भारतीय मानक

Indian Standard

IS/ISO/IEC 15693-3 : 2019

व्यक्तिगत पहचान के लिए कार्ड और सुरक्षा उपकरण — संपर्क रहित विसिनिटी ऑब्जेक्ट्स

भाग 3 एंटीकोलिजन और ट्रांसमिशन प्रोटोकॉल

Cards and Security Devices for Personal Identification — Contactless Vicinity Objects

Part 3 Anticollision and Transmission Protocol

ICS 35.240.15

भारतीय मानक ब्यूरो
BUREAU OF INDIAN STANDARDS
मानक भवन, 9 बहादुर शाह ज़फर मार्ग, नई दिल्ली - 110002
MANAK BHAVAN, 9 BAHADUR SHAH ZAFAR MARG
NEW DELHI - 110002
www.bis.gov.in www.standardsbis.in

December 2022 **Price Group 16**

Identification and Data Capture Techniques, Cards and Security Devices Sectional Committee, LITD 16

NATIONAL FOREWORD

This Indian Standard (Part 3) which is identical with ISO/IEC 15693-3 : 2019 ‘Cards and security devices for personal identification — Contactless vicinity objects — Part 3 : Anticollision and transmission protocol’ issued by the International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) was adopted by the Bureau of Indian Standards on the recommendation of the Identification & Data Capture Techniques, Cards and Security Devices Sectional Committee and approval of the Electronics and Information Technology Division Council.

Other parts in this series are:-

Part 1 Physical characteristics
Part 2 Air interface and initialization

The text of ISO/IEC Standard has been approved as suitable for publication as an Indian Standard without deviations. Certain conventions are, however, not identical to those used in Indian Standards. Attention is particularly drawn to the following:

a) Wherever the words ‘International Standard’ appear referring to this standard, they should be read as ‘Indian Standard’.
b) Comma (,) has been used as a decimal marker while in Indian Standards, the current practice is touse a point (.) as the decimal marker.

In this adopted standard, reference appears to certain International Standards for which the Indian Standard also exists. The corresponding Indian Standard which is to be substituted in its respective place is listed below along with its degree of equivalence for the edition indicated:

International Standard	*Corresponding Indian Standard*	*Degree of Equivalence*
ISO/IEC 15693-1 Cards and security devices for personal identification — Contactless vicinity objects — Part 1: Physical characteristics	IS/ISO/IEC 15693-1 Cards and security devices for personal identification contactless vicinity objects: Part 1 Physical characteristics	Identical with ISO/IEC 15693-1 : 2019
ISO/IEC 15693-2 Cards and security devices for personal identification — Contactless vicinity objects — Part 2: Air interface and initialization	ISO/IEC 15693-2 Cards and security devices for personal identification contactless vicinity objects: Part 2 Air interface and initialization	Identical with ISO/IEC 15693-2 : 2019

The technical committee has reviewed the provisions of the following International Standard referred in this adopted standard and has decided that it is acceptable for use in conjunction with this standard:

International Standard	*Title*
ISO/IEC 13239	Information technology — Telecommunications and information exchange between systems — High-level data link control (HDLC) procedures

For the purpose of deciding whether a particular requirement of this standard is complied with, the final value, observed or calculated, expressing the result of a test or analysis, shall be rounded off in accordance with IS 2 : 2022 ‘Rules for rounding off numerical values (*second revision*). The number of significant places retained in the rounded off value should be the same as that of the specified value in this standard.

Contents

Page

Introduction

ISO/IEC 15693 (all parts) is one of a series of International Standards describing the parameters for identification cards as defined in ISO/IEC 7810 and the use of such cards for international interchange.

This document describes the anticollision and transmission protocols.

This document does not preclude the incorporation of other standard technologies on the card.

Contactless card standards cover a variety of types as embodied in the ISO/IEC 10536 series (close-coupled cards), the ISO/IEC 14443 series (proximity cards) and the ISO/IEC 15693 series (vicinity cards). These are intended for operation when very near, nearby and at a longer distance from associated coupling devices, respectively.

Indian Standard

CARDS AND SECURITY DEVICES FOR PERSONAL IDENTIFICATION — CONTACTLESS VICINITY OBJECTS
PART 3 ANTICOLLISION AND TRANSMISSION PROTOCOL

1 Scope

This document specifies:

— protocols and commands;

— other parameters required to initialize communications between a vicinity integrated circuit card and a vicinity coupling device;

— methods to detect and communicate with one card among several cards ("anticollision");

— optional means to ease and speed up the selection of one among several cards based on application criteria.

This document does not preclude the incorporation of other standard technologies on the card as described in Annex A.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 13239, *Information technology — Telecommunications and information exchange between systems — High-level data link control (HDLC) procedures*

ISO/IEC 15693-1, *Cards and security devices for personal identification — Contactless vicinity objects — Part 1: Physical characteristics*

ISO/IEC 15693-2, *Cards and security devices for personal identification — Contactless vicinity objects — Part 2: Air interface and initialization*

3 Terms, definitions, symbols and abbreviated terms

3.1 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 15693-1, ISO/IEC 15693-2 and the following apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1.1
anticollision loop
algorithm used to prepare for and handle a dialogue between a VCD and one or more VICCs from several in its energizing field

3.1.2
byte
string that consists of 8 bits of data designated b1 to b8, from the most significant bit (MSB, b8) to the least significant bit (LSB, b1)

3.1.3
payload
part of the message data which conveys information relating to the use of the security commands defined in this document

Note 1 to entry: The message data is defined in the ISO/IEC 29167 series.

3.1.4
ResponseBuffer
VICC memory area where the result of a cryptographic operation is stored which may be retrieved using a ReadBuffer command

3.1.5
Write alike
command or request resulting in a non-volatile change to the contents of the VICC memory

3.2 Symbols and abbreviated terms

f_C	frequency of operating field (carrier frequency)
AFI	application family identifier
CRC	cyclic redundancy check
CS	Cryptographic Suite
CSI	Cryptographic Suite Identifier
DSFID	data storage format identifier
EOF	end of frame
LSB	least significant bit
LSByte	least significant byte
MSB	most significant bit
MSByte	most significant byte
RFU	reserved for future use
SOF	start of frame
UID	unique identifier
VCD	vicinity coupling device
VICC	vicinity integrated circuit card

4 Definition of data elements

4.1 UID

The VICCs are uniquely identified by a 64 bits UID. This is used for addressing each VICC uniquely and individually, during the anticollision loop and for one-to-one exchange between a VCD and a VICC.

The UID shall be set permanently by the IC manufacturer in accordance with Table 1.

Table 1 — UID format

MSB LSB

64 57	56 49	48 1
'E0'	IC Mfg code	IC manufacturer serial number

The UID comprises:

— the MSByte (bits 64 – 57) which shall be 'E0';

— the IC manufacturer code (bits 56 – 49) defined in ISO/IEC 7816-6;

— a unique serial number (bits 48 – 1) assigned by the IC manufacturer.

4.2 AFI

The AFI represents the type of application targeted by the VCD and is used to extract from all the VICCs present only the VICCs meeting the required application criteria.

It may be programmed and locked by the respective commands.

The AFI is coded on one byte, which constitutes 2 nibbles of 4 bits each.

The most significant nibble of the AFI is used to code one specific or all application families, as defined in Table 2.

The least significant nibble of the AFI is used to code one specific or all application sub-families. Sub-family codes different from 0 are proprietary.

Table 2 — AFI coding

AFI most significant nibble	AFI least significant nibble	Meaning VICCs respond from	Examples/comments
‘0’	‘0’	All families and subfamilies	No applicative preselection
X	'0'	All sub-families of family X	Wide applicative preselection
X	Y	Only the Yth sub-family of family X	
‘0’	Y	Proprietary sub-family Y only	
‘1'	‘0’, Y	Transport	Mass transit, bus, airline
'2'	‘0’, Y	Financial	IEP, banking, retail
'3'	‘0’, Y	Identification	Access control
'4'	‘0’, Y	Telecommunication	Public telephony, GSM
‘5’	‘0’, Y	Medical	
'6'	‘0’, Y	Multimedia	Internet services
'7'	‘0’, Y	Gaming	
'8'	‘0’, Y	Data storage	Portable files
NOTE X = ‘1’ to ‘F’, Y = ‘1’ to ‘F’.			

Table 2 *(continued)*

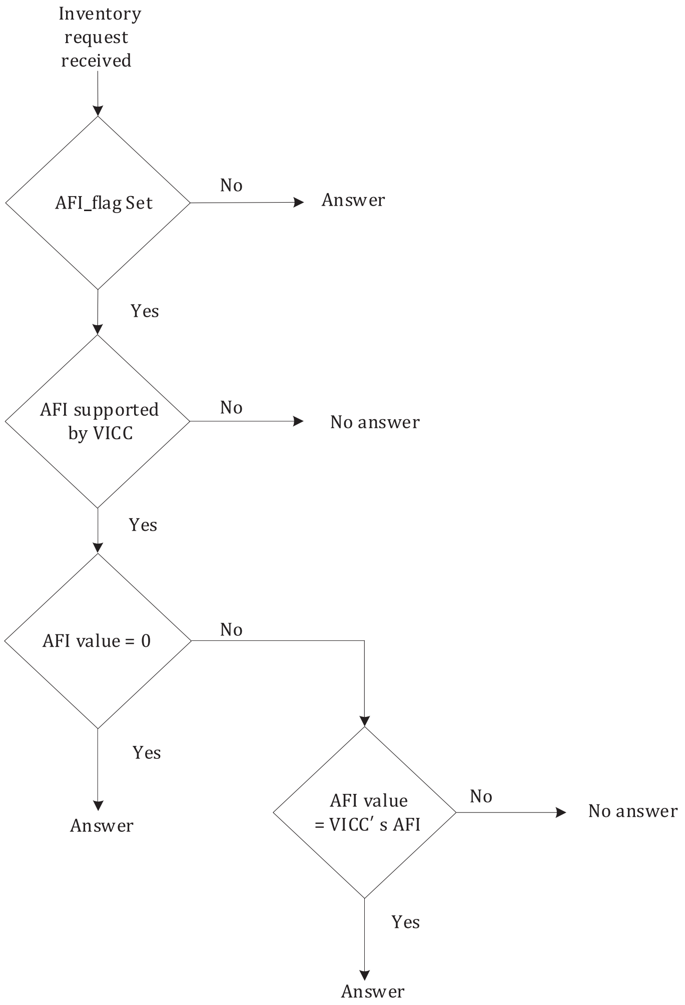
AFI most significant nibble	AFI least significant nibble	Meaning VICCs respond from	Examples/comments
'9'	‘0’, Y	EAN-UCC system for Application Identifiers	
'A'	‘0’, Y	Data Identifiers as defined in ISO/IEC 15418	
'B'	‘0’, Y	UPU (Universal Postal Union)	
'C'	‘0’, Y	IATA (International Air Transport Association)	
'D'	‘0’, Y	RFU	
'E'	‘0’, Y	RFU	
‘F’	‘0’, Y	RFU	
NOTE X = ‘1’ to ‘F’, Y = ‘1’ to ‘F’.			

The support of the AFI by the VICC is optional.

If the AFI is not supported by the VICC and if the AFI_flag is set, the VICC shall not answer whatever the AFI value is in the request.

If the AFI is supported by the VICC, it shall answer according to the matching rules described in Table 2.

Figure 1 shows the VICC decision tree for the AFI.



NOTE "Answer" means that the VICC answers to the Inventory request.

Figure 1 — VICC decision tree for the AFI

4.3 DSFID

The DSFID indicates how the data is structured in the VICC memory.

It may be programmed and locked by the respective commands. It is coded on one byte. It allows for instant knowledge on the logical organisation of the data.

If its programming is not supported by the VICC, the VICC shall respond with the value zero ('00').

4.4 CRC

The CRC shall be calculated in accordance with ISO/IEC 13239.

The initial register content shall be all ones: 'FFFF'.

For examples, refer to Annex C.

The two bytes CRC are appended to each request and each response, within each frame, before the EOF. The CRC is calculated on all the bytes after the SOF up to but not including the CRC field.

Upon reception of a request from the VCD, the VICC shall verify that the CRC value is valid. If it is invalid, it shall discard the frame and shall not answer (modulate).

Upon reception of a response from the VICC, it is recommended that the VCD verifies that the CRC value is valid. If it is invalid, actions to be performed are left to the responsibility of the VCD designer.

The CRC is transmitted least significant byte first (see Table 3).

Each byte is transmitted least significant bit first.

Table 3 — CRC bits and bytes transmission rules

LSByte		MSByte	
LSB	MSB	LSB	MSB
CRC 16 (8 bits)		CRC 16 (8 bits)	
↑ first transmitted bit of the CRC			

NOTE The probability that CRC 16 detects an error depends on the frame length and bit error rate. With a bit error rate of 1E-4 the maximum frame length is less than 512 bytes.

4.5 Security framework

The security framework provides an interface to the crypto suites defined in the ISO/IEC 29167 series. Crypto suites are identified by an 8-bit CSI defined in ISO/IEC 29167-1.

The security framework includes optional security features such as VICC or VCD Authentication, Mutual Authentication, key update or secure messaging.

5 VICC memory organization

The commands specified in this document assume that the physical memory is organized in blocks (or pages) of fixed size.

— Up to 65 536 blocks can be addressed.

— The block size can be of up to 256 bits.

— This leads to a maximum memory capacity of up to 2 MBytes (16 MBits).

The commands described in this document allow the access (read and write) by block(s). There is no implicit or explicit restriction regarding other access method, e.g. by byte or by logical object in future revision(s) of this document or in custom commands.

6 Block security status

The block security status is sent back by the VICC as a parameter in the response to a VCD request as specified in Clause 10 (e.g. Read single block). It is currently coded on one byte but may be coded in 2, 4 and 8 as defined in future revisions of this document (see Table 4).

It is an element of the protocol. There is no implicit or explicit assumption that the 8 bits are actually implemented in the physical memory structure of the VICC.

Table 4 — Block security status

Bit	Flag name	Value	Description
b1	Lock_flag	0	Not locked
		1	Locked
b2 to b5	Proprietary	X	Not defined in this document

Table 4 *(continued)*

Bit	Flag name	Value	Description
b6		0	Unless otherwise specified in future revisions of this document
		1	See warning for legacy commands listed in Annex E.
b7		0	Unless otherwise specified in future revisions of this document
		1	See warning for legacy commands listed in Annex E.
b8		0	Unless otherwise specified in future revisions of this document
		1	See warning for legacy commands listed in Annex E.
b9 to b16		RFU	Only present if specified in future revisions of this document and the block security status length_flag is set to (0,1)b
b9 to b32		RFU	Only present if specified in future revisions of this document and the block security status length_flag is set to (1, 0)b
b9 to b64		RFU	Only present if specified in future revisions of this document and the block security status length_flag is set to (1, 1)b

7 Overall protocol description

7.1 Protocol concept

The transmission protocol (or protocol) defines the mechanism to exchange instructions and data between the VCD and the VICC, in both directions.

It is based on the concept of "VCD talks first".

This means that any VICC shall not start transmitting (i.e. modulating according to ISO/IEC 15693-2) unless it has received and properly decoded an instruction sent by the VCD.

a) The protocol is based on an exchange of:

- a request from the VCD to the VICC;
- a response from the VICC(s) to the VCD.

The conditions under which the VICC sends a response are defined in Clause 10.

b) Each request and each response are contained in a frame. The frame delimiters (SOF, EOF) shall be implemented as specified in ISO/IEC 15693-2. The maximum frame length is 8 192 bytes.

c) Each request consists of the following fields:

- flags;
- command code;
- mandatory and optional parameters fields, depending on the command;

- application data fields;
- CRC.

d) Each response consists of the following fields:

- flags;
- mandatory and optional parameters fields, depending on the command;
- application data fields;
- CRC.

e) The protocol is bit-oriented. The number of bits transmitted in a frame is a multiple of eight (8), i.e. an integer number of bytes.

f) A single-byte field is transmitted LSB first.

g) A multiple-byte field is transmitted LSByte first, each byte is transmitted LSB first.

h) The setting of the flags indicates the presence of the optional fields. When the flag is set (to one), the field is present. When the flag is reset (to zero), the field is absent.

i) RFU flags shall be set to zero (0).

