ETSI TS 131 101 V3.0.0 (2000-01)

Technical Specification

Universal Mobile Telecommunications System (UMTS); UICC-Terminal Interface; Physical and Logical Characteristics (3G TS 31.101 version 3.0.0 Release 1999)

Reference

DTS/TSGT-0331101U

Keywords

UMTS

ETSI

Postal address

F-06921 Sophia Antipolis Cedex - FRANCE

Office address

650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16
Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Internet

secretariat@etsi.fr
Individual copies of this ETSI deliverable
can be downloaded from
http://www.etsi.org
If you find errors in the present document, send your
comment to: editor@etsi.fr

Important notice

This ETSI deliverable may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Copyright Notification

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://www.etsi.org/ipr).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by the ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables. The mapping of document identities is as follows:

For 3GPP documents:

3G TS | TR nn.nnn "<title>" (with or without the prefix 3G)

is equivalent to

ETSI TS | TR 1nn nnn "[Digital cellular telecommunications system (Phase 2+) (GSM);] Universal Mobile Telecommunications System; <title>

For GSM document identities of type "GSM xx.yy", e.g. GSM 01.04, the corresponding ETSI document identity may be found in the Cross Reference List on www.etsi.org/key

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 3G 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 3G telecommunication networks is specified in document 3G TS 31.102 [2].

1 Scope

The present document specifies the interface between the UICC and the Terminal for 3G 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 following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies.

[1] 3G TS 23.038: "Alphabets and language-specific information".

[2] 3G TS 31.102: "Characteristics of the USIM application".

[3] 3G TS 31.110: "Numbering system for telecommunication IC card applications"

[4] 3G TS 31.111: "USIM Application Toolkit (USAT)".

[5] ITU-T E.118: "The international telecommunication charge card".

[6] ISO 639 (1988): "Code for the representation of names of languages".

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

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

[9] ISO/IEC 7811-3 (1995): "Identification cards - Recording technique - Part 3: Location of embossed characters on ID-1 cards".

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

[11] ISO/IEC 7816-2 (1999): "Identification cards - Integrated circuit(s) cards with contacts, Part 2: Dimensions and locations of the contacts".

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

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

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

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

[16] ISO/IEC 7816-8 (1999): "Identification cards - Integrated circuit(s) cards with contacts, Part 8: Security related Interindustry commands".

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

[18] ISO/IEC 10646-1 (1993): "Information technology -- Universal Multiple-Octet Coded Character Set (UCS) -- Part 1: Architecture and Basic Multilingual Plane".

3 Definitions, symbols, abbreviations and coding

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.

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

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 deselection: TBD

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 DFor ADF selected.

Current EF: The latest EF selected.

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, DF and ADF.

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

File: A directory or an organised set of bytes or records in the UICC.

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

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.

ID-1 UICC: The UICC having the format of an ID-1 card (see ISO/IEC 7816-1 [10]).

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

Multi-application card: A card that can have more than one selectable application.

Multi-session card: A card that supports more than one concurrent selectable application session during a card session.

Normal USIM operation: Relating to general, PIN related, 3G and or GSM security 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 UICC: A Second format of UICC

Proactive UICC : A UICC , which is capable of issuing commands to the Terminal. Part of USAT.

Record: A string of bytes within an EF handled as a single entity

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

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

UICC application toolkit procedures: Defined in 3G TS 31.111 [4].

Selectable application: An application that is selectable by an AID according to the process described in ISO/IEC 7816-4 [13] over the Terminal-UICC interface.

Selectable application session: A link between the application and the external world during a card session starting with the application selection and ending with de-selection or termination of the card session.

USIM session: A USIM session is a selectable application session for a USIM application.

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:

AC	Access Condition
ACK	ACKnowledge
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
f	frequency
FCI	File Control Information
Fi	Clock rate conversion factor
FID	File IDentifier
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
INS	INStruction
I/O	Input/Output
ISO	International Organization for Standardization
Lc	Length of Command data sent by the application layer in a case 3 or 4 Command.
Le	Maximum length of data Expected by the application layer in response to a case 2 or 4 Command.
LEN	LENgth
LRC	Longitudinal Redundancy Check
LSB	Least Significant Bit
Luicc	Exact Length of data available in the UICC to be returned in response to the case 2 or 4 Command received by the UICC
MF	Master File
MMI	Man Machine Interface
MSB	Most Significant Bit
NAD	Node Address byte
NEV	NEVer
NPI	Numbering Plan Identifier
OSI	Open System Interconnection
P1	Parameter 1
P2	Parameter 2
P3	Parameter 3
PCB	Protocol Control Byte
PIN	Personal Identification Number
PPS	Protocol and Parameter Selection
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 (elementary) File Identifier

State H	High state logic level
State L	Low state logic level
SW	Status Word
TE	Terminal Equipment
TLV	Tag Length Value
TPDU	Transfer Protocol Data Unit
USIM	Universal Subscriber Identity Module
WI	Waiting time Integer
VPP	Programming power input, optional use by the card
WTX	Waiting Time eXtenstion
WWT	Work Waiting Time

3.4 Coding Conventions

The following coding conventions apply to the present document.

All lengths are presented in bytes, unless otherwise stated. Each byte is represented by bits 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.

In the UICC, all bytes specified as RFU shall be set to '00' and all bits specifies as RFU shall be set to 0. If the GSM and/or USIM application exists on a UICC or is built on a generic telecommunications card, then other values may apply for the non- GSM or non-USIM 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.

4 Physical Characteristics

Two physical types of UICCs are currently specified by the present document. These are the "ID-1 UICC" and the "Plug-in UICC". At least one of the card sizes shall supported by a terminal that is compliant to the present document.

The physical characteristics of both types of UICCs shall be in accordance with ISO/IEC 7816-1,2 [9,10] unless otherwise specified by the present document. The following additional requirements shall be applied to ensure correct operation in a Telecom environment.

4.1 ID-1 UICC

The physical characteristics of the ID-1 UICC shall conform to ISO/IEC 7816-1,2 [9,10].

The Terminal shall accept embossed ID-1 UICCs. The embossing shall be in accordance with ISO/IEC 7811-1 [8] and ISO/IEC 7811-3 [9]. The contacts of the ID-1 UICC shall be located on the front (embossed face, see ISO/IEC 7810 [7]) of the card.

4.2 Plug-in UICC

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/IEC 7816-2 [11] 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 figure 2 of ISO/IEC 7816-2 [11] are replaced by the corresponding values of figure 4.1.

The physical characteristics of the Plug-in UICC (Plug-in card) are defined in the present document.

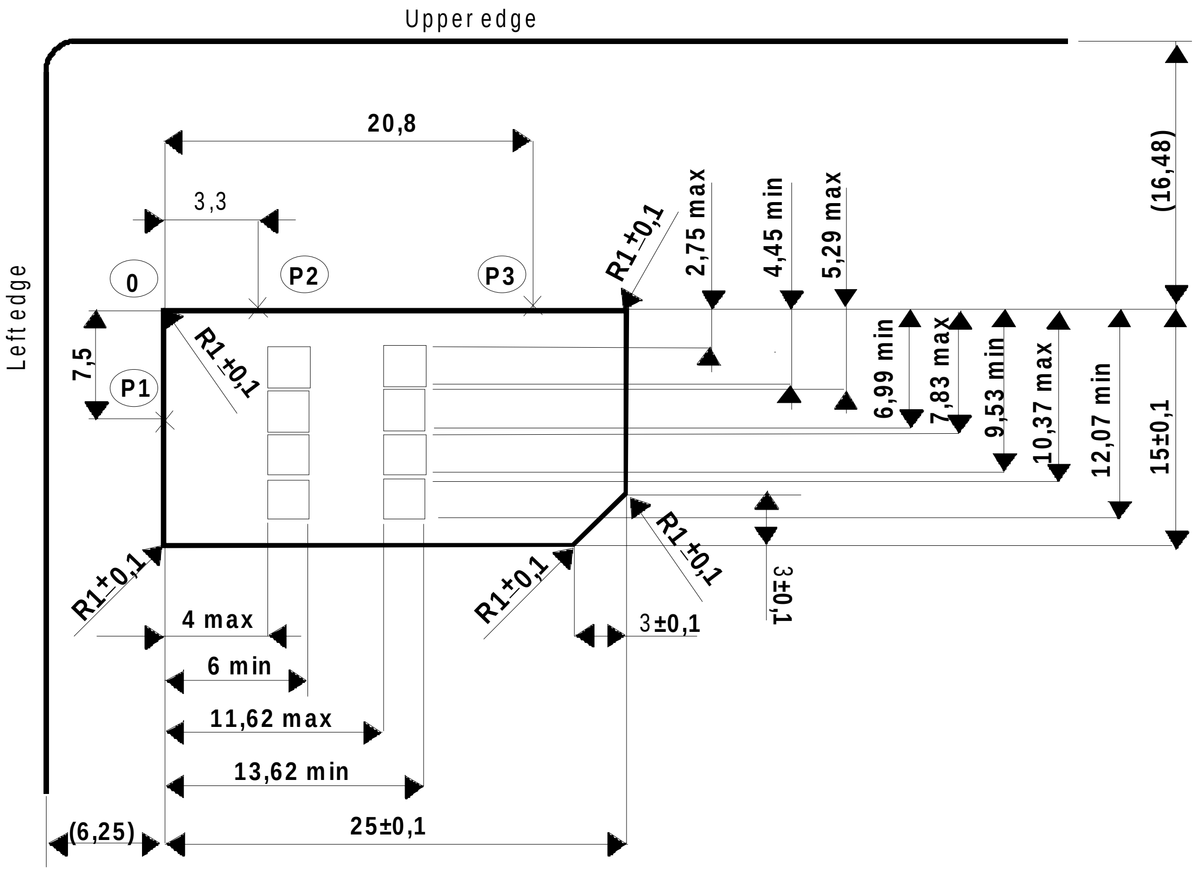


Figure 4.1: Plug-in UICC

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

4.4.1.1 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 a 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 cards.

