cases of C-APDU structure are possible as defined Table 0.1:

Case	Structure
1	CLA INS P1 P2
2	CLA INS P1 P2 Le
3	CLA INS P1 P2 Lc Data
4	CLA INS P1 P2 Lc Data Le

Table 0.1: Cases of C-APDU's

10.2.1 Coding of Class Byte

The most significant nibble of the Class byte (b8-b5) codes the type of the command as stated in table 10.2. Bits b4 and b3 are used for indication of secure messaging format (see table 10.3). Bits b2 and b1 indicates the logical channel used. Logical channels are numbered from 0 to 3. If the card supports the logical channel mechanism, the maximum number of available logical channels is indicated in the card capabilities data object of historical bytes of an ATR (refer to ISO/IEC 7816-4 [2]). If the card capabilities data object is missing, logical channel b2=b1=0 is supported only.

Table 10.2: Coding of Class Byte

B8	b7	b6	b5	b4	b3	b2	b1	Value	Meaning
0	0	0	0	-	-	-	-	'0X'	The coding is according to 7816-4 [2]
1	0	1	0	-	-	-	-	'AX'	Coded as 7816-4 [2] unless stated otherwise
-	-	-	-	x	x	-	-	-	Secure Messaging indication (see table 12.3)
-	-	-	-	-	-	X	x	-	Logical channel number (see 10.4)

Table 10.3: Coding of Security Messaging Indication

b4	b3	Meaning
0	0	No SM used between Terminal and card
1	x	Secure messaging according to ISO/IEC 7816-4 [2] used
1	0	Command header not authenticated
1	1	Command header authenticated

Editor's note: Is b4=0 and b3=1 needed (proprietary SM format)? Now it is RFU. It might not be wise to allow everything in this spec.

10.2.2 Coding of Instruction Byte

Table 10.4 depicts coding of instruction byte of the commands.

For telecom applications (e.g. GSM and USIM) the class byte shall always be coded as ‘AX’.

Editor's note: Is the last sentence valid, i.e. shall we mandate that the instruction byte is coded as ‘AX’ for telecom applications?

Editor’s note: For USIM application the CLA can be also some other value. It might be better to leave the CLA column out of the following table, since it depends on the application. Another way is to add the application in the table, as follows:

Table 10.4: Coding of Instruction Byte of the Commands for a telecom application

COMMAND 7816-4	COMMAND(GSM 11.11)	CLA (GSM)	CLA (3GPP)	INS
SELECT FILE	SELECT FILE	AX	'AX' or '0x'	'A4'
NA	STATUS	AX	Only AX is valid	'F2'
READ BINARY	READ BINARY	AX		'B0'
UPDATE BINARY	UPDATE BINARY	AX		'D6'
READ RECORD	READ RECORD	AX		'B2'
UPDATE RECORD	UPDATE RECORD	AX		'DC'
NA	SEEK	AX	Only AX is valid	'A2'
NA	INCREASE	AX	Only AX is valid	'32'
VERIFY	VERIFY	AX		'20'
CHANGE PIN	CHANGE PIN	AX		'24'
DISABLE PIN	DISABLE PIN	AX		'26'
ENABLE PIN	ENABLE PIN	AX		'28'
UNBLOCK PIN	UNBLOCK PIN	AX		'2C'
DEACTIVATE FILE	INVALIDATE	AX		'04'
ACTIVATE FILE	REHABILITATE	AX		'44'
INTERNAL AUTHENTICATE	RUN GSM ALGORITHM	AX		'88'
GET RESPONSE	GET RESPONSE	AX		'C0'
	TERMINAL PROFILE	AX		'10'
ENVELOPE	ENVELOPE	AX		'C2'
NA	FETCH	AX	Only AX is valid	'12'
NA	TERMINAL RESPONSE	AX	Only AX is valid	'14'
MANAGE CHANNEL	NA	-		70

Editor’s note: an AP to card manufacturers to investigate the implications of having mixed ‘AX’ or ‘0X’ in a session.

Secondly we should decide on using ‘0X’ in the future is the command is ISO compliant.

10.2.3 Coding of Parameter Bytes

The value of the parameters P1 and P2 depends on the command. If the parameter is not used, the value is set to '00'. Coding of the parameter bytes is presented in the command definition sections.

10.2.4 Coding of Lc Byte

The number of data bytes present in the data field of the command APDU is presented in the parameter Lc. Lc is optional, in the command APDU, however if the Lc is present in the command APDU, data field consists of the Lc subsequent bytes.

Table 10.5: Coding of Lc

Range	Byte 1
1 – 255	Binary value

10.2.5 Coding of Data Part

When present in a command or response APDU the data field consists of a string of proprietary or TLV coded data.

10.2.6 Coding of Le Byte

The maximum number of bytes expected in the data part of the response APDU is presented in the parameter Le, which is optional meaning that if the Terminal does not expect any data in the response APDU Le is absent from the command APDU. However, if Le is present in the command APDU, the data field of the response APDU is expected to consist of the Le bytes.

Le set to ‘00’ indicates that the Terminal expects to receive at most the maximum number of bytes, i.e. 256, in the response ADPU. The UICC may return any number of bytes in the range 0 to 256.

Table 10.6: Coding of Le

Range	Byte 1
1 – 256	Binary value

10.3 Response APDU Structure

The response APDU consists of an optional data field and a mandatory status part divided into two bytes; SW1 and SW2. The parameter Le of the command APDU indicates the length of the data part of the response APDU. The structure of the response APDU is shown in table 10.7.

Table 10.7: Contents of Response APDU

Code	Length	Description
Data	Le (in command APDU)	Response data string
SW1	1	Status byte 1
SW2	1	Status byte 2

Coding of SW1 and SW2 is presented in 10.3.1.

10.3.1 Status Conditions Returned by the UICC

Status of the card after processing of the command is coded in the status bytes SW1 and SW2. This subclause specifies coding of the status bytes in the following tables.

10.3.1.1 Responses to commands which are correctly executed

SW1	SW2	Description
'90'	'00'	- normal ending of the command
'91'	'XX'	- Normal ending of the command, with extra information from the proactive UICC containing a command for the Terminal. Length 'XX' of the response data

10.3.1.2 Responses to commands which are postponed

SW1	SW2	Error description
'93'	'00'	- SIM Application Toolkit is busy. Command cannot be executed at present, further normal commands are allowed.

10.3.1.3 Memory management

SW1	SW2	Error description
'63'	'Cx'	- command successful but after using an internal update retry routine 'X' times
'65'	'81'	- memory problem

10.3.1.4 Referencing management

SW1	SW2	Error description
'69'	'86'	- no EF selected
'94'	'02'	- out of range (invalid address)
'6A'	'82'	- file ID not found - pattern not found
'69'	'81'	- file is inconsistent with the command

10.3.1.5 Security management

SW1	SW2	Error description
'98'	'02'	- no PIN initialised
'98'	'04'	- access condition not fulfilled - unsuccessful PIN verification, at least one attempt left - unsuccessful UNBLOCK PIN verification, at least one attempt left - authentication failed (see note)
'98'	'08'	- in contradiction with PIN status
'98'	'10'	- in contradiction with invalidation status
'98'	'40'	- unsuccessful PIN verification, no attempt left - unsuccessful UNBLOCK PIN verification, no attempt left - PIN blocked - UNBLOCK PIN blocked
'98'	'50'	- increase cannot be performed, Max value reached

10.3.1.6 Additional Return Values of USIM Applications

SW1	SW2	Error description
'62'	'00'	- No information given, processing completed
'62'	'81'	- Part of returned data may be corrupted
'62'	'82'	- End of file/record reached before reading Le bytes
'62'	'83'	- Selected file invalidated
'62'	'84'	- FCI not formatted according to chapter **9.**4.1??
'69'	'82'	- Security status not satisfied
'69'	'84'	- Referenced data invalidated
'69'	'85'	- Conditions of use not satisfied
'6A'	'81'	- Function not supported
'6A'	'83'	- Record not found
'6A'	'84'	- Not enough memory space in the file
'6A'	'85'	- Lc inconsistent with TLV structure
'6A'	'86'	- Incorrect parameters P1-P2
'6A'	'87'	- Lc inconsistent with P1-P2
'6A'	'88'	- Referenced data not found

10.3.2 Status Words of the Commands

The following table shows for each command the possible status conditions returned (marked by an asterisk *).

Table 10.8: Commands and status words

Status Words	SELECT	STATUS	UPDATE BINARY	UPDATE RECORD	READ BINARY	READ RECORD	SEARCH RECORD	INCREASE	VERIFY PIN	CHANGE PIN	DISABLE PIN	ENABLE PIN	UNBLOCK PIN	DEACTIVATE FILE	ACTIVATE FILE	INTERNAL AUTHENTICATE	GET RESPONSE	TERMINAL PROFILE	ENVELOPE	FETCH	TERMINAL RESPONSE	MANAGE CHANNEL
90 00		*	*	*	*	*	*		*	*	*	*	*	*	*		*	*	*	*	*	*
91 XX		*	*	*	*	*			*	*	*	*	*	*	*		*	*	*		*	
93 00																			*			
63 CX			*	*				*	*	*	*	*	*	*	*			*	*		*	
65 81	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	
69 86			*	*	*	*	*	*						*	*							
94 02				*		*																
6A 82	*		*	*	*	*	*							*	*							
69 81			*	*	*	*	*	*								*						
98 02									*	*	*	*	*									
69 82			*	*	*	*	*	*	*	*	*	*	*	*	*	*						
98 08									*	*	*	*	*									
98 10			*	*	*	*	*	*						*	*							
98 40									*	*	*	*	*									
98 50																						
67 XX	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
6B XX	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
6D XX																						
6E XX	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
6F XX	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
62 81					*	*																
62 83	*																					
62 82					*	*																
62 84	*																					
62 00																						*
69 84																*						
69 85																*						
69 86			*	*	*																	
6A 81	*		*	*	*	*																
6A 83						*																
6A 84				*																		
6A 85				*																		
6A 86	*															*						
6A 87	*																					
6A 88																*						

DLR NOTE: The meaning of status 67XX, 6BXX, 6DXX, 6EXX, 6FXX is not defined.