VICC or VCD receiving RFU bits set incorrectly shall disregard the command or response except for commands listed in Annex E. The VICC may respond with an error code.

7.2 Modes

7.2.1 General

The term mode refers to the mechanism to specify in a request the set of VICCs that shall answer to the request.

7.2.2 Addressed mode

When the Address_flag is set to 1 (Addressed mode), the request shall contain the UID of the addressed VICC.

Any VICC receiving a request with the Address_flag set to 1 shall compare the received unique ID (address) to its own ID.

If it matches, it shall execute it (if possible) and return a response to the VCD as specified by the command description.

If it does not match, it shall remain silent.

7.2.3 Non-addressed mode

When the Address_flag is set to 0 (Non-addressed mode), the request shall not contain a unique ID.

Any VICC receiving a request with the Address_flag set to 0 shall execute it (if possible) and shall return a response to the VCD as specified by the command description.

7.2.4 Select mode

When the Select_flag is set to 1 (Select mode), the request shall not contain a VICC unique ID.

The VICC in the Selected state or Selected Secure state receiving a request with the Select_flag set to 1 shall execute it (if possible) and shall return a response to the VCD as specified by the command description.

Only the VICC in the Selected state or Selected Secure state shall answer to a request having the Select flag set to 1.

7.3 Request format

7.3.1 General

The request consists of the following fields (see Figure 2):

- flags;
- command code (see Clause 10);
- parameters and data fields;
- CRC (see 4.4).

SOF	Flags	Command code	Parameters	Data	CRC	EOF

Figure 2 — General request format

7.3.2 Request flags

In a request, the field "flags" specifies the actions to be performed by the VICC and whether corresponding fields are present or not. Table 5 to Table 7 show the Request flags definition.

It consists of eight bits.

Table 5 — Request flags 1 to 4 definition

Bit	Flag name	Value	Description
b1	Sub-carrier_flag	0	A single sub-carrier frequency shall be used by the VICC.
		1	Two sub-carriers shall be used by the VICC.
b2	Data_rate_flag	0	Low data rate shall be used unless specified otherwise in the definition of the command.
		1	High data rate shall be used unless specified otherwise in the definition of the command.
b3	Inventory_flag	0	The meaning for flags 5 to 8 is according to Table 6.
		1	The meaning for flags 5 to 8 is according to Table 7.
b4	Protocol Extension_flag	0	No protocol format extension.
		1	Protocol format is extended. — See warning for legacy commands listed in Annex E. — Reserved for future use for all other commands.
NOTE 1 Sub-carrier_flag refers to the VICC-to-VCD communication as specified in ISO/IEC 15693-2.			
NOTE 2 Data_rate_flag refers to the VICC-to-VCD communication as specified in ISO/IEC 15693-2.			

Table 6 — Request flags 5 to 8 definition when Inventory_flag is not set

Bit	Flag name	Value	Description
b5	Select_flag	0	The request shall be executed by any VICC according to the setting of Address_flag.
		1	The request shall be executed only by the VICC in Selected state. The Address_flag shall be set to 0 and the UID field shall not be included in the request.
b6	Address_flag	0	The request is not addressed. The UID field is not included. It shall be executed by any VICC.
		1	The request is addressed. The UID field is included. It shall be executed only by the VICC whose UID matches the UID specified in the request.
b7	Option_flag	0	The meaning is defined by the command description. It shall be set to 0 if not otherwise defined by the command.
		1	The meaning is defined by the command description.
b8		0	Unless otherwise specified in the command definition.
		1	See warning for legacy commands listed in Annex E.

Table 7 — Request flags 5 to 8 definition when Inventory_flag is set

Bit	Flag name	Value	Description
b5	AFI_flag	0	AFI field is not present.
		1	AFI field is present.
b6	Nb_slots_flag	0	16 slots.
		1	1 slot.
b7	Option_flag	0	The meaning is defined by the command description. It shall be set to 0 if not otherwise defined by the command.
		1	The meaning is defined by the command description.
b8		0	Unless otherwise specified in the command definition.
		1	See warning for legacy commands listed in Annex E.

7.4 Response format

7.4.1 General

The response consists of the following fields (see Figure 3):

- flags;
- one or more parameter fields;
- data;
- CRC (see 4.4).

SOF	Flags	Parameters	Data	CRC	EOF

Figure 3 — General response format

7.4.2 Response flags

The eight bit response flags field indicates how actions have been performed by the VICC and whether corresponding fields are present or not. Table 8 shows the Response flags definition.

Where the value "1b" for bits b2 to b8 may be used see warning for the legacy commands listed in Annex E.

Table 8 — Response flags 1 to 8 definition

<table>
<tr><th>Bit</th><th>Flag name</th><th>Value</th><th colspan="4">Description</th></tr>
<tr><td rowspan="2">b1</td><td rowspan="2">Error_flag</td><td>0</td><td colspan="4">No error.</td></tr>
<tr><td>1</td><td colspan="4">Error detected. Error code is in the "Error" field.</td></tr>
<tr><td rowspan="2">b2</td><td rowspan="2">ResponseBuffer Validity_flag</td><td>0</td><td colspan="4">In any response if the ResponseBuffer does not contain a valid result of a (cryptographic) calculation or if the ResponseBuffer is not supported.</td></tr>
<tr><td>1</td><td colspan="4">In any response if the ResponseBuffer contains a valid result of a (cryptographic) calculation.</td></tr>
<tr><td rowspan="2">b3</td><td rowspan="2">Final response_flag</td><td>0</td><td colspan="4">In the Final response of an In-process reply if this reply does not contain the result of a (cryptographic) calculation.</td></tr>
<tr><td>1</td><td colspan="4">In the Final response of an In-process reply if this reply contains the result of a (cryptographic) calculation.</td></tr>
<tr><td rowspan="2">b4</td><td rowspan="2">Extension_flag</td><td>0</td><td colspan="4">No protocol format extension.</td></tr>
<tr><td>1</td><td colspan="4">Protocol format is extended. Reserved for future use.</td></tr>
<tr><td rowspan="5">b5
b6</td><td rowspan="5">Block security status length_flag</td><td rowspan="5"></td><th>b5</th><th>b6</th><th colspan="2">Block security status Length</th></tr>
<tr><td>0</td><td>0</td><td colspan="2">1 byte</td></tr>
<tr><td>0</td><td>1</td><td>2 bytes</td><td rowspan="3">These values shall not be used unless the content of the additional bytes of the block security status is defined in future revisions of this document.</td></tr>
<tr><td>1</td><td>0</td><td>4 bytes</td></tr>
<tr><td>1</td><td>1</td><td>8 bytes</td></tr>
<tr><td rowspan="2">b7</td><td rowspan="2">Waiting time extension request_flag</td><td>0</td><td colspan="4">No Waiting time extension request.</td></tr>
<tr><td>1</td><td colspan="4">Waiting time extension request.</td></tr>
<tr><td>b8</td><td>RFU</td><td>0</td><td colspan="4"></td></tr>
</table>

The ResponseBuffer Validity_flag shall be set or reset as specified in the command description.

For each of the following commands, block security status b5 and b6 in Table 8 shall be set to zero (1 byte).

— Read single block with option flag = 1

— Read multiple blocks with option flag = 1

— Extended Read single block with option flag = 1

— Extended Read multiple blocks with option flag = 1

— Get multiple block security status

— Extended get multiple block security status

7.4.3 Response error code

When the Error_flag is set by the VICC, the error code field shall be included and provides information about the error that occurred. Error codes are defined in Table 9.

If the VICC does not support specific error code(s) listed in Table 9, it shall answer with the error code '0F' ("Error with no information given").

Table 9 — Response error code definition

Error code	Meaning
'01'	The command is not supported, i.e. the request code is not recognized.
'02'	The command is not recognized, for example, a format error occurred.
'03'	The command option is not supported.
'04'	The command cannot be processed in time.
'0F'	Error with no information given or a specific error code is not supported.
'10'	The specified block is not available (doesn't exist).
'11'	The specified block is already locked and thus cannot be locked again.
'12'	The specified block is locked and its content cannot be changed.
'13'	The specified block was not successfully programmed.
'14'	The specified block was not successfully locked.
'15'	The specified block is protected.
'40'	Generic cryptographic error.
'A0' – 'DF'	Custom command error codes.
all others	RFU.

7.4.4 In-process reply response formats

7.4.4.1 Barker field

The Barker field contains the Done flag and a Barker code.

7.4.4.2 Barker code

The Barker code is included to facilitate the synchronisation between the VCD and the VICC. It is a fixed 7-bit value as defined in Table 10.

Table 10 — Barker field

b8	b7	b6	b5	b4	b3	b2	b1
X	0	1	0	0	1	1	1
Done flag	Barker code						

7.4.4.3 Done flag

The Done flag indicates whether the VICC is still processing a command. Done flag = 0 means the VICC is still processing a command; Done flag = 1 means that the VICC has finished the command processing.

7.4.4.4 Barker response

If no error occurs and the Done flag is set to 0, the Barker response contains the fields defined in Table 11.

Table 11 — Barker response

SOF	Flags	Barker field	CRC16	EOF
	8 bits	8 bits	16 bits	

If an error occurs, the response contains the error code and is the Final response (see Table 13).

7.4.4.5 Final response

If no error occurs and the Done flag is set to 1, the Final response contains the fields defined in Table 12.

Table 12 — Final response

SOF	Flags	Barker field	Data	CRC16	EOF
	8 bits	8 bits	multiple of 8 bits	16 bits	

The Data field shall be padded with least significant 0 bits as required to a minimum multiple of 8 bits or not be present.

If an error occurs, the Final response contains the fields defined in Table 13.

Table 13 — Final response if error flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

7.4.4.6 Initial response

If no error occurs and the Done flag is set to 0, the Initial response contains the fields defined in Table 14.

Table 14 — Initial response

SOF	Flags	Barker field	Data	CRC16	EOF
	8 bits	8 bits	16 bits	16 bits	

The Done flag is set to 0. The Data field contains the timing information. Timing information is coded as a binary integer multiple of 4 096/f_C (302 µs), a value of 0 indicates that the feature is not supported.

If an error occurs, the response contains the error code and is the Final response (see Table 13).

7.4.5 Waiting time extension request formats

If no error occurs, the VICC Waiting time extension request contains the fields defined in Table 15.

Table 15 — Waiting time extension request

SOF	Flags	Data	CRC16	EOF
	8 bits	16 bits	16 bits	

The Waiting time extension request_flag shall be set to 1.

The Data field contains the Waiting time extension delay information. Waiting time extension delay is coded as a binary integer multiple of 4 096/f_C (302 µs).

If an error occurs, the response shall use the format described in the command description.

7.5 VICC states

7.5.1 General

A VICC can be in one of the 5 following states:

— Power-off;

- Ready;
- Quiet;
- Selected;
- Selected Secure.

The transition between these states is specified in Figure 4.

The support of the Power-off, Ready and Quiet states is mandatory.

The support of the Selected and Selected Secure states is optional as represented with a dotted line in Figure 4.

The VICC state transition diagram shows only valid transitions. In all other cases the current VICC state remains unchanged. When the VICC cannot process a VCD request (e.g. CRC error, etc.), it shall stay in its current state.

The intention of the state transition method is that only one VICC should be in the Selected or Selected Secure state at a time.

NOTE The case where one VICC is in a Selected state and another is in a Selected Secure state can happen if a first VICC is selected, and a second VICC moves in the Selected Secure state when the VCD does a VCD or Mutual Authentication. Only the VCD can prevent this situation to occur by deselecting the first VICC before doing the VCD or Mutual Authentication with the second one.

7.5.2 Power-off state

The VICC is in the Power-off state when it cannot be activated by the VCD.

7.5.3 Ready state

The VICC is in the Ready state when it is activated by the VCD. It shall process any request unless otherwise specified in the command definition where the Select_flag is not set.

In a Ready state, a VCD can perform a VICC Authentication by a successful Challenge, ReadBuffer or Authenticate command sequence. After a VICC Authentication, the VICC remains in the Ready state.

For a transition from the Ready state to the Selected Secure state, perform a VCD Authentication or Mutual Authentication in Addressed mode containing the correct UID as specified by the crypto suites.

See Annex D for additional information on crypto command sequence.

7.5.4 Quiet state

When in the Quiet state, the VICC shall process any request unless otherwise specified in the command definition where the Inventory_flag is not set and where the Address_flag is set.

For a transition from the Quiet state to the Selected Secure state, perform a VCD Authentication or Mutual Authentication in Addressed mode containing the correct UID as specified in the crypto suites.

7.5.5 Selected state

The VICC in the Selected state shall process any request unless otherwise specified in the command definition where the Select_flag is set.

For a transition from the Selected state to the Ready state, a VCD shall perform:

- Reset to Ready command with the Select_flag set;
- Select command with different VICC UID.

For a transition from the Selected state to the Quiet state, a VCD shall perform a Stay quiet command with the correct UID number.

For a transition from the Selected state to the Selected Secure state, perform a VCD Authentication or Mutual Authentication as specified by the crypto suites with the Select_flag set.

7.5.6 Selected Secure state

A VICC may execute any optional commands and the mandatory Stay quiet command. All commands shall be executed with the Select_flag set except the Stay quiet or Select commands which shall be executed in Addressed mode.

For a transition from the Selected Secure state to the Ready state, a VCD shall perform:

- Reset to Ready command with the Select_flag set;
- Challenge command;
- any Authenticate command starting a new authentication process;
- Select command with different VICC UID.

In addition the VICC shall return to the Ready state in case of cryptographic errors as specified in the crypto suites.

For a transition from the Selected Secure state to the Quiet state, a VCD shall perform a Stay quiet command with the correct UID number.

For a transition from the Selected Secure state to the Selected state, the VCD shall perform a Select command in Addressed mode containing the correct UID.

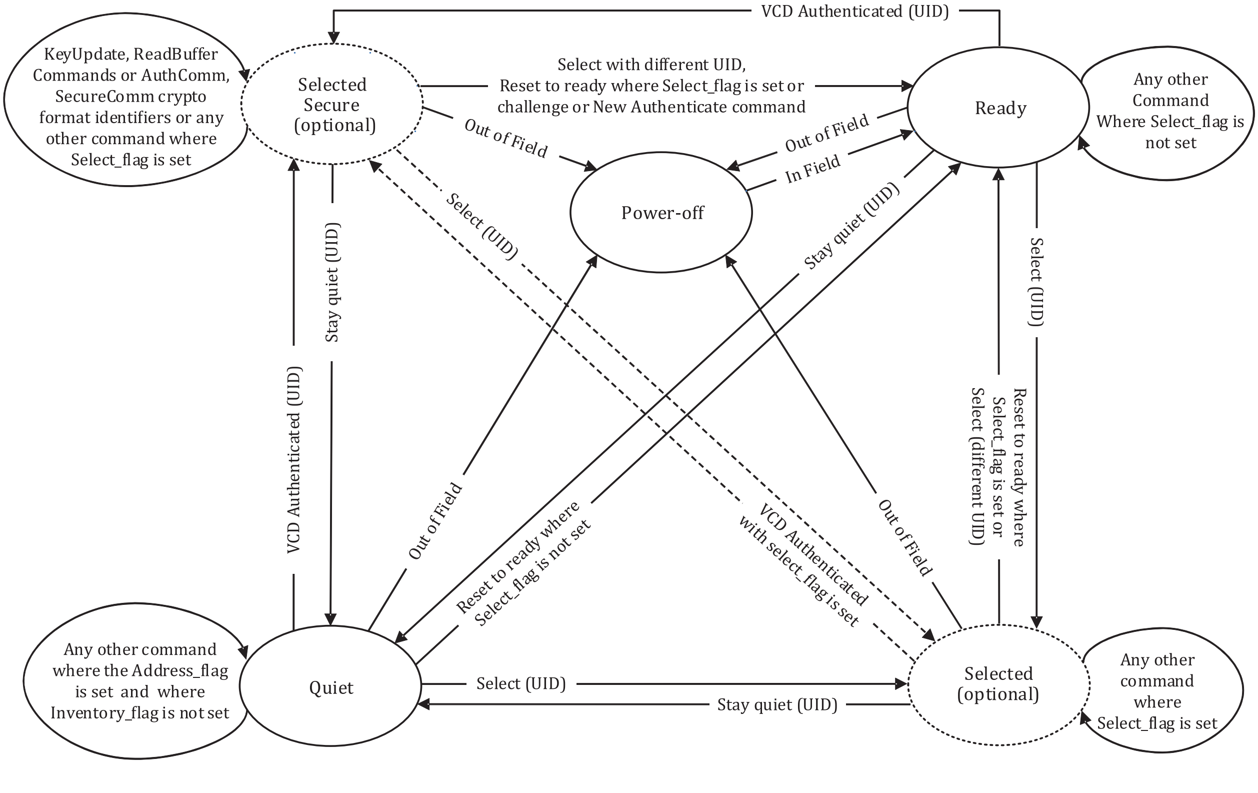

VCD Authenticated (UID)
KeyUpdate, ReadBuffer Commands or AuthComm, SecureComm crypto format identifiers or any other command where Select_flag is set
Selected Secure (optional)
Select with different UID, Reset to ready where Select_flag is set or challenge or New Authenticate command
Ready
Any other Command Where Select_flag is not set
Out of Field
Out of Field
In Field
Power-off
Select (UID)
Stay quiet (UID)
Stay quiet (UID)
VCD Authenticated (UID)
Select (UID)
Reset to ready where Select_flag is set or Select (different UID)
Out of Field
Reset to ready where Select_flag is not set
VCD Authenticated with select_flag is set
Out of Field
Any other command where the Address_flag is set and where Inventory_flag is not set
Quiet
Select (UID)
Stay quiet (UID)
Selected (optional)
Any other command where Select_flag is set

Figure 4 — VICC state transition diagram

8 Anticollision

8.1 General

The purpose of the anticollision sequence is to make an inventory of the VICCs present in the VCD field by their UID.