4.4.1.2 UICC

Contacts C4 and C8 need not be provided by the UICC. If provided, 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 [12].

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 [12] 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 [12] 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 [12] 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 in the range 0 ± 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

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.

5.1 Specification of the 5V UICC – Terminal Interface

5.1.1 Supply voltage Vcc (contact C1)

The Terminal shall operate the UICC 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 6.4 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 Specification of the 3V UICC – Terminal Interface

5.2.1 Supply voltage Vcc (contact C1)

The Terminal shall operate the UICC within the following limits:

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)

The Terminal shall operate the UICC within the following limits:

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)

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 5.7: 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 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 5.8: 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 Specification of the 1.8V UICC – Terminal Interface

5.3.1 Supply voltage Vcc (contact C1)

The Terminal shall operate the UICC within the following limits:

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)

The Terminal shall operate the UICC within the following limits:

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)

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 5.11: 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 5.12 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 5.12: 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

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).

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 [12]. Class C and D are a further evolution of values specified in ISO/IEC 7816-3 [12]. It is possible to support a range of classes. The support must be consecutive e.g. AB, BC. A combination like AC is not allowed. Note 2: x indicates '0'.					

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 6.2: 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
NOTE: All values assume the maximum external clock.				

6.2.3 Application Related Electrical Parameters

The power consumption of a 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 table 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 6.3: 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	

Applications may specify their own maximum power consumption values.

NOTE: A negotiation mechanism between the Terminal and the UICC may be included in a future release of this specification.

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 ATR is defined in ISO/IEC 7816-3 [12].

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.

T=15 parameters shall be returned by the UICC.

Example ATRs are listed in Annex E of the present document.

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 clause 8 of ISO/IEC 7816-4 [13].

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 tag '31'. The second information sent by the card shall be the "card capabilities" data object. This data object is introduced by tag '73'. The other data objects are optional.

6.3.2 Speed enhancement

The Terminal and the UICC shall at least support (F,D) = (512,8) and (512,16) in addition to (372,1), the default values. However, other values may also be supported. 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 of the ATR.

6.4 PPS procedure

The Terminal and the UICC shall support the PPS procedure in order to use other transmission parameters as default values. The alternative parameters are indicated in the ATR. The interpretation of these parameters is according to ISO/IEC 7816-3 [12].

6.5 Cold Reset

The cold reset is performed according to clause 5.3.2 of ISO/IEC 7816-3 [12] and the UICC shall enter the negotiable mode. After a cold reset, the security status shall be reset.

6.6 Warm reset

The warm reset is performed according to clause 5.3.3 of ISO/IEC 7816-3 [12] and the UICC shall enter either the negotiable or the specific mode. After a warm reset, the security status shall be reset.

6.7 Clock stop mode

The UICC shall support the clock stop procedure as defined in this clause.

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.

6.8 Bit/character duration and sampling time

The bit/character duration and sampling time specified in ISO/IEC 7816-3 [12], subclause 6.3.2 are valid for all communications.

6.9 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 [12], subclause 6.3.3, 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.

6.10 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 or a 1.8V 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 6.2.3) shall support this procedure in addition to the supply voltage class indication in the ATR as defined in this standard.

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

7 Transmission Protocols

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 T=0 protocol is mandatory for the UICC and the Terminal. The T=1 protocol is mandatory for the Terminal and optional for the UICC.

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.

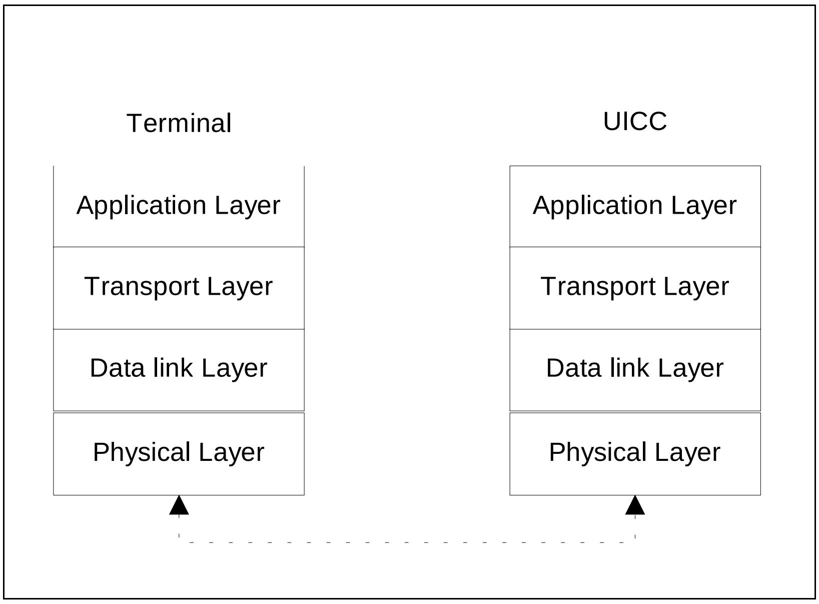


Figure 7.1: Layers

7.1 Physical Layer

Both protocols T=0 and T=1 shall 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 the protocols T=0 and T=1.

7.2.1 Character Frame

A character that is 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 7.2):

- 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 UICC and the terminal shall be in reception mode (I/O line in state H).

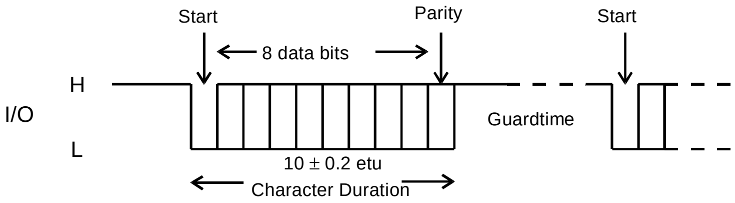


Figure 7.2: Character Frame

The data shall always be passed over the I/O line with the most significant byte first. The order of bits within a byte (that is, whether the least significant or most significant 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 protocol 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 × WI × 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.

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

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 C-APDU onto the C-TPDU is described in clause 7.3.

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

7.2.2.3 Command Processing

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.

Procedure bytes are used to keep up the communication between the terminal and the UICC. They shall not be transmitted to the Application Layer.

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

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 wait for a further status word (SW2). 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 T=0 protocol except for the Terminal 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 7.2 "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

The T=1 protocol is a half-duplex asynchronous block based transmission protocol. The protocol may be initiated after an ATR due to a Warm Reset,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 subclause 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 TA3 of the ATR.

The IFSD defines the maximum length of the information field of blocks that the terminal can receive. IFSD has a default value of 32 bytes and may be adjusted during the card session. The maximum value of the IFSD is 254 bytes.

7.2.3.1.2 Character waiting integer

CWI is used to calculate CWT and shall be in the range from 0 to 5. 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.

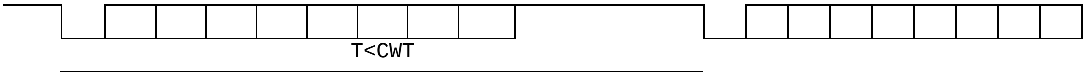


Figure 7.3: Character waiting time

The value of CWT may be calculated from the following equation: 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.

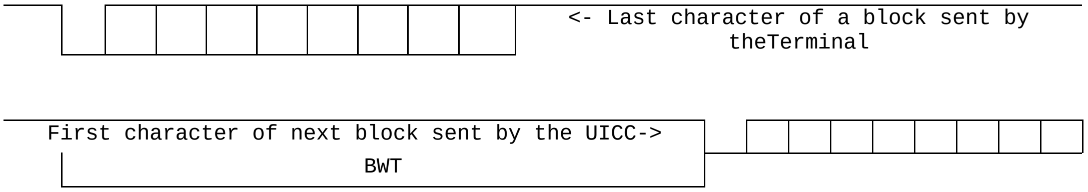

<- Last character of a block sent by theTerminal
First character of next block sent by the UICC->
BWT

Figure 7.4: Block waiting time

BWT is used to detect an unresponsive card.