DLR NOTE: How will the USIM choose:
- between 67XX and 6CXX for SELECT and READ BINARY,
- or between 6200 and 9000 for MANAGE CHANNEL,
- etc ?

The responses '91 XX', and '93 00' can only be given by an UICC supporting SIM Application Toolkit to a Terminal also supporting SIM Application Toolkit.

10.4 Logical channels

This clause defines the logical channel concept.

Editor’s note: more introductional text needed.

10.4.1 Logical channel basics

A logical channel works like a link between an application residing in the card and an application in the Terminal. Each logical channel is assigned a unique channel number, indicated in b1 and b2 of the class byte, that is used by the Terminal to identify to which application a command is directed.

The basic logical channel ‘00’ shall always exist, and is implicitly selected after ATR.

If other than the basic logical channel exists the number (at most 4) is indicated in the card capabilities sent as part of the historical bytes in the ATR.

Editor’s note: a reference to the ATR chapter/standard is needed here!

There shall be independence between the activity on each logical channel.

There can be more than one logical channel between the Terminal and an application.

At any time there can only be one response pending for all logical channels, this implies that when one command has been sent on one logical channel there can only be sent a command when the first command has received its response.

When a logical channel has been opened it remains open for the rest of the card session unless it is explicitly closed as defined in 10.5.3.

A logical channel number can only be assigned by an application residing in the Terminal.

Editor’s note: should it be stated that there is a relation between the logical channel (conceived as a link between applications residing in the UICC and the Terminal) and a PIN verification, i.e. the PIN status is set to not verified when the application residing in the Terminal sends a SELECT command to the UICC selecting another application (DF). As far as the editor knows there is no such requirement or definition in 7816-4!

DLR NOTE: The issue of PIN sharing is just one of the to be defined security status aspects.

10.4.2 Opening of logical channels

A logical channel can be opened in one of the following ways:

- by sending a SELECT command to the UICC indicating a logical channel, in b2 and b1 in the class byte, that is not yet opened;
- by sending a MANAGE CHANNEL command to the UICC indicating that a new channel is to be opened.

NOTE: The basic channel is always open and thus it can never be opened or closed.

10.4.3 Closing of logical channels

A logical channel is closed by sending a MANAGE CHANNEL command to the UICC explicitly stating that a specific logical channel shall be closed.

11 Generic Commands

This subclause lists the basic command and response APDU formats that are supported by applications residing on a UICC.

Commands used to manage an application are not defined in this standard.

In the subsequent subclauses only the response data is listed, for the coding of the status words see 10.4.2.

11.1.1 SELECT

11.1.1.1 Functional description

This function selects a file according to the methods described in clause 8.4. After a successful selection the record pointer in a linear fixed file is undefined. The record pointer in a cyclic file shall address the last record, which has been updated or increased.

Input:
- file ID, application ID, path or empty.

Output:
- if the selected file is the MF, a DF or an ADF:
  file ID, total memory space available, PIN enabled/disabled indicator, PIN status and other application specific data;
- if the selected file is an EF:
  file ID, file size, access conditions, invalidated/not invalidated indicator, structure of EF and length of the records in case of linear fixed structure or cyclic structure.

Note I: The select method with application ID, path or file name is only applicable for 3G systems.

Note II: Selection with a DF-filename can only be used if the DF-file is an application DF e.g. DF_{GSM}

Editor's note: It shall be possible to select a file by Application ID's, path and name in 3G. This has been added on the input to the command. Shall the application ID etc be sent in the response as well?

11.1.1.2 Command Parameters and Data

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	Selection control, see table 11.1
P2	Selection control, see table 11.2
Lc	Length of subsequent data field or empty
Data	AID, file ID, DF name, or path to file, according to P1
Le	Empty, '00', or maximum length of data expected in response

Table 11.1: Coding of P1

b8	b7	b6	b5	b4	b3	B2	b1	Meaning
0	0	0	0	0	0	0	0	Select DF, EF or MF by file id
0	0	0	0	0	0	0	1	Select child DF of the current DF
0	0	0	0	0	0	1	1	Select parent DF of the current DF
0	0	0	0	0	1	0	0	Selection by DF name – see NOTE
0	0	0	0	1	0	0	0	Select by path from MF
0	0	0	0	1	0	0	1	Select by path from current DF
NOTE: This is selection by AID – see chapter 8.2								

Table 11.2: Coding of P2

b8	b7	b6	B5	b4	b3	b2	b1	Meaning
0	0	0	0	0	0	0	0	Return FCI template.
NOTE: The FCI template shall be included in the response data of USIM applications								

11.1.1.3 Response Data in case of MF, DF or ADF

Table 11.3: Response with FCI Template

Byte(s)	Description	Status
1	FCI template tag = '6F'	M
2	Length of FCI template	M
3	DF name tag = '84'	C
4	Length of the DF name	C
5 – 20	DF name	C
21	Proprietary information tag = ‘85’	M
22	Length of response information	M
23 – X	Response information – see NOTE	M
NOTE: The content of the proprietary information shall be defined in application specific documents M: Mandatory C: IF a Application DF is selected this field is mandatory ELSE not present		

DLR NOTE: This is not the kind of answer to be returned to a 2G terminal. Something is missing to fulfil the 21.111 requirements on 3GPP/GSM interworking.

DLR NOTE: Compatibility problem in clause 8.3, for record pointer after SELECT.

DLR NOTE: The card shall return more information. Some TLVs can be added as defined in ISO 7816-4: number of data bytes in the file (tag = '80'), file descriptor byte (tag = '82'), file identifier (tag = '83'), security attributes (tag = '87')...

11.1.1.4 Response Data in case of an EF

Table 11.5 Response with FCI Template

Byte(s)	Description	Status
1	FCI template tag = '6F'	M
2	Length of FCI template	M
3	Proprietary information tag = ‘85’	M
4	Length of response information	M
5 – X	Response information – see NOTE1	M
5+X+1	Short File Identifier tag=‘88’	O
5+X+2	Length = 1	O
5+X+3	SFI value	O
NOTE1: The content of the proprietary information shall be defined in application specific documents. NOTE2: The TLV object is coded according to table 13.2 M: Mandatory O: Optional		

DLR NOTE: The card shall return more information. Some TLVs can be added as defined in ISO 7816-4: number of data bytes in the file (tag = '80'), file descriptor byte (tag = '82'), file identifier (tag = '83'), security attributes (tag = '87')...

11.1.2 STATUS

11.1.2.1 functional description

This function returns information concerning the current directory.

Input:

- none.

Output:

- ID of the current directory, total memory space available, PIN enabled/disabled indicator, PIN status and other system specific data (identical to SELECT).

11.1.2.2 Command parameters

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	00
P2	00
Le	Maximum length of data expected in response

Response parameters and data are identical to the response parameters and data of the SELECT command in case of MF or DF.

11.1.3 READ BINARY

11.1.3.1 Functional description

This function reads a string of bytes from the current transparent EF. This function shall only be performed if the READ access condition for this EF is satisfied.

If the command is applied to an EF without transparent structure then the command shall be aborted.

When the command contains a valid short file identifier, this sets the current directory.

Input:

- relative address and the length of the string.

Output:

- string of bytes.

Editors note: Short file identifier, is that included in the concept of relative address?

DLR NOTES:

- In 7816-4, the relation between the 30 SFIs and the corresponding EFs is not specified. It may be interesting to draw the attention of the reader on the fact that, even if SFIs are supported, there may be an SFI allocated for an EF of an Application and no SFI available for an equivalent EF of another similar application.

- Question: If the availability of an SFI for an application cannot be guaranteed, will the selection by SFI be used by the terminals at the end ?

11.1.3.2 Command parameters:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	See table 11.6
P2	Offset low
Lc	Not present
Data	Not present
Le	Number of bytes to be read

Table 11.6: Coding of P1

b8	B7	b6	B5	b4	b3	b2	B1	Meaning
0	x	x	x	x	x	x	X	b7-b1 is the offset to the first byte to read – P2 is the low part of the offset
1	0	0	X	X	X	X	X	SFI referencing used, b1-b5 are the SFI and P2 is the offset to the first byte to read

Response data:

Byte(s)	Description	Length
1 – Le	Data to be read	Le

11.1.4 UPDATE BINARY

11.1.4.1 Functional parameters

This function updates the current transparent EF with a string of bytes. This function shall only be performed if the UPDATE access condition for this EF is satisfied. An update can be considered as a replacement of the string already present in the EF by the string given in the update command.

Input:
- relative address and the length of the string;
- string of bytes.

Output:
- none

11.1.4.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	See table 11.6
P2	Offset low
Lc	Length of the subsequent data field
Data	String of data to be updated
Le	Not present

Coding of parameter P1 and P2 are identical to the coding of P1 and P2 in the READ BINARY command.

11.1.5 READ RECORD

11.1.5.1 Functional description

This function reads one complete record in the current linear fixed or cyclic EF. The record to be read is described by the modes below. This function shall only be performed if the READ access condition for this EF is satisfied. The record pointer shall not be changed by an unsuccessful READ RECORD function.

Four modes are defined:

CURRENT: The current record is read. The record pointer is not affected.

ABSOLUTE: The record given by the record number is read. The record pointer is not affected.

NEXT: The record pointer is incremented before the READ RECORD function is performed and the pointed record is read. If the record pointer has not been previously set within the selected EF, then READ RECORD (next) shall read the first record and set the record pointer to this record.