The VCD is the master of the communication with one or multiple VICCs. It initiates card communication by issuing the Inventory request. See Annex B for additional information how the anticollision could be implemented on the VCD.

The VICC shall send its response in the slot determined or shall not respond, according to the algorithm described in 8.3.

8.2 Request parameters

When issuing the Inventory command, the VCD shall set the Nb_slots_flag to the desired setting and add after the command field the mask length and the mask value defined in Table 16.

The mask length indicates the number of significant bits of the mask value. It can have any value between 0 and 60 when 16 slots are used and any value between 0 and 64 when 1 slot is used. LSB shall be transmitted first.

The mask value is contained in an integer number of bytes. LSB shall be transmitted first.

If the mask length is not a multiple of 8 (bits), the mask value MSB shall be padded with the required number of null (set to 0) bits so that the mask value is contained in an integer number of bytes.

The next field starts on the next byte boundary.

Table 16 — Inventory request format

SOF	Flags	Command	Mask length	Mask value	CRC16	EOF
	8 bits	8 bits	8 bits	0 to 8 bytes	16 bits	

Table 17 — Example of the padding of the mask

MSB	LSB
0000	0100 1100 1111
Pad	Mask value

In the example of Table 17, the mask length is 12 bits. The mask value MSB is padded with four bits set to 0.

The AFI field shall be present if the AFI_flag is set.

The pulse shall be generated according to the definition of the EOF in ISO/IEC 15693-2.

The first slot starts immediately after the reception of the request EOF.

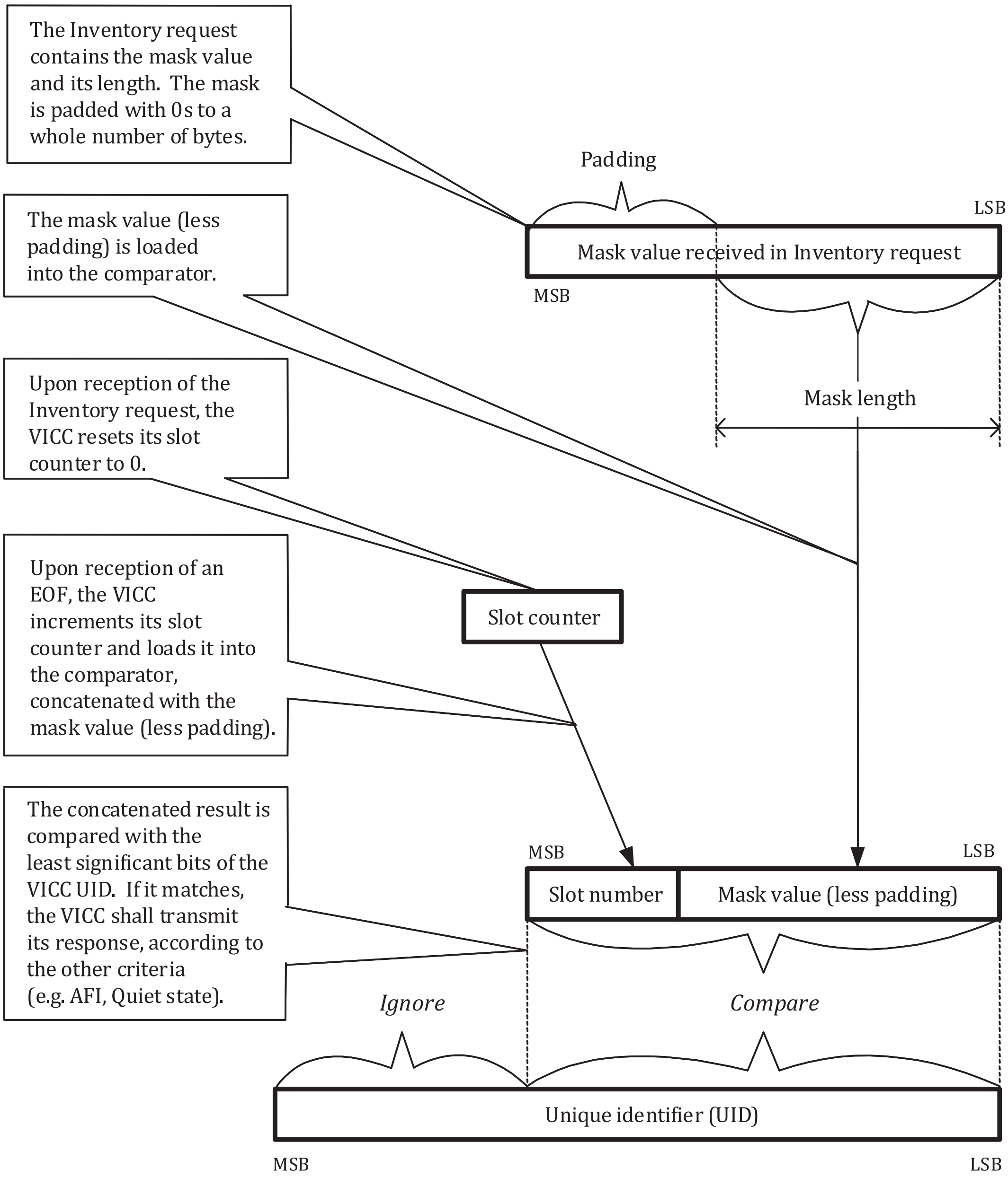
To switch to the next slot, the VCD sends an EOF. The rules, restrictions and timing are specified in Clause 9.

8.3 Request processing by the VICC

Upon reception of a valid request, the VICC shall process it by executing the operation sequence specified in the following text in italics. The step sequence is also graphically represented in Figure 5.

NbS is the total number of slots (1 or 16)

```
SN is the current slot number (0 to 15)
SN_length is set to 0 when 1 slot is used and set to 4 when 16 slots are used
LSB (value, n) function returns the n less significant bits of value
"&" is the concatenation operator
Slot_Frame is either a SOF or an EOF
    SN= 0
    if Nb_slots_flag then
        NbS =1 SN_length=0
        else NbS = 16 SN_length=4
    endif
label1: if LSB(UID, SN_length + Mask_length) = LSB(SN, SN_length)&LSB(Mask, Mask_length) then
        transmit response to inventory request
    endif
    wait (Slot_Frame)
    if Slot_Frame= SOF then
        Stop anticollision and decode/process request
        exit
    endif
    if SN<NbS-1 then
        SN = SN +1
        goto label1
        exit
    endif
    exit
```


The Inventory request contains the mask value and its length. The mask is padded with 0s to a whole number of bytes.
The mask value (less padding) is loaded into the comparator.
Upon reception of the Inventory request, the VICC resets its slot counter to 0.
Upon reception of an EOF, the VICC increments its slot counter and loads it into the comparator, concatenated with the mask value (less padding).
The concatenated result is compared with the least significant bits of the VICC UID. If it matches, the VICC shall transmit its response, according to the other criteria (e.g. AFI, Quiet state).
Padding
LSB
Mask value received in Inventory request
MSB
Mask length
Slot counter
MSB
LSB
Slot number
Mask value (less padding)
Ignore
Compare
Unique identifier (UID)
MSB
LSB

NOTE When the slot number is 1 (Nb_slots_flag is set to 1), the comparison is made only on the mask (without padding).

Figure 5 — Principle of comparison between the mask value, slot number and UID

8.4 Explanation of an anticollision sequence

Figure 6 summarises the main cases that can occur during a typical anticollision sequence where the number of slots is 16.

The different steps are:

a) the VCD sends the Inventory request, in a frame, terminated by an EOF; the number of slots is 16;

b) VICC 1 transmits its response in slot 0; it is the only one to do so, therefore no collision occurs and its UID is received and registered by the VCD;

c) the VCD sends an EOF, meaning to switch to the next slot;

d) in slot 1, two VICCs 2 and 3 transmits their response, this generates a collision; the VCD detects it and remembers that a collision was detected in slot 1;

e) the VCD sends an EOF, meaning to switch to the next slot;

f) in slot 2, no VICC transmits a response; therefore the VCD does not detect a VICC SOF and decides to switch to the next slot by sending an EOF;

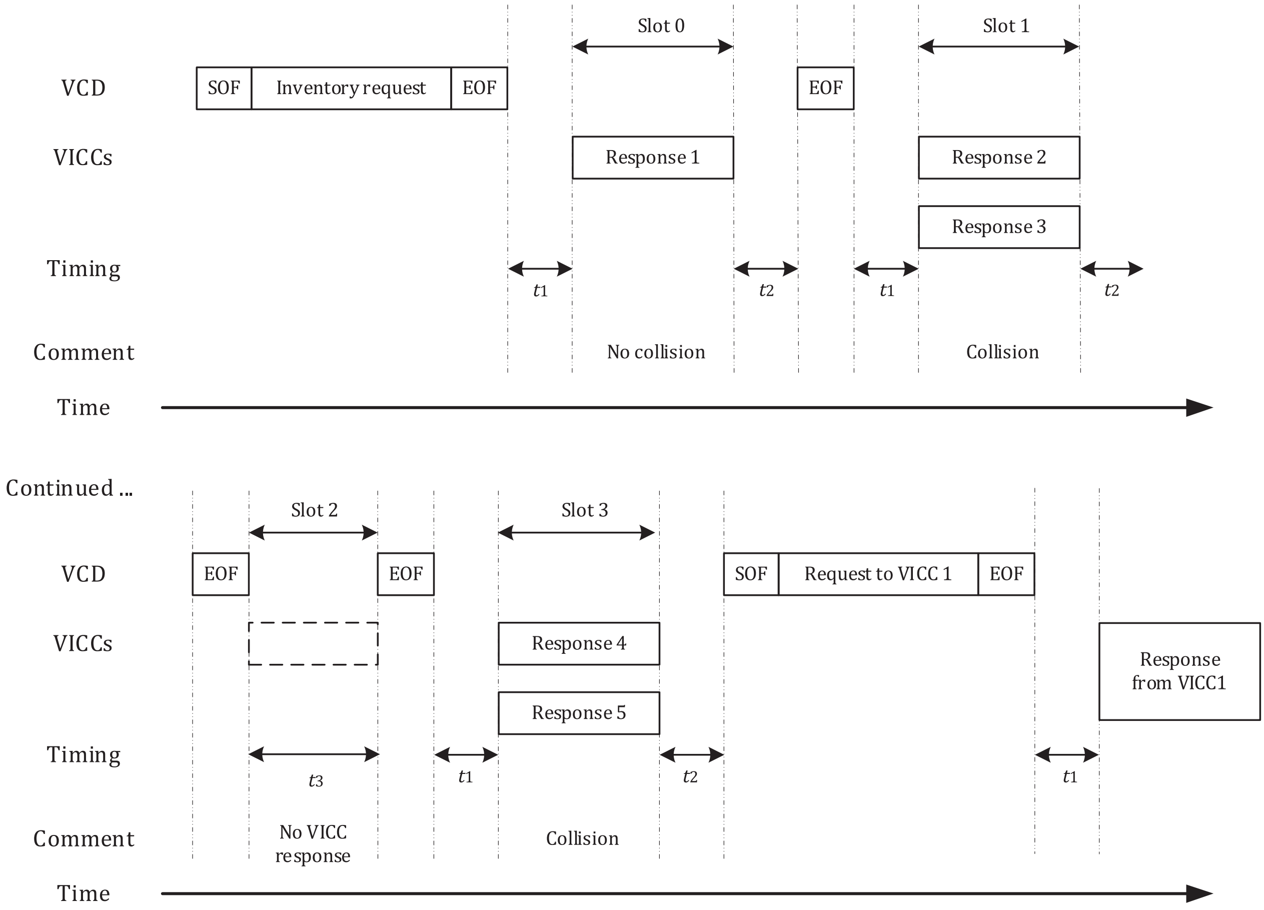
g) in slot 3, there is another collision caused by responses from VICC 4 and 5;

h) the VCD then decides to send an addressed request (for instance a Read Block) to VICC 1, which UID was already correctly received;

i) all VICCs detect an SOF and exit the anticollision sequence; they process this request and since the request is addressed to VICC 1, only VICC1 transmits its response;

j) all VICCs are ready to receive another request; if it is an Inventory command, the slot numbering sequence restarts from 0.

NOTE The decision to interrupt the anticollision sequence is up to the VCD. It could have continued to send EOFs till slot 15 and then send the request to VICC 1.



NOTE t_1, t_2 and t_3 are specified in Clause 9.

Figure 6 — Description of a possible anticollision sequence

9 Timing specifications

9.1 General

The VCD and the VICC shall comply with the following timing specifications.

9.2 VICC waiting time before transmitting its response after reception of an EOF from the VCD

When the VICC has detected an EOF of a valid VCD request or when this EOF is in the normal sequence of a valid VCD request, it shall wait for a time t_1 before starting to transmit its response to a VCD request or before switching to the next slot when in an inventory process (see 8.3 and 8.4).

t_1 starts from the detection of the rising edge of the EOF received from the VCD (see ISO/IEC 15693-2).

NOTE The synchronisation on the rising edge of the VCD-to-VICC EOF is needed for ensuring the required synchronization of the VICC responses.

The minimum value of t_1 is $t_{1_{\text{min}}}$ = 4 320/f_c (318,6 µs)

The nominal value of t_1 is $t_{1_{\text{nom}}}$ = 4 352/f_c (320,9 µs)

The maximum value of t_1 is $t_{1_{\text{max}}}$ = 4 384/f_c (323,3 µs)

$t_{1_{\text{max}}}$ does not apply for Write alike requests. Timing conditions for Write alike requests are defined in the command descriptions.

$t_{1_{\text{max}}}$ does not apply to commands which may use the Waiting time extension reply as indicated in the command description. Timing conditions for those commands are defined in 9.9.

If the VICC detects a carrier modulation during this time t_1, it shall reset its t_1 timer and wait for a further time t_1 before starting to transmit its response to a VCD request or to switch to the next slot when in an inventory process.