7.2.3.1.5 Block guard time

BGT is defined as the minimum delay between the leading edge of two consecutive characters sent in opposite directions. The value of BGT shall be 22 etu.

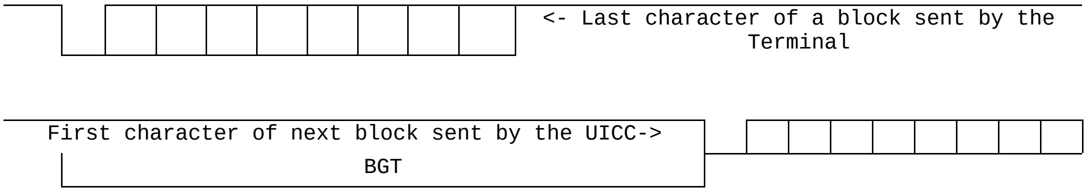

<- Last character of a block sent by the Terminal
First character of next block sent by the UICC->
BGT

Figure 7.5: Block guard time

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 TCi in the ATR is used to define which error detection code to use. LRC shall be used (b1=0). All other bits in TCi are RFU and shall be set to 0.

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:

Table 7.3: Block frame structure

Prologue field			Information field	Epilogue field
NAD	PCB	LEN	INF	EDC
1 byte	1 byte	1 byte	0-254 bytes	1 byte

The prologue field and the epilogue field are mandatory. 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:

Table 7.4: Node address byte

b8	b7	b6	b5	b4	b3	b2	b1
Unused	DAD			Unused	SAD		

In the first block sent from the Terminal, a logical connection is 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.

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 of block. In the T=1 protocol the following three different types of blocks are supported:

- Information block, I-block which is used to transfer command and response APDUs;
- 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 7.5: 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	RFU				

Table 7.6: 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 7.7: 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 errors
X	X	X	X	'X'	Other values are RFU

Table 7.8: Coding of PCB for an S-block

b8	b7	b6	b5	b4	b3	b2	b1
1	1	X	See table 7.7				

Table 7.9: Bits b5-b1 of PCB for an S-block

b5	b4	b3	b2	b1	Value	Meaning
0	0	0	0	0	'0'	Resynchronization
0	0	0	0	1	'1'	Information field
0	0	0	1	0	'2'	Abortion
0	0	0	1	1	'3'	Extension of BWT
0	0	1	0	0	'4'	Error on VPP State, see NOTE
X	X	X	X	X	'X'	Other values are RFU
NOTE: Not used by UICCs and Terminals conforming to this specification.						

The combination of b6 and b5 to 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 the number of bytes in the Information field of the block. The number of bytes in the information field may vary in the range from 0 to 254 bytes, depending on the type of block.

The value LEN='00' indicates that the information field is absent and the value 'FF' is RFU.

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.

Table 7.10: Information field

Type of block	INF used for
I-block	Transfer command and response APDUs.
R-block	Not used
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.

The LRC as defined in ISO/IEC 7816-3 [12] shall be used.

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 follows: 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-blocks are denoted as follows:

- S(RESYNCH request), a request of a resynchronization;
- S(RESYNCH response), an acknowledge of the resynchronization;
- 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;
- S(ABORT response), an acknowledge of the abortion of the chain function;
- 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 first block sent to the UICC shall 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 $N_s(S) \neq N_r(R)$, 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 value of IFSD, 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.

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 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).

Resynchronisation of the protocol may be attempted at three consecutive levels. If one level is unsuccessful, then the next level is tried.

- For the terminal, the three levels are:
  - Retransmission of blocks;
  - Use of S(RESYNCH request);
  - Card reset or deactivation.

- For the UICC, the three levels are:
  - Retransmission of blocks;
  - Use of S(RESYNCH response);
  - Without action by the terminal, the UICC becomes unresponsive.

7.2.3.4.1 Protocol initialisation

After an ATR due to a Warm reset 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 is not sent within BWT, the terminal shall 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.

7.2.3.4.2 Block dependent errors

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

When 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 (with the terminal) or the received response is not 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 (with the terminal), an R-block will be sent.

When 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, each has a length ≤ IFSC or IFSD. 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 0 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.

- When the UICC is the sender, all I-blocks of a chain shall have LEN ≤ IFSD bytes per block.

- When 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.

For reasons of efficiency, it is not recommended to send empty I-blocks.

7.3 Transport Layer

This subclause describes how the APDU are transported between the terminal and the UICC. For definition of the cases for Data in APDUs see subclause 7.4.

7.3.1 Transportation of an APDU using T=0

This subclause describes the mapping of C-APDUs and R-APDUs for T=0 protocol, the APDU exchange and the use of the GET RESPONSE command for case 2 and case 4.

7.3.1.1 Mapping of APDUs to TPDUs

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 UICC onto the R-APDU is dependent upon the length of the data returned.

Procedure bytes '61XX' and '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 '61XX' or '6CXX'.

Normal status on completion of processing a command is indicated if the UICC returns status words '9000' to the Transport Layer of the Terminal. Any other value of status words returned by the UICC indicates that the UICC has terminated the processing of the command, and that the processing was unsuccessful for the reasons indicated in the status words. 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 are for information, and 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 (P3 = '00').

	Terminal		UICC
Application Layer (C-APDU)	CLA INS P1 P2		
Transport Layer (C-TPDU)	CLA INS P1 P2 P3 = '00'	➔	Command Processing
Transport Layer (R-TPDU)	SW1||SW2	🡐	SW1||SW2
Application Layer (R-APDU)	SW1||SW2		

Figure 7.6

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.

The UICC 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 C-APDU is mapped onto the C-TPDU without any change.

	Terminal		UICC
Application Layer (C-APDU)	CLA INS P1 P2 Le		
Transport Layer (C-TPDU)	CLA INS P1 P2 P3=Le	→	Command Processing
Transport Layer (R-TPDU)	Le bytes of DATA||SW1||SW2	←	Le bytes of DATA||SW1||SW2
Application Layer (R-APDU)	Le bytes of DATA||SW1||SW2		

Figure 7.7

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:

The data returned (if present) is mapped onto the conditional body of the R-APDU. If no data is returned, the conditional body of R-APDU is left empty.

The status returned is mapped onto the mandatory trailer of the R-APDU without change.

7.3.1.1.3 Case 3

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

	Terminal		UICC
Application Layer (C-APDU)	CLA INS P1 P2 Lc [Lc bytes of DATA]		
Transport Layer (C-TPDU)	CLA INS P1 P2 P3=Lc [Lc bytes of DATA]	→	Command Processing
Transport Layer (R-TPDU)	SW1||SW2	←	SW1||SW2
Application Layer (R-APDU)	SW1||SW2		

Figure 7.8

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

	Terminal		UICC
Application Layer (C-APDU)	CLA INS P1 P2 Lc [Lc bytes of DATA] Le		
Transport Layer (C-TPDU)	CLA INS P1 P2 P3=Lc [Lc bytes of DATA]	→	Command Processing
	SW1||SW2 (61XX, XX=Luicc)	←	SW1||SW2 (61XX)
	Get Response, P3 = Luicc	→	Supply of Response Data
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		

Figure 7.9

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

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 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 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'),

then 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 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 of an I-block according to the four different cases described below. Each case is described in detail in the following sections.

7.3.2.1 Case 1

C-APDU is mapped to the INF of the I-block without any changes

	Terminal		UICC
Application Layer (C-APDU)	CLA INS P1 P2		
Transport Layer (C-TPDU, INF)	CLA INS P1 P2	→	Command Processing
Transport Layer (R-TPDU, INF)	SW1||SW2	←	SW1||SW2
Application Layer (R-APDU)	SW1||SW2		

Figure 7.10

The response received from the INF in the I-block is mapped unchanged to the R-APDU.

7.3.2.2 Case 2

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

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

Figure 7.11

The R-APDU consists of either the INF of the I-block or the concatenation of the INF of successive I-blocks all received in the same response, which all shall be chained.

7.3.2.3 Case 3

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

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

Figure 7.12

The INF 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 INF or is concatenated onto several successive I-blocks, which all shall be chained.

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

Figure 7.13

The response consists of either the INF of an I-block received in the response or the concatenation of INF of successive I-blocks in response, which all 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 7.11.

Table 7.11: Definition of Cases for Data in APDUs

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

7.4.1 Exchange of APDUs

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

Terminal		UICC
C-APDU	→	
	←	R-APDU
C-APDU	→	
	←	R-APDU

Figure 7.14

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.