If the record pointer addresses the last record in a linear fixed EF, READ RECORD (next) shall not cause the record pointer to be changed, and no data shall be read.

If the record pointer addresses the last record in a cyclic EF, READ RECORD (next) shall set the record pointer to the first record in this EF and this record shall be read.

PREVIOUS: The record pointer is decremented before the READ RECORD function is performed and the pointed record is read. If the record pointer has not been previously set within the selected EF, then READ RECORD (previous) shall read the last record and set the record pointer to this record.

If the record pointer addresses the first record in a linear fixed EF, READ RECORD (previous) shall not cause the record pointer to be changed, and no data shall be read.

If the record pointer addresses the first record in a cyclic EF, READ RECORD (previous) shall set the record pointer to the last record in this EF and this record shall be read.

DLR NOTE: Impact of SFIs on record pointer to be specified.

Input:
- mode, record number (absolute mode only) and the length of the record.

Output:
- the record.

11.1.5.2 Command parameters:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	Record number
P2	Mode, see table 11.7
Lc	Not present
Data	Not present
Le	Number of bytes to be read

Table 11.7: Coding of P2

b8	b7	b6	B5	b4	b3	b2	B1	Meaning
0	0	0	0	0	-	-	-	Currently selected EF
x	x	x	x	x	-	-	-	Short EF identifier (from 1 to 30)
-	-	-	-	-	0	1	0	Next record
-	-	-	-	-	0	1	1	Previous record
-	-	-	-	-	1	0	0	Absolute/ current mode, the record number is given in P1 with P1='00' denoting the current record

For the modes "next" and "previous" P1 has no significance and shall be set to '00' by the **TE**. To ensure backward compatibility, the UICC shall not interpret the value given by the **TE**.

DLR NOTE: If P1 is not 0, a card compliant with 7816-4 would perform a read via record identifier, which is not compatible with GSM. The remark above leads to incompatibility between 31.101 and 7816-4.

Response data:

Byte(s)	Description	Length
1 – Le	Data to be read	Le

DLR NOTE: 7816-4 allows Le=0. It should be more explicit that 31.101does not interpret ‘00’ as “whole record”.

11.1.6 UPDATE RECORD

11.1.6.1 Functional description

This function updates one specific, complete record in the current linear fixed or cyclic EF. This function shall only be performed if the UPDATE access condition for this EF is satisfied. The UPDATE can be considered as a replacement of the relevant record data of the EF by the record data given in the command. The record pointer shall not be changed by an unsuccessful UPDATE RECORD function.

The record to be updated is described by the modes below. Four modes are defined of which only PREVIOUS is allowed for cyclic files:

CURRENT: The current record is updated. The record pointer is not affected.

ABSOLUTE: The record given by the record number is updated. The record pointer is not affected.

NEXT: The record pointer is incremented before the UPDATE RECORD function is performed and the pointed record is updated. If the record pointer has not been previously set within the selected EF, then UPDATE RECORD (next) shall set the record pointer to the first record in this EF and this record shall be updated. If the record pointer addresses the last record in a linear fixed EF, UPDATE RECORD (next) shall not cause the record pointer to be changed, and no record shall be updated.

PREVIOUS: For a linear fixed EF the record pointer is decremented before the UPDATE RECORD function is performed and the pointed record is updated. If the record pointer has not been previously set within the selected EF, then UPDATE RECORD (previous) shall set the record pointer to the last record in this EF and this record shall be updated. If the record pointer addresses the first record in a linear fixed EF, UPDATE RECORD (previous) shall not cause the record pointer to be changed, and no record shall be updated.

For a cyclic EF the record containing the oldest data is updated, the record pointer is set to this record and this record becomes record number 1.

Input:
- mode, record number (absolute mode only) and the length of the record;
- the data used for updating the record.

Output:

- none.

11.1.6.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	Record number
P2	Mode, see table 11.7
Lc	Length of the subsequent data field
Data	String of data to be updated
Le	Not present

Coding of parameter P2 is identical to the coding of P2 in READ RECORD command.

For the modes "next" and "previous" P1 has no significance and shall be set to '00' by the Terminal. To ensure backward compatibility, the UICC shall not interpret the value given by the Terminal.

11.1.7 SEARCH RECORD

Editor's note: this command will be updated according to the outcome of the 7816-9 voting currently ongoing. This is because there are some enhancement under discussion that may come into 7816-9.

11.1.7.1 Functional description

This function searches through a linear fixed or cyclic EF to find record(s) containing a specific pattern. This function shall only be performed if the READ access condition for this EF is satisfied. The search starts:

- either at the first byte of the record(s) (simple search) or
- from a given offset in the record(s) or
- from the first occurrence of a given byte in the record(s)

The response is either empty or contains the, up to the Le specified number of, record number(s) of the records that matches the search in the selected EF.

Input:
- search mode (simple/enhanced);
- offset,
- pattern;

Output:
- either none, if Le is empty or no matches where found, or
- at most the number of record(s) number(s) defined in Le

11.1.7.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	Record number
P2	See table 11.X
Lc	Length of the subsequent data field
Data	Offset indication followed by search string
Le	Empty or maximum length of response data

Table 11.X: Coding of P2

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	0	0	-	-	-	Currently selected EF
X	X	X	X	X	-	-	-	Short File Identifier
1	1	1	1	1	-	-	-	RFU
-	-	-	-	-	0	X	X	RFU – see NOTE
-	-	-	-	-	1	X	X	Usage of P1 as a record number
-	-	-	-	-	1	0	0	Start forward search form record indicated in P1
-	-	-	-	-	1	0	1	Start backward search form record indicated in P1
-	-	-	-	-	1	1	0	Enhanced search – see table 11.X+1
-	-	-	-	-	1	1	1	Proprietary
NOTE: This value is reserved by 7816-9 [8]								

Table 11.X+1: Coding of the first byte in the data field in enhanced mode.

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	0	-	-	-	-	RFU
-	-	-	-	0	-	-	-	Offset, the subsequent byte indicates the absolute position within the record form where the search starts
-	-	-	-	1	-	-	-	Offset, indicated as a character. The character (first occurrence) within the record after which the search starts is indicated in the subsequent byte
-	-	-	-	-	0	X	X	RFU – see NOTE
-	-	-	-	-	1	X	X	Usage of value in first byte as a record number
-	-	-	-	-	1	0	0	Start forward search form record indicated in first data byte
-	-	-	-	-	1	0	1	Start backward search form record indicated in first data byte
-	-	-	-	-	1	1	0	Start forward search from next record
-	-	-	-	-	1	1	1	Start backward search form previous record
NOTE: This value is reserved by 7816-9 [8]								

Response data:

Byte(s)	Description	Length
0 – Le	Record number(s)	Le
NOTE: If Le is empty no record numbers will be returned		

11.1.8 INCREASE

11.1.8.1 Functional description

This function adds the value given by the Terminal to the value of the last increased/updated record of the current cyclic EF, and stores the result into the oldest record. The record pointer is set to this record and this record becomes record number 1. This function shall be used only if this EF has an INCREASE access condition assigned and this condition is fulfilled. The selected application shall not perform the increase if the result would exceed the maximum value of the record (represented by all bytes set to 'FF').

Input:
- value to be added.

Output:
- value of the increased record;
- value which has been added.

11.1.8.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	See table 11.X+2
P2	'00'
Lc	Length of the subsequent data field
Data	Value to be added
Le	Length of the response data

Table 11.X+2: Coding of P1

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
1	0	0	X	X	X	X	X	SFI referencing used, b1-b5 are the SFI
NOTE: All other values are RFU								

Response data:

Byte(s)	Description	Length
1 – X	Value of the increased record	X
X+1 – X+Lc	Value which has been added	Lc

11.1.9 VERIFY PIN

11.1.9.1 Functional description

This function verifies the PIN presented by the Terminal by comparing it with the relevant one stored in the selected application. The verification process is subject to the following conditions being fulfilled:

- PIN is not disabled;
- PIN is not blocked.

If the access condition for a function to be performed on the last selected file is PIN, then a successful verification of the relevant PIN is required prior to the use of the function on this file unless the PIN is disabled.

If the PIN presented is correct, the number of remaining PIN attempts for that PIN shall be reset to its initial value 3.

If the PIN presented is false, the number of remaining PIN attempts for that PIN shall be decremented. After 3 consecutive false PIN presentations, not necessarily in the same card session, the respective PIN shall be blocked and the access condition can never be fulfilled until the UNBLOCK PIN function has been successfully performed on the respective PIN.

Input:
- indication PIN.

Output:
- none.

11.1.9.2 Command parameters:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	'00'
P2	Qualifier, see table 11.8
Lc	Length of the subsequent data field = '08'
Data	PIN value
Le	Not present

Table 11.8: Coding of P2

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	-	-	-	-	-	Global reference data (e.g. card password)
1	0	0	-	-	-	-	-	Specific reference data (e.g. DF password)
-	0	0	x	x	x	x	X	PIN number

DLR NOTES:

- We do not need b8=1 for compatibility with ISO. To be checked when the PIN model is stable whether the global or specific level are meaningful. Moreover, if we decide to include such a table, we need to define the use and coding of card or DF passwords.

- We of course need to specify the range for PIN numbers, even if there may be some application dependent aspects.

Five least significant bits of parameter P2 specify the PIN number. The following values are reserved for backward compatibility:

- 'X1' = PIN;
- 'X2' = PIN2.

For GSM application bit8 of parameter P2 shall be set to '0'.

Command data:

Byte(s)	Description	Length
1 - 8	PIN value	8

11.1.10 CHANGE PIN

11.1.10.1 Functional description

This function assigns a new value to the relevant PIN subject to the following conditions being fulfilled:

- PIN is not disabled;
- PIN is not blocked.