9.3 VICC modulation ignore time after reception of an EOF from the VCD

When the VICC has detected an EOF of a valid VCD request or when this EOF is in the normal sequence of a valid VCD request, it shall ignore any received 10 % modulation during a time t_{mit}.

t_{mit} starts from the detection of the rising edge EOF received from the VCD (see ISO/IEC 15693-2).

The minimum value of t_{mit} is $t_{\text{mit}_{\text{min}}}$ = 4 384/f_c (323,3 µs) + t_{nrt}, where t_{nrt} is the nominal response time of a VICC.

t_{nrt} is dependent on the VICC-to-VCD data rate and subcarrier modulation mode (see ISO/IEC 15693-2).

NOTE The synchronisation on the rising edge of the VCD-to-VICC EOF is needed for ensuring the required synchronization of the VICC responses.

9.4 VCD waiting time before sending a subsequent request

a) When the VCD has received a VICC response to a previous request other than Inventory and Quiet, it shall wait a time t_2 before sending a subsequent request. t_2 starts from the time the EOF has been received from the VICC.

b) When the VCD has sent a Quiet request (which causes no VICC response), it shall wait a time t_2 before sending a subsequent request. t_2 starts from the end of the Quiet request EOF (rising edge of the EOF plus 9,44 µs, see ISO/IEC 15693-2).

The minimum value of t_2 is $t_{2_{\text{min}}}$ = 4 192/f_c (309,2 µs).

NOTE 1 This ensures that the VICCs are ready to receive this subsequent request (see ISO/IEC 15693-2).

NOTE 2 The VCD wait at least 1 ms after it activated the powering field before sending the first request, to ensure that the VICCs are ready to receive it (see ISO/IEC 15693-2).

c) When the VCD has sent the Inventory request, it is in an inventory process. See 9.5.

9.5 VCD waiting time before switching to the next slot during an inventory process

9.5.1 General

An inventory process is started when the VCD sends the Inventory request. (See 8.3, 8.4, 10.3.1).

To switch to the next slot, the VCD may send either a 10 % or a 100 % modulated EOF independent of the modulation index it used for transmitting its request to the VICC, after waiting a time specified in 9.5.2 and 9.5.3).

9.5.2 When the VCD has started to receive one or more VICC responses

During an inventory process, when the VCD has started to receive one or more VICC responses (i.e. it has detected a VICC SOF and/or a collision), it shall:

— wait for the complete reception of the VICC responses (i.e. when a VICC EOF has been received or when the VICC nominal response time t_{nrt} has elapsed);

— wait an additional time t_2;

— then send a 10 % or 100 % modulated EOF to switch to the next slot.

t_2 starts from the time the EOF has been received from the VICC (see ISO/IEC 15693-2).

The minimum value of t_2 is $t_{2_{\text{min}}}$ = 4 192/f_c (309,2 µs).

t_{nrt} is dependent on the VICC-to-VCD data rate and subcarrier modulation mode (see ISO/IEC 15693-2).

9.5.3 When the VCD has received no VICC response

During an inventory process, when the VCD has received no VICC response, it shall wait a time t_3 before sending a subsequent EOF to switch to the next slot.

t_3 starts from the time the VCD has generated the rising edge of the last sent EOF:

a) if the VCD sends a 100 % modulated EOF,

the minimum value of t_3 is $t_{3_{\text{min}}}$ = 4 384/f_c (323,3 µs) + t_{sof};

b) if the VCD sends a 10 % modulated EOF,

the minimum value of t_3 is $t_{3_{\text{min}}}$ = 4 384/f_c (323,3 µs) + t_{nrt} + $t_{2_{\text{min}}}$;

where

t_{sof} is the time duration for a VICC to transmit an SOF to the VCD;

t_{nrt} is the nominal response time of a VICC.

t_{nrt} and t_{sof} are dependent on the VICC-to-VCD data rate and subcarrier modulation mode (see ISO/IEC 15693-2).

9.6 Clarification of use of Option_flag in Write alike commands

VICC may or may not support Write alike commands with all Option_flag settings.

If the Option_flag is not set, the VICC shall return its response when it has completed the operation starting after: ${t_1}_{\mathrm{nom}}$ [4 352/f_c (320,9 µs), see 9.2] + a multiple of 4 096/f_c (302 µs) with a total tolerance of ±32/f_c and latest after 20 ms upon detection of the rising edge of the EOF of the VCD request.

If the Option_flag is set and supported by the VICC, the VICC shall wait for the reception of an EOF from the VCD and upon such reception shall return its response.

The VCD shall transmit an EOF at least 10 ms and no later than 20 ms after transmitting the command.

If the VCD transmits an EOF within 10 ms, the VICC may not be able to execute the command or it may reset.

Upon reception of an EOF the VICC shall wait for a duration of t_1 before transmitting its response (see 9.2).

If the Option_flag is set and not supported by the VICC, the VICC shall return an error in the timing defined in 9.2 in the Select or Addressed mode.

9.7 Security timeout as used in the CS

A VICC may use a security timeout for each of the following commands:

— Challenge;

— Authenticate;

— KeyUpdate.

Or for the following crypto format indicators:

— SecureComm;

— AuthComm.

If implemented, a security timeout shall be triggered by specific errors as specified in the CS and shall cause a VICC to reject those commands or crypto format indicators for which a VICC implements a security timeout until the end of the security timeout period.

A security timeout shall be a minimum of 20 ms and a maximum of 200 ms in any state including the Power-off state.

9.8 VICC replies as used in CS or extended functionalities

9.8.1 General

In all responses for Authenticate, KeyUpdate commands and SecureComm, AuthComm crypto format indicators, the immediate VICC reply or in-process VICC reply shall be used by the VICC.

If a CS specifies a Delayed reply an in-process VICC reply shall be used.

The specified timing mechanisms may also be used for custom commands or future extensions.

9.8.2 Immediate VICC reply

See 9.2 for a specification of the immediate VICC reply as requested by the CS.

9.8.3 In-process reply

9.8.3.1 General

The In-process reply allows a VICC to spend longer than t_1 and to notify the VCD that it is still processing that command.

The In-process reply is composed of two modes called Synchronous (see Figure 7) and Asynchronous (see Figure 8) modes.

The Asynchronous and Synchronous modes are selected using the Option_flag (OF) within the request flags:

- OF = 0 : Synchronous mode;
- OF = 1 : Asynchronous mode.

9.8.3.2 Synchronous mode

- The VCD sends a command which may require an In-process reply (as specified in CS).
- The VICC maintains a continuous communication until its response is ready by sending the Barker response in accordance with the response grid. The response grid is defined as:
  - t_{res1} = $t_{1_{nom}}$ [4 352/f_c (320,9 µs), see 9.2] + a multiple of 4 096/f_c (302 µs) with a total tolerance of ±32/f_c and no later than 20 ms from the detection of the rising edge of the EOF of the VCD request;
  - t_{res2} = $t_{1_{nom}}$ [4 352/f_c (320,9 µs), see 9.2] + a multiple of 4 096/f_c (302 µs) with a total tolerance of ±32/f_c from the rising edge of the EOF of the VCD request;
  - t_{res3} = no later than 20ms from the logical end of EOF of the previous Baker response.
- The VICC has not completed the operation if the Done flag is set to 0.
- The VICC sends the Final response when the execution of the command is completed or whenever an error occurs in accordance with the response grid. The error response does not include the Barker field.
- If the Final response is available within the 20 ms, the Barker response may be skipped and only the Final response is sent.
- The VICC has completed the operation if the Done flag is set to 1.
- The VICC decides whether the data field is included in the Final response or stored in the ResponseBuffer.
  - If response flag b3 is set to 1, the Final response includes the data field with valid cryptographic results. The VICC may also store the results inside a ResponseBuffer and shall set b2 to 1.
  - If response flag b3 is set to 0, the Final response does not include the data field. The VICC shall store the cryptographic results inside the ResponseBuffer and shall set b2 to 1.

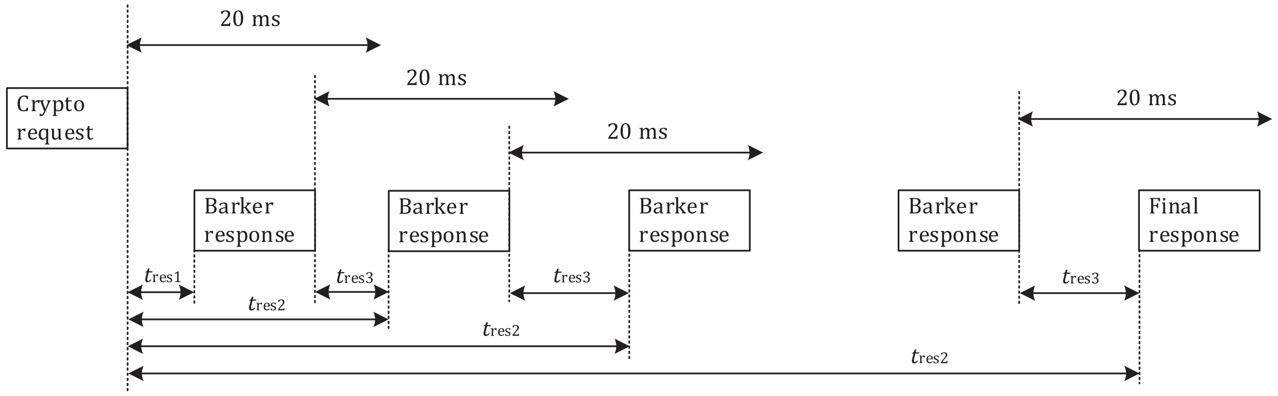


Figure 7 — VICC in-reply process — Synchronous mode

9.8.3.3 Asynchronous mode

- The VCD sends a command which may require an In-Process reply (as specified in CS).
- The VICC sends an Initial response in accordance with the response grid. The response grid is defined as t_{res1} = $t_{1_{nom}}$ [4 352/f_c (320,9 µs), see 9.2] + a multiple of 4 096/f_c (302 µs) with a total tolerance of ±32/f_c and no later than 20 ms from the detection of the rising edge of the EOF of the VCD request for the Initial response.
- Optionally the VICC may provide in the Initial response the estimated time required for the computation of the Final response.
- The VICC sends the Final response when the execution of the command is completed or whenever an error occurs in accordance with the response grid. The response grid is defined as t_{res2} = $t_{1_{nom}}$ [4 352/f_c (320,9 µs), see 9.2] + a multiple of 4 096/f_c (302 µs) with a total tolerance of ±32/f_c from the detection of the rising edge of the EOF of the VCD request. The error response does not include the Barker field.
- If the Final response is available within the 20 ms, the Initial response may be skipped and only the Final response is sent.
- The VICC decides whether the data field is included in the Final response or stored in the ResponseBuffer.
  - If response flag b3 is set to 1, the Final response includes the data field with valid cryptographic results. The VICC may also store the results inside a ResponseBuffer and shall set b2 to 1.
  - If response flag b3 is set to 0, the Final response does not include the data field. The VICC shall store the cryptographic results inside the ResponseBuffer and shall set b2 to 1.

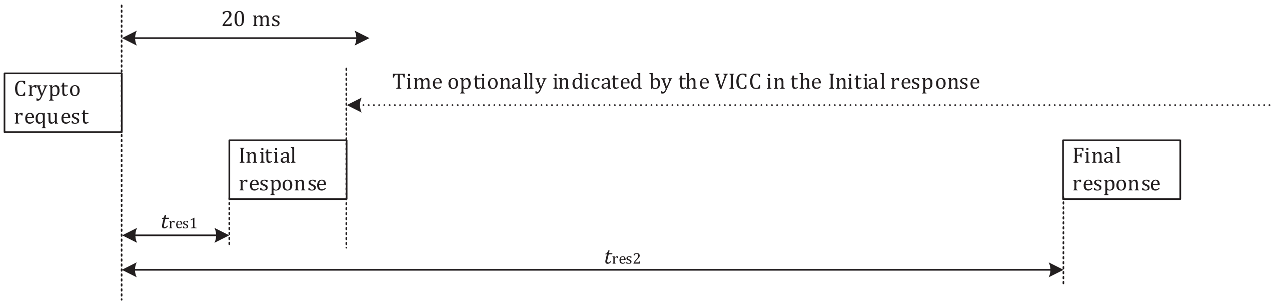


Figure 8 — VICC in-reply process — Asynchronous mode

A VICC shall ignore any VCD commands while processing a prior command.

If a VCD transmits a command while the VICC is processing a prior command the VICC may undergo a power-on reset.

NOTE The optionally indicated time by the VICC in the Initial response is a rough estimation and can be inaccurate, it starts from the logical end of the Initial response of the VICC.

9.9 Waiting time extension reply

Where a command definition supports a Waiting time extension reply (see Figure 9) a VICC may respond to that command with a request to the VCD indicating the extension time required. The VCD shall then resend the previous command.

The Waiting time extension sequence shall be:

- The VCD sends a command for which the Waiting time extension reply is allowed;
- In case no waiting time extension delay is required, the VICC sends the response to the command in accordance with the response grid. The response grid is defined as t_{res1} = $t_{1_{\mathrm{nom}}}$ [4 352/f_{c} (320,9 µs), see 9.2] + a multiple of 4 096/f_{c} (302 µs) with a total tolerance of ±32/f_{c} and no later than 20 ms from the detection of the rising edge of the EOF of the VCD request;
- In case a waiting time extension delay is required for providing the response to the command, the VICC sends a Waiting time extension request in accordance with the response grid. The Waiting time extension request indicates the waiting time extension delay required by the VICC;
- VCD shall wait for the waiting time extension time t_{wtx} indicated by the VICC in the Waiting time extension request before resending the previous command;
- VICC shall respond with the Final response to the command, which terminates the Waiting time extension sequence in accordance to the response grid or send an error after t_{res1}.

If specified in the command definition the Waiting time extension sequence may be repeated.

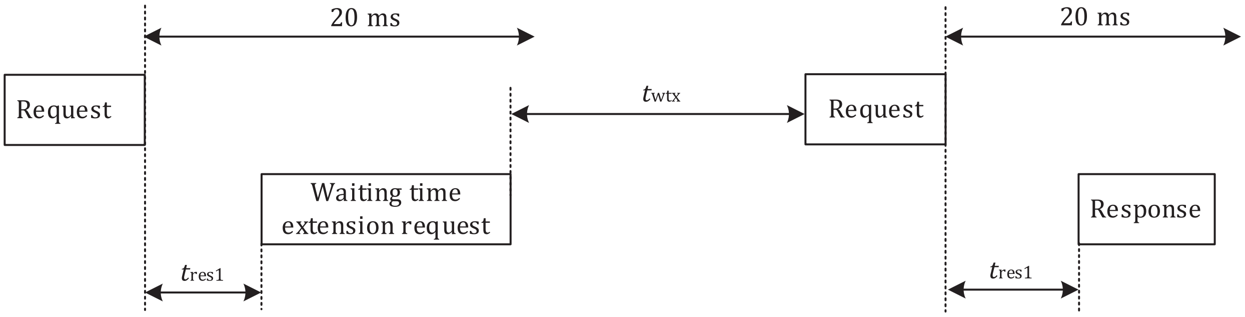


Figure 9 — Waiting time extension reply

The VCD shall not send any command within t_{wtx}.

If an error occurs, the Waiting time extension sequence shall be terminated. The VICC may answer with an error in accordance with the response grid.

If the VCD request following a Waiting time extension request is different from the initial VCD request, the Waiting time extension sequence shall be terminated. The VICC may answer with an error in accordance with the response grid or may ignore the last request or process the last request.

If the VCD request following a Waiting time extension request is sent before t_{wtx}, the Waiting time extension sequence may be terminated. The VICC may answer with an error in accordance with the response grid or may ignore the last request or process the last request.

The usage of the Waiting time extension reply mechanism is described in the command description and may also be used for custom commands or future extensions.

10 Commands

10.1 Command types

10.1.1 General

Four sets of commands are defined: mandatory, optional, custom, proprietary.

All VICCs with the same IC manufacturer code and same IC version number shall behave the same functionalities.

10.1.2 Mandatory

The command codes range from '01' to '1F'.

All VICCs shall support them.

10.1.3 Optional

The command codes range from '20' to '9F'.