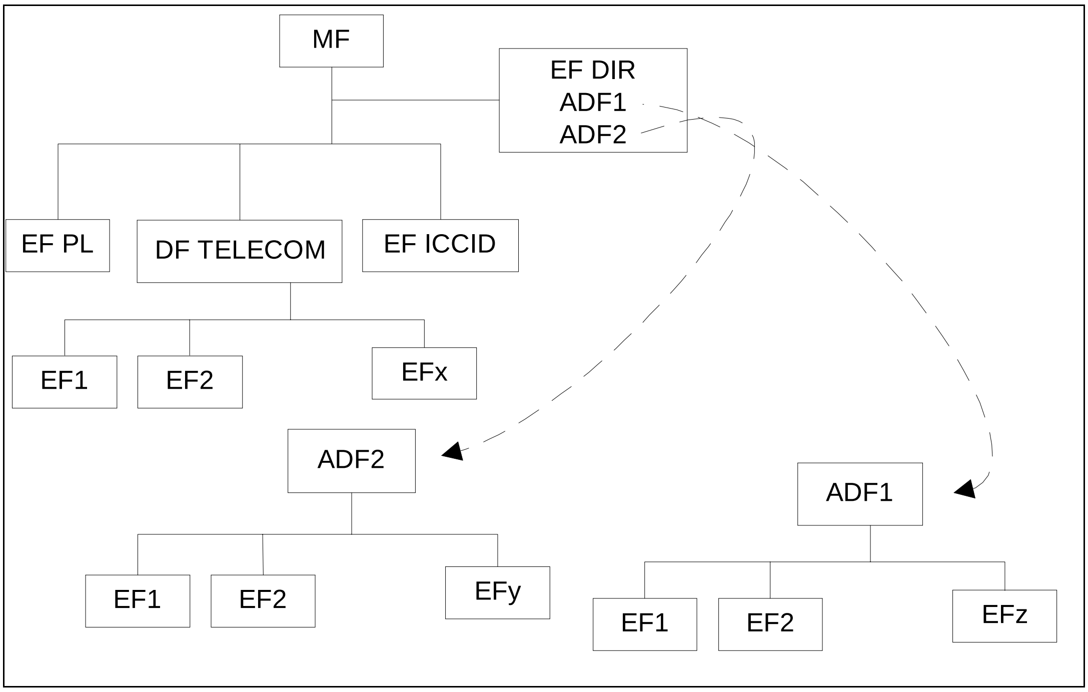


Figure 8.1: Example of an application structure

The present document 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.

$EF_{DIR,}$ EF_{PL} and EF_{ICCID} are all mandatory and reside directly under the Master File. See clause 13 of the present document for details.

$DF_{TELECOM}$ is optional. If present it resides under the MF and use the reserved FID '7F 10'. $DF_{TELECOM}$ contains application independent information.

8.2 File types

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

8.2.1 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 contains all the DFs and EFs of an application.

8.2.2 Elementary files

8.2.2.1 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.

8.2.2.2 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.

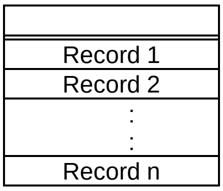


Figure 8.2: Structure of a linear fixed file

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 search.

If an action following selection of a record is aborted (e.g. due to an unsuccessful execution of a command), then the record pointer shall remain set at the record at which it was set prior to the action.

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.

8.2.2.3 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.

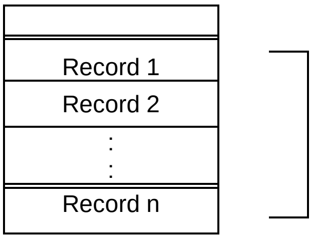


Figure 8.3: Structure of a cyclic file

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

If an action following selection of a record is aborted (e.g. due to an unsuccessful execution of a command), then the record pointer shall remain set at the record at which it was set prior to the action.

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.

8.3 File referencing

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

FIDs shall be subject to the following conditions:

- the FID shall be assigned at the time of creation of the file concerned;
- no two files under the same parent shall have the same ID;
- the immediate children of the current DF, the parent DF or the immediate children of the parent DF shall not have the same FID.

A path is a concatenation of FIDs. The path starts from MF or 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. The order of the FIDs is always in the direction from father to child.

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

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

8.4 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 5 file referencing methods defined in this sub-clause.

8.4.1 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. Only the ADF of the current application can be selected by FID.

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 (FID) referencing, from the last selected file:

- any file which is an immediate child of the Current Directory;
- any DF 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.

Figure 8.4 is an example of the logical structure for an application conforming to this document.

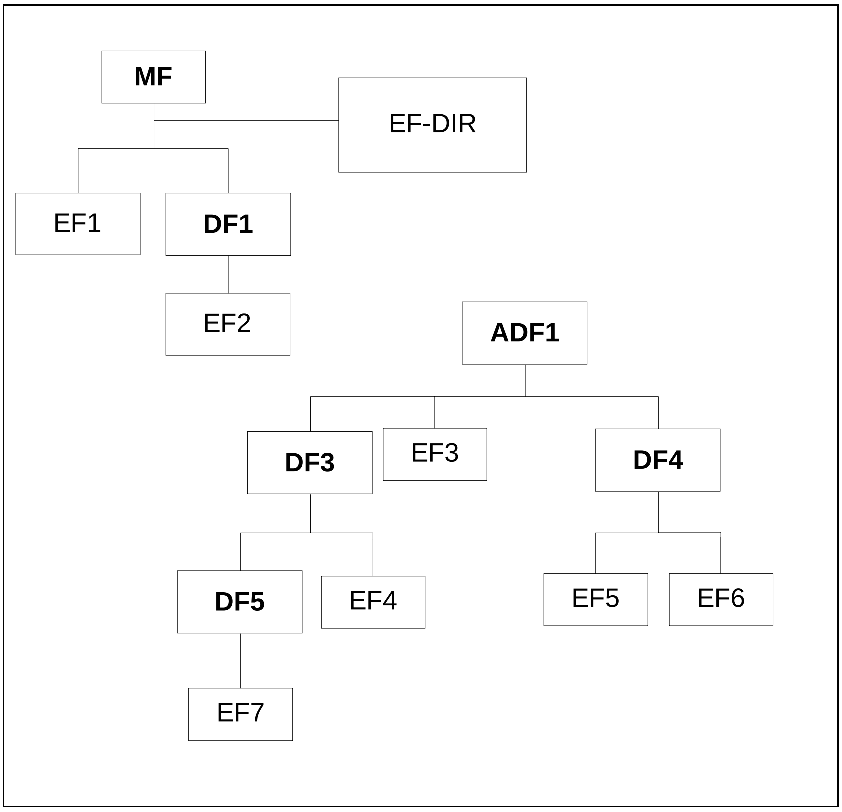


Figure 8.4: Example of a logical structure

The following table gives the valid selections for an application complying to this document for the logical structure in figure 8.4, if the FID 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

8.4.2 SELECT by Path Referencing

A file, DF or EF, may be referenced by path, as defined in clause 8.3. Table 8.2 contains examples of selection by path from figure 8.4.

It is not possible to select an ADF by path.

8.4.2 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. In this example, it is considered that the current application (ADF1) has been previously selected by DF name. The FID of ADF1 is noted '7FFF' (see clause 8.5) in the table.

It is not possible to select an ADF by path.

Table 8.2: Examples of file selection by path

<table>
<tr><th>Last selected DF</th><th>Beginning of the path</th><th>Example Selections</th></tr>
<tr><td>MF</td><td>MF</td><td>'EF1', 'EF-DIR', 'DF1', 'DF1||EF2'

'7FFF||DF3', '7FFF ||DF3||EF4',
'7FFF||DF3||DF5', '7FFF ||DF3||DF5||EF7'
'7FFFF||DF4', '7FFF ||DF4||EF5',
'7FFFF||DF4||EF6,
'7FFF||EF3'</td></tr>
<tr><td>DF1</td><td>Current DF</td><td>'EF2'</td></tr>
<tr><td>DF3</td><td>Current DF</td><td>'DF5', 'DF5||EF7', 'EF4'</td></tr>
<tr><td>DF4</td><td>Current DF</td><td>'EF5', 'EF6'</td></tr>
<tr><td>DF5</td><td>Current DF</td><td>'EF7'</td></tr>
</table>

8.4.5 Short File 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 File Identifier (SFI) as a part of the command:

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

Support of SFI for a specific file is indicated if the FCI of the file contains a TLV D0 with tag '88'.

If the length is 0 it indicates that the file does not support the SFI.

TLV DO is absent indicates that the 5 least significant bits of the FID may be used as SFI.

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

When the READ RECORD command contains a valid SFI, it sets the file as the current EF and resets the current record pointer. Subsequent records are read with the READ RECORD command without SFI.

When the UPDATE RECORD command contains a valid SFI, it sets the file as the current EF and resets the current record pointer. Subsequent records are updates with the UPDATE RECORD command without SFI.

When the INCREASE command contains a valid SFI, it sets the file as the current EF and resets the current record pointer. Subsequent records are increased with the INCREASE command without SFI.

When the SEARCH RECORD command contains a valid SFI, it sets the file as the current EF and resets the current record pointer. Subsequent records are searched with the SEARCH RECORD command without SFI.

8.5 Application characteristics

An application may be either explicitly or implicitly referenced.

An application is activated by explicit selecting it with the AID. This sets the application's ADF as the current ADF.

A current ADF can be referenced by FID with the implicit reference value '7F FF'.

8.5.1 Explicit Application selection

8.5.1.1 SELECT by DF Name

A selectable 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 a selectable application.