The old and new PIN shall be presented.

If the old PIN presented is correct, the number of remaining PIN attempts for that PIN shall be reset to its initial value 3 and the new value for the PIN becomes valid.

If the old PIN presented is false, the number of remaining PIN attempts for that PIN shall be decremented and the value of the PIN is unchanged. After 3 consecutive false PIN presentations, not necessarily in the same card session, the respective PIN shall be blocked and the access condition can never be fulfilled until the UNBLOCK PIN function has been performed successfully on the respective PIN.

Input:
- indication of PIN, old PIN, new PIN.

Output:
- none.

11.1.10.2 Command parameters:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	‘00’
P2	PIN number (TBD)
Lc	'10'
Data	Old PIN value, new PIN value
Le	Not present

DLR NOTE: “Change PIN” is named “exchange reference data” in 7816-9.

Coding of P2 is identical to the P2 of VERIFY PIN command.

Byte(s)	Description	Length
1 – 8	Old PIN value	8
9 – 16	New PIN value	8

11.1.11 DISABLE PIN

11.1.11.1 Functional description

The successful execution of this function has the effect that files protected by PIN are now accessible as if they were marked "ALWAYS". The function DISABLE PIN shall not be executed by the selected application when PIN is already disabled or blocked.

NOTE: Every application must specify whether this function is applicable to all PIN's defined for the application.

If the PIN presented is correct, the number of remaining PIN attempts shall be reset to its initial value 3 and PIN shall be disabled.

If the PIN presented is false, the number of remaining PIN attempts shall be decremented and PIN remains enabled. After 3 consecutive false PIN presentations, not necessarily in the same card session, PIN1 shall be blocked and the access condition can never be fulfilled until the UNBLOCK PIN function has been successfully performed on PIN.

Input:
- PIN.

Output:
- none.

Editor's note: Multi-applications, an application needs to be selected before? Only apply this for PIN (for what application)?

11.1.11.2 Command parameters:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	'00'
P2	PIN number (TBD)
Lc	'08'
Data	P1= '00': Data field contains the PIN P1= '01': Data field empty
Le	Not present

Table 11.9: Coding of P2

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	-	-	-	-	-	Global reference data (e.g. card password)
1	0	0	-	-	-	-	-	Specific reference data (e.g. DF password)
-	0	0	x	x	x	X	X	PIN number

Editor's note: It is to be studied what PIN's can be disable

DLR NOTE: The allowed range for PIN number has to be specified in 31.101, but some applications could specify different requirements in their dedicated part.

Command data in case P1 = '00':

Byte(s)	Description	Length
1 - 8	PIN value	8

11.1.12 ENABLE PIN

11.1.12.1 Functional description

This function may only be applied to PIN. It is the reverse function of DISABLE PIN. The function ENABLE PIN shall not be executed by the SIM when PIN is already enabled or blocked.

NOTE: Every application must specify whether this function is applicable to all PIN’s defined for the application.

If the PIN presented is correct, the number of remaining PIN attempts shall be reset to its initial value 3 and PIN shall be enabled.

If the PIN presented is false, the number of remaining PIN attempts shall be decremented and PIN remains disabled. After 3 consecutive false PIN presentations, not necessarily in the same card session, PIN shall be blocked and may optionally be set to "enabled". Once blocked, the PIN can only be unblocked using the UNBLOCK PIN function. If the PIN is blocked and "disabled", the access condition shall remain granted. If the PIN is blocked and "enabled", the access condition can never be fulfilled until the UNBLOCK PIN function has been successfully performed on PIN.

Editor’s note: Multiple applications: Doesn’t this command require that an application is selected before this command can be executed? How will this be solved in the multiple application case? Only apply this for PIN, what application?

Input:
- PIN.

Output:
- none.

11.1.12.2 Command parameters:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	'00'
P2	PIN number (TBD)
Lc	‘08’
Data	PIN value
Le	Not present

Table 11.10: Coding of P2

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	-	-	-	-	-	Global reference data (e.g. card password)
1	0	0	-	-	-	-	-	Specific reference data (e.g. DF password)
-	0	0	x	x	x	X	X	PIN number

Editor’s note: It is to be studied what PIN's can be disabled

Command data:

Byte(s)	Description	Length
1 - 8	PIN value	8

11.1.13 UNBLOCK PIN

11.1.13.1 Functional description

The command unblocks a PIN, which has been blocked by 3 consecutive wrong PIN presentations. This function may be performed whether or not the relevant PIN is blocked.

If the UNBLOCK PIN presented is correct, the value of the PIN, presented together with the UNBLOCK PIN, is assigned to that PIN, the number of remaining UNBLOCK PIN attempts for that UNBLOCK PIN is reset to its initial value 10 and the number of remaining PIN attempts for that PIN is reset to its initial value 3. After a successful unblocking attempt the PIN is enabled and the relevant access condition level is satisfied.

If the presented UNBLOCK PIN is false, the number of remaining UNBLOCK PIN attempts for that UNBLOCK PIN shall be decremented. After 10 consecutive false UNBLOCK PIN presentations, not necessarily in the same card session, the respective UNBLOCK PIN shall be blocked. A false UNBLOCK PIN shall have no effect on the status of the respective PIN itself.

Input:
- indication PIN, the UNBLOCK PIN and the new PIN.

Output:
- none.

11.1.13.2 Command parameters:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	'00'
P2	Qualifier, see table 11.8
Lc	Length of the subsequent data field = '10'
Data	UNBLOCK PIN value, new PIN value
Le	Not present

Editor’s note: do we need the new P1 options as defined in 7816-8?

Coding of P2 is identical to the P2 of VERIFY PIN command.

Coding of PIN number in parameter P2:
- 00 = PIN1;
- 02 = PIN2.

NOTE: The coding '00' for PIN differs from the coding of PIN used for other commands.

Command data:

Byte(s)	Description	Length
1 - 8	UNBLOCK PIN value	8
9 - 16	New PIN value	8

11.1.14 DEACTIVATE FILE

11.1.14.1 Functional description

This function initiates a reversible deactivation of an EF. After a DEACTIVATE FILE function the respective flag in the file status shall be changed accordingly. This function shall only be performed if the DEACTIVATE FILE access condition for the EF is satisfied.

An deactivated file shall no longer be available within the selected application for any function except for the SELECT and the ACTIVATE FILE functions.

Input:
- none.

Output:
- none.

11.1.14.2 Command parameters:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	As specified for the SELECT (see 11.1.1)
P2	As specified for the SELECT (see 11.1.1)
Lc	Length of subsequent data field or empty
Data	AID, file ID, DF name, or path to file, according to P1
Le	Not present

11.1.15 ACTIVATE FILE

11.15.1 Functional description

This function reactivates a deactivated EF. After an ACTIVATE FILE function the respective flag in the file status shall be changed accordingly. This function shall only be performed if the ACTIVATE FILE access condition for the current EF is satisfied

Input:
- none.

Output:
- none.

11.1.15.2 Command parameters:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	As specified for the SELECT (see 11.1.1)
P2	As specified for the SELECT (see 11.1.1)
Lc	Length of subsequent data field or empty
Data	AID, file ID, DF name, or path to file, according to P1
Le	Not present

11.1.16 INTERNAL AUTHENTICATE

11.1.16.1 Functional description

An appropriate application shall be selected in the UICC before issuing this command. This command is used during the procedure for authenticating a specific subscription to a network and to calculate a cipher key.

The command initiates the computation of the authentication data by using the challenge data sent from the Terminal together with a secret key, that is stored in the card.

Input:
- challenge data

Output:
- authentication and ciphering data

11.1.16.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	Reference to the algorithm in the card – see table 11.11
P2	Reference to the secret – see table 11.12
Lc	Length of the subsequent data field
Data	Authentication related data
Le	Length of the response data

Table 11.11: Coding of P1

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	0	0	0	0	0	Default algorithm used – see NOTE
x	X	x	x	x	x	x	x	Application specific algorithm reference
NOTE: The default algorithm must be specified by each application specific document.								

Table 11.12: Coding of P2

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	0	0	0	0	0	No information given
0	-	-	-	-	-	-	-	Global reference data (e.g. an MF specific key)
1	-	-	-	-	-	-	-	Specific reference data (e.g. an application specific key)
-	X	x	-	-	-	-	-	'00' (other values are RFU)
-	-	-	x	x	x	x	x	Number of the secret

Command data:

Byte(s)	Description	Length
1 – Lc	Authentication related data (see NOTE)	Lc
NOTE: The command data must be specified by each application specific document.		

Response data (generic):

Byte(s)	Description	Length
1 – Le	Authentication related data (see NOTE)	Le
NOTE: The response data must be specified by each application specific document.		

11.1.17 TERMINAL PROFILE

11.1.17.1 Functional description

This function is used by the Terminal to transmit its capabilities to the selected application concerning the SIM Application Toolkit functionality.

Input:
- terminal profile.

Output:
- none.

11.1.17.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	'00'
P2	'00'
Lc	Length of the subsequent data field
Data	Terminal profile data defined in GSM 11.14
Le	Not present

DLR NOTE: Our position toward SAT and GSM 11.14 in Phase 1 should clarified and consistent in all specifications.

Editor's note: Are there any needs to use this command for sending some other profile than SIM ATK profile to the card? The profile could be selected e.g. with parameter P1 or P2, so that P1='00' means SIM ATK profile, P1='01' means some other profile, etc.

11.1.18 ENVELOPE

11.1.18.1 Functional description

DLR NOTE: The above definition tries to merge the purpose of ENVELOPE in GSM and in ISO, but it seems there is a real difference (application protocol issue for GSM identical for T=0 or T=1, transmission protocol issue for ISO), even if that difference is not reflected in the coding of the command in the respective standards.