VICCs may support them, at their option. If supported, request and response formats shall comply with the definition given in this document.

If the VICC does not support an optional command and if the Address_flag or the Select_flag is set, it may return an error code or remain silent. If neither the Address_flag nor the Select_flag is set, the VICC shall remain silent.

If a command has different options they may be supported by the VICC otherwise an error code shall be returned.

10.1.4 Custom

The command codes range from 'A0' to 'DF'.

VICCs support them, at their option, to implement manufacturer-specific functions. The function of flags (including reserved bits) shall not be modified except the Option_flag. The only fields that can be customized are the parameters and the data fields.

Each custom command contains as its first parameter the IC manufacturer code. This allows IC manufacturers to implement custom commands without risking duplication of command codes and thus misinterpretation.

If the VICC does not support a custom command and if the Address_flag or the Select_flag is set, it may return an error code or remain silent. If neither the Address_flag nor the Select_flag is set, the VICC shall remain silent.

If a command has different options they may be supported by the VICC otherwise an error code shall be returned.

10.1.5 Proprietary

The command codes range from 'E0' to 'FF'.

These commands are used by IC and VICC manufacturers for various purposes such as tests, programming of system information, etc. They are not specified in this document. The IC manufacturer may at its option document them or not. It is allowed that these commands are disabled after IC and/or VICC manufacturing.

10.2 Command codes

Table 18 shows a list of Command codes.

Table 18 — Command codes

Command code	Type	Write alike (See 9.6)	Usage in AuthComm crypto format possible	Usage in SecureComm crypto format possible	Function
'01'	Mandatory	No	No	No	Inventory
'02'	Mandatory	No	No	No	Stay quiet
'03' – '1F'	Mandatory				RFU
'20'	Optional	No	Yes	Yes	Read single block
'21'	Optional	Yes	Yes	Yes	Write single block
'22'	Optional	Yes	Yes	Yes	Lock block
'23'	Optional	No	Yes	Yes	Read multiple blocks
'24'	Optional	Yes	Yes	Yes	Write multiple blocks
'25'	Optional	No	Yes	Yes	Select
'26'	Optional	No	Yes	Yes	Reset to ready
'27'	Optional	Yes	Yes	Yes	Write AFI
'28'	Optional	Yes	Yes	Yes	Lock AFI
'29'	Optional	Yes	Yes	Yes	Write DSFID
'2A'	Optional	Yes	Yes	Yes	Lock DSFID
'2B'	Optional	No	No	No	Get system information
'2C'	Optional	No	No	No	Get multiple block security status
‘2D’	Optional	No	Yes	Yes	Fast read multiple blocks
'30'	Optional	No	Yes	Yes	Extended read single block
'31'	Optional	Yes	Yes	Yes	Extended write single block
'32'	Optional	Yes	Yes	Yes	Extended lock block
'33'	Optional	No	Yes	Yes	Extended read multiple blocks
'34'	Optional	Yes	Yes	Yes	Extended write multiple blocks
'35'	Optional		No	No	Authenticate
'36'	Optional		Yes	Yes	KeyUpdate
'37'	Optional		No	No	AuthComm crypto format indicator
'38'	Optional		No	No	SecureComm crypto format indicator
'39'	Optional		No	No	Challenge
'3A'	Optional	No	Yes	No	ReadBuffer
'3B'	Optional	No	No	No	Extended get system information
'3C'	Optional	No	No	No	Extended get multiple block security status

Table 18 *(continued)*

Command code	Type	Write alike (See 9.6)	Usage in AuthComm crypto format possible	Usage in SecureComm crypto format possible	Function
'3D'	Optional	No	Yes	Yes	Fast extended read multiple blocks
'2E' – '2F' '3E' – '9F'	Optional				RFU
'A0' – 'DF'	Custom				IC Mfg dependent
'E0' – 'FF'	Proprietary				IC Mfg dependent

10.3 Mandatory commands

10.3.1 Inventory

Command code = '01'

When receiving the Inventory request defined in Table 19, the VICC shall perform the anticollision sequence.

The request contains:

- the flags;
- the Inventory command code;
- the AFI if the AFI_flag is set;
- the mask length;
- the mask value;
- the CRC.

The Inventory_flag shall be set to 1.

The meaning of flags 5 to 8 is according to Table 7.

Table 19 — Inventory request format

SOF	Flags	Inventory	Optional AFI	Mask length	Mask value	CRC16	EOF
	8 bits	8 bits	8 bits	8 bits	0 - 64 bits	16 bits	

The response shall contain (see Table 20):

- the DSFID;
- the unique ID.

If the VICC detects an error, it shall remain silent.

Table 20 — Inventory response format

SOF	Flags	DSFID	UID	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	

10.3.2 Stay quiet

Command code = '02'

When receiving the Stay quiet command defined in Table 21, the VICC shall enter the quiet state and shall NOT send back a response. There is NO response to the Stay quiet command.

When in quiet state:

- the VICC shall not process any request where Inventory_flag is set;
- the VICC shall process any addressed request.

The VICC shall exit the quiet state when:

- reset (power off);
- receiving a Select request; it shall then go to the Selected state if supported or return an error if not supported;
- receiving a Reset to ready request; it shall then go to the Ready state.

Table 21 — Stay quiet request format

SOF	Flags	Stay quiet	UID	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	

Request parameter:

UID (mandatory).

The Stay quiet command shall always be executed in ddressed mode (Select_flag is set to 0 and Address_flag is set to 1).

10.4 Optional commands

10.4.1 Read single block

Command code = '20'

When receiving the Read single block command defined in Table 22, the VICC shall read the requested block and send back its value in the response (see Table 23 and Table 24).

If the Option_flag is set in the request, the VICC shall return the block security status, followed by the block value.

If it is not set, the VICC shall return only the block value.

Table 22 — Read single block request format

SOF	Flags	Read single block	UID	Block number	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	16 bits	

Request parameter:

(optional) UID;

block number.

Table 23 — Read single block response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 24 — Read single block response format when Error_flag is NOT set

SOF	Flags	Block security status	Data	CRC16	EOF
	8 bits	8 bits	Block length	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set

Block security status (if Option_flag is set in the request)

Block data

10.4.2 Write single block

Command code = '21'

When receiving the Write single block command defined in Table 25, the VICC shall write the requested block with the data contained in the request and report the success of the operation in the response (see Table 26 and Table 27).

For the Option_flag definition, see 9.6.

Table 25 — Write single block request format

SOF	Flags	Write single block	UID	Block number	Data	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	Block length	16 bits	

Request parameter:

(optional) UID;

block number;

data.

Table 26 — Write single block response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 27 — Write single block response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.3 Lock block

Command code = '22'

When receiving the Lock block command defined in Table 28, the VICC shall lock permanently the requested block and report the success of the operation in the response (see Table 29 and Table 30).

For the Option_flag definition, see 9.6.

Table 28 — Lock single block request format

SOF	Flags	Lock block	UID	Block number	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	16 bits	

Request parameter:

(optional) UID;

block number.

Table 29 — Lock block response format Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 30 — Lock block response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.4 Read multiple blocks

Command code = '23'

When receiving the Read multiple blocks command defined in Table 31, the VICC shall read the requested block(s) and send back their value(s) in the response (see Table 32 and Table 33).

If the Option_flag is set in the request, the VICC shall return the block security status, followed by the block value sequentially block by block.

If the Option_flag is not set in the request, the VICC shall return only the block value.

The blocks are numbered from '00' to 'FF' (0 to 255).

The number of blocks in the request is one less than the number of blocks that the VICC shall return in its response.

EXAMPLE A value of '06' in the "Number of blocks" field requests to read 7 blocks. A value of '00' requests to read a single block.

Table 31 — Read multiple blocks request format

SOF	Flags	Read multiple block	UID	First block number	Number of blocks	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	8 bits	16 bits	

Request parameter:

(optional) UID;

first block number;

number of blocks.

Table 32 — Read multiple blocks response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 33 — Read multiple block response format when Error_flag is NOT set

SOF	Flags	Block security status	Data	CRC16	EOF
	8 bits	8 bits	Block length	16 bits	
		Repeated as needed			

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set (the following order shall be respected in the VICC response)

Block security status N (if Option_flag is set in the request)

Block value N

Block security status N+1 (if Option_flag is set in the request)

Block value N+1

etc.

where N is the first requested (and returned) block.

10.4.5 Write multiple blocks

Command code = '24'

When receiving the Write multiple blocks command defined in Table 34, the VICC shall write the requested block(s) with the data contained in the request and report the success of the operation in the response (see Table 35 and Table 36).

For the Option_flag definition, see 9.6.

The blocks are numbered from '00' to 'FF' (0 to 255).

The number of blocks in the request is one less than the number of blocks that the VICC shall write.

EXAMPLE A value of '06' in the "Number of blocks" field requests to write 7 blocks. The "Data" field contains 7 blocks. A value of '00' in the "Number of blocks" field requests to write 1 block. The "Data" field contains 1 block.

Table 34 — Write multiple blocks request format

SOF	Flags	Write multiple block	UID	First block number	Number of blocks	Data	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	8 bits	Block length	16 bits	
						Repeated as needed		

Request parameter:

(optional) UID;

first block number;

number of blocks;

block data (repeated as defined in Table 34).

Table 35 — Write multiple blocks response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 36 — Write multiple block response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.6 Select

Command code = '25'

When receiving the Select command defined in Table 37:

- If the UID is equal to its own UID, the VICC shall enter the Selected state and shall send a response (see Table 39).
- If the UID is different to its own and in Selected state, the VICC shall return to the Ready state and shall not send a response. The Select command shall always be executed in Addressed mode. (The Select_flag is set to 0. The Address_flag is set to 1.)
- If the UID is different to its own and not in Selected state, the VICC shall remain in its state and shall not send a response.

Table 37 — Select request forma

SOF	Flags	Select	UID	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	

Request parameter:

UID (mandatory).

Table 38 — Select response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 39 — Select response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.7 Reset to ready

Command code = '26'

When receiving a Reset to ready command defined in Table 40, the VICC shall return to the Ready state and report the success of the operation in the response (see Table 41 and Table 42).

Table 40 — Reset to ready request format

SOF	Flags	Reset to ready	UID	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	

Request parameter:

(Optional) UID.

Table 41 — Reset to ready response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 42 — Reset to ready response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.8 Write AFI

Command code = '27'

When receiving the Write AFI request defined in Table 43, the VICC shall write the AFI value into its memory and report the success of the operation in the response (see Table 44 and Table 45).

For the Option_flag definition, see 9.6.

Table 43 — Write AFI request format

SOF	Flags	Write AFI	UID	AFI	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	16 bits	

Request parameter:

(optional) UID;

AFI.

Table 44 — Write AFI response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 45 — Write AFI response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.9 Lock AFI

Command code = '28'

When receiving the Lock AFI request defined in Table 46, the VICC shall lock the AFI value permanently into its memory and report the success of the operation in the response (see Table 47 and Table 48).

For the Option_flag definition, see 9.6.

Table 46 — Lock AFI request format

SOF	Flags	Lock AFI	UID	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	

Request parameter:

(Optional) UID.

Table 47 — Lock AFI response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 48 — Lock AFI response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.10 Write DSFID command

Command code = '29'

When receiving the Write DSFID request defined in Table 49, the VICC shall write the DSFID value into its memory and report the success of the operation in the response (see Table 50 and Table 51).

For the Option_flag definition, see 9.6.

Table 49 — Write DSFID request format

SOF	Flags	Write DSFID	UID	DSFID	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	16 bits	

Request parameter:

(optional) UID;

DSFID.

Table 50 — Write DSFID response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 51 — Write DSFID response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.11 Lock DSFID

Command code = '2A'

When receiving the Lock DSFID request defined in Table 52, the VICC shall lock the DSFID value permanently into its memory and report the success of the operation in the response (see Table 53 and Table 54).

For the Option_flag definition, see 9.6.

Table 52 — Lock DSFID request format

SOF	Flags	Lock DSFID	UID	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	

Request parameter:

(Optional) UID.

Table 53 — Lock DSFID response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 54 — Lock DSFID response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.12 Get system information

Command code = '2B'

The Get system information request command defined in Table 55 allows for retrieving the system information value from the VICC. When receiving this command, the VICC shall report the success of the operation in the response (see Table 56 and Table 57).

Table 55 — Get system information request format

SOF	Flags	Get system info	UID	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	

Request parameter:

(Optional) UID.

Table 56 — Get system information response when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 57 — Get system information response format when Error_flag is NOT set

SOF	Flags	Info flags	UID	DSFID	AFI	Other fields	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	8 bits	See below	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set

Information flag

UID (mandatory)

Information fields, in the order of their corresponding flag, as defined in

Table 57 and Table 58, if their corresponding flag is set.

Table 58 — Information flags definition

Bit	Flag name	Value	Description
b1	DSFID	0	DSFID is not supported. DSFID field is not present.
		1	DSFID is supported. DSFID field is present.
b2	AFI	0	AFI is not supported. AFI field is not present.
		1	AFI is supported. AFI field is present.
b3	VICC memory size	0	Information on VICC memory size defined in Table 59 is not supported. Memory size field is not present.
		1	Information on VICC memory size defined in Table 59 is supported. Memory size field is present.
b4	IC reference	0	Information on IC reference is not supported. IC reference field is not present.
		1	Information on IC reference is supported. IC reference field is present.
b5	Proprietary	x	Not defined in this document.
b6	Proprietary	x	Not defined in this document.
b7	Proprietary	x	Not defined in this document.
b8	Proprietary	x	Not defined in this document.

Table 59 — VICC memory size information

MSB		LSB
16 14	13 9	8 1
Proprietary	Block size in bytes	Number of blocks

Proprietary bits b14 to b16 are not defined in this document.

The block size is expressed in number of bytes on 5 bits, allowing to specify up to 32 bytes i.e. 256 bits. It is one less than the actual number of bytes.

EXAMPLE value of '1F' indicates 32 bytes, a value of '00' indicates 1 byte.

The number of blocks is on 8 bits, allowing to specify up to 256 blocks. It is one less than the actual number of blocks.

EXAMPLE A value of 'FF' indicates 256 blocks, a value of '00' indicates 1 block.

The three most significant bits are reserved for future use and shall be set to zero.

The IC reference is on 8 bits and its meaning is defined by the IC manufacturer.

10.4.13 Get multiple block security status

Command code = '2C'

When receiving the Get multiple block security status command defined in Table 60, the VICC shall send back the block security status (see Table 61 and Table 62).

The blocks are numbered from '00 to 'FF' (0 to 255).

The number of blocks in the request is one less than the number of block security status that the VICC shall return in its response.

EXAMPLE A value of '06' in the "Number of blocks" field requests to return 7 Block security status. A value of '00' in the "Number of blocks" field requests to return a single Block security status.

Table 60 — Get multiple block security status request format

SOF	Flags	Get multiple block security status	UID	First block number	Number of blocks	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	8 bits	16 bits	

Request parameter:

(optional) UID;

first block number;

number of blocks.

Table 61 — Get multiple block security status response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 62 — Get multiple block security status response format when Error_flag is NOT set

SOF	Flags	Block security status	CRC16	EOF
	8 bits	8 bits	16 bits	
		Repeated as needed		

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set Block security status (repeated as per Table 61)

10.4.14 Fast read multiple blocks

Command code = ‘2D’

When receiving the Fast read multiple blocks command defined in Table 64, the VICC shall read the requested block(s) and send back their value in the response (see Table 65 and Table 66).

The VICC shall send back this response using the fast response data rate as determined by the 2nd and 8th bits in Request flag as defined in Table 63.

Table 63 — Fast response data rates supported by VICC

b2	b8	Meaning
1	0	High data rate [26,48 kbits/s (f_c /512)]
0	1	X2 [52,97 kbits/s (f_c /256)]
1	1	X4 [105,94 kbits/s (f_c /128)]
0	0	X8 [211,88 kbits/s (f_c /64)]

Sub-carrier_flag shall be set to 0.

If two subcarriers are requested and the VICC responds with an error code it shall be with two subcarriers at the Low or High data rate according to the value of b2 as defined in Table 5.