8.5.1.2 SELECT by partial DF Name

Selection by partial DF Name is for further study within 3GPP TSG-T WG3.

8.5.2 Application session activation

An application may need a session activation procedure to be performed after the selection. This procedure is outside the scope of this specification.

8.5.3 Application session deactivation

A selectable application shall have a session deactivation procedure. After this procedure has been executed the application is in a well-defined state and the session has ended.

8.5.4 GSM/USIM application interaction and restrictions

Activation of a USIM session excludes the activation of a GSM session. In particular, this implies that once a USIM application session has been activated, commands sent to the UICC with CLAss byte set to 'A0' shall return SW1SW2 '6E 00' (class not supported) to the terminal.

Similarly, activation of a GSM session excludes the activation of a USIM session.

At most one USIM session can be active at the same time.

8.6 Reservation of file IDs

The following FIDs are reserved by the present document.

ADF:
- operational use:
  '7F FF'

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 23' (DF_{FP-CTS}), '7F24' ($DF_{TIA/EIA-136'}$), '7F25' ($DF_{TIA/EIA-95'}$) and '7F 2X', where X ranges from '6' to 'F'.
  Note: '7F80' (DF_{PDC}) is used for the Japanese PDC specification
- reserved under '7F10':
  '5F50' ($DF_{GRAPHICS}$); '5F 3A' ($DF_{PHONEBOOK}$)

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 SEARCH RECORD are identical;
- the access conditions for the commands SELECT and STATUS are ALWays.

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 NOTE 1
3	Reserved for Future Use
4 to 14	see NOTE 2
15	NEVer

NOTE 1: This level is reserved for a second PIN that may be defined by an application.

NOTE 2: 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 an 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 $DF_{TELECOM}$.

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

9.2 PIN access condition

The UICC will have only one PIN. This PIN can either be enabled, disabled, blocked or unblocked. The access to any file by a command such as READ, SEARCH RECORD, UPDATE, INCREASE, DEACTIVATE or ACTIVATE can be protected by this PIN.

It is the responsibility of the UICC to store and remember the status of the PIN.

10 Structure of commands and responses

This clause defines the command and response APDUs supported by the UICC.

10.1 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 table10.1 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 in table 10.2:

Table 10.2: Cases of C-APDUs

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

10.1.1 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 10.4). 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 [13]). If the card capabilities data object is missing, logical channel b2=b1=0 is supported only.

Table 10.3: Coding of Class Byte

b8	b7	b6	b5	b4	b3	b2	b1	Value	Meaning
0	0	0	0	-	-	-	-	'0X'	The coding is according to ISO/IEC 7816-4 [13]
1	0	1	0	-	-	-	-	'AX'	Coded as ISO/IEC 7816-4 [13] unless stated otherwise
1	0	0	0	-	-	-	-	'8X'	Structured as ISO/IEC 7816-4 [13], coding and meaning is defined in this specification
-	-	-	-	X	X	-	-	-	Secure Messaging indication (see table 10.4)
-	-	-	-	-	-	X	X	-	Logical channel number (see clause 10.3)

Table 10.4: Coding of Security Messaging Indication

b4	b3	Meaning
0	0	No SM used between Terminal and card
0	1	Proprietary SM format
1	x	Secure messaging according to ISO/IEC 7816-4 [13] used
1	0	Command header not authenticated
1	1	Command header authenticated

By default no secure messaging is supported by the card, i.e. b4=b3=0, unless it is stated otherwise by an application.

10.1.2 Coding of Instruction Byte

Table 10.5 depicts coding of instruction byte of the commands.

Table 10.5: Coding of Instruction Byte of the Commands for a telecom application

COMMAND	CLA	INS
Command APDUs		
SELECT FILE	0X	'A4'
STATUS	8X	'F2'
READ BINARY	0X	'B0'
UPDATE BINARY	0X	'D6'
READ RECORD	0X	'B2'
UPDATE RECORD	0X	'DC'
SEARCH RECORD	0X	'A2'
INCREASE	8X	'32'
VERIFY	0X	'20'
CHANGE PIN	0X	'24'
DISABLE PIN	0X	'26'
ENABLE PIN	0X	'28'
UNBLOCK PIN	0X	'2C'
DEACTIVATE FILE	0X	'04'
ACTIVATE FILE	0X	'44'
AUTHENTICATE	0X	'88'
TERMINAL PROFILE	AX	'10'
ENVELOPE	AX	'C2'
FETCH	AX	'12'
TERMINAL RESPONSE	AX	'14'
MANAGE CHANNEL	0X	'70'
Transmission oriented APDUs		
GET RESPONSE	0X	'C0'

10.1.3 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.

10.1.4 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. The terminal may send from 1 to 255 bytes of command data.

10.1.5 Coding of Data Part

When present in a command or response APDU the structure of the data field is specific to each command.

10.1.6 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 1 to 256.

10.2 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 number of bytes received in the response APDU is denoted Lr (length of the response data field). The structure of the response APDU is shown in table 10.6.

Table 10.6: Contents of Response APDU

Code	Length	Description
Data	Lr	Response data string
SW1	1	Status byte 1
SW2	1	Status byte 2

Coding of SW1 and SW2 is presented in 10.2.1.

10.2.1 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 the coding of the status bytes.

10.2.1.1 Normal processing

Table 10.7: Status byte coding - normal processing

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

10.2.1.2 Postponed processing

Table 10.8: Status byte coding - postponed processing

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

10.2.1.3 Warnings

Table 10.9: Status byte coding - warnings

SW1	SW2	Description
'62'	'00'	- No information given, state of non volatile memory unchanged
'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 11.1.3.1
'63'	'00'	- No information given, state of non volatile memory changed
'63'	'81'	- File filled up by the last write
'63'	'CX'	- Command successful but after using an internal update retry routine 'X' times - Verification failed, 'X' retries remaining

10.2.1.4 Execution errors

Table 10.10: Status byte coding - execution errors

SW1	SW2	Description
'64'	'00'	- No information given, state of non-volatile memory unchanged
'65'	'00'	- No information given, state of non-volatile memory changed
'65'	'81'	- Memory problem

10.2.1.5 Checking errors

Table 10.11: Status byte coding - checking errors

SW1	SW2	Description
'67'	'XX'	- Wrong length
'6B'	'00'	- Wrong parameter(s) P1-P2
'6D'	'00'	- Instruction code not supported or invalid
'6E'	'00'	- Class not supported
'6F'	'XX'	- Technical problem, no precise diagnosis

10.2.1.5.1 Functions in CLA not supported

Table 10.12: Status byte coding - functions in CLA not supported

SW1	SW2	Description
'68'	'00'	- No information given
'68'	'81'	- Logical channel not supported
'68'	'82'	- Secure messaging not supported

10.2.1.5.2 Command not allowed

Table 10.13: Status byte coding - command not allowed

SW1	SW2	Description
'69'	'00'	- No information given
'69'	'81'	- Command incompatible with file structure
'69'	'82'	- Security status not satisfied
'69'	'83'	- Authentication method blocked
'69'	'84'	- Referenced data invalidated
'69'	'85'	- Conditions of used not satisfied
'69'	'86'	- Command not allowed (no EF selected)

10.2.1.5.3 Wrong parameters

Table 10.14: Status byte coding - wrong parameters

SW1	SW2	Description
'6A'	'80'	- Incorrect parameters in the data field
'6A'	'81'	- Function not supported
'6A'	'82'	- File not found
'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

10.2.1.6 Application errors

Table 10.15: Status byte coding - application errors

SW1	SW2	Error description
'98'	'50'	- INCREASE cannot be performed, max value reached.
'98'	'62'	- Authentication error, application specific

NOTE: Applications may define their own error codes.

10.2.2 Status Words of the Commands

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

Table 10.16: 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	AUTHENTICATE	TERMINAL PROFILE	ENVELOPE	FETCH	TERMINAL RESPONSE	MANAGE CHANNEL
90 00	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
91 XX	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*		*	*
93 00																		*			
98 50								*													
98 62																*					
62 00																					*
62 81					*	*															
62 82					*	*	*														
62 83	*																				
62 84	*																				
63 00										*	*	*	*								
63 CX			*	*				*		*	*	*	*								
64 00															*						
65 81			*	*				*		*	*	*	*			*					
67 XX			*	*	*			*		*	*	*	*								
69 81			*	*	*	*		*													
69 82			*	*	*	*	*	*		*	*	*	*	*	*						
69 83										*	*	*	*								
69 84										*	*	*	*			*					
69 85																*					
69 86			*	*	*			*													
6A 81	*		*	*	*	*		*		*	*	*	*	*							
6A 82	*		*	*	*	*				*	*	*	*								
6A 83				*		*															
6A 84				*																	
6A 86	*									*	*	*	*			*					
6A 87	*																				
6A 88										*	*	*	*			*					
6B 00		*	*		*												*	*	*	*	*
6E 00	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*
6F XX	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

The responses '91 XX', and '93 00' can only be given by a UICC to a Terminal supporting USAT (see 3G TS 31.111 [4]).