The command is used to transfer information, which otherwise could not be transferred by the available protocols e.g. SIM Application Toolkit applications to the selected application.

Input:
- data string.

Output:
- The structure of the data is defined in GSM 11.14 [27].

Editor's note: SAT will probably be available in the USIM application, but this issue is FFS.

11.1.18.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	'00'
P2	'00'
Lc	Length of the subsequent data field
Data	Data string
Le	Length of expected data

Response data:

Structure of the response data is defined in GSM 11.14

11.1.19 FETCH

11.1.19.1 Functional description

This function is used to transfer an SIM Application Toolkit command from the selected application to the Terminal.

Input:
- none.

Output:
- data string containing an SIM Application Toolkit command for the Terminal.

11.1.19.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	'00'
P2	'00'
Lc	Not present
Data	Not present
Le	Length of expected data

Response data:

Structure of the response data is defined in GSM 11.14

11.1.20 TERMINAL RESPONSE

11.1.20.1 Functional description

DLR NOTE: More details (e.g. definition and security) and more compliance with 7816-4 (e.g. P1 coding) required.

This function is used to transfer from the Terminal to the selected application the response to a previously fetched SIM Application Toolkit command.

Input:
- data string containing the response.

Output:
- none.

11.1.20.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	'00'
P2	'00'
Lc	Length of the subsequent data field
Data	Terminal response data defined in GSM 11.14
Le	Not present

11.1.21 MANAGE CHANNEL

11.1.21.1 Functional description

This command opens and closes logical channels.
The open function opens a new logical channel than the basic channel ‘00’ of the card. Options for defining number of logical channels to be provided by the card are as well as options for the card to assign to a logical channel provided.
The close function explicitly closes a logical channel other than the basic logical channel ‘00’ of the card. When a channel has been successfully closed, the channel shall be ready for re-usage.

The basic channel is always available.

11.1.21.2 Command parameters and data:

Code	Value
CLA	As specified in 10.2.1
INS	As specified in 10.2.2
P1	Logical channel operation code
P2	Logical channel number, or '00'
Lc	Not present
Data	Not present
Le	Length of expected data (see NOTE)

Values for parameter P1:

- '00': Open logical channel
- '01': Close logical channel

Parameter P2 indicates logical channel number: '01', '02' or '03'. Value '00' of P2 does not mean logical channel number, but advises the card internally assign the channel number and return it as a response. With other values of P2 the channel number is externally assigned.

DLR NOTE: 7816-4 does not specify whether the channel assigned by the card when P2=0 is open or has to be opened issuing another MANAGE CHANNEL command. 31.101 should be clear on this aspect.

Response data (see NOTE):

Byte(s)	Description	Length
1	Logical channel number	1

NOTE: Response data is available only if the value of the parameters P1-P2 is '0000'.

12 Transmission Oriented Commands

This clause lists all transport specific commands

12.1 T=0 specific commands

12.1.1 GET RESPONSE

12.1.1.1 Functional description

The command is used to transmit from the card to the Terminal APDU(s), which otherwise not could be transferred by the protocol.

The response data depends on the preceding command. Response data is available after the commands INTERNAL AUTHENTICATION, SEEK (type 2), SELECT and INCREASE. If the command GET RESPONSE is executed, it is required that it is executed immediately after the command it is related to (no other command shall come between the command/response pair and the command GET RESPONSE). If the sequence is not respected, the selected application shall send the status information "technical problem with no diagnostic given" as a reaction to the GET RESPONSE.

Since the MF is implicitly selected after activation of the selected application, GET RESPONSE is also allowed as the first command after activation.

The response data itself is defined in the sub-clause for the corresponding command.

The command shall only be used when the T=0 transmission protocol has been selected.

12.1.1.2 Command parameters:

Code	Value
CLA	As specified in 12.2.1
INS	As specified in 12.2.2
P1	'00'
P2	'00'
Lc	Not present
Data	Not present
Le	'00' or value of SW2 of the previous command

Response parameters and data:

The response data is defined in each subclause of the corresponding command.

The response data depends on the preceding command. Response data is available after the commands INTERNAL AUTHENTICATE, SEEK (type 2), SELECT, INCREASE, ENVELOPE, and MANAGE CHANNEL. If the command GET RESPONSE is executed, it is required that it is executed immediately after the command it is related to (no other command shall come between the command/response pair and the command GET RESPONSE). If the sequence is not respected, the UICC shall send the status information "technical problem with no diagnostic given" as a reaction to the GET RESPONSE.

Since the MF is implicitly selected after activation of the UICC, GET RESPONSE is also allowed as the first command after activation.

13 Application independent files

13.1 EF_{DIR}

EF_{DIR} is a linear fixed file. This file is under the responsibility of the issuer at the MF level, but can be under other responsibilities, when situated at a lower level.

Table 13.1: EF_{DIR} at MF-level

Identifier: '2F00'	Structure: Linear fixed		Mandatory
File size: X bytes		Update activity: low	
Access Conditions: READ ALWAYS/PIN UPDATE ADM			
Bytes	Description	M/O	Length
1 – X	Application template TLV object	M	X bytes

Each entry in the EF_{DIR} is an application template Data Object (DO) as defined in 7816-5 [5]. An application template DO is a constructed BER-TLV object (see Annex A) with a maximum length of 127 bytes and has a mandatory AID DO, all other DO's are optional.

In table 13.2 the coding of the mandatory DO's and the optional DO's that has special meaning to this document. All other DO's are according to 7816-5.

Table 13.2 coding of an entry application template

Length	Description	Status
1	Application template tag = '61'	M
03 - '7F'	Length of the application template	M
1	Application IDentifier tag	M
1	AID length	M
'01' – '10'	AID value	M
1	Application label tag	O
1	Application label length	O
NOTE	Application label value	O
1	Discretionary data label tag	O
1	Discretionary data label length	O
N	Discretionary data – see table 13.3	O
NOTE: The application label is a DO that contains a string of bytes provided by the application provider to be shown to the user for information, e.g. operator name. The value part of the application label shall be coded according to Annex D. In 7816-5 [5] the application label is limited to 16 octets, this does not apply to 31.101 it is however recommended that the number of octets does not exceed 32.		

Table 13.3: Coding of value of the discretionary data

Byte	b8	b7	b6	b5	b4	b3	b2	b1	Meaning
1	-	-	-	-	-	-	-	0	Not a default application
	-	-	-	-	-	-	-	1	Default application
…..									RFU
N	-	-	-	-	-	-	-	-	RFU

13.2 EF$_{ICCID}$ (ICC Identification)

This EF provides a unique identification number for the UICC.

Table 13.4: EF$_{ICCID}$ at MF-level

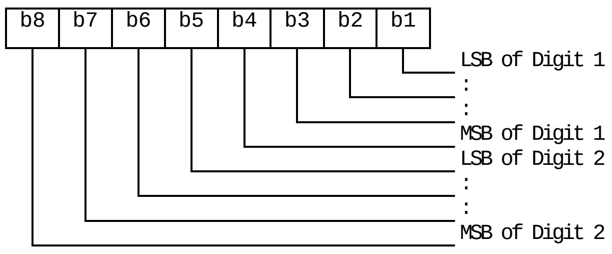
Identifier: '2FE2'	Structure: transparent		Mandatory
File size: 10 bytes		Update activity: low	
Access Conditions: READ ALWAYS UPDATE NEVER			
Bytes	Description	M/O	Length
1 – 10	Identification number	M	10 bytes

- Identification number

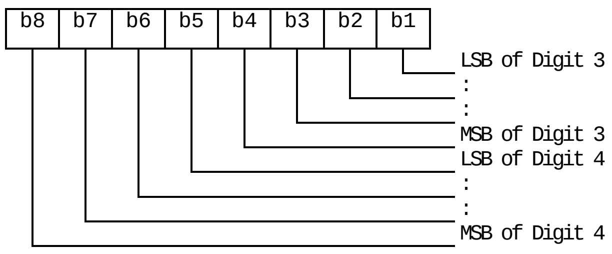
Contents:
according to CCITT Recommendation E.118 [18].Purpose:
card identification number.

Coding:
BCD, left justified and padded with 'F'; after padding the digits within a byte are swapped (see below).

Byte 1:



Byte 2:



etc.

13.3 EF_{PL} (Preferred languages)

This EF contains the codes for up to n languages. This information, determined by the user/operator, defines the preferred languages of the user in order of priority. This information may be used by the Terminal for MMI purposes and for short message handling (e.g. screening of preferred languages in SMS-CB).

Table 13.5: EF_{PL} at MF-level

Identifier: '2F 05'	Structure: transparent		Mandatory
File size: 2n bytes		Update activity: low	
Access Conditions: READ ALW UPDATE PIN			
Bytes	Description	M/O	Length
1 – 2	1st language code (highest prior.)	M	2 bytes
3 – 4	2nd language code	O	2 bytes
2n-1 – 2n	nth language code (lowest prior.)	O	2 bytes

Coding:

each language code is a pair of alpha-numeric characters, defined in ISO 639 [30]. Each alpha-numeric character shall be coded on one byte using the SMS default 7-bit coded alphabet as defined in GSM 03.38 [12] with bit 8 set to 0.

Unused language entries shall be set to 'FF FF'.

14 Application independent protocol

When the UICC is in any other than the inactive state, as defined in 9.X the UICC interfaces with an Terminal by means of functions. A procedure consists of one or more functions. A procedure shall be considered as a whole, that is to say that the corresponding task is achieved if and only if the procedure is completed. The Terminal shall ensure that, when operated according to the manufacturer's manual, any unspecified interruption of the sequence of command/response pairs which realize the procedure, leads to the abortion of the procedure itself.