NOTE 1 If the command is not supported and the VICC responds with an error code it will be at the Low or High data rate according to the value of b2 and the selection of one or two subcarrier according to the value of b1 as defined in Table 5.

If the command is supported and Sub-carrier_flag is set to 0, the data rate for the response with or without an error code shall be according to the value of b2 and b8.

The VICC shall respond to the Fast read multiple blocks command accordingly to 9.9. If the Waiting time extension sequence is required it shall only be used once. The timing specifications are independent of the data rate.

If the Option_flag is set in the request, the VICC shall return the block security status, followed by the block value sequentially block by block.

If the Option_flag is not set in the request, the VICC shall return only the block value.

The blocks are numbered from ‘00’ to ‘FF’ (0 to 255).

The number of blocks in the request is one less than the number of blocks that the VICC shall return in its response.

EXAMPLE A value of '06' in the "Number of blocks" field requests to read 7 blocks. A value of '00' requests to read a single block.

Table 64 — Fast read multiple blocks request format

SOF	Flags	Fast read multiple block	UID	First block number	Number of blocks	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	8 bits	16 bits	

Request parameter:

(optional) UID;

first block number;

number of blocks.

Table 65 — Fast read multiple blocks response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Response format when Error_flag is not set and response is available within 20 ms.

Table 66 — Fast read multiple blocks response format when Error_flag is NOT set and response is available within 20ms

SOF	Flags	Block security status	Data	CRC16	EOF
	8 bits	8 bits	Block length	16 bits	
		Repeated as needed			

NOTE 2 The block security status can be extended to 2, 4 or 8 bytes in future revisions of this document.

If the response is not available within 20 ms, VICC can request a waiting time extension by using the Waiting time extension reply as defined in 7.4.5.

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set (the following order shall be respected in the VICC response)

Block security status N (if Option_flag is set in the request)

Block value N

Block security status N+1 (if Option_flag is set in the request)

Block value N+1

etc.

where N is the first requested (and returned) block.

10.4.15 Extended read single block

Command code = '30'

When receiving the Extended read single block command defined in Table 67, the VICC shall read the requested block and send back its value in the response (see Table 68 and Table 69).

If a VICC supports Extended read single block command, it shall also support Read single block command for the first 256 blocks of memory.

If the Option_flag is set in the request, the VICC shall return the block security status, followed by the block value.

If it is not set, the VICC shall return only the block value.

Table 67 — Extended read single block request format

SOF	Flags	Extended read single block	UID	Block number	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	16 bits	

Request parameter:

(optional) UID;

block number.

Table 68 — Extended read single block response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 69 — Extended read single block response format when Error_flag is NOT set

SOF	Flags	Block security status	Data	CRC16	EOF
	8 bits	8 bits	Block length	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set

Block security status (if Option_flag is set in the request)

Block data

10.4.16 Extended write single block

Command code = '31'

When receiving the Extended write single block command defined in Table 70, the VICC shall write the requested block with the data contained in the request and report the success of the operation in the response (see Table 71 and Table 72).

If a VICC supports Extended write single block command, it shall also support Write single block command for the first 256 blocks of memory.

For the Option_flag definition, see 9.6.

Table 70 — Extended write single block request format

SOF	Flags	Extended write single block	UID	Block number	Data	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	Block length	16 bits	

Request parameter:

(optional) UID;

block number;

data.

Table 71 — Extended write single block response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 72 — Extended write single block response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.17 Extended lock block

Command code = '32

When receiving the Extended lock block command defined in Table 73, the VICC shall lock permanently the requested block and report the success of the operation in the response (see Table 74 and Table 75).

If a VICC supports Extended lock block command, it shall also support Lock block command for the first 256 blocks of memory.

For the Option_flag definition, see 9.6.

Table 73 — Extended lock single block request format

SOF	Flags	Extended lock block	UID	Block number	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	16 bits	

Request parameter:

(optional) UID;

block number.

Table 74 — Extended lock block response format Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 75 — Extended lock block response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.18 Extended read multiple block

Command code = '33'

When receiving the Extended read multiple blocks command defined in Table 76, the VICC shall read the requested block(s) and send back their value in the response (see Table 77 and Table 78).

If a VICC supports Extended read multiple blocks command, it shall also support Read multiple blocks command for the first 256 blocks of memory.

If the Option_flag is set in the request, the VICC shall return the block security status, followed by the block value sequentially block by block.

If the Option_flag is not set in the request, the VICC shall return only the block value.

The blocks are numbered from '0000' to 'FFFF' (0 to 65 535).

The number of blocks in the request is one less than the number of blocks that the VICC shall return in its response.

EXAMPLE A value of ‘0006’ in the "Number of blocks" field requests to read 7 blocks. A value of ‘0000’ requests to read a single block.

Table 76 — Extended read multiple blocks request format

SOF	Flags	Extended read multiple block	UID	First block number	Number of blocks	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	16 bits	16 bits	

Request parameter:

(optional) UID;

first block number;

number of blocks.

Table 77 — Extended read multiple blocks response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 78 — Extended read multiple block response format when Error_flag is NOT set

SOF	Flags	Block security status	Data	CRC16	EOF
	8 bits	8 bits	Block length	16 bits	
		Repeated as needed			

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set (the following order shall be respected in the VICC response)

Block security status N (if Option_flag is set in the request)

Block value N

Block security status N+1 (if Option_flag is set in the request)

Block value N+1

etc.

where N is the first requested (and returned) block.

10.4.19 Extended write multiple blocks

Command code = ‘34’

When receiving the Extended write multiple blocks command defined in Table 79, the VICC shall write the requested block(s) with the data contained in the request and report the success of the operation in the response (see Table 80 and Table 81).

If a VICC supports Extended write multiple blocks command, it shall also support Write multiple blocks command for the first 256 blocks of memory.

For the Option_flag definition, see 9.6.

The blocks are numbered from ‘0000’ to ‘FFFF’ (0 to 65 535).

The number of blocks in the request is one less than the number of blocks that the VICC shall write.

EXAMPLE A value of ‘0006’ in the "Number of blocks" field requests to write 7 blocks. The "Data" field contains 7 blocks. A value of ‘0000’ in the "Number of blocks" field requests to write 1 block. The "Data" field contains 1 block.

Table 79 — Extended write multiple blocks request format

SOF	Flags	Extended write multiple block	UID	First block number	Number of blocks	Data	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	16 bits	Block length	16 bits	
						Repeated as needed		

Request parameter:

(optional) UID;

first block number;

number of blocks;

block data (repeated as defined in Table 79).

Table 80 — Extended write multiple blocks response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 81 — Extended write multiple block response format when Error_flag is NOT set

SOF	Flags	CRC16	EOF
	8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

10.4.20 Extended get multiple block security status

Command code = ‘3C’

When receiving the Extended get multiple block security status command defined in Table 82, the VICC shall send back the block security status (see Table 83 and Table 84).

The blocks are numbered from ‘0000’ to ‘FFFF’ (0 to 65 535).

The number of blocks in the request is one less than the number of block security status that the VICC shall return in its response.

EXAMPLE A value of ‘0006’ in the "Number of blocks" field requests to return 7 Block security status. A value of ‘0000’ in the "Number of blocks" field requests to return a single Block security status.

Table 82 — Extended get multiple block security status request format

SOF	Flags	Extended get multiple block security status	UID	First block number	Number of blocks	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	16 bits	16 bits	

Request parameter:

(optional) UID;

first block number;

number of blocks.

Table 83 — Extended get multiple block security status response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 84 — Extended get multiple block security status response format when Error_flag is NOT set

SOF	Flags	Block security status	CRC16	EOF
	8 bits	8 bits	16 bits	
		Repeated as needed		

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set Block security status (repeated as per Table 84)

10.4.21Fast extended read multiple blocks

Command code = ‘3D’

When receiving the Fast extended read multiple blocks command defined Table 85, the VICC shall read the requested block(s) and send back their value in the response (see Table 86 and Table 87).

The VICC shall send back this response using the fast response data rate as determined by the 2nd and 8th bits in Request flag as defined in Table 63.

Sub-carrier_flag shall be set to 0.

If two sub-carriers are requested and the VICC responds with an error code it shall be with two subcarriers at the Low or High data rate according to the value of b2 as defined in Table 5.

NOTE 1 If the command is not supported and the VICC responds with an error code it will be at the Low or High data rate according to the value of b2 and the selection of one or two subcarrier according to the value of b1 as defined in Table 5.

If the command is supported and Sub-carrier_flag is set to 0, the data rate for the response with or without an error code shall be according to the value of b2 and b8.

The VICC shall respond to the Fast extended read multiple blocks command accordingly to 9.9. If the Waiting time extension sequence is required, it shall only be used once. The timing specifications are independent of the data rate.

If the Option_flag is set in the request, the VICC shall return the block security status, followed by the block value sequentially block by block.

If the Option_flag is not set in the request, the VICC shall return only the block value.

The blocks are numbered from ‘0000’ to ‘FFFF’ (0 to 65 535).

The number of blocks in the request is one less than the number of blocks that the VICC shall return in its response.

EXAMPLE A value of '0006' in the "Number of blocks" field requests to read 7 blocks. A value of '0000' requests to read a single block.

Table 85 — Fast extended read multiple blocks request format

SOF	Flags	Fast extended read multiple block	UID	First block number	Number of blocks	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	16 bits	16 bits	

Request parameter:

(optional) UID;

first block number;

number of blocks.

Table 86 — Fast extended read multiple blocks response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 87 — Fast extended read multiple blocks response format when Error_flag is NOT set and response is available within 20 ms

SOF	Flags	Block security status	Data	CRC16	EOF
	8 bits	8 bits	Block length	16 bits	
		Repeated as needed			

NOTE 2 The block security status can be extended to 2, 4 or 8 bytes in future revisions of this document.

If the response is not available within 20 ms, VICC can request a waiting time extension by using the Waiting time extension reply as defined in 7.4.5.

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set (the following order shall be respected in the VICC response)

Block security status N (if Option_flag is set in the request)

Block value N

Block security status N+1 (if Option_flag is set in the request)

Block value N+1

etc.

where N is the first requested (and returned) block.

10.4.22 Authenticate

Command code = ’35’

The Authenticate command defined in Table 88 is used to perform VICC, VCD or Mutual Authentication.

The Authenticate command supports a variety of CS as indicated by the CSI field. The supported authenticate method, number and sequence of Authenticate commands depends on the selected CS and are defined inside payload field.

CS may define VICC Authentication only, VCD Authentication only or Mutual Authentication. The VCD shall not integrate an Authenticate command in a SecureComm or AuthComm crypto format indicator.

A VICC receiving an Authenticate command with an unsupported CSI shall not execute the Authenticate command and remain in the current state.

If the VICC receives an Authenticate command but there is a cryptographic error, and the cryptographic suite specifies that the error requires a security timeout, the VICC shall set a security timeout as specified in 9.7.

If the VICC supports security timeout for the Authenticate command and receives an Authenticate during a timeout then it shall reject the command, send the generic crypto error code and remain in its current state.

The VICC decides whether the data field is included in the Final response to this command or stored in the ResponseBuffer.

If response flag b3 is set to 1, the Final response includes the data field with valid cryptographic results. The VICC may also store the results inside a ResponseBuffer and shall set b2 to 1.

If the response flag b3 is set to 0, the Final response does not include the data field. The VICC shall store the cryptographic results inside the ResponseBuffer and shall set b2 to 1.

If a ResponseBuffer is supported, the VICC shall clear the Validity flag (flag b2) immediately after receiving the Authenticate command and shall set this flag after a successful execution of Authenticate command and successful storage of the results of the cryptographic calculation in the VICC ResponseBuffer.

In case of an in-process reply:

— The Initial response or the Barker response shall always contain the Barker field with Done flag set to 0.

— The Final response to this command shall always contain the Barker field with Done flag set to 1.

The reply timing for this command is defined by the CS (Immediate or In-Process reply).

Table 88 — Authenticate request format

SOF	Flags	Authenticate	UID	CSI	Message	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	Size and content defined by the crypto suite specified by the CSI	16 bits	

Request parameter:

(optional) UID;

CSI;

message.

Response parameter:

The response formats for this command are defined in 7.4. The response payload is defined by the CS.

10.4.23 KeyUpdate

Command code =’36’

The KeyUpdate command defined in Table 89 allows the VCD to write or overwrite a key in the VICC. The KeyUpdate shall only be executed when the VICC is in the Selected Secure state. The payload message and reply formatting are specified by the CSI used by the previous Authenticate command.

The VCD may send the KeyUpdate command in a SecureComm crypto format indicator or an AuthComm crypto format indicator. If the CS requires to send the KeyUpdate command in a SecureComm crypto format then the payload message shall not be encrypted. If the CS allows to send the KeyUpdate command in an AuthComm crypto format or without usage of a crypto format then the payload message shall be encrypted.

The VICC shall answer the KeyUpdate command using the In-process reply as defined in 9.8.

If an error occurs, the response may be an Immediate reply.

Table 89 — Key Update request format

SOF	Flags	KeyUpdate	UID	Key ID	Message	CRC16	EOF
	8 bits	8 bits	64 bits	8 bits	Size and content defined by the crypto suite specified by the CSI	16 bits	

The VICC shall update its current key value with a new key value. If the VICC does not write the new key successfully then it shall revert to the prior key. Any KeyUpdate shall be atomic in nature. For example a VICC may meet this requirement by double-buffering the write operation.

A VICC shall only accept a KeyUpdate command in Selected Secure state. If a KeyUpdate command is received in another state, the KeyUpdate command shall be ignored and the VICC shall remain in the current state.

If a cryptographic error occurs when processing a properly formatted KeyUpdate command of a CS requiring a security timeout the VICC shall set a security timeout if supported and shall reply with an error response (generic crypto error code).

During a security timeout the VICC shall reject Authenticate, Challenge or KeyUpdate commands or any command sent in a crypto format. The VICC shall reply with an error response except for the Challenge command (generic crypto error code) and shall remain in its current state.

Request parameter:

(optional) UID;

key ID;

message.

Response parameter:

The response formats for this command are defined in 7.4. The response payload is defined by the CS.

10.4.24Challenge

Command code =’39’

The Challenge command defined in Table 90 allows the VCD to instruct single or multiple VICCs to simultaneously and independently precompute cryptographic values for use in a subsequent VICC, VCD or mutual Authentication. The VICC shall not reply and shall store its response (computed values) for subsequent authentication in the ResponseBuffer.

The VCD may subsequently read the ResponseBuffer using a ReadBuffer command. If an error occurs during the execution of a Challenge command, the VICC shall respond to this ReadBuffer command with a standard error response (e.g. generic crypto error code).

The Challenge command shall only be executed in Non-addressed mode. If the Address or Select_flag is set, the VICC shall ignore it and keep the current state. A VICC shall only execute a Challenge command in ready state.

After receiving a Challenge command, the VICC shall immediately clear the validity response flag (flag b2) and shall set this flag after a successful execution of the challenge and successful storage of the result of the cryptographic calculation in the VICC ResponseBuffer.

If a VCD sends any command while the VICC is processing a Challenge command, the VICC may suspend its processing and execute the new incoming command, or in environments with limited power availability, undergo a power-on reset.

After transmitting a Challenge command, VCD should send un-modulated carrier for a sufficient period of time defined by CS allowing all VICCs to compute and store their results.

Table 90 — Challenge request format

SOF	Flags	Challenge	CSI	Message	CRC 16	EOF
	8 bits	8 bits	8 bits	Size and content defined by the crypto suite specified by the CSI	16 bits	

Request parameter:

CSI;

message.

Response parameter:

No answer from the VICC

10.4.25 ReadBuffer

Command code =’3A’

If a ResponseBuffer is implemented the ReadBuffer command defined in Table 91 shall be supported. After receiving the ReadBuffer command, the VICC shall send back to the VCD the data stored in the ResponseBuffer (see Table 92 and Table 93).

If a ResponseBuffer is not implemented, the VICC may return an error code after receiving an addressed or selected ReadBuffer command.

The VCD may use a ReadBuffer command in an AuthComm crypto format indicator but shall not use it in a SecureComm crypto format indicator.