10.3 Logical channels

Only the basic channel is currently used.

11 Generic Commands

This clause 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 subclause 10.2.

11.1 SELECT

11.1.1 Functional description

This function selects a file according to the methods described in clause 8.4. After a successful selection the record pointer is undefined.

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.

11.1.2 Command Parameters and Data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
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								

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, optional template
NOTE: Whether the FCI information is returned or not depend on the type of APDU								

To avoid ambiguities when P1=P2='00' the following search order applies when selecting a file with a file ID (FID) as a parameter:

- immediate children of the current DF
- the parent DF
- the immediate children of the parent DF

11.1.3 Response Data

Returns the File Control Information (FCI) template of the selected file. FCI are coded as a BER-TLV as defined in table 11.X.

Table 11.3: Response data

Description	Section	Applies to	Length
FCI template tag = '6F'			1
Length of FCI template			1 or 2
File size	11.1.3.1	EFs	2
Total file size	11.1.3.2	Any file	2
File Descriptor	11.1.3.3	Any file	1 - 6
File Identifier	11.1.3.4	Any file	2
DF name	11.1.3.5	ADFs	1 - 16
Proprietary information	11.1.3.6	Any file	X
Security attributes	11.1.3.7	Any file	Y
Short file identifier	11.1.3.8	EFs	1

Other TLV objects may be present in the response data, ISO/IEC 7816-9 [17].

11.1.3.1 File size

Byte(s)	Description	Value	Length
1	Tag	'80'	1
2	Length	'01' or '02'	1
3 to 4	Number of data bytes in the file, excluding structural information		1 or 2

For a transparent EF, the file size is the length of the body part of the EF. For a linear fixed or cyclic EF, the file size is the record length multiplied by the number of records of the EF.

11.1.3.2 Total file size

Byte(s)	Description	Value	Length
1	Tag	'81'	1
2	Length	'01' or '02'	1
3 to 4	Number of data bytes in the file, including structural information if any		1 or 2

The total file size is the physical memory size occupied by the file on the card.

11.1.3.3 File Descriptor

Byte(s)	Description	Applies to	Value	Length
1	Tag		'82'	1
2	Length		'02' or '03'	1
3	File descriptor byte (see table 11.4)	Any file		1
4	RFU	Any file	RFU	1
5	Maximum record length	EFs with record structure		1

Table 11.4: File descriptor byte

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	X	-	-	-	-	-	-	**File accessibility**
0	0	-	-	-	-	-	-	Not shareable file
0	1	-	-	-	-	-	-	Shareable file
0	-	X	X	X	-	-	-	**File type**
0	-	0	0	0	-	-	-	Working EF
0	-	0	0	1	-	-	-	Internal EF
0	-	0	1	0	-	-	-	RFU
0	-	0	1	1	-	-	-	
0	-	1	0	0	-	-	-	
0	-	1	0	1	-	-	-	
0	-	1	1	0	-	-	-	
0	-	1	1	1	-	-	-	
0	-	-	-	-	X	X	X	**EF structure**
0	-	-	-	-	0	0	0	No information given
0	-	-	-	-	0	0	1	Transparent
0	-	-	-	-	0	1	0	Linear fixed
0	-	-	-	-	0	1	1	RFU
0	-	-	-	-	1	0	0	
0	-	-	-	-	1	0	1	
0	-	-	-	-	1	1	0	Cyclic
0	-	-	-	-	1	1	1	RFU
1	X	X	X	X	X	X	X	RFU

11.1.3.4 File identifier

Byte(s)	Description	Value	Length
1	Tag	'83'	1
2	Length	'02'	1
3 to 4	File identifier		2

11.1.3.5 DF name

Byte(s)	Description	Value	Length
1	Tag	'84'	1
2	Length	X	1
3 to 2+X	DF name		X

DF name is a string of bytes which is used to uniquely identify a dedicated file in the card.

11.1.3.6 Proprietary information

Byte(s)	Description	Value	Length
1	Tag	'85'	1
2	Length	X	1
3 4 to 2+X	UICC characteristics byte (see table 11.5)		1

Table 11.5: UICC characteristics byte

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
-	-	-	-	X	X	-	1	Clock stop allowed
-	-	-	-	0	0	-	1	No preferred level
-	-	-	-	0	1	-	1	High level preferred
-	-	-	-	1	0	-	1	Low level preferred
-	-	-	-	1	1	-	1	RFU
-	-	-	-	X	X	-	0	Clock stop not allowed
-	-	-	-	0	0	-	0	Never
-	-	-	-	0	1	-	0	Unless at high level
-	-	-	-	1	0	-	0	Unless at low level
-	-	-	-	1	1	-	0	RFU
-	X	X	X	-	-	-	-	Supply voltage class (see section 6.2.1)
X	-	-	-	-	-	X	-	RFU (shall be set to 0)

If bit b1 is coded 1, stopping the clock is allowed at high or low level. In this case, bit b3 and b4 give information about the preferred level (high or low, respectively) at which the clock may be stopped.

If b1 is coded 0, the clock may be stopped only if the mandatory condition b3 = 1 (i.e. stop at high level) or b4 = 1 (i.e. stop at low level) is fulfilled. If all 3 bits are coded 0, then the clock shall not be stopped.

11.1.3.7 Security attributes

Byte(s)	Description	Value	Length
1	Tag	'86'	1
2	Length	X	1
3 to 2+X	TBD		X

11.1.3.8 Short file identifier

When this TLV is present in the template, it indicates that the file has a Short File Identifier (SFI). The value of the SFI depends on the length of the TLV (see below).

Byte(s)	Description	Value	Length
1	Tag	'88'	1
2	Length	'00' or '01'	1
3	Short file identifier		0 or 1

If the length of the TLV is 1, the SFI value is indicated in the 5 most significant bits (bits b8 to b4) of the TLV value field. In this case, bits b3 to b1 shall be set to 0.

11.2 STATUS

11.2.1 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).

11.2.2 Command parameters

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	00
P2	00
Le	Empty, '00', or 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.

11.3 READ BINARY

11.3.1 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.

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

Output:
- string of bytes.

11.3.2 Command parameters

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
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 read	Le

11.4 UPDATE BINARY

11.4.1 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.

11.4.2 Command parameters and data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
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.

11.5 READ RECORD

11.5.1 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.

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

Output:
- the record.

11.5.2 Command parameters

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
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 File 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 within the scope of this specification and shall be set to '00' by the Terminal.

Response data:

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

11.6 UPDATE RECORD

11.6.1 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.

11.6.2 Command parameters and data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
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.

11.7 SEARCH RECORD

11.7.1 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.

11.7.2 Command parameters and data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	Record number
P2	See table 11.8
Lc	Length of the subsequent data field
Data	Offset indication followed by search string
Le	Empty or maximum length of response data

Table 11.8: 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 11.9
-	-	-	-	-	1	1	1	Proprietary
NOTE: This value is reserved by ISO/IEC 7816-9 [17]								

Table 11.9: 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 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	Start forward search from next record
-	-	-	-	-	1	1	1	Start backward search form previous record
NOTE: This value is reserved by ISO/IEC 7816-9 [17]								

Response data:

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

11.8 INCREASE

11.8.1 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 function does 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.

11.8.2 Command parameters and data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	See table 11.10
P2	'00'
Lc	Length of the subsequent data field
Data	Value to be added
Le	Length of the response data

Table 11.10: Coding of P1

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	0	0	0	0	0	Increase the currently selected EF
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
NOTE: X denotes the length of the record.		

11.9 VERIFY PIN

11.9.1 Functional description

The Verify PIN command initiates the comparison in the UICC of the PIN verification data sent from the Terminal with the PIN reference data stored in the card. 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.

11.9.2 Command parameters:

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	'00'
P2	Qualifier of the reference data, see table 11.11
Lc	Length of the subsequent data field = '08'
Data	PIN value
Le	Not present

Table 11.11: Coding of P2

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	0	0	0	0	0	Not supported
0	-	-	-	-	-	-	-	Global reference data (e.g. MF specific PIN)
1	-	-	-	-	-	-	-	Specific reference data (e.g. DF specific/application dependent PIN)
-	X	X	-	-	-	-	-	'00' (other values are RFU)
-	-	-	X	X	X	X	X	Reference data number ('01' to '1F')

The five least significant bits of parameter P2 specify the PIN number.

Command data:

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

11.10 CHANGE PIN

11.10.1 Functional description

The Change PIN command is used to initiate the comparison of the verification data with the PIN, and then to conditionally replace the existing PIN with the new PIN sent to the UICC in the command.

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.

11.10.2 Command parameters

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	'00'
P2	As specified for the VERIFY PIN command, see 11.9
Lc	Length of the subsequent data field = '10'
Data	Old PIN value, new PIN value
Le	Not present

NOTE: "Change PIN" is named "exchange reference data" in ISO/IEC 7816-8 [16].

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

11.11 DISABLE PIN

11.11.1 Functional description

The Disable PIN command is used to switch off the requirement to compare the PIN verification data with the PIN reference data.