Some procedures at the UICC/Terminal interface require MMI interactions. The descriptions hereafter do not intend to infer any specific implementation of the corresponding MMI. When MMI interaction is required, it is marked "MMI" in the list given below.

Some procedures are automatically initiated by the Terminal. They are marked "Terminal" in the list given below.

The list of generic application procedures that are; either common to all applications or non-application specific at the UICC/Terminal interface is as follows:

File related procedures:

- Reading an EF	Terminal
- Updating an EF	Terminal
- Increasing an EF	Terminal

PIN related procedures:

- PIN verification	MMI
- PIN value substitution	MMI
- PIN disabling	MMI
- PIN enabling	MMI
- PIN unblocking	MMI

General application related procedures

- Application selection	Terminal/MMI
- Application session activation	MMI
- UICC Application interrogation	MMI
- Default application	MMI
- UICC application session termination	MMI

Miscellaneous procedures:

- UICC activation	Terminal
- UICC presence detection	Terminal
- UICC Language Indication request	Terminal/MMI

14.1 File related procedures

14.1.1 Reading an EF

Reading of an EF can be done in two different ways:

If the short file identifiers are used the following procedure applies:

- If short file identifiers are used every EF of the Current Directory be read without explicitly selecting the EF. The Terminal selects the DF or ADF and sends a READ command. This contains the short file identifier of the EF to be read and the location of the data to be read. If the access condition for READ is fulfilled, the application sends the requested data contained in the EF to the Terminal. If the access condition is not fulfilled, no data will be sent and an error code will be returned.

If the short file identifiers are not used the following procedure applies:

- The Terminal selects the EF and sends a READ command. This contains the location of the data to be read. If the access condition for READ is fulfilled, the application sends the requested data contained in the EF to the Terminal. If the access condition is not fulfilled, no data will be sent and an error code will be returned.

14.1.2 Updating an EF

Updating of an EF can be done in two different ways:

If the short file identifiers are used the following procedure applies:

- If short file identifiers are used every EF of the Current Directory be updated without explicitly selecting the EF. The Terminal selects the DF or ADF and sends an UPDATE command. This contains the short file identifier of the EF and the location of the data to be updated and the new data to be stored. If the access condition for UPDATE is fulfilled, the application updates the selected EF by replacing the existing data in the EF with that contained in the command. If the access condition is not fulfilled, the data existing in the EF will be unchanged, the new data will not be stored, and an error code will be returned.

If the short file identifiers are not used the following procedure applies:

- The Terminal selects the EF and sends an UPDATE command. This contains the location of the data to be updated and the new data to be stored. If the access condition for UPDATE is fulfilled, the application updates the selected EF by replacing the existing data in the EF with that contained in the command. If the access condition is not fulfilled, the data existing in the EF will be unchanged, the new data will not be stored, and an error code will be returned.

14.1.3 Increasing an EF

The Terminal selects the EF and sends an INCREASE command. This contains the value which has to be added to the contents of the last updated/increased record. If the access condition for INCREASE is fulfilled, the application increases the existing value of the EF by the data contained in the command, and stores the result. If the access condition is not fulfilled, the data existing in the EF will be unchanged and an error code will be returned**.**

NOTE: The identification of the data within an EF to be acted upon by the above procedures is specified within the command. For the procedures in subclauses 14.1.1 and 14.1.2 this data may have been previously identified using a SEEK command, e.g. searching for an alphanumeric pattern.

14.2 PIN related procedures

NOTE: This document specifies only the generic behaviour of a PIN. An application may create a set of PIN's each with a specific behaviour.

A successful completion of one of the following procedures grants the access right of the corresponding PIN for an application session. This right is valid for all files within the application protected by this PIN.

After a third consecutive presentation of a wrong PIN to an application, not necessarily in the same application session, the PIN status becomes "blocked" and if the PIN is "enabled", the access right previously granted by this PIN is lost immediately.

An access right is not granted if any of the following procedures are unsuccessfully completed or aborted.

14.2.1 PIN verification

The Terminal checks the PIN status and the following procedure applies:

If the PIN status is "blocked" and PIN is "enabled", the procedure ends and is finished unsuccessfully.

If the PIN status is "blocked" but PIN is "disabled", the procedure ends and is finished successfully. The Terminal shall, however, accept applications which do not grant access rights when PIN is "blocked" and "disabled". In that case Terminal shall consider those applications as "blocked".

If the PIN status is not "blocked" and PIN is "disabled", the procedure is finished successfully.

If the PIN status is not "blocked" and PIN is "enabled", the Terminal uses the VERIFY PIN function. If the PIN presented by the Terminal is equal to the corresponding PIN stored in the application, the procedure is finished successfully. If the PIN presented by the Terminal is not equal to the corresponding PIN stored in the application, the procedure ends and is finished unsuccessfully.

Note: At the T3 plenary in Lund it was decided that there should be no indication in this document regarding different types of PIN's this shall be defined in the application specific documents, e.g. 31.102. this decision was taken as the only difference between CHV1 and CHV2 in GSM is the fact that CHV2 can not be disabled but otherwise works completely as CHV1!

14.2.3 PIN value substitution

The Terminal checks the PIN status. If the PIN status is "blocked" or "disabled", the procedure ends and is finished unsuccessfully.

If the PIN status is not "blocked" and the enabled/disabled indicator is set "enabled", the Terminal uses the CHANGE PIN function. If the old PIN presented by the Terminal is equal to the corresponding PIN stored in the application, the new PIN presented by the Terminal is stored in the application and the procedure is finished successfully.

If the old PIN and the PIN in memory are not identical, the procedure ends and is finished unsuccessfully.

14.2.4 PIN disabling

PIN enabling and disabling may be disallowed by an application. If it is allowed then the following procedures must be followed:

The Terminal checks the PIN status. If the PIN status is "blocked", the procedure ends and is finished unsuccessfully.

If the PIN status is not "blocked", the Terminal reads the PIN enabled/disabled indicator. If this is set "disabled", the procedure ends and is finished unsuccessfully.

If the PIN status is not "blocked" and the enabled/disabled indicator is set "enabled", the Terminal uses the DISABLE PIN function. If the PIN presented by the Terminal is equal to the PIN stored in the SIM, the status of PIN is set "disabled" and the procedure is finished successfully. If the PIN presented by the Terminal is not equal to the PIN stored in the application, the procedure ends and is finished unsuccessfully.

14.2.5 PIN enabling

PIN enabling and disabling may be disallowed by an application. If it is allowed then the following procedures must be followed:

The Terminal checks the PIN status. If the PIN status is "blocked", the procedure ends and is finished unsuccessfully.

If the PIN status is not "blocked", the Terminal reads the PIN enabled/disabled indicator. If this is set "enabled", the procedure ends and is finished unsuccessfully.

If the PIN status is not "blocked" and the enabled/disabled indicator is set "disabled", the Terminal uses the ENABLE PIN function. If the PIN presented by the Terminal is equal to the PIN stored in the SIM, the status of PIN is set "enabled" and the procedure is finished successfully. If the PIN presented by the Terminal is not equal to the PIN stored in the application, the procedure ends and is finished unsuccessfully.

14.2.6 PIN unblocking

The execution of the PIN unblocking procedure is independent of the corresponding PIN status, i.e. being blocked or not.

The Terminal checks the UNBLOCK PIN status. If the UNBLOCK PIN status is "blocked", the procedure ends and is finished unsuccessfully.

If the UNBLOCK PIN status is not "blocked", the Terminal uses the UNBLOCK PIN function. If the UNBLOCK PIN presented by the Terminal is equal to the corresponding UNBLOCK PIN stored in the application, the relevant PIN status becomes "unblocked" and the procedure is finished successfully. If the UNBLOCK PIN presented by the Terminal is not equal to the corresponding UNBLOCK PIN stored in the application, the procedure ends and is finished unsuccessfully.

14.3 Application selection procedures

NOTE: All the procedures must be updated after the Korea meeting.

14.3.1 Application selection by use of the EF_{DIR} file

Application selection by use of the EF_{DIR} file is the procedure where the Terminal reads the content of the EF_{DIR} file and presents the list of applications to the user whom can then make select one or more applications to activate.

The Terminal performs the read procedure with EF_{DIR} and does one of the following:

1 if indicated by the user the default application(s), as defined in xx, are activated automatically one by one by performing a series of application activation procedures, or

2 the Terminal presents the applications that it supports to the user who may make a manual selection. If only one supported application is found this may be implicitly selected. For both cases the application activation procedure is performed

14.3.2 Direct application selection

An application may be selected, without reading the content of the EF_{DIR} file, by performing the SELECT procedure with the AID of the application to be selected.

14.3.2 Direct application selection with partial AID

The Terminal performs the SELECT procedure with the part of the AID of the application to be selected. The UICC will select the first application that matches the partial AID. The Terminal can select other applications that match the AID by performing the SELECT procedure indicating that the next matching application should be selected.

14. 4 General application related procedures

14.4.1 Application session activation

The Terminal performs the SELECT function with the AID of the selected application as a parameter.

If the SELECT function ends successfully the selected application's initialisation procedure is executed. If the initialisation procedure end successfully the UICC enters the operation state otherwise the UICC remains in the application management state and sends and indication to the user that it was not possible to activate the selected application.

NOTE: This text should go into 31.102" If no EF_{DIR} file is found or no USIM's are listed in the EF_{DIR} file the Terminal then tries to select the GSM application as specified in GSM 11.11."

14.4.2 UICC Application interrogation

The list of applications residing in the UICC can be read at anytime when the UICC is not inactive.

Request: The Terminal performs the read procedure with EF_{DIR}.

14.4.3 Default application setting

The Terminal performs the write procedure on the discretionary data TLV DO of the application template belonging to the selected application with an indication that the application is default.

14.4.4 Default application deleting

The Terminal performs the write procedure on the discretionary data TLV DO of the application template belonging to the selected application with an indication that the application is not default.