The VICC shall return its response as Immediate reply or as In-process reply if used in an AuthComm crypto format indicator.

The response shall contain the calculated data padded with least significant 0 bits as required to a minimum multiple of 8 bits or an error code.

Table 91 — ReadBuffer request format

SOF	Flags	ReadBuffer	UID	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	

Request parameter:

(Optional) UID.

Table 92 — ReadBuffer response format when Error_Flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 93 — ReadBuffer response format when Error_Flag is NOT set

SOF	Flags	ReadBuffer Data	CRC16	EOF
	8 bits	multiple of 8 bits	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set

ReadBuffer Data bits (minimum multiple of 8 bits)

10.4.26 Extended get system information

Command code = '3B'

The Extended get system information request command defined in Table 94 and Table 95 allows for retrieving the system information value from the VICC and shall be supported by the VICC if extended memory, security functionalities or Fast response data rate are supported by the VICC. When receiving this command, the VICC shall report the success of the operation in the response (see Table 96 and Table 97).

Table 94 — Extended get system information request format

SOF	Flags	Extended Get System info	Get System info parameter request field	UID	CRC16	EOF
	8 bits	8 bits	8 bits or 16 bits	64 bits	16 bits	

Request parameter:

Get System info parameter request field;

Table 95 — Get System info parameter request field

Bit	Flag name	Value	Description
b1	DSFID	0	No request of DSFID
		1	Request of DSFID
b2	AFI	0	No request of AFI
		1	Request of AFI
b3	VICC memory size	0	No request of data field on VICC memory size
		1	Request of data field on VICC memory size
b4	IC reference	0	No request of Information on IC reference
		1	Request of Information on IC reference
b5	MOI	1	Information on MOI always returned in response flag
b6	VICC Command list	0	No request of Data field of all supported commands
		1	Request of Data field of all supported commands
b7	CSI Information	0	No request of CSI list
		1	Request of CSI list
b8	Get System Info parameter request field length	0	One byte
		1	Two bytes

The bit b8, Get System Info parameter request field length shall be set to 0. When receiving a get system information command with info parameter bit b8 set to 1, the VICC shall send an error code.

(optional) UID.

Table 96 — Extended get system information response when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 97 — Extended get system information response format when Error_flag is NOT set

SOF	Flags	Info flags	UID	DSFID	AFI	Other fields	CRC16	EOF
	8 bits	8 bits or 16 bits	64 bits	8 bits	8 bits	See below	16 bits	

Response parameter:

Error_flag (and Error code if Error_flag is set)

if Error_flag is not set

Information flags

UID (mandatory)

Information fields, in the order of their corresponding flag, as defined in Table 95 and Table 98, if their corresponding flag is set.

Table 98 — Information flags definition

Bit	Flag name	Value	Description
b1	DSFID	0	DSFID field is not present.
		1	DSFID field is present.
b2	AFI	0	AFI field is not present.
		1	AFI field is present.
b3	VICC memory size	0	Data field on VICC memory size defined in Table 99 is not present.
		1	Data field on VICC memory size defined in Table 99 is present.
b4	IC reference	0	Information on IC reference field is not present.
		1	Information on IC reference field is present.
b5	MOI	0	1 byte memory addressing
		1	2 bytes memory addressing
b6	VICC Command list	0	Data field of all supported commands defined in Table 100 to Table 104 is not present.
		1	Data field of all supported commands defined in Table 100 to Table 104 is present.
b7	CSI Information	0	CSI list defined in Table 105 is not present.
		1	CSI list defined in Table 105 is present.
b8	Info flag length	0	One byte
		1	Two byte

The bit b8, Info flag length shall be set to 0. If set to 1, the response shall be ignored.

If the DSFID information was requested by the VCD but is not returned by the VICC then the DSFID functionality is not supported by the VICC.

If the AFI information was requested by the VCD but is not returned by the VICC then the AFI functionality is not supported by the VICC.

The VICC memory size information when requested by the VCD shall always be returned by the VICC if extended memory or security functionalities are supported by the VICC.

The IC reference information when requested by the VCD shall always be returned by the VICC if extended memory or security functionalities are supported by the VICC.

The VICC Command list when requested by the VCD shall always be returned by the VICC if extended memory or security functionalities are supported by the VICC.

The CSI list when requested by the VCD shall always be returned by the VICC if security functionality is supported by the VICC.

Other field information description is as follows.

The VICC memory size information is shown in Table 99.

Table 99 — VICC memory size information

MSB		LSB
24 22	21 17	16 1
RFU	Block size in bytes	Number of blocks

The block size shall be expressed in number of bytes on 5 bits, allowing to specify up to 32 bytes i.e. 256 bits. It is one less than the actual number of bytes.

EXAMPLE A value of '1F' indicates 32 bytes, a value of '00' indicates 1 byte.

The number of blocks is on 16 bits, allowing to specify up to 65 536 blocks. It is one less than the actual number of blocks.

EXAMPLE A value of 'FFFF' indicates 63 536 blocks, a value of '0000' indicates 1 block.

The three most significant bits are reserved for future use and shall be set to zero.

The IC reference shall be coded as an 8 bits binary integer and is defined by the IC manufacturer.

For the supported command list, Byte 1, 2, 3 and 4 are mandatory.

Table 100 — VICC memory supported commands information

MSB			LSB
32 25	24 17	16 09	08 01
Byte 4	Byte 3	Byte 2	Byte 1

Table 101 — Byte 1 definition

Bit	Meaning if bit is set	Comment
b1	Read single block is supported.	
b2	Write single block is supported.	
b3	Lock single block is supported.	
b4	Read multiple blocks is supported.	
b5	Write multiple blocks is supported.	
b6	Select is supported.	Including Selected state.
b7	Reset to Ready is supported.	
b8	Get multiple block security status is supported.	

Table 102 — Byte 2 definition

Bit	Meaning if bit is set	Comment
b1	Write AFI is supported.	
b2	Lock AFI is supported.	
b3	Write DSFID is supported.	
b4	Lock DSFID is supported.	
b5	Get System Information is supported.	
b6	Custom commands are supported.	
b7	Fast read multiple blocks is supported.	
b8	RFU	0 shall be returned.

Table 103 — Byte 3 definition

Bit	Meaning if bit is set	Comment
b1	Extended read single block is supported.	
b2	Extended write single block is supported.	
b3	Extended lock single Block is supported.	
b4	Extended read multiple blocks is supported.	
b5	Extended write multiple blocks is supported.	
b6	Extended get multiple block security status is supported.	
b7	Fast Extended read multiple blocks is supported.	
b8	RFU	0 shall be returned.

Table 104 — Byte 4 definition

Bit	Meaning if bit is set	Comment
b1	ReadBuffer is supported.	Meaning that Response Buffer is supported.
b2	Selected Secure State is supported.	Meaning that VCD or Mutual Authentication are supported.
b3	Final response always includes crypto result.	Meaning that flag b3 will be set in the Final response.
b4	AuthComm crypto format is supported.	
b5	SecureComm crypto format is supported.	
b6	KeyUpdate is supported.	
b7	Challenge is supported.	
b8	If set to 1 a further Byte is transmitted.	Allowing future extensions.

Table 105 — CSI list definition

MSB		LSB
... ...	16 09	08 01
Byte ...	Byte 2	Byte 1

Byte 1 indicates as a binary integer the number of different CS supported by the VICC. Subsequent bytes indicate the CSI values as specified in ISO/IEC 29167-1.

A VICC not supporting any CS shall set bit 7 of Information flags byte to 0.

NOTE CS list is not present.

10.5 Custom commands

The format of custom commands is generic and allows unambiguous attribution of custom command codes to each VICC manufacturer.

The custom command code defined in Table 106 is the combination of a custom command code and of the VICC manufacturer code.

The format of response to the custom commands is generic as shown in Table 107 and Table 108.

The custom request and response parameters definition is the responsibility of the VICC manufacturer.

Table 106 — Custom request format

SOF	Flags	Custom	IC Mfg code	Custom request parameters	CRC16	EOF
	8 bits	8 bits	8 bits	Custom defined	16 bits	

Request parameter:

IC manufacturer code defined in ISO/IEC 7816-6.

Table 107 — Custom response format when Error_flag is set

SOF	Flags	Error code	CRC16	EOF
	8 bits	8 bits	16 bits	

Table 108 — Custom response format when Error_flag is NOT set

SOF	Flags	Custom response parameters	CRC16	EOF
	8 bits	Custom defined	16 bits	

Response parameter:

Error_flag (and code if Error_flag is set)

if Error_flag is not set

Custom parameters

10.6 Proprietary commands

The format of these commands is not defined in this document.

11 Secured Communication

11.1 General

The following subclauses describe crypto format identifiers. These identifiers contain information about the payload content.

11.2 AuthComm

Crypto format indicator ='37'

The AuthComm crypto format indicator defined in Table 109 provides information about the payload which includes the command to be executed and optional security features. A VICC may accept the AuthComm crypto format indicator in any state. The usage of an AuthComm crypto format indicator shall always be preceded by a VICC, VCD or Mutual Authentication via an Authenticate or a Challenge command. The CS indicated by the CSI in the Authenticate or Challenge command that preceded the AuthComm specifies the payload message and reply formatting.

A VICC that receives an AuthComm crypto format indicator followed by a command in the payload that it does not support or is not allowed in the current state or is not supported by the AuthComm crypto format indicator shall not execute the command in the payload.

If a cryptographic error occurs when processing a properly formatted AuthComm crypto format indicator of a CS requiring a security timeout, the VICC shall set a security timeout if supported and shall reply with an error response (generic crypto error code).

During security timeout, the VICC shall reject Authenticate, Challenge or KeyUpdate commands or any command sent in a crypto format indicator, The VICC shall reply with an error response except for the Challenge command (generic crypto error code) and shall remain in its current state.

The VICC may answer with an Immediate reply or with the In-process reply as defined in 9.7 (depending on the command in the payload and on the CS).

Table 109 — AuthComm request format

SOF	Flags	AuthComm Format indicator	UID	Message	CRC16	EOF
	8 bits	8 bits	64 bits	ISO/IEC15693-3 command including flags, payload, CRC excluding SOF and EOF and optional security features as described in CS	16 bits	

Request parameters:

(optional) UID;

message.

Response parameters:

The response formats for this command are defined in 7.4.

The response payload is defined by the CS.

The Payload is composed of the complete message response (including Flags and CRC16) to the command in the AuthComm crypto format indicator payload plus the security feature [(eg. Message Authenticate Code (MAC)] calculated over this complete message.

NOTE The response frame contains flags and a CRC16 as defined in 7.4.

11.3 SecureComm

Crypto format indicator ='38'

The SecureComm crypto format indicator defined in Table 110 provides information about the payload which is encrypted including the command to be executed. A VICC may accept the SecureComm crypto format indicator in any state. The usage of a SecureComm crypto format indicator shall always be preceded by a VICC, VCD or Mutual Authentication via an Authenticate or a Challenge command. The CS indicated by the CSI in the Authenticate or Challenge command that preceded the SecureComm specifies the payload message and reply formatting.

A VICC that receives a SecureComm crypto format followed by a command in the payload that it does not support or is not allowed in the current state or is not supported by the SecureComm crypto format shall not execute the command in the payload.

If a cryptographic error occurs when processing a properly formatted SecureComm crypto format of a CS requiring a security timeout, the VICC shall set a security timeout if supported and shall reply with an error response (generic crypto error code).

During security timeout, the VICC shall reject Authenticate, Challenge or KeyUpdate commands or any command sent in a crypto format indicator. The VICC shall reply with an error response except for the Challenge command (generic crypto error code) and shall remain in its current state.

The VICC may answer with an Immediate reply or with the In-process reply as defined in 9.7 (depending on the command in the payload and on the CS).

Table 110 — SecureComm request format

SOF	Flags	SecureComm Format indicator	UID	Length	Message	CRC16	EOF
	8 bits	8 bits	64 bits	16 bits	Encrypted ISO/IEC15693-3 command including flags, payload, CRC excluding SOF and EOF and optional security features as described in CS	16 bits	

Request parameter:

(optional) UID;

length: describes length of message in bytes (e.g. a length of 5 indicates a message length of 40 bits);

message.

Response parameters:

The response formats for this command are defined in 7.4. The response payload is defined by the CS.

The Payload is composed of the complete encrypted message response (including Flags and CRC16) to the command in the SecureComm crypto format indicator payload.

NOTE The response frame contains flags and a CRC16 as defined in 7.4.

Annex A
(informative)

Compatibility with other card standards

This document does not preclude the addition of other existing card standards on the VICC, such as ISO/IEC 7816 or others listed in the bibliography.

Annex B
(informative)

VCD pseudo-code for anticollision

The following pseudo-code in italics describes how the anticollision can be implemented on the VCD, using recursivity. It does not describe the collision detection mechanism

```
Algorithm for 16 slots

     function push (mask, address)      ; pushes on private stack
     function pop (mask, address)       ; pops from private stack
     function pulse_next_pause          ; generates a power pulse
     function store(VICC_UID)           ; stores VICC_UID
     function poll_loop (sub_address_size as integer)

; address length must be four (4) bits.
pop (mask, address)
mask = address & mask                   ; generates new mask

; send the Request
mode = anticollision
send_Request(Request_cmd, mode, mask length, mask)

for address = 0 to (2^sub_address_size - 1)
     if no_collision_is_detected then ; VICC is inventoried
          store (VICC_UID)
     else                             ; remember a collision was detected
          push(mask,address)
     endif

pulse_next_pause

next sub_address
; if some collisions have been detected and not yet processed,
; the function calls itself recursively to process the last
; stored collision
if stack_not_empty then poll_loop (sub_address_size)

end poll_loop

main_cycle:
mask = null
address = null
push (mask, address)
poll_loop(sub_address_size)
end_main_cycle
```

Annex C
(informative)

Cyclic redundancy check (CRC)

C.1 The CRC error detection method

The CRC is calculated on all data contained in a message, from the start of the flags through to the end of data. This CRC is used from VCD to VICC and from VICC to VCD. Table C.1 shows the definition of the CRC.

Table C.1 — CRC Definition

CRC Definition					
CRC type	**Length**	**Polynomial**	**Direction**	**Preset**	**Residue**
ISO/IEC 13239	16 bits	$X^{16} + X^{12} + X^5 + 1$	Backward	'FFFF'	'F0B8'

To add extra protection against shifting errors, a further transformation is made on the calculated CRC. The value attached to the message for transmission is the 1's complement of the calculated CRC. This transformation is included in the example below.

For checking of received messages the 2 CRC bytes are often also included in the re-calculation, for ease of use. In this case, for the expected value for the generated CRC the value of the residue is 'F0B8'.

The following text in italics is an example which illustrates one method in C language of calculating the CRC on a given set of bytes comprising a message.

```
#include <stdio.h>

#define POLYNOMIAL 0x8408 // x^16 + x^12 + x^5 + 1
#define PRESET_VALUE 0xFFFF
#define CHECK_VALUE 0xF0B8
#define NUMBER_OF_BYTES 4 // Example: 4 data bytes
#define CALC_CRC 1
#define CHECK_CRC 0

void main()
{
unsigned int current_crc_value;
unsigned char array_of_databytes[NUMBER_OF_BYTES + 2] = {1, 2, 3, 4, 0x91, 0x39};
int number_of_databytes = NUMBER_OF_BYTES;
int calculate_or_check_crc;
int i, j;
calculate_or_check_crc = CALC_CRC;
// calculate_or_check_crc = CHECK_CRC; // This could be an other example
if (calculate_or_check_crc == CALC_CRC)
{
    number_of_databytes = NUMBER_OF_BYTES;
}
else // check CRC
{
    number_of_databytes = NUMBER_OF_BYTES + 2;
}
current_crc_value = PRESET_VALUE;
for (i = 0; i < number_of_databytes; i++)
{
 current_crc_value = current_crc_value ^ ((unsigned int)array_of_databytes[i]);
 for (j = 0; j < 8; j++)
 {
   if (current_crc_value & 0x0001)
   {
      current_crc_value = (current_crc_value >> 1) ^ POLYNOMIAL;
   }
   else
   {
      current_crc_value = (current_crc_value >> 1);
   }
  }
}
if (calculate_or_check_crc == CALC_CRC)
{
    current_crc_value = ~current_crc_value;
    printf ("CRC-ISO/IEC 13239 of { 1, 2, 3, 4 } is '3991'\n");
    printf ("Generated CRC is '%04X'\n", current_crc_value);
    printf ("The Least Significant Byte (transmitted first) is: '%02X'\n",
            current_crc_value & 0xFF);
    printf( "The Most Significant Byte (transmitted second) is: '%02X'\n",
           (current_crc_value >> 8) & 0xFF);
    printf( "Executing this program when CHECK_CRC generates: 'F0B8'\n");
// current_crc_value is now ready to be appended to the data stream
// (first LSByte, then MSByte)
}
else // check CRC
{
    if (current_crc_value == CHECK_VALUE)
    {
      printf ("Checked CRC is ok (0x%04X)\n", current_crc_value);
    }
    else
    {
      printf ("Checked CRC is NOT ok (0x%04X)\n", current_crc_value);
    }
}
```

The above C-program generates the following printout when executed.

```
CRC-ISO/IEC 13239 of { 1, 2, 3, 4 } is '3991'
Generated CRC is '3991'
The Least Significant Byte (transmitted first) is: '91'
The Most Significant Byte (transmitted second) is: '39'
Executing this program when CHECK_CRC generates: 'F0B8'
```

C.2 CRC calculation example

This example refers to a Read single block request for reading block '0B' (see Table C.2).

The modes selected by the VCD are: single subcarrier, high VICC-to-VCD data rate, addressed.

The UID of the VICC is: 'E0 04 AB 89 67 45 23 01'.

The request therefore consists of the following fields:

— the flags: '22';

— the command code: '20';

— the UID: 'E0 04 AB 89 67 45 23 01' where 'E0' is the most significant byte;

— the block number: '0B';

— the CRC: 'BAE3' where 'BA' is the most significant byte.

The CRC is calculated on the request fields assembled in a frame according to the transmission rules defined in this document:

'22' '20' '01' '23' '45' '67' '89' 'AB' '04' 'E0' '0B'

NOTE 1 The UID is transmitted least significant byte first.

NOTE 2 Table C.2 describes the different steps of the calculation.

The request is then transmitted as follows:

SOF '22' '20' '01' '23' '45' '67' '89' 'AB' '04' 'E0' '0B' 'E3' 'BA' EOF

NOTE 3 The CRC is transmitted least significant byte first.

NOTE 4 Each byte is transmitted least significant bit first.

Table C.2 — Practical example of CRC calculation

Step	Input	Calculated CRC in VCD	Calculated CRC in VICC for check
1	'22'	'F268	'0D97'
2	'20'	'3EC6'	'C139'
3	'01'	'42F5'	'BD0A'
4	'23'	'4381'	'BC7E'
5	'45'	'7013'	'8FEC'
6	'67'	'C5AB'	'3A54'
7	'89'	'F2AD'	'0D52'
8	'AB'	'95BC'	'6A43'
9	'04'	'C92E'	'36D1'
10	'E0'	'DFC3'	'203C'
11	'0B'	'BAE3'	'451C'
12	'E3'		'0F3D'
13	'BA'		'F0B8'

Annex D
(informative)

Examples of crypto command sequence

This Annex illustrates a few examples of possible crypto command sequences (see Figure D.1 to Figure D.3).

These examples are not exhaustive and do not preclude any other reasonable sequences.

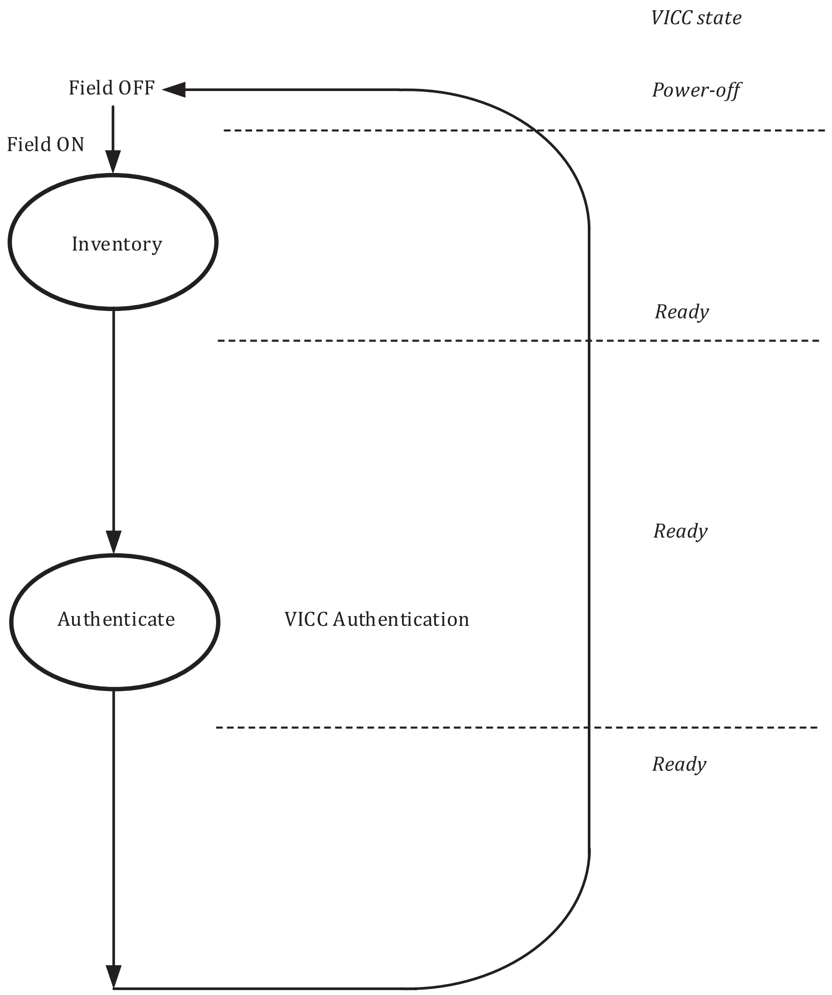


Figure D.1 — Example of VICC Authentication

If there is only one VICC in the field the Inventory may not be needed.

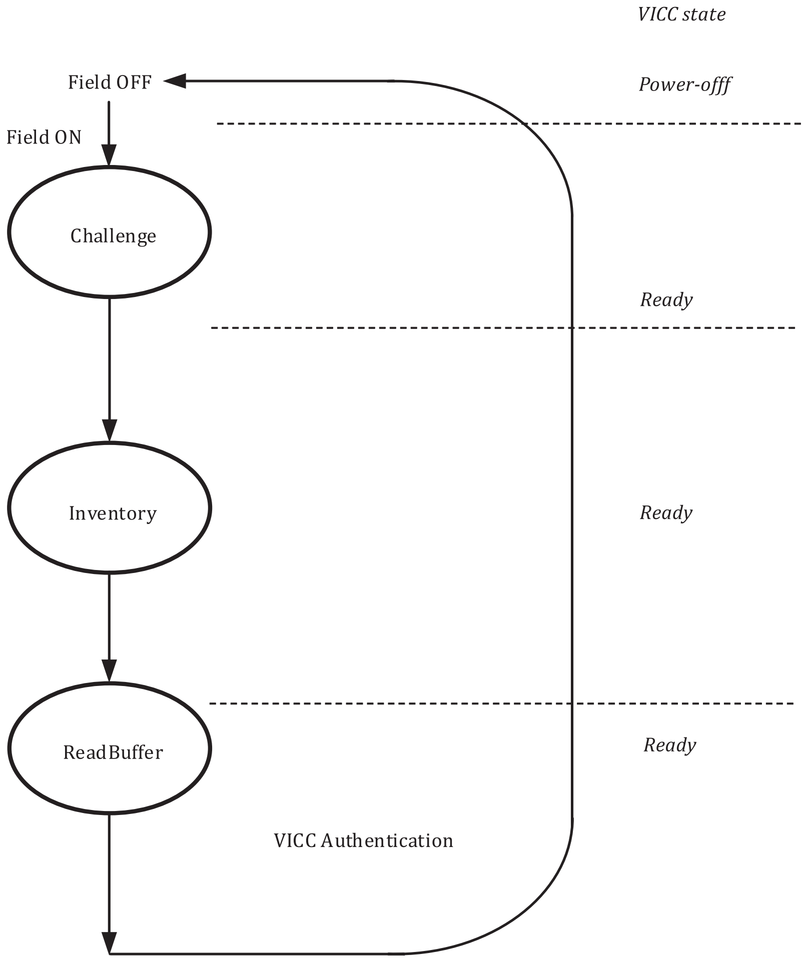


Figure D.2 — Example of VICC Authentication after a Challenge

If there is only one VICC in the field the Inventory may not be needed.

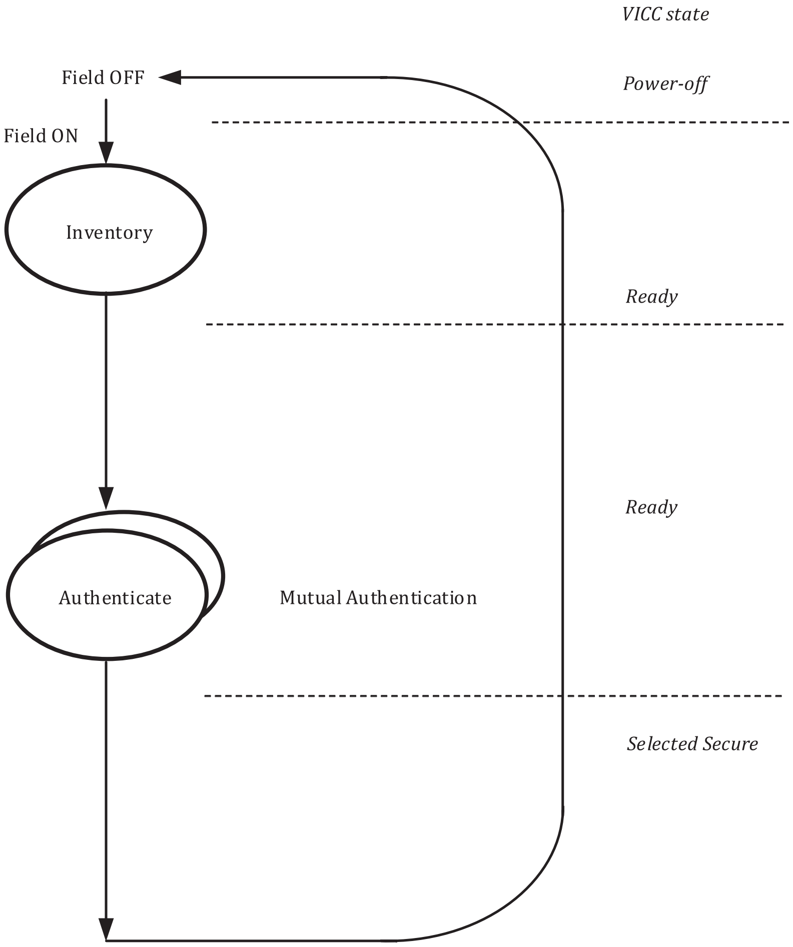


Figure D.3 — Example of a Mutual Authentication

Several iterations of the Authenticate command may take place as determined by the CSI.

Annex E
(normative)

List of legacy commands

WARNING — This list of Legacy commands shown in Table E.1 are those commands defined in earlier versions of this document where RFU bits may have been used in a proprietary way and therefore may not be compatible with this version of the document.

Table E.1 — Legacy use command code

Command code	Type	Function
'01'	Mandatory	Inventory
'02'	Mandatory	Stay quiet
'03' – '1F'	Mandatory	RFU
'20'	Optional	Read single block
'21'	Optional	Write single block
'22'	Optional	Lock block
'23'	Optional	Read multiple blocks
'24'	Optional	Write multiple blocks
'25'	Optional	Select
'26'	Optional	Reset to ready
'27'	Optional	Write AFI
'28'	Optional	Lock AFI
'29'	Optional	Write DSFID
'2A'	Optional	Lock DSFID
'2B'	Optional	Get system information
'2C'	Optional	Get multiple block security status
'30'	Optional	Extended read single block
'31'	Optional	Extended write single block
'32'	Optional	Extended lock block
'33'	Optional	Extended read multiple blocks
'34'	Optional	Extended write multiple blocks
'3C'	Optional	Extended get multiple block security status

Bibliography

[1] ISO/IEC 7810, *Identification cards — Physical characteristics*

[2] ISO/IEC 7811 (all parts), *Identification cards — Recording technique*

[3] ISO/IEC 7812-1, *Identification cards — Identification of issuers — Part 1: Numbering system*

[4] ISO/IEC 7812-2, *Identification cards — Identification of issuers — Part 2: Application and registration procedures*

[5] ISO/IEC 7813, *Information technology — Identification cards — Financial transaction cards*

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

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

[8] ISO/IEC 7816-6, *Identification cards — Integrated circuit cards — Part 6: Interindustry data elements for interchange*

[9] ISO/IEC 10373-7, *Identification cards — Test methods — Part 7: Vicinity cards*

[10] ISO/IEC 10536 (all parts), *Identification cards — Contactless integrated circuit(s) cards*

[11] ISO/IEC 14443 (all parts), *Cards and security devices for personal identification — Contactless proximity objects*

[12] ISO/IEC 15418, *Information technology — Automatic identification and data capture techniques — GS1 Application Identifiers and ASC MH10 Data Identifiers and maintenance*

[13] ISO/IEC 29167 (all parts), *Information technology — Automatic identification and data capture techniques*

Bureau of Indian Standards

BIS is a statutory institution established under the *Bureau of Indian Standards Act*, 2016 to promote harmonious development of the activities of standardization, marking and quality certification of goods and attending to connected matters in the country.

Copyright

Review of Indian Standards

Amendments are issued to standards as the need arises on the basis of comments. Standards are also reviewed periodically; a standard along with amendments is reaffirmed when such review indicates that no changes are needed; if the review indicates that changes are needed, it is taken up for revision. Users of Indian Standards should ascertain that they are in possession of the latest amendments or edition by referring to the website-www.bis.gov.in or www.standardsbis.in.

This Indian Standard has been developed from Doc No.: LITD 16 (17417).

Amendments Issued Since Publication

Amend No.	Date of Issue	Text Affected

BUREAU OF INDIAN STANDARDS

Headquarters:

Manak Bhavan, 9 Bahadur Shah Zafar Marg, New Delhi 110002
Telephones: 2323 0131, 2323 3375, 2323 9402 *Website*: www.bis.gov.in

Regional Offices: *Telephones*

Central : 601/A, Konnectus Tower -1, 6th Floor,
DMRC Building, Bhavbhuti Marg, New
Delhi 110002 — 2323 7617

Eastern : 8th Floor, Plot No 7/7 & 7/8, CP Block, Sector V,
Salt Lake, Kolkata, West Bengal 700091 — 2367 0012, 2320 9474

Northern : Plot No. 4-A, Sector 27-B, Madhya Marg,
Chandigarh 160019 — 265 9930

Southern : C.I.T. Campus, IV Cross Road, Taramani, Chennai 600113 — 2254 1442, 2254 1216

Western : Plot No. E-9, Road No.-8, MIDC, Andheri
(East), Mumbai 400093 — 2821 8093

Branches : AHMEDABAD. BENGALURU. BHOPAL. BHUBANESHWAR. CHANDIGARH. CHENNAI. COIMBATORE. DEHRADUN. DELHI. FARIDABAD. GHAZIABAD. GUWAHATI. HIMACHAL PRADESH. HUBLI. HYDERABAD. JAIPUR. JAMMU & KASHMIR. JAMSHEDPUR. KOCHI. KOLKATA. LUCKNOW. MADURAI. MUMBAI. NAGPUR. NOIDA. PANIPAT. PATNA. PUNE. RAIPUR. RAJKOT. SURAT. VISAKHAPATNAM.

Published by BIS, New Delhi