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 PINs 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, the PIN 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.

11.11.2 Command parameters

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	'00'
P2	As specified for the VERIFY PIN command, see 11.9
Lc	Length of the subsequent data = '08'
Data	PIN value
Le	Not present

Command data:

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

11.12 ENABLE PIN

11.12.1 Functional description

The Enable PIN command is used to switch on the requirement to compare the PIN verification data with the PIN reference data. It is the reverse function of DISABLE PIN.

The function ENABLE PIN shall not be executed by the selected application when PIN is already enabled or blocked.

Every application shall specify whether this function is applicable to all PINs 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.

Input:
- PIN.

Output:
- none.

11.12.2 Command parameters

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	'00'
P2	As specified for the VERIFY PIN command, see 11.9
Lc	Length of the subsequent data = '08'
Data	PIN value
Le	Not present

Command data:

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

11.13 UNBLOCK PIN

11.13.1 Functional description

The Unblock PIN command is used to reset the PIN retry counter to its initial value and then to conditionally set a new PIN value . This function may be performed whether or not the relevant PIN is blocked (e.g. by 3 consecutive wrong PIN presentations).

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.

11.13.2 Command parameters

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	'00'
P2	As specified for the VERIFY PIN command (see 11.9)
Lc	Length of the subsequent data field = '10'
Data	UNBLOCK PIN value, new PIN value
Le	Not present

Command data:

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

11.14 DEACTIVATE FILE

11.14.1 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.

11.14.2 Command parameters

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	As specified for the SELECT command (see 11.1)
P2	As specified for the SELECT command (see 11.1)
Lc	Length of subsequent data field or empty
Data	File ID, DF name (AID), or path to file, according to P1
Le	Not present

11.15 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.

11.15.2 Command parameters

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	As specified for the SELECT command (see 11.1)
P2	As specified for the SELECT command (see 11.1)
Lc	Length of subsequent data field or empty
Data	File ID, DF name (AID), or path to file, according to P1
Le	Not present

11.16 AUTHENTICATE

11.16.1 Functional description

An appropriate application shall be selected in the UICC before issuing this command. The function initiates the computation of authentication data by the UICC using a challenge sent from the terminal and a secret stored in the UICC.

Input:

- challenge data.

Output:

- authentication and ciphering data.

11.16.2 Command parameters and data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	'00'
P2	See table 11.12
Lc	Length of the subsequent data field
Data	Authentication related data
Le	Length of the response data

NOTE 1: Parameter P1='00' indicates that no information on the algorithm is given. The algorithm is implicitly known in the context of the selected application.

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. MF specific KEY)
1	-	-	-	-	-	-	-	Specific reference data (e.g. DF specific/application dependent KEY)
-	X	X	-	-	-	-	-	'00' (other values are RFU)
-	-	-	X	X	X	X	X	Reference data number ('01' to '1F')

NOTE 2: Parameter P2='00' indicates that no information on the key is given. The key is implicitly known in the context of the selected application.

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.		

11.17 TERMINAL PROFILE

11.17.1 Functional description

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

Input:
- terminal profile.

Output:
- none.

11.17.2 Command parameters and data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	'00'
P2	'00'
Lc	Length of the subsequent data field
Data	Terminal profile data defined in 3G TS 31.111 [4]
Le	Not present

11.18 ENVELOPE

11.18.1 Functional description

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

Input:
- data string.

Output:
- the structure of the data is defined in 3G TS 31.111 [4].

11.18.2 Command parameters and data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	'00'
P2	'00'
Lc	Length of the subsequent data field
Data	Data string, see 3G TS 31.111 [ ]
Le	Length of expected data

Response data:

Structure of the response data is defined in 3G TS 31.111 [4] for USAT applications.

11.19 FETCH

11.19.1 Functional description

This function is used to transfer a command from the selected application to the Terminal (e.g. from a USIM Application Toolkit application).

Input:
- none.

Output:
- data string containing a command for the Terminal (e.g. a USIM Application Toolkit command).

11.19.2 Command parameters and data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
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 3G TS 31.111 [4] for USAT applications.

11.20 TERMINAL RESPONSE

11.20.1 Functional description

This function is used to transfer from the Terminal to the selected application the response to a previously fetched application command (e.g. a USIM Application Toolkit command).

Input:
- data string containing the response.

Output:
- none.

11.20.2 Command parameters and data

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
P1	'00'
P2	'00'
Lc	Length of the subsequent data field
Data	Terminal response data defined in 3G TS 31.111 [4]
Le	Not present

11.21 MANAGE CHANNEL

The MANAGE CHANNEL command is for further study by 3GPP TSG-T WG3.

12 Transmission Oriented Commands

This clause lists all transport specific commands. Currently there is only one such command.

12.1 T=0 specific commands

12.1.1 GET RESPONSE

12.1.1.1 Functional description

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

The response data depends on the preceding command. Response data is available after the commands AUTHENTICATE, SEARCH RECORD, 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, no precise diagnosis" as a reaction to the GET RESPONSE.

Since the MF is implicitly selected after activation, 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.

12.1.1.2 Command parameters

Table 12.1: Parameters for GET RESPONSE

Code	Value
CLA	As specified in 10.1.1
INS	As specified in 10.1.2
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.

13 Application independent files

13.1 EF$_{DIR}$

EF$_{DIR}$ is a linear fixed file. This file is under the responsibility of the issuer when at the MF level. It can be under a different responsibility, 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 ALW UPDATE ADM			
Bytes	Description	M/O	Length
1 – X	Application template TLV object	M	X bytes

The EF consists of one or more records, with each record able to hold one entry. Each entry in the EF_{DIR} is an application template Data Object (DO) as defined in ISO/IEC 7816-5 [14]. 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. Within the scope of this specification, all other DOs are optional.

In table 13.2 the coding of the mandatory DOs and the optional DOs that has special meaning to this document. All other DOs are according to ISO/IEC 7816-5 [14].

Table 13.2: Coding of an application template entry

Length	Description	Status
1	Application template tag = '61'	M
1	Length of the application template = '03'-'7F'	M
1	Application Identifier tag = '4F'	M
1	AID length = '01'-'10'	M
'01' – '10'	AID value. For 3G applications, see 3G TS 31.110 [3]	M
1	Application label tag = '50'	O
1	Application label length	O
NOTE 1	Application label value	O
NOTE 1: 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 ISO/IEC 7816-5 [14] the application label is limited to 16 octets, this does not apply to this specification it is however recommended that the number of octets does not exceed 32. NOTE 2: Other DOs from ISO/IEC 7816-5 [14] may, at the application issuer's discretion, be present as well.		

13.2 EF_{ICCID} (ICC Identification)

This EF provides a unique identification number for the UICC.

Table 13.3: EF_{ICCID} at MF-level

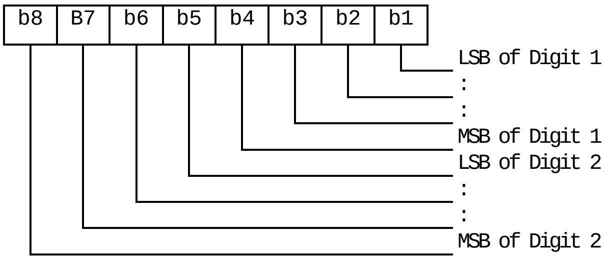
Identifier: '2FE2'	Structure: transparent		Mandatory
File size: 10 bytes		Update activity: low	
Access Conditions: READ ALW UPDATE NEV			
Bytes	Description	M/O	Length
1 – 10	Identification number	M	10 bytes

- Identification number

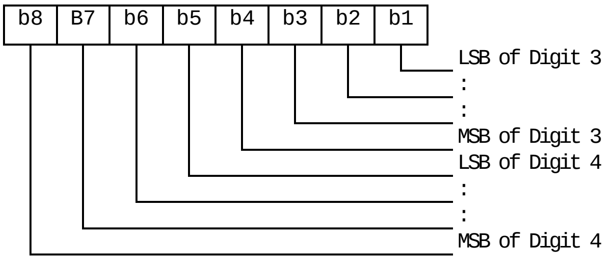
Contents:
according to CCITT Recommendation E.118 [5].
Purpose:
card identification number.

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 13.4: 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 [6]. Each alpha-numeric character shall be coded on one byte using the SMS default 7-bit coded alphabet as defined in 3G TS 23.038 [1] with bit 8 set to 0.

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

14 Application independent protocol

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, file EF of the Current Directory, that support the SFI, can 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 , file EF of the Current Directory, that support the SFI, can 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

Increasing 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, file EF of the Current Directory, that support the SFI, can be increased without explicitly selecting the EF. The Terminal selects the DF or ADF and sends an INCREASE command. This contains the short file identifier of the EF and 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**.**

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

- 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 SEARCH RECORD 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 PINs 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 PIN which protects the application, the procedure ends and is finished unsuccessfully.

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 PIN which protects the application, the new PIN presented by the Terminal is stored instead of the old one 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 shall 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 which protects the application, 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 which protects 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 shall 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 which protects the application, 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 which protects 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 of 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 of the application, the procedure ends and is finished unsuccessfully.

14.3 Application selection procedures

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 presents the applications that it supports to the user who may make a selection. If only one supported application is found this may be implicitly selected.

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.

14.3.3 Direct application selection with partial AID

This is ffs.

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 ends successfully the UICC enters the operation state. If the initialisation procedure does not end successfully, the UICC remains in the application management state and sends an indication to the user that it was not possible to activate the selected application.

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 UICC application session termination

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

14.5 Miscellaneous procedures

14.5.1 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 Terminal's default language is selected. The Terminal then performs an application selection procedure according to clause 14.3.

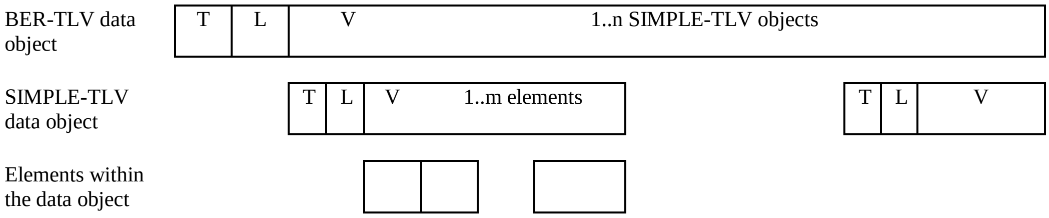
14.5.2 UICC presence detection

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.

14.5.3 UICC Preferred Language request

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

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



Commands and responses which are sent across the interface may contain BER-TLV data objects. Commands and responses shall only contain one BER-TLV object.

The tag is a constant value, length one byte.

The length is coded onto 1 or 2 bytes according to ISO/IEC 7816-6 [15]. 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 [15].

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, in accordance with ISO/IEC 7816-6 [15], never used as tag values for BER-TLVs. Padding characters are not allowed.

See ISO/IEC 7816-6 [15] 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" – In this state 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. 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;
- "Operation" - This state is entered after the application session activation procedure for at least one application has ended successfully.

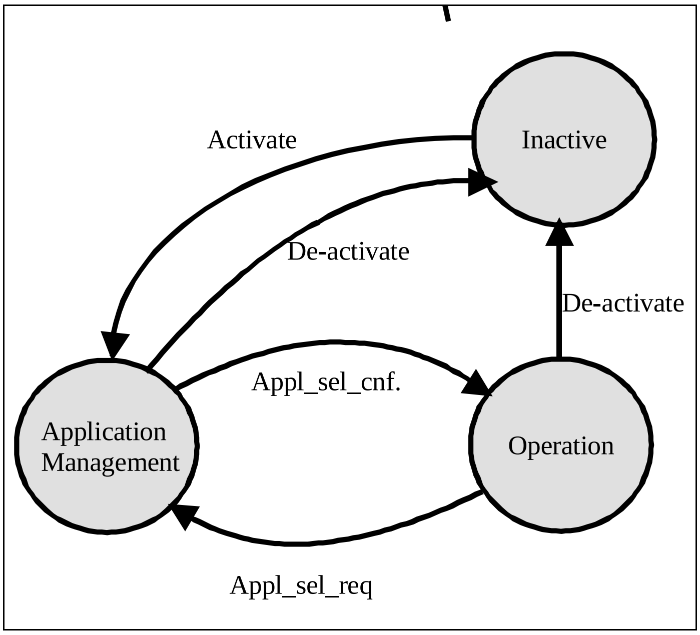


Figure B.1: UICC states

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 $\overline{\text{INS}}$ is not illustrated.
- [Data(X)] means X bytes of data.
- Case 2 and 4 commands may 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 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 P3=00]	⇒		
			⇐	90 00

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

C.1.2 Case 2 Command

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

	Terminal			UICC
C-TPDU	[CLA INS P1 P2 P3=00]	⇒		
			⇐	6C Luicc
C-TPDU	[CLA INS P1 P2 Luicc]	⇒		
			⇐	INS [Data(Luicc)] 90 00

An 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 be forced to re-issue the command. A more practical approach is the following:

	Terminal			UICC
C-TPDU	[CLA INS P1 P2 P3=00]	⇒		
			⇐	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

An R-APDU of {[Data(YY + ZZ)] 90 00} is returned from the UICC to the Terminal.

C.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 P3=Lc]	⇒		
			⇐	[INS]
	[Data(Lc)]	⇒		
			⇐	90 00

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

C.1.4 Case 4 Command

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

	Terminal			**UICC**
C-TPDU	[CLA INS P1 P2 P3=Lc]	⇒		
			⇐	[INS]
	[Data(Lc)]	⇒		
			⇐	61 Luicc
GET RESPONSE	[00 C0 00 00 Luicc]	⇒		
			⇐	C0 [Data(Luicc)] 90 00

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

C.1.5 Case 2 Commands Using the '61' and '6C' Procedure Bytes

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

	Terminal			**UICC**
C-TPDU	[CLA INS P1 P2 P3=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

An R-APDU of {[Data(YY + ZZ)] 90 00} is returned from the UICC to the Terminal.

C.1.6 Case 4 Command Using the '61' Procedure Byte

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

	Terminal			**UICC**
C-TPDU	[CLA INS P1 P2 P3=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

An R-APDU of {[Data(XX + YY)] 90 00} is returned from the UICC to the Terminal.

C.1.7 Case 4 Command with Warning Condition

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

	Terminal			UICC
C-TPDU	[CLA INS P1 P2 P3=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

An 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 [18] are 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 the GSM default alphabet (see 3G TS 23.038 [1]), 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.

Annex E (informative): ATR Examples

This annex gives examples of ATRs that can be returned by a UICC after a reset.

1) Example 1: Cold reset for a T=0 protocol only UICC.

Character	Value	Description
TS	'3B' or '3F'	Indicates direct or inverse convention
T0	'97'	TA1 and TD1 are present 7 bytes of historical bytes
TA1	'94'	Clock rate conversion factor FI=9 (F=512) Baud rate adjustment factor DI=4 (D=8)
TD1	'80'	TD2 only is present
TD2	'1F'	TA3 only is present Global interface bytes following
TA3	'42'	Clock stop supported (low electrical state) 3V UICC
T1	'80'	
T2	'31'	Card data services

T3	'E0'	SELECT by AID supported
		SELECT by partial AID supported
		EFDIR present

T4	'73'	Card capabilities
T5	'FE'	SFI supported
T6	'20'	
T7	'00'	No extended Lc and Le No Logical channels supported
TCK	'XX'	Check byte

2) Example 2: Cold reset for a T=0 and T=1 protocol UICC.

Character	Value	Description
TS	'3B' or '3F'	Indicates direct or inverse convention
T0	'97'	TA1, and TD1 are present 7 bytes of historical bytes
TA1	'94'	Clock rate conversion factor FI=9 (F=512) Baud rate adjustment factor DI=4 (D=8)
TD1	'80'	Only TD2 is present Protocol T=0 supported by UICC
TD2	'B1'	TA3, TB3 and TD3 are present Protocol T=1 supported by UICC
TA3	'FE'	IFSC is 254 bytes long
TB3	'00'	Block Waiting Integer=0 Character Waiting Integer=0
TD3	'1F'	Only TA4 is present Global interface bytes following
TA4	'42'	Clock stop supported (low electrical state) 3V UICC
T1	'80'	
T2	'31'	Card data services

T3	'E0'	SELECT by AID supported
		SELECT by partial AID supported
		EFDIR present

T4	'73'	Card capabilities
T5	'FE'	SFI supported
T6	'20'	
T7	'00'	No extended Lc and Le No Logical channels supported
TCK	'XX'	Check byte

3) Example 3: Warm reset and T=1 protocol ask by UICC

Character	Value	Description
TS	'3B' or '3F'	Indicates direct or inverse convention
T0	'97'	TA1, and TD1 are present 7 bytes of historical bytes
TA1	'94'	Clock rate conversion factor FI=9 (F=512) Baud rate adjustment factor DI=4 (D=8)
TD1	'91'	TA2 and TD2 are present Protocol T=1 supported by UICC
TA2	'81'	Protocol T=1 used in specific mode Parameters indicated by the interface bytes, and card is enable to change mode
TD2	'B1'	TA3, TB3 and TD3 are present Protocol T=1 supported by UICC
TA3	'FE'	IFSC is 254 bytes long
TB3	'00'	Block Waiting Integer=0 Character Waiting Integer=0
TD3	'0F'	Global interface bytes following (none so use previous ones)
T1	'80'	
T2	'31'	Card data services

T3	'E0'	SELECT by AID supported
		SELECT by partial AID supported
		EF$_{DIR}$ present

T4	'73'	Card capabilities
T5	'FE'	SFI supported
T6	'20'	
T7	'00'	No extended Lc and Le No Logical channels supported
TCK	'XX'	Check byte

History

Document history		
V3.0.0	January 2000	Publication