14.4.5 UICC application session termination

An application session can be terminated at any time when the UICC is not inactive.

The Terminal performs the read procedure with EF_{DIR} , marks the application as being inactive and performs the applications session termination procedure.

When an application session is terminated the access condition PIN for the current application must be invalidated. If the application acts as a parent to other applications with disabled or invalidated PIN's then these applications access condition PIN is also invalidated. See the example below:

EXAMPLE: Please refer to figure 9.1. ADF1, has an active session and, has PIN enabled and verified, ADF2 and ADF3 has invalidated their PIN's but have active sessions. The ADF1 session is now terminated, this implies that the PIN access condition for ADF2 and ADF3 is not valid anymore.

It is the responsibility of the UICC to ensure that the access condition PIN always has the correct value for each file.

NOTE: The example and the paragraphs above MUST be checked.

NOTE: It is the responsibility of the UICC to mark an application as being inactive in the application status information bytes.

NOTE: This must be updated according to the decisions taken a the T3 meeting in Korea.

14.4 Miscellaneous procedures

14.4.1 UICC activation

After activation of the UICC , as defined in subclause 4.5, the Terminal requests the Preferred Language (EF_{PL}), if available, otherwise the Terminals default language is selected. The Terminal then performs the application selection procedure.

NOTE: An application that resides on the UICC must define an initialisation procedure.

14.4.2 UICC presence detection

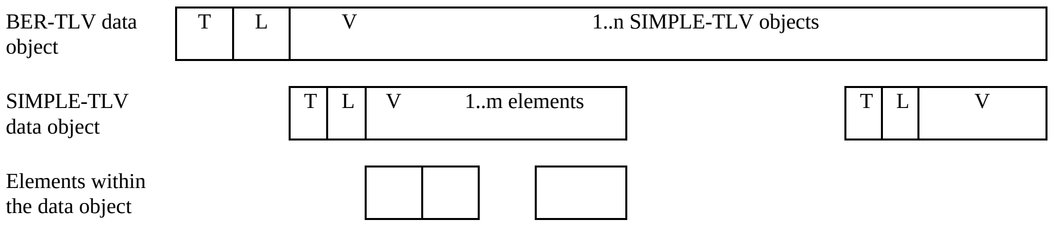
To ensure that the UICC has not been removed during a card session, the Terminal sends, at frequent intervals, a STATUS command during each call. A STATUS command shall be issued within all 30 second periods of inactivity on the UICC-Terminal interface during a call. Inactivity in this case is defined as starting at the end of the last communication or the last issued STATUS command. If no response data is received to this STATUS command, then the call shall be terminated as soon as possible but at least within 5 seconds after the STATUS command has been sent. If the DF indicated in response to a STATUS command is not the same as that which was indicated in the previous response, or accessed by the previous command, then the call shall be terminated as soon as possible but at least within 5 seconds after the response data has been received. This procedure shall be used in addition to a mechanical or other device used to detect the removal of a UICC.

14.4.3 UICC Language Indication request

Request:	The Terminal performs the read procedure with EF_{LI}.
Update:	The Terminal performs the update procedure with EF_{LI}

Annex A (informative): Coding of BER-TLV data objects.

Editor's note: This Annex is a cut from GSM 11.14 and must be modified to reflect the coding used for e.g. FCI templates and such



SIM Application Toolkit commands and responses are sent across the interface as BER-TLV data objects. Each APDU shall only contain one BER-TLV object.

The tag is a constant value, length one byte, indicating it is a SIM Application Toolkit command.

The length is coded onto 1,or 2 bytes according to ISO/IEC 7816-6 [17]. The following table details this coding:

Length	Byte 1	Byte 2
0-127	length ('00' to '7F')	not present
128-255	'81'	length ('80' to 'FF')

Any length within the APDU limits (up to 255 bytes) can thus be encoded on two bytes. This coding is chosen to remain compatible with ISO/IEC 7816-6 [17].

Any values for byte 1 or byte 2 that are not shown above shall be treated as an error and the whole message shall be rejected.

The value part of the BER-TLV data object consists of SIMPLE-TLV data objects, as shown in the description of the SIMPLE-TLV data objects on individual commands. It is mandatory for SIMPLE-TLV data objects to be provided in the order given in the description of each command. New SIMPLE-TLV data objects can be added to the end of a command.

The M/O columns specify whether it is mandatory or optional for the sender to send that particular SIMPLE-TLV data object for compliance with the current version of this TS. The Min (Minimum Set) column describes whether it is necessary for the receiver to have received that particular SIMPLE-TLV data object to be able to attempt at least the most basic form of this command. The procedure for dealing with incomplete messages is described in subclause 6.10.

'00' and 'FF' are never used as tag values for BER-TLVs. This is in accordance with ISO/IEC 7816-6 [17]. Padding characters are not allowed.

See ISO/IEC 7816-6 [17] for more information on data objects.

Annex B (informative): Main states of a UICC

A UICC complying to this standard has the following states of operation:

- Inactive, when the UICC is powered off.
- Application management - In this state it is possible to start/end one or more application-session(s) as well as retrieve the list of applications in the UICC. In this state all files in any application with the access condition ALWAYS can be accessed. This state can be entered at any time when the UICC is not in the inactive state and left when the PIN access condition for all the selected applications has been verified or the .
- Operation, this state is entered after the application session activation procedure for at least one application has ended successfully.

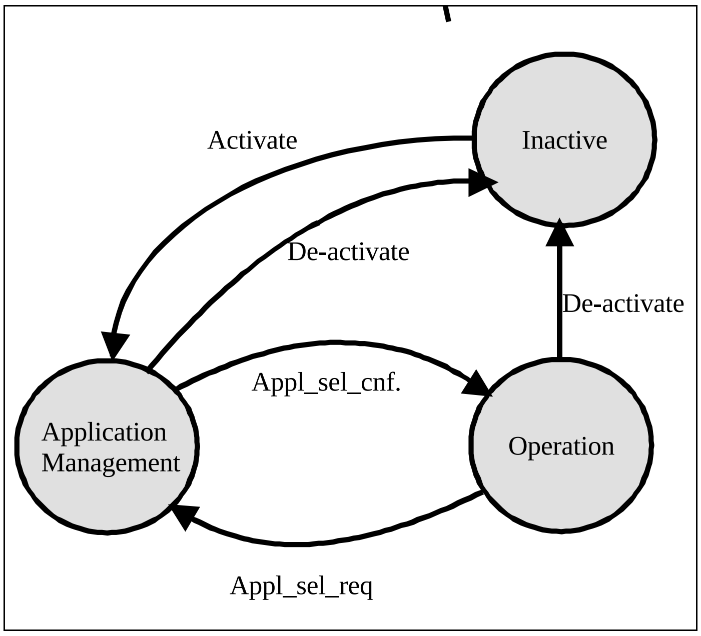


Figure 9.2: UICC states

NOTE: This chapter should be moved to an informative annex – please make a note stating that the information in the AIP takes precedence in case of ambiguities. Changes are need on the headings.

Annex C (informative): APDU Protocol Transmission Examples

C.1 Exchanges Using T=0

The following examples illustrate exchanges of data and procedure bytes between the terminal and the UICC.

Note the following:

- The use of procedure bytes '60' and INS is not illustrated
- [Data(x)] means x bytes of data
- Case 2 and 4 commands have Le = '00' requesting the return of all data from the UICC up to the maximum available.

The examples in clauses C.1.1.1 to C1.1.4 illustrate typical exchanges using case 1 to 4 commands. The examples in the subclauses C.1.1.5 and C.1.1.6 illustrate the more extensive use of procedure bytes '61 xx' when used with case 2 and 4 commands. The example in subclause C.1.1.7 illustrates a warning condition with a case 4 command.

C.1.1.1 Case 1 Command

A C-APDU of {CLA INS P1 P2} is passed from the Terminal to the UICC (note that P3 of the C-TPDU is set to '00').

	Terminal		**UICC**
C-TPDU	[CLA INS P1 P2 00]	⇒	
		⇐	90 00

A R-APDU of {90 00} is returned from the UICC to the Terminal

C.1.1.2 Case 2 Command

A C-APDU of {CLA INS P1 P2 00} is passed from the Terminal to the UICC.

	Terminal		**UICC**
C-TPDU	[CLA INS P1 P2 00]	⇒	
		⇐	6C Luicc
C-TPDU	[CLA INS P1 P2 Luicc]	⇒	
		⇐	INS [Data(Luicc)] 90 00

A R-APDU of {[Data(Luicc)] 90 00} is returned from the UICC to the Terminal.

(i) If Le ≥ Luicc, the data returned is mapped onto the conditional body of the R-TPDU, and the status returned is mapped onto the mandatory trailer of the R-APDU without change.

(ii) If Le < Luicc, the first Le bytes of the data returned are mapped onto the conditional body of the R-TPDU, and the status returned is mapped onto the mandatory trailer of the R-APDU without change.

Since the procedure defined above for Le ≥ Luicc is inefficient. One will forced to re-issue the command. A more practical approach should be following:

	Terminal			**UICC**
C-TPDU	[CLA INS P1 P2 00]	⇒		
			⇐	61 Luicc
GET RESPONSE	[00 C0 00 00 yy]	⇒		
			⇐	C0 [Data(yy)] 61 zz
	[00 C0 00 00 zz]	⇒		
			⇐	C0 [Data(zz)] 90 00

Where yy ≤ xx

A R-APDU of {[Data(yy + zz)] 90 00} is returned from the UICC to the Terminal.

Note: Since all case 2 commands issued according to this specification shall have Le set to ‘00’, case (ii) should not occur and is for information only.

C.1.1.3 Case 3 Command

A C-APDU of {CLA INS P1 P2 Lc [Data(Lc)]} is passed from the Terminal to the UICC.

	Terminal			**UICC**
C-TPDU	[CLA INS P1 P2 Lc]	⇒		
			⇐	[INS]
	[Data(Lc)]	⇒		
			⇐	90 00

A R-APDU of {90 00} is returned from the UICC to the Terminal.

C1.1.4 Case 4 Command

A C-APDU of {CLA INS P1 P2 Lc [Data (Lc)] 00} is passed from the Terminal to the UICC.

	Terminal			**UICC**
C-TPDU	[CLA INS P1 P2 Lc]	⇒		
			⇐	[INS]
	[Data(Lc)]	⇒		
			⇐	61 Luicc
GET RESPONSE	[00 C0 00 00 Luicc]	⇒		
			⇐	C0 [Data(Luicc)] 90 00

A R-APDU of {[Data(Luicc)] 90 00} is returned from the UICC to the Terminal.

C1.1.5 Case 2 Commands Using the ‘61’ and ‘6C’ Procedure Bytes

A C-APDU of {CLA INS P1 P2 00} is passed from the Terminal to the UICC.

	Terminal			**UICC**
C-TPDU	[CLA INS P1 P2 00]	⇒		
			⇐	6C Luicc
	[CLA INS P1 P2 Luicc]	⇒		
			⇐	61 xx
GET RESPONSE	[00 C0 00 00 yy]	⇒		
			⇐	C0 [Data(yy)] 61 zz
	[00 C0 00 00 zz]	⇒		
			⇐	C0 [Data(zz)] 90 00

Where yy ≤ xx

A R-APDU of {[Data(yy + zz)] 90 00} is returned from the UICC to the Terminal.

C1.1.6 Case 4 Command Using the ‘61’ Procedure Byte

A C-APDU of {CLA INS P1 P2 Lc [Data Lc] 00} is passed from the Terminal to the UICC.

	Terminal			**UICC**
C-TPDU	[CLA INS P1 P2 Lc]	⇒		
			⇐	[INS]
	[Data(Lc)]	⇒		
			⇐	61 xx
GET RESPONSE	[00 C0 00 00 xx]	⇒		
			⇐	C0 [Data(xx)] 61 yy
	[00 C0 00 00 yy]	⇒		
			⇐	C0 [Data(yy)] 90 00

A R-APDU of {[Data(xx + yy)] 90 00} is returned from the UICC to the Terminal.

C1.1.7 Case 4 Command with Warning Condition

A C-APDU of {CLA INS P1 P2 Lc [Data Lc] 00} is passed from the Terminal to the UICC.

	Terminal			**UICC**
C-TPDU	[CLA INS P1 P2 Lc]	⇒		
			⇐	[INS]
	[Data(Lc)]	⇒		
			⇐	62 xx
GET RESPONSE	[00 C0 00 00 00]	⇒		
			⇐	6C Luicc
	[00 C0 00 00 Luicc]	⇒		
			⇐	C0 [Data(Luicc)] 90 00

A R-APDU of {[Data(Luicc)] 62 xx} is returned from the Terminal to the UICC containing the data returned together with the warning status bytes.

Annex D (normative): UCS2 Coding of Alpha fields for files residing on the UICC

If 16 bit UCS2 characters as defined in ISO/IEC 10646 [31] are being used in an alpha field, the coding can take one of three forms. If the Terminal supports UCS2 coding of alpha fields in the UICC, the Terminal shall support all three coding schemes for character sets containing 128 characters or less; for character sets containing more than 128 characters, the Terminal shall at least support the first coding scheme. If the alpha field record contains GSM default alphabet characters only, then none of these schemes shall be used in that record. Within a record, only one coding scheme, either GSM default alphabet, or one of the three described below, shall be used.

1) If the first octet in the alpha string is '80', then the remaining octets are 16 bit UCS2 characters, with the more significant octet (MSO) of the UCS2 character coded in the lower numbered octet of the alpha field, and the less significant octet (LSO) of the UCS2 character is coded in the higher numbered alpha field octet, i.e. octet 2 of the alpha field contains the more significant octet (MSO) of the first UCS2 character, and octet 3 of the alpha field contains the less significant octet (LSO) of the first UCS2 character (as shown below). Unused octets shall be set to 'FF', and if the alpha field is an even number of octets in length, then the last (unusable) octet shall be set to 'FF'.

Example 1

Octet 1	Octet 2	Octet 3	Octet 4	Octet 5	Octet 6	Octet 7	Octet 8	Octet 9
'80'	$Ch1_{MSO}$	$Ch1_{LSO}$	$Ch2_{MSO}$	$Ch2_{LSO}$	$Ch3_{MSO}$	$Ch3_{LSO}$	'FF'	'FF'

2) If the first octet of the alpha string is set to '81', then the second octet contains a value indicating the number of characters in the string, and the third octet contains an 8 bit number which defines bits 15 to 8 of a 16 bit base pointer, where bit 16 is set to zero, and bits 7 to 1 are also set to zero. These sixteen bits constitute a base pointer to a "half-page" in the UCS2 code space, to be used with some or all of the remaining octets in the string. The fourth and subsequent octets in the string contain codings as follows; if bit 8 of the octet is set to zero, the remaining 7 bits of the octet contain a GSM Default Alphabet character, whereas if bit 8 of the octet is set to one, then the remaining seven bits are an offset value added to the 16 bit base pointer defined earlier, and the resultant 16 bit value is a UCS2 code point, and completely defines a UCS2 character.

Example 2

Octet 1	Octet 2	Octet 3	Octet 4	Octet 5	Octet 6	Octet 7	Octet 8	Octet 9
'81'	'05'	'13'	'53'	'95'	'A6'	'XX'	'FF'	'FF'

In the above example;

- Octet 2 indicates their 5 characters in the string
- Octet 3 indicates bits 15 to 8 of the base pointer, and indicates a bit pattern of 0hhh hhhh h000 0000 as the 16 bit base pointer number. Bengali characters for example start at code position 0980 (0*000 1001 1*000 0000), which is indicated by the coding '13' in octet 3 (shown by the italicised digits).
- Octet 4 indicates GSM Default Alphabet character '53', i.e. "S".
- Octet 5 indicates a UCS2 character offset to the base pointer of '15', expressed in binary as follows 001 0101, which, when added to the base pointer value results in a sixteen bit value of 0000 1001 1001 0101, i.e. '0995', which is the Bengali letter KA.

  Octet 8 contains the value 'FF', but as the string length is 5, this a valid character in the string, where the bit pattern 111 1111 is added to the base pointer, yielding a sixteen bit value of 0000 1001 1111 1111 for the UCS2 character (i.e. '09FF').

3) If the first octet of the alpha string is set to '82', then the second octet contains a value indicating the number of characters in the string, and the third and fourth octets contain a 16 bit number which defines the complete 16 bit base pointer to a "half-page" in the UCS2 code space, for use with some or all of the remaining octets in the string. The fifth and subsequent octets in the string contain codings as follows; if bit 8 of the octet is set to zero, the remaining 7 bits of the octet contain a GSM Default Alphabet character, whereas if bit 8 of the octet is set to one, the remaining seven bits are an offset value added to the base pointer defined in octets three and four, and the resultant 16 bit value is a UCS2 code point, and defines a UCS2 character.

Example 3

Octet 1	Octet 2	Octet 3	Octet 4	Octet 5	Octet 6	Octet 7	Octet 8	Octet 9
'82'	'05'	'05'	'30'	'2D'	'82'	'D3'	'2D'	'31'

In the above example

- Octet 2 indicates there are 5 characters in the string
- Octets 3 and 4 contain a sixteen bit base pointer number of '0530', pointing to the first character of the Armenian character set.
- Octet 5 contains a GSM Default Alphabet character of '2D', which is a dash "-".
- Octet 6 contains a value '82', which indicates it is an offset of '02' added to the base pointer, resulting in a UCS2 character code of '0532', which represents Armenian character Capital BEN.

- Octet 7 contains a value 'D3', an offset of '53', which when added to the base pointer results in a UCS2 code point of '0583', representing Armenian Character small PIWR.

History

Document history		
V0.1.0	April 1999	1ST draft version for comments before TSG T3 #4 meeting, 19-21 April, 1999.
V0.2.0	April 1999	Inclusion of some material discussed at T3 #4 (includes, in particular, the electrical and mechanical parameters)
V0.3.0	May 1999	Preparation to the Ad-hoc meeting in Copenhagen 27-28 May 1999.
V0.4.0	May 1999	Updates after the Ad-hoc meeting in Copenhagen 27-28 May 1999.
V0.5.0	June 1999	Updates after the T3 #5 in Mariehamn.
V0.6.0	June 1999	Updates after the T3 #6 in Miami.
V0.7.0	August 1999	Updates after T3#7 in Lund.
V0.8.0	September 1999	Updates after T3 #8 in Bonn.
V0.9.0	September 1999	Updates before the editing meeting in Turku
V0.10.0	October 1999	Updates before the T3#9 meeting in Korea on the following chapters General all chapters (except the old Ch. 8) are now in sequential order Ch 7: restructuring of start-up, changes to the ATR Speed enhancements Ch. 10: alignment of status words with ISO Ch.11: general update of commands Ch.13: the content of the EF_{DIR} file is modified Ch.14: New application selection procedures added.
V0.11.0	October 1999	Yet another update before the T3#9 meeting in Korea The old Ch. 8 Transmission protocols is renamed to Ch. 7. Modifications to Annex C
V1.0.0	October 1999	Version for information to 3GPP TSG -T #5 (7 - 8 October, 1999)

Rapporteurs: Peter Vestergaard E-mail:peter.vestergaard@nokia.com

Rune Lindholm E-mail:rune.lindholm@.nokia